

2006



COLING • ACL

COLING • ACL 2006

21st International Conference on
Computational Linguistics
and
44th Annual Meeting of the
Association for Computational Linguistics

Proceedings of the Conference

Volume 2

17-21 July 2006
Sydney, Australia

Production and Manufacturing by
BPA Digital
11 Evans St
Burwood VIC 3125
AUSTRALIA

©2006 The Association for Computational Linguistics

Order copies of this and other ACL proceedings from:

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

ISBN 1-932432-66-3

Tree-to-String Alignment Template for Statistical Machine Translation

Yang Liu , Qun Liu , and Shouxun Lin

Institute of Computing Technology

Chinese Academy of Sciences

No.6 Kexueyuan South Road, Haidian District

P. O. Box 2704, Beijing, 100080, China

{yliu, liuqun, sxlin}@ict.ac.cn

Abstract

We present a novel translation model based on *tree-to-string alignment template* (TAT) which describes the alignment between a source parse tree and a target string. A TAT is capable of generating both terminals and non-terminals and performing reordering at both low and high levels. The model is linguistically syntax-based because TATs are extracted automatically from word-aligned, source side parsed parallel texts. To translate a source sentence, we first employ a parser to produce a source parse tree and then apply TATs to transform the tree into a target string. Our experiments show that the TAT-based model significantly outperforms Pharaoh, a state-of-the-art decoder for phrase-based models.

1 Introduction

Phrase-based translation models (Marcu and Wong, 2002; Koehn et al., 2003; Och and Ney, 2004), which go beyond the original IBM translation models (Brown et al., 1993)¹ by modeling translations of phrases rather than individual words, have been suggested to be the state-of-the-art in statistical machine translation by empirical evaluations.

In phrase-based models, phrases are usually strings of adjacent words instead of syntactic constituents, excelling at capturing local reordering and performing translations that are localized to

substrings that are common enough to be observed on training data. However, a key limitation of phrase-based models is that they fail to model reordering at the phrase level robustly. Typically, phrase reordering is modeled in terms of offset positions at the word level (Koehn, 2004; Och and Ney, 2004), making little or no direct use of syntactic information.

Recent research on statistical machine translation has led to the development of syntax-based models. Wu (1997) proposes Inversion Transduction Grammars, treating translation as a process of parallel parsing of the source and target language via a synchronized grammar. Alshawi et al. (2000) represent each production in parallel dependency tree as a finite transducer. Melamed (2004) formalizes machine translation problem as synchronous parsing based on multi-text grammars. Graehl and Knight (2004) describe training and decoding algorithms for both generalized tree-to-tree and tree-to-string transducers. Chiang (2005) presents a hierarchical phrase-based model that uses hierarchical phrase pairs, which are formally productions of a synchronous context-free grammar. Ding and Palmer (2005) propose a syntax-based translation model based on a probabilistic synchronous dependency insert grammar, a version of synchronous grammars defined on dependency trees. All these approaches, though different in formalism, make use of synchronous grammars or tree-based transduction rules to model both source and target languages.

Another class of approaches make use of syntactic information in the target language alone, treating the translation problem as a parsing problem. Yamada and Knight (2001) use a parser in the target language to train probabilities on a set of

¹The mathematical notation we use in this paper is taken from that paper: a source string $f_1^J = f_1, \dots, f_j, \dots, f_J$ is to be translated into a target string $e_1^I = e_1, \dots, e_i, \dots, e_I$. Here, I is the length of the target string, and J is the length of the source string.

operations that transform a target parse tree into a source string.

Paying more attention to source language analysis, Quirk et al. (2005) employ a source language dependency parser, a target language word segmentation component, and an unsupervised word alignment component to learn treelet translations from parallel corpus.

In this paper, we propose a statistical translation model based on tree-to-string alignment template which describes the alignment between a source parse tree and a target string. A TAT is capable of generating both terminals and non-terminals and performing reordering at both low and high levels. The model is linguistically syntax-based because TATs are extracted automatically from word-aligned, source side parsed parallel texts. To translate a source sentence, we first employ a parser to produce a source parse tree and then apply TATs to transform the tree into a target string.

One advantage of our model is that TATs can be automatically acquired to capture linguistically motivated reordering at both low (word) and high (phrase, clause) levels. In addition, the training of TAT-based model is less computationally expensive than tree-to-tree models. Similarly to (Galley et al., 2004), the tree-to-string alignment templates discussed in this paper are actually transformation rules. The major difference is that we model the syntax of the source language instead of the target side. As a result, the task of our decoder is to find the best target string while Galley’s is to seek the most likely target tree.

2 Tree-to-String Alignment Template

A tree-to-string alignment template z is a triple $\langle \tilde{T}, \tilde{S}, \tilde{A} \rangle$, which describes the alignment $\tilde{A}$ between a source parse tree $\tilde{T} = T(F_1^{J'})$ ² and a target string $\tilde{S} = E_1^{I'}$. A source string $F_1^{J'}$, which is the sequence of leaf nodes of $T(F_1^{J'})$, consists of both terminals (source words) and non-terminals (phrasal categories). A target string $E_1^{I'}$ is also composed of both terminals (target words) and non-terminals (placeholders). An alignment $\tilde{A}$ is defined as a subset of the Cartesian product of source and target symbol positions:

$$\tilde{A} \subseteq \{(j, i) : j = 1, \dots, J'; i = 1, \dots, I'\} \quad (1)$$

²We use $T(\cdot)$ to denote a parse tree. To reduce notational overhead, we use $T(z)$ to represent the parse tree in z . Similarly, $S(z)$ denotes the string in z .

Figure 1 shows three TATs automatically learned from training data. Note that when demonstrating a TAT graphically, we represent non-terminals in the target strings by blanks.

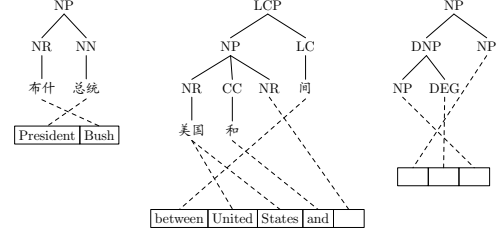


Figure 1: Examples of tree-to-string alignment templates obtained in training

In the following, we formally describe how to introduce tree-to-string alignment templates into probabilistic dependencies to model $Pr(e_1^I | f_1^J)$ ³.

In a first step, we introduce the hidden variable $T(f_1^J)$ that denotes a parse tree of the source sentence f_1^J :

$$Pr(e_1^I | f_1^J) = \sum_{T(f_1^J)} Pr(e_1^I, T(f_1^J) | f_1^J) \quad (2)$$

$$= \sum_{T(f_1^J)} Pr(T(f_1^J) | f_1^J) Pr(e_1^I | T(f_1^J), f_1^J) \quad (3)$$

Next, another hidden variable D is introduced to detach the source parse tree $T(f_1^J)$ into a sequence of K subtrees $\tilde{T}_1^K$ with a preorder transversal. We assume that each subtree $\tilde{T}_k$ produces a target string $\tilde{S}_k$. As a result, the sequence of subtrees $\tilde{T}_1^K$ produces a sequence of target strings $\tilde{S}_1^K$, which can be combined serially to generate the target sentence e_1^I . We assume that $Pr(e_1^I | D, T(f_1^J), f_1^J) \equiv Pr(\tilde{S}_1^K | \tilde{T}_1^K)$ because e_1^I is actually generated by the derivation of $\tilde{S}_1^K$. Note that we omit an explicit dependence on the detachment D to avoid notational overhead.

$$Pr(e_1^I | T(f_1^J), f_1^J) = \sum_D Pr(e_1^I, D | T(f_1^J), f_1^J) \quad (4)$$

$$= \sum_D Pr(D | T(f_1^J), f_1^J) Pr(e_1^I | D, T(f_1^J), f_1^J) \quad (5)$$

$$= \sum_D Pr(D | T(f_1^J), f_1^J) Pr(\tilde{S}_1^K | \tilde{T}_1^K) \quad (6)$$

$$= \sum_D Pr(D | T(f_1^J), f_1^J) \prod_{k=1}^K Pr(\tilde{S}_k | \tilde{T}_k) \quad (7)$$

³The notational convention will be as follows. We use the symbol $Pr(\cdot)$ to denote general probability distribution with no specific assumptions. In contrast, for model-based probability distributions, we use generic symbol $p(\cdot)$.

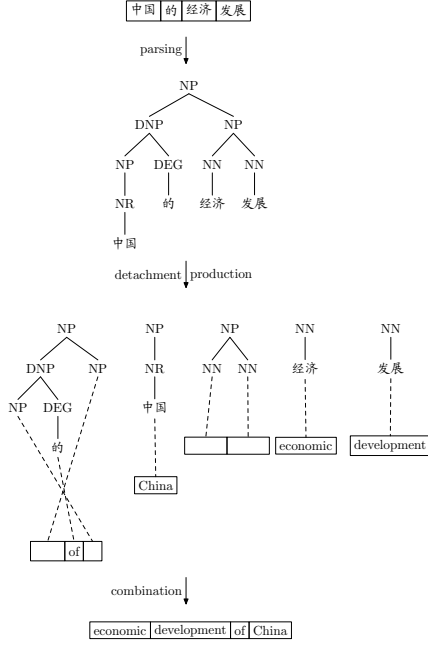


Figure 2: Graphic illustration for translation process

To further decompose $Pr(\tilde{S}|\tilde{T})$, the tree-to-string alignment template, denoted by the variable z , is introduced as a hidden variable.

$$Pr(\tilde{S}|\tilde{T}) = \sum_z Pr(\tilde{S}, z|\tilde{T}) \quad (8)$$

$$= \sum_z Pr(z|\tilde{T}) Pr(\tilde{S}|z, \tilde{T}) \quad (9)$$

Therefore, the TAT-based translation model can be decomposed into four sub-models:

1. parse model: $Pr(T(f_1^J)|f_1^J)$
2. detachment model: $Pr(D|T(f_1^J), f_1^J)$
3. TAT selection model: $Pr(z|\tilde{T})$
4. TAT application model: $Pr(\tilde{S}|z, \tilde{T})$

Figure 2 shows how TATs work to perform translation. First, the input source sentence is parsed. Next, the parse tree is detached into five subtrees with a preorder transversal. For each subtree, a TAT is selected and applied to produce a string. Finally, these strings are combined serially to generate the translation (we use X to denote the non-terminal):

$$\begin{aligned} X_1 &\Rightarrow X_2 \text{ of } X_3 \\ &\Rightarrow X_2 \text{ of China} \end{aligned}$$

$$\begin{aligned} &\Rightarrow X_3 \text{ } X_4 \text{ of China} \\ &\Rightarrow \text{economic } X_4 \text{ of China} \\ &\Rightarrow \text{economic development of China} \end{aligned}$$

Following Och and Ney (2002), we base our model on log-linear framework. Hence, all knowledge sources are described as feature functions that include the given source string f_1^J , the target string e_1^I , and hidden variables. The hidden variable $T(f_1^J)$ is omitted because we usually make use of only single best output of a parser. As we assume that all detachment have the same probability, the hidden variable D is also omitted. As a result, the model we actually adopt for experiments is limited because the parse, detachment, and TAT application sub-models are simplified.

$$\begin{aligned} &Pr(e_1^I, z_1^K | f_1^J) \\ &= \frac{\exp[\sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J, z_1^K)]}{\sum_{e_1^I, z_1^K} \exp[\sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J, z_1^K)]} \\ &\hat{e}_1^I = \operatorname{argmax}_{e_1^I, z_1^K} \left\{ \sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J, z_1^K) \right\} \end{aligned}$$

For our experiments we use the following seven feature functions⁴ that are analogous to default feature set of Pharaoh (Koehn, 2004). To simplify the notation, we omit the dependence on the hidden variables of the model.

$$\begin{aligned} h_1(e_1^I, f_1^J) &= \log \prod_{k=1}^K \frac{N(z) \cdot \delta(T(z), \tilde{T}_k)}{N(T(z))} \\ h_2(e_1^I, f_1^J) &= \log \prod_{k=1}^K \frac{N(z) \cdot \delta(T(z), \tilde{T}_k)}{N(S(z))} \\ h_3(e_1^I, f_1^J) &= \log \prod_{k=1}^K \text{lex}(T(z)|S(z)) \cdot \delta(T(z), \tilde{T}_k) \\ h_4(e_1^I, f_1^J) &= \log \prod_{k=1}^K \text{lex}(S(z)|T(z)) \cdot \delta(T(z), \tilde{T}_k) \\ h_5(e_1^I, f_1^J) &= K \\ h_6(e_1^I, f_1^J) &= \log \prod_{i=1}^I p(e_i | e_{i-2}, e_{i-1}) \\ h_7(e_1^I, f_1^J) &= I \end{aligned}$$

⁴When computing lexical weighting features (Koehn et al., 2003), we take only terminals into account. If there are no terminals, we set the feature value to 1. We use $\text{lex}(\cdot)$ to denote lexical weighting. We denote the number of TATs used for decoding by K and the length of target string by I .

Tree	String	Alignment
(NR 布什)	Bush	1:1
(NN 总统)	President	1:1
(VV 发表)	made	1:1
(NN 演讲)	speech	1:1
(NP (NR) (NN))	$X_1 \mid X_2$	1:2 2:1
(NP (NR 布什) (NN))	$X \mid \text{Bush}$	1:2 2:1
(NP (NR) (NN 总统))	President $\mid X$	1:2 2:1
(NP (NR 布什) (NN 总统))	President $\mid \text{Bush}$	1:2 2:1
(VP (VV) (NN))	$X_1 \mid a \mid X_2$	1:1 2:3
(VP (VV 发表) (NN))	made $\mid a \mid X$	1:1 2:3
(VP (VV) (NN 演讲))	$X \mid a \mid \text{speech}$	1:1 2:3
(VP (VV 发表) (NN 演讲))	made $\mid a \mid \text{speech}$	1:1 2:3
(IP (NP) (VP))	$X_1 \mid X_2$	1:1 2:2

Table 1: Examples of TATs extracted from the TSA in Figure 3 with $h = 2$ and $c = 2$

3 Training

To extract tree-to-string alignment templates from a word-aligned, source side parsed sentence pair $\langle T(f_1^J), e_1^I, A \rangle$, we need first identify TSAs (Tree-String-Alignment) using similar criterion as suggested in (Och and Ney, 2004). A TSA is a triple $\langle T(f_{j_1}^{j_2}), e_{i_1}^{i_2}, \bar{A} \rangle$ that is in accordance with the following constraints:

1. $\forall (i, j) \in A : i_1 \leq i \leq i_2 \leftrightarrow j_1 \leq j \leq j_2$
2. $T(f_{j_1}^{j_2})$ is a subtree of $T(f_1^J)$

Given a TSA $\langle T(f_{j_1}^{j_2}), e_{i_1}^{i_2}, \bar{A} \rangle$, a triple $\langle T(f_{j_3}^{j_4}), e_{i_3}^{i_4}, \hat{A} \rangle$ is its *sub TSA* if and only if:

1. $T(f_{j_3}^{j_4}), e_{i_3}^{i_4}, \hat{A}$ is a TSA
2. $T(f_{j_3}^{j_4})$ is rooted at the direct descendant of the root node of $T(f_{j_1}^{j_2})$
3. $i_1 \leq i_3 \leq i_4 \leq i_2$
4. $\forall (i, j) \in \bar{A} : i_3 \leq i \leq i_4 \leftrightarrow j_3 \leq j \leq j_4$

Basically, we extract TATs from a TSA $\langle T(f_{j_1}^{j_2}), e_{i_1}^{i_2}, \bar{A} \rangle$ using the following two rules:

1. If $T(f_{j_1}^{j_2})$ contains only one node, then $\langle T(f_{j_1}^{j_2}), e_{i_1}^{i_2}, \bar{A} \rangle$ is a TAT
2. If the height of $T(f_{j_1}^{j_2})$ is greater than one, then build TATs using those extracted from sub TSAs of $\langle T(f_{j_1}^{j_2}), e_{i_1}^{i_2}, \bar{A} \rangle$.

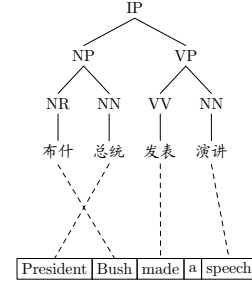


Figure 3: An example of TSA

Usually, we can extract a very large amount of TATs from training data using the above rules, making both training and decoding very slow. Therefore, we impose three restrictions to reduce the magnitude of extracted TATs:

1. A third constraint is added to the definition of TSA:
 $\exists j', j'' : j_1 \leq j' \leq j_2 \text{ and } j_1 \leq j'' \leq j_2$
and $(i_1, j') \in \bar{A}$ and $(i_2, j'') \in \bar{A}$
This constraint requires that both the first and last symbols in the target string must be aligned to some source symbols.
2. The height of $T(z)$ is limited to no greater than h .
3. The number of direct descendants of a node of $T(z)$ is limited to no greater than c .

Table 1 shows the TATs extracted from the TSA in Figure 3 with $h = 2$ and $c = 2$.

As we restrict that $T(f_{j_1}^{j_2})$ must be a subtree of $T(f_1^J)$, TATs may be treated as syntactic hierar-

chical phrase pairs (Chiang, 2005) with tree structure on the source side. At the same time, we face the risk of losing some useful non-syntactic phrase pairs. For example, the phrase pair

布什 总统 发表 $\longleftrightarrow$ President Bush made

can never be obtained in form of TAT from the TSA in Figure 3 because there is no subtree for that source string.

4 Decoding

We approach the decoding problem as a bottom-up beam search.

To translate a source sentence, we employ a parser to produce a parse tree. Moving bottom-up through the source parse tree, we compute a list of *candidate translations* for the input subtree rooted at each node with a postorder transversal. Candidate translations of subtrees are placed in stacks. Figure 4 shows the organization of candidate translation stacks.

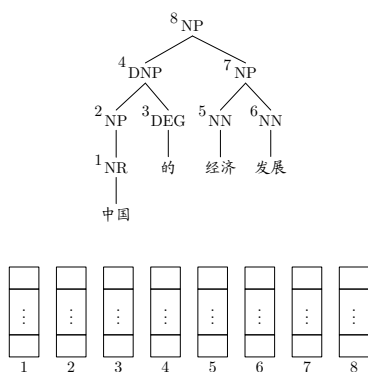


Figure 4: Candidate translations of subtrees are placed in stacks according to the root index set by postorder transversal

A candidate translation contains the following information:

1. the partial translation
2. the accumulated feature values
3. the accumulated probability

A TAT z is *usable* to a parse tree T if and only if $T(z)$ is rooted at the root of T and *covers* part of nodes of T . Given a parse tree T , we find all usable TATs. Given a usable TAT z , if $T(z)$ is equal to T , then $S(z)$ is a candidate translation of T . If $T(z)$ covers only a portion of T , we have

to compute a list of candidate translations for T by replacing the non-terminals of $S(z)$ with candidate translations of the corresponding uncovered subtrees.

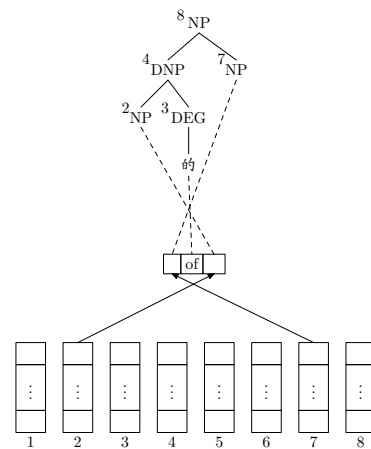


Figure 5: Candidate translation construction

For example, when computing the candidate translations for the tree rooted at node 8, the TAT used in Figure 5 covers only a portion of the parse tree in Figure 4. There are two uncovered subtrees that are rooted at node 2 and node 7 respectively. Hence, we replace the third symbol with the candidate translations in stack 2 and the first symbol with the candidate translations in stack 7. At the same time, the feature values and probabilities are also accumulated for the new candidate translations.

To speed up the decoder, we limit the search space by reducing the number of TATs used for each input node. There are two ways to limit the TAT table size: by a fixed limit (*tatTable-limit*) of how many TATs are retrieved for each input node, and by a probability threshold (*tatTable-threshold*) that specify that the TAT probability has to be above some value. On the other hand, instead of keeping the full list of candidates for a given node, we keep a top-scoring subset of the candidates. This can also be done by a fixed limit (*stack-limit*) or a threshold (*stack-threshold*). To perform recombination, we combine candidate translations that share the same leading and trailing bigrams in each stack.

5 Experiments

Our experiments were on Chinese-to-English translation. The training corpus consists of 31, 149 sentence pairs with 843, 256 Chinese words and

System	Features	BLEU4
Pharaoh	$d + \phi(e f)$	0.0573 ± 0.0033
	$d + lm + \phi(e f) + wp$	0.2019 ± 0.0083
	$d + lm + \phi(f e) + lex(f e) + \phi(e f) + lex(e f) + pp + wp$	0.2089 ± 0.0089
Lynx	h_1	0.1639 ± 0.0077
	$h_1 + h_6 + h_7$	0.2100 ± 0.0089
	$h_1 + h_2 + h_3 + h_4 + h_5 + h_6 + h_7$	0.2178 ± 0.0080

Table 2: Comparison of Pharaoh and Lynx with different feature settings on the test corpus

949,583 English words. For the language model, we used SRI Language Modeling Toolkit (Stolcke, 2002) to train a trigram model with modified Kneser-Ney smoothing (Chen and Goodman, 1998) on the 31,149 English sentences. We selected 571 short sentences from the 2002 NIST MT Evaluation test set as our development corpus, and used the 2005 NIST MT Evaluation test set as our test corpus. We evaluated the translation quality using the BLEU metric (Papineni et al., 2002), as calculated by mteval-v11b.pl with its default setting except that we used case-sensitive matching of n -grams.

5.1 Pharaoh

The baseline system we used for comparison was Pharaoh (Koehn et al., 2003; Koehn, 2004), a freely available decoder for phrase-based translation models:

$$p(e|f) = p_\phi(f|e)^{\lambda_\phi} \times p_{LM}(e)^{\lambda_{LM}} \times p_D(e, f)^{\lambda_D} \times \omega^{\text{length}(e)\lambda_W(e)} \quad (10)$$

We ran GIZA++ (Och and Ney, 2000) on the training corpus in both directions using its default setting, and then applied the refinement rule “diag-and” described in (Koehn et al., 2003) to obtain a single many-to-many word alignment for each sentence pair. After that, we used some heuristics, which including rule-based translation of numbers, dates, and person names, to further improve the alignment accuracy.

Given the word-aligned bilingual corpus, we obtained 1,231,959 bilingual phrases (221,453 used on test corpus) using the training toolkits publicly released by Philipp Koehn with its default setting.

To perform minimum error rate training (Och, 2003) to tune the feature weights to maximize the system’s BLEU score on development set, we used optimizeV5IBMBLEU.m (Venugopal and Vogel,

2005). We used default pruning settings for Pharaoh except that we set the distortion limit to 4.

5.2 Lynx

On the same word-aligned training data, it took us about one month to parse all the 31,149 Chinese sentences using a Chinese parser written by Deyi Xiong (Xiong et al., 2005). The parser was trained on articles 1 – 270 of Penn Chinese Treebank version 1.0 and achieved 79.4% (F1 measure) as well as a 4.4% relative decrease in error rate. Then, we performed TAT extraction described in section 3 with $h = 3$ and $c = 5$ and obtained 350,575 TATs (88,066 used on test corpus). To run our decoder Lynx on development and test corpus, we set tatTable-limit = 20, tatTable-threshold = 0, stack-limit = 100, and stack-threshold = 0.00001.

5.3 Results

Table 2 shows the results on test set using Pharaoh and Lynx with different feature settings. The 95% confidence intervals were computed using Zhang’s significance tester (Zhang et al., 2004). We modified it to conform to NIST’s current definition of the BLEU brevity penalty. For Pharaoh, eight features were used: distortion model d , a trigram language model lm , phrase translation probabilities $\phi(f|e)$ and $\phi(e|f)$, lexical weightings $lex(f|e)$ and $lex(e|f)$, phrase penalty pp , and word penalty wp . For Lynx, seven features described in section 2 were used. We find that Lynx outperforms Pharaoh with all feature settings. With full features, Lynx achieves an absolute improvement of 0.006 over Pharaoh (3.1% relative). This difference is statistically significant ($p < 0.01$). Note that Lynx made use of only 88,066 TATs on test corpus while 221,453 bilingual phrases were used for Pharaoh.

The feature weights obtained by minimum er-

System	Features							
	d	lm	$\phi(f e)$	$\text{lex}(f e)$	$\phi(e f)$	$\text{lex}(e f)$	pp	wp
Pharaoh	0.0476	0.1386	0.0611	0.0459	0.1723	0.0223	0.3122	-0.2000
Lynx	-	0.3735	0.0061	0.1081	0.1656	0.0022	0.0824	0.2620

Table 3: Feature weights obtained by minimum error rate training on the development corpus

	BLEU4
tat	0.2178 ± 0.0080
tat + bp	0.2240 ± 0.0083

Table 4: Effect of using bilingual phrases for Lynx

ror rate training for both Pharaoh and Lynx are shown in Table 3. We find that $\phi(f|e)$ (i.e. h_2) is not a helpful feature for Lynx. The reason is that we use only a single non-terminal symbol instead of assigning phrasal categories to the target string. In addition, we allow the target string consists of only non-terminals, making translation decisions not always based on lexical evidence.

5.4 Using bilingual phrases

It is interesting to use bilingual phrases to strengthen the TAT-based model. As we mentioned before, some useful non-syntactic phrase pairs can never be obtained in form of TAT because we restrict that there must be a corresponding parse tree for the source phrase. Moreover, it takes more time to obtain TATs than bilingual phrases on the same training data because parsing is usually very time-consuming.

Given an input subtree $T(F_{j_1}^{j_2})$, if $F_{j_1}^{j_2}$ is a string of terminals, we find all bilingual phrases that the source phrase is equal to $F_{j_1}^{j_2}$. Then we build a TAT for each bilingual phrase $\langle f_1^{j'}, e_1^{j'}, \hat{A} \rangle$: the tree of the TAT is $T(F_{j_1}^{j_2})$, the string is $e_1^{j'}$, and the alignment is $\hat{A}$. If a TAT built from a bilingual phrase is the same with a TAT in the TAT table, we prefer to the greater translation probabilities.

Table 4 shows the effect of using bilingual phrases for Lynx. Note that these bilingual phrases are the same with those used for Pharaoh.

5.5 Results on large data

We also conducted an experiment on large data to further examine our design philosophy. The training corpus contains 2.6 million sentence pairs. We used all the data to extract bilingual phrases and a portion of 800K pairs to obtain TATs. Two tri-

gram language models were used for Lynx. One was trained on the 2.6 million English sentences and another was trained on the first 1/3 of the Xinhua portion of Gigaword corpus. We also included rule-based translations of named entities, dates, and numbers. By making use of these data, Lynx achieves a BLEU score of 0.2830 on the 2005 NIST Chinese-to-English MT evaluation test set, which is a very promising result for linguistically syntax-based models.

6 Conclusion

In this paper, we introduce tree-to-string alignment templates, which can be automatically learned from syntactically-annotated training data. The TAT-based translation model improves translation quality significantly compared with a state-of-the-art phrase-based decoder. Treated as special TATs without tree on the source side, bilingual phrases can be utilized for the TAT-based model to get further improvement.

It should be emphasized that the restrictions we impose on TAT extraction limit the expressive power of TAT. Preliminary experiments reveal that removing these restrictions does improve translation quality, but leads to large memory requirements. We feel that both parsing and word alignment qualities have important effects on the TAT-based model. We will retrain the Chinese parser on Penn Chinese Treebank version 5.0 and try to improve word alignment quality using log-linear models as suggested in (Liu et al., 2005).

Acknowledgement

This work is supported by National High Technology Research and Development Program contract “Generally Technical Research and Basic Database Establishment of Chinese Platform”(Subject No. 2004AA114010). We are grateful to Deyi Xiong for providing the parser and Haitao Mi for making the parser more efficient and robust. Thanks to Dr. Yajuan Lv for many helpful comments on an earlier draft of this paper.

References

- Hiyan Alshawi, Srinivas Bangalore, and Shona Douglas. 2000. Learning dependency translation models as collections of finite-state head transducers. *Computational Linguistics*, 26(1):45-60.
- 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.
- Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University Center for Research in Computing Technology.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of 43rd Annual Meeting of the ACL*, pages 263-270.
- Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insert grammars. In *Proceedings of 43rd Annual Meeting of the ACL*, pages 541-548.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of NAACL-HLT 2004*, pages 273-280.
- Jonathan Graehl and Kevin Knight. 2004. Training tree transducers. In *Proceedings of NAACL-HLT 2004*, pages 105-112.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL 2003*, pages 127-133.
- 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*, pages 115-124.
- Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *Proceedings of 43rd Annual Meeting of the ACL*, pages 459-466.
- Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 133-139.
- Dan Melamed. 2004. Statistical machine translation by parsing. In *Proceedings of 42nd Annual Meeting of the ACL*, pages 653-660.
- Franz J. Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of 38th Annual Meeting of the ACL*, pages 440-447.
- Franz J. Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of 40th Annual Meeting of the ACL*, pages 295-302.
- Franz J. Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417-449.
- Franz J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of 41st Annual Meeting of the ACL*, pages 160-167.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the ACL*, pages 311-318.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of 43rd Annual Meeting of the ACL*, pages 271-279.
- Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*, volume 2, pages 901-904.
- Ashish Venugopal and Stephan Vogel. 2005. Considerations in maximum mutual information and minimum classification error training for statistical machine translation. In *Proceedings of the Tenth Conference of the European Association for Machine Translation (EAMT-05)*.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377-403.
- Deyi Xiong, Shuanglong Li, Qun Liu, Shouxun Lin, and Yueliang Qian. 2005. Parsing the Penn Chinese treebank with semantic knowledge. In *Proceedings of IJCNLP 2005*, pages 70-81.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of 39th Annual Meeting of the ACL*, pages 523-530.
- Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system? In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*, pages 2051-2054.

Incorporating speech recognition confidence into discriminative named entity recognition of speech data

Katsuhito Sudoh Hajime Tsukada Hideki Isozaki

NTT Communication Science Laboratories

Nippon Telegraph and Telephone Corporation

2-4 Hikaridai, Seika-cho, Keihanna Science City, Kyoto 619-0237, Japan

{sudoh, tsukada, isozaki}@cs-lab.kecl.ntt.co.jp

Abstract

This paper proposes a named entity recognition (NER) method for speech recognition results that uses confidence on automatic speech recognition (ASR) as a feature. The ASR confidence feature indicates whether each word has been correctly recognized. The NER model is trained using ASR results with named entity (NE) labels as well as the corresponding transcriptions with NE labels. In experiments using support vector machines (SVMs) and speech data from Japanese newspaper articles, the proposed method outperformed a simple application of text-based NER to ASR results in NER F-measure by improving precision. These results show that the proposed method is effective in NER for noisy inputs.

1 Introduction

As network bandwidths and storage capacities continue to grow, a large volume of speech data including broadcast news and PodCasts is becoming available. These data are important information sources as well as such text data as newspaper articles and WWW pages. Speech data as information sources are attracting a great deal of interest, such as DARPA's global autonomous language exploitation (GALE) program. We also aim to use them for information extraction (IE), question answering, and indexing.

Named entity recognition (NER) is a key technique for IE and other natural language processing tasks. Named entities (NEs) are the proper expressions for things such as peoples' names, locations' names, and dates, and NER identifies those

expressions and their categories. Unlike text data, speech data introduce automatic speech recognition (ASR) error problems to NER. Although improvements to ASR are needed, developing a robust NER for noisy word sequences is also important. In this paper, we focus on the NER of ASR results and discuss the suppression of ASR error problems in NER.

Most previous studies of the NER of speech data used generative models such as hidden Markov models (HMMs) (Miller et al., 1999; Palmer and Ostendorf, 2001; Horlock and King, 2003b; Béchet et al., 2004; Favre et al., 2005). On the other hand, in text-based NER, better results are obtained using discriminative schemes such as maximum entropy (ME) models (Borthwick, 1999; Chieu and Ng, 2003), support vector machines (SVMs) (Isozaki and Kazawa, 2002), and conditional random fields (CRFs) (McCallum and Li, 2003). Zhai et al. (2004) applied a text-level ME-based NER to ASR results. These models have an advantage in utilizing various features, such as part-of-speech information, character types, and surrounding words, which may be overlapped, while overlapping features are hard to use in HMM-based models.

To deal with ASR error problems in NER, Palmer and Ostendorf (2001) proposed an HMM-based NER method that explicitly models ASR errors using ASR confidence and rejects erroneous word hypotheses in the ASR results. Such rejection is especially effective when ASR accuracy is relatively low because many misrecognized words may be extracted as NEs, which would decrease NER precision.

Motivated by these issues, we extended their approach to discriminative models and propose an NER method that deals with ASR errors as fea-

tures. We use NE-labeled ASR results for training to incorporate the features into the NER model as well as the corresponding transcriptions with NE labels. In testing, ASR errors are identified by ASR confidence scores and are used for the NER. In experiments using SVM-based NER and speech data from Japanese newspaper articles, the proposed method increased the NER F-measure, especially in precision, compared to simply applying text-based NER to the ASR results.

2 SVM-based NER

NER is a kind of chunking problem that can be solved by classifying words into NE classes that consist of name categories and such chunking states as **PERSON-BEGIN** (the beginning of a person’s name) and **LOCATION-MIDDLE** (the middle of a location’s name). Many discriminative methods have been applied to NER, such as decision trees (Sekine et al., 1998), ME models (Borthwick, 1999; Chieu and Ng, 2003), and CRFs (McCallum and Li, 2003). In this paper, we employ an SVM-based NER method in the following way that showed good NER performance in Japanese (Isozaki and Kazawa, 2002).

We define three features for each word: the word itself, its part-of-speech tag, and its character type. We also use those features for the two preceding and succeeding words for context dependence and use 15 features when classifying a word. Each feature is represented by a binary value (1 or 0), for example, “whether the previous word is *Japan*,” and each word is classified based on a long binary vector where only 15 elements are 1.

We have two problems when solving NER using SVMs. One, SVMs can solve only a two-class problem. We reduce multi-class problems of NER to a group of two-class problems using the *one-against-all* approach, where each SVM is trained to distinguish members of a class (e.g., **PERSON-BEGIN**) from non-members (**PERSON-MIDDLE**, **MONEY-BEGIN**, ...). In this approach, two or more classes may be assigned to a word or no class may be assigned to a word. To avoid these situations, we choose class c that has the largest SVM output score $g_c(x)$ among all others.

The other is that the NE label sequence must be consistent; for example, **ARTIFACT-END** must follow **ARTIFACT-BEGIN** or

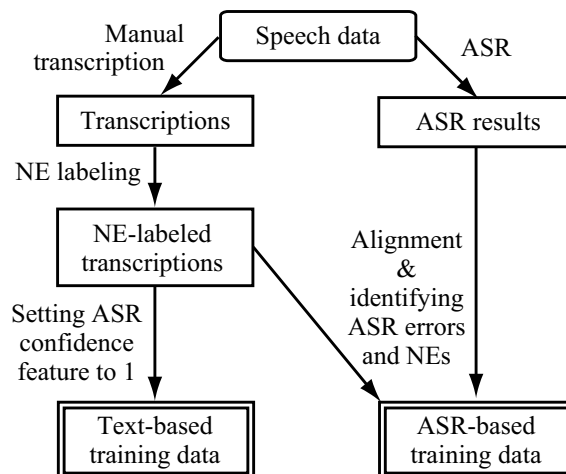


Figure 1: Procedure for preparing training data.

ARTIFACT-MIDDLE. We use a Viterbi search to obtain the best and consistent NE label sequence after classifying all words in a sentence, based on probability-like values obtained by applying sigmoid function $s_n(x) = 1/(1 + \exp(-\beta_n x))$ to SVM output score $g_c(x)$.

3 Proposed method

3.1 Incorporating ASR confidence into NER

In the NER of ASR results, ASR errors cause NEs to be missed and erroneous NEs to be recognized. If one or more words constituting an NE are misrecognized, we cannot recognize the correct NE. Even if all words constituting an NE are correctly recognized, we may not recognize the correct NE due to ASR errors on context words. To avoid this problem, we model ASR errors using additional features that indicate whether each word is correctly recognized. Our NER model is trained using ASR results with a feature, where feature values are obtained through alignment to the corresponding transcriptions. In testing, we estimate feature values using ASR confidence scores. In this paper, this feature is called the *ASR confidence feature*.

Note that we only aim to identify NEs that are correctly recognized by ASR, and NEs containing ASR errors are not regarded as NEs. Utilizing erroneous NEs is a more difficult problem that is beyond the scope of this paper.

3.2 Training NER model

Figure 1 illustrates the procedure for preparing training data from speech data. First, the speech

data are manually transcribed and automatically recognized by the ASR. Second, we label NEs in the transcriptions and then set the ASR confidence feature values to 1 because the words in the transcriptions are regarded as correctly recognized words. Finally, we align the ASR results to the transcriptions to identify ASR errors for the ASR confidence feature values and to label correctly recognized NEs in the ASR results. Note that we label the NEs in the ASR results that exist in the same positions as the transcriptions. If a part of an NE is misrecognized, the NE is ignored, and all words for the NE are labeled as non-NE words (OTHER). Examples of text-based and ASR-based training data are shown in Tables 1 and 2. Since the name *Murayama Tomiichi* in Table 1 is misrecognized in ASR, the correctly recognized word *Murayama* is also labeled OTHER in Table 2. Another approach can be considered, where misrecognized words are replaced by word error symbols such as those shown in Table 3. In this case, those words are rejected, and those part-of-speech and character type features are not used in NER.

3.3 ASR confidence scoring for using the proposed NER model

ASR confidence scoring is an important technique in many ASR applications, and many methods have been proposed including using word posterior probabilities on word graphs (Wessel et al., 2001), integrating several confidence measures using neural networks (Schaaf and Kemp, 1997), using linear discriminant analysis (Kamppari and Hazen, 2000), and using SVMs (Zhang and Rudnicki, 2001).

Word posterior probability is a commonly used and effective ASR confidence measure. Word posterior probability $p([w; \tau, t] | \mathbf{X})$ of word w at time interval $[\tau, t]$ for speech signal $\mathbf{X}$ is calculated as follows (Wessel et al., 2001):

$$p([w; \tau, t] | \mathbf{X}) = \sum_{\mathbf{W} \in \mathbf{W}_{[w; \tau, t]}} \frac{\{p(\mathbf{X} | \mathbf{W}) (p(\mathbf{W}))^\beta\}^\alpha}{p(\mathbf{X})}, \quad (1)$$

where $\mathbf{W}$ is a sentence hypothesis, $\mathbf{W}_{[w; \tau, t]}$ is the set of sentence hypotheses that include w in $[\tau, t]$, $p(\mathbf{X} | \mathbf{W})$ is a acoustic model score, $p(\mathbf{W})$ is a language model score, α is a scaling parameter ($\alpha < 1$), and β is a language model weight. α is used for scaling the large dynamic range of

Word	Confidence	NE label
<i>Murayama</i>	1	PERSON-BEGIN
<i>Tomiichi</i>	1	PERSON-END
<i>shusho</i>	1	OTHER
<i>wa</i>	1	OTHER
<i>nento</i>	1	DATE-SINGLE

Table 1: An example of text-based training data.

Word	Confidence	NE label
<i>Murayama</i>	1	OTHER
<i>shi</i>	0	OTHER
<i>ni</i>	0	OTHER
<i>ichi</i>	0	OTHER
<i>shiyo</i>	0	OTHER
<i>wa</i>	1	OTHER
<i>nento</i>	1	DATE-SINGLE

Table 2: An example of ASR-based training data.

Word	Confidence	NE label
<i>Murayama</i>	1	OTHER
(error)	0	OTHER
(error)	0	OTHER
(error)	0	OTHER
(error)	0	OTHER
<i>wa</i>	1	OTHER
<i>nento</i>	1	DATE-SINGLE

Table 3: An example of ASR-based training data with word error symbols.

$p(\mathbf{X} | \mathbf{W}) (p(\mathbf{W}))^\beta$ to avoid a few of the top hypotheses dominating posterior probabilities. $p(\mathbf{X})$ is approximated by the sum over all sentence hypotheses and is denoted as

$$p(\mathbf{X}) = \sum_{\mathbf{W}} \{p(\mathbf{X} | \mathbf{W}) (p(\mathbf{W}))^\beta\}^\alpha. \quad (2)$$

$p([w; \tau, t] | \mathbf{X})$ can be efficiently calculated using a forward-backward algorithm.

In this paper, we use SVMs for ASR confidence scoring to achieve a better performance than when using word posterior probabilities as ASR confidence scores. SVMs are trained using ASR results, whose errors are known through their alignment to their reference transcriptions. The following features are used for confidence scoring: the word itself, its part-of-speech tag, and its word posterior probability; those of the two preceding and succeeding words are also used. The word itself and its part-of-speech are also represented

by a set of binary values, the same as with an SVM-based NER. Since all other features are binary, we reduce real-valued word posterior probability p to ten binary features for simplicity: (if $0 < p \leq 0.1$, if $0.1 < p \leq 0.2$, ... , and if $0.9 < p \leq 1.0$). To normalize SVMs' output scores for ASR confidence, we use a sigmoid function $s_w(x) = 1/(1 + \exp(-\beta_w x))$. We use these normalized scores as ASR confidence scores. Although a large variety of features have been proposed in previous studies, we use only these simple features and reserve the other features for further studies.

Using the ASR confidence scores, we estimate whether each word is correctly recognized. If the ASR confidence score of a word is greater than threshold t_w , the word is estimated as correct, and we set the ASR confidence feature value to 1; otherwise we set it to 0.

3.4 Rejection at the NER level

We use the ASR confidence feature to suppress ASR error problems; however, even text-based NERs sometimes make errors. NER performance is a trade-off between missing correct NEs and accepting erroneous NEs, and requirements differ by task. Although we can tune the parameters in training SVMs to control the trade-off, it seems very hard to find appropriate values for all the SVMs. We use a simple NER-level rejection by modifying the SVM output scores for the non-NE class (OTHER). We add constant offset value t_o to each SVM output score for OTHER. With a large t_o , OTHER becomes more desirable than the other NE classes, and many words are classified as non-NE words and *vice versa*. Therefore, t_o works as a parameter for NER-level rejection. This approach can also be applied to text-based NER.

4 Experiments

We conducted the following experiments related to the NER of speech data to investigate the performance of the proposed method.

4.1 Setup

In the experiment, we simulated the procedure shown in Figure 1 using speech data from the NE-labeled text corpus. We used the training data of the Information Retrieval and Extraction Exercise (IREX) workshop (Sekine and Eriguchi, 2000) as the text corpus, which consisted of 1,174

Japanese newspaper articles (10,718 sentences) and 18,200 NEs in eight categories (artifact, organization, location, person, date, time, money, and percent). The sentences were read by 106 speakers (about 100 sentences per speaker), and the recorded speech data were used for the experiments. The experiments were conducted with 5-fold cross validation, using 80% of the 1,174 articles and the ASR results of the corresponding speech data for training SVMs (both for ASR confidence scoring and for NER) and the rest for the test.

We tokenized the sentences into words and tagged the part-of-speech information using the Japanese morphological analyzer ChaSen¹ 2.3.3 and then labeled the NEs. Unreadable tokens such as parentheses were removed in tokenization. After tokenization, the text corpus had 264,388 words of 60 part-of-speech types. Since three different kinds of characters are used in Japanese, the character types used as features included: *single-kanji* (words written in a single Chinese character), *all-kanji* (longer words written in Chinese characters), *hiragana* (words written in *hiragana* Japanese phonograms), *katakana* (words written in *katakana* Japanese phonograms), *number*, *single-capital* (words with a single capitalized letter), *all-capital*, *capitalized* (only the first letter is capitalized), *roman* (other roman character words), and *others* (all other words). We used all the features that appeared in each training set (no feature selection was performed). The chunking states included in the NE classes were: *BEGIN* (beginning of a NE), *MIDDLE* (middle of a NE), *END* (ending of a NE), and *SINGLE* (a single-word NE). There were 33 NE classes (eight categories * four chunking states + OTHER), and therefore we trained 33 SVMs to distinguish words of a class from words of other classes. For NER, we used an SVM-based chunk annotator YamCha² 0.33 with a quadratic kernel $(1 + \vec{x} \cdot \vec{y})^2$ and a soft margin parameter of SVMs $C=0.1$ for training and applied sigmoid function $s_n(x)$ with $\beta_n=1.0$ and Viterbi search to the SVMs' outputs. These parameters were experimentally chosen using the test set.

We used an ASR engine (Hori et al., 2004) with a speaker-independent acoustic model. The lan-

¹<http://chasen.naist.jp/hiki/ChaSen/> (in Japanese)

²<http://www.chasen.org/~taku/software/yamcha/>

guage model was a word 3-gram model, trained using other Japanese newspaper articles (about 340 M words) that were also tokenized using ChaSen. The vocabulary size of the word 3-gram model was 426,023. The test-set perplexity over the text corpus was 76.928. The number of out-of-vocabulary words was 1,551 (0.587%). 223 (1.23%) NEs in the text corpus contained such out-of-vocabulary words, so those NEs could not be correctly recognized by ASR. The scaling parameter α was set to 0.01, which showed the best ASR error estimation results using word posterior probabilities in the test set in terms of receiver operator characteristic (ROC) curves. The language model weight β was set to 15, which is a commonly used value in our ASR system. The word accuracy obtained using our ASR engine for the overall dataset was 79.45%. In the ASR results, 82.00% of the NEs in the text corpus remained. Figure 2 shows the ROC curves of ASR error estimation for the overall five cross-validation test sets, using SVM-based ASR confidence scoring and word posterior probabilities as ASR confidence scores, where

$$\begin{aligned} \text{True positive rate} &= \frac{\# \text{ correctly recognized words estimated as correct}}{\# \text{ correctly recognized words}} \\ \text{False positive rate} &= \frac{\# \text{ misrecognized words estimated as correct}}{\# \text{ misrecognized words}}. \end{aligned}$$

In SVM-based ASR confidence scoring, we used the quadratic kernel and $C=0.01$. Parameter β_w of sigmoid function $s_w(x)$ was set to 1.0. These parameters were also experimentally chosen. SVM-based ASR confidence scoring showed better performance in ASR error estimation than simple word posterior probabilities by integrating multiple features. Five values of ASR confidence threshold t_w were tested in the following experiments: 0.2, 0.3, 0.4, 0.5, and 0.6 (shown by black dots in Figure 2).

4.2 Evaluation metrics

Evaluation was based on an averaged NER F-measure, which is the harmonic mean of NER precision and recall:

$$\begin{aligned} \text{NER precision} &= \frac{\# \text{ correctly recognized NEs}}{\# \text{ recognized NEs}} \\ \text{NER recall} &= \frac{\# \text{ correctly recognized NEs}}{\# \text{ NEs in original text}}. \end{aligned}$$

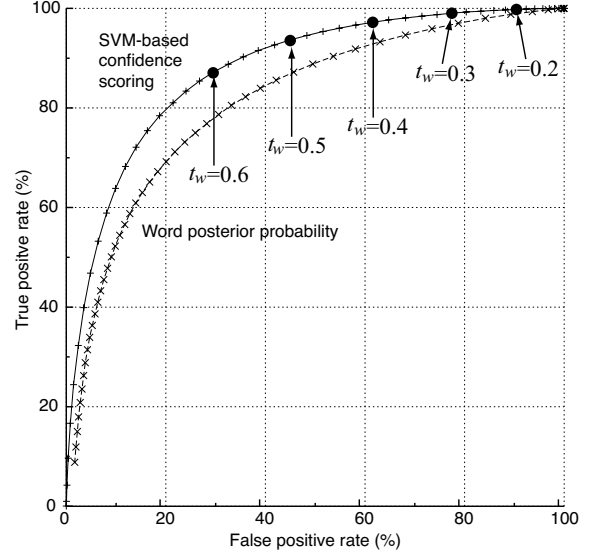


Figure 2: SVM-based confidence scoring outperforms word posterior probability for ASR error estimation.

A recognized NE was accepted as correct if and only if it appeared in the same position as its reference NE through alignment, in addition to having the correct NE surface and category, because the same NEs might appear more than once. Comparisons of NE surfaces did not include differences in word segmentation because of the segmentation ambiguity in Japanese. Note that NER recall with ASR results could not exceed the rate of the remaining NEs after ASR (about 82%) because NEs containing ASR errors were always lost.

In addition, we also evaluated the NER performance in NER precision and recall with NER-level rejection using the procedure in Section 3.4, by modifying the non-NE class scores using offset value t_o .

4.3 Compared methods

We compared several combinations of features and training conditions for evaluating the effect of incorporating the ASR confidence feature and investigating differences among training data: text-based, ASR-based, and both.

Baseline does not use the ASR confidence feature and is trained using text-based training data only.

NoConf-A does not use the ASR confidence feature and is trained using ASR-based training data only.

Method	Confidence	Training	Test	F-measure (%)	Precision (%)	Recall (%)
Baseline	Not used	Text	ASR	67.00	70.67	63.70
NoConf-A		ASR	ASR	65.52	78.86	56.05
NoConf-TA		Text+ASR	ASR	66.95	77.55	58.91
Conf-A	Used	ASR	ASR*	67.69	76.69	60.59
Proposed		Text+ASR	ASR*	69.02	78.13	61.81
Conf-Reject	Used [†]	Text+ASR	ASR*	68.77	77.57	61.78
Conf-UB	Used	Text+ASR	ASR**	73.14	87.51	62.83
Transcription	Not used	Text	Text	84.04	86.27	81.93

Table 4: NER results in averaged NER F-measure, precision, and recall without considering NER-level rejection ($t_o = 0$). ASR word accuracy was 79.45%, and 82.00% of NEs remained in ASR results. ([†]Unconfident words were rejected and replaced by word error symbols, $*t_w = 0.4$, **ASR errors were known.)

NoConf-TA does not use the ASR confidence feature and is trained using both text-based and ASR-based training data.

Conf-A uses the ASR confidence feature and is trained using ASR-based training data only.

Proposed uses the ASR confidence feature and is trained using both text-based and ASR-based training data.

Conf-Reject is almost the same as Proposed, but misrecognized words are rejected and replaced with word error symbols, as described at the end of Section 3.2.

The following two methods are for reference.

Conf-UB assumes perfect ASR confidence scoring, so the ASR errors in the test set are known. The NER model, which is identical to Proposed, is regarded as the upper-boundary of Proposed.

Transcription applies the same model as Baseline to reference transcriptions, assuming word accuracy is 100%.

4.4 NER Results

In the NER experiments, Proposed achieved the best results among the above methods. Table 4 shows the NER results obtained by the methods without considering NER-level rejection (i.e., $t_o = 0$), using threshold $t_w = 0.4$ for Conf-A, Proposed, and Conf-Reject, which resulted in the best NER F-measures (see Table 5). Proposed showed the best F-measure, 69.02%. It outperformed Baseline by 2.0%, with a 7.5% improvement in precision, instead of a recall decrease of 1.9%. Conf-Reject showed slightly worse results

Method	t_w	F (%)	P (%)	R (%)
Conf-A	0.2	66.72	71.28	62.71
	0.3	67.32	73.68	61.98
	0.4	67.69	76.69	60.59
	0.5	67.04	79.64	57.89
	0.6	64.48	81.90	53.14
Proposed	0.2	68.08	72.54	64.14
	0.3	68.70	75.11	63.31
	0.4	69.02	78.13	61.81
	0.5	68.17	80.88	58.93
	0.6	65.39	83.00	53.96
Conf-Reject	0.2	68.06	72.49	64.14
	0.3	68.61	74.88	63.31
	0.4	68.77	77.57	61.78
	0.5	67.93	80.23	58.91
	0.6	64.93	82.05	53.73

Table 5: NER results with varying ASR confidence score threshold t_w for Conf-A, Proposed, and Conf-Reject. (F: F-measure, P: precision, R: recall)

than Proposed. Conf-A resulted in 1.3% worse F-measure than Proposed. NoConf-A and NoConf-TA achieved 7-8% higher precision than Baseline; however, their F-measure results were worse than Baseline because of the large drop of recall. The upper-bound results of the proposed method (Conf-UB) in F-measure was 73.14%, which was 4% higher than Proposed.

Figure 3 shows NER precision and recall with NER-level rejection by t_o for Baseline, NoConf-TA, Proposed, Conf-UB, and Transcription. In the figure, black dots represent results with $t_o = 0$, as shown in Table 4. By all five methods, we

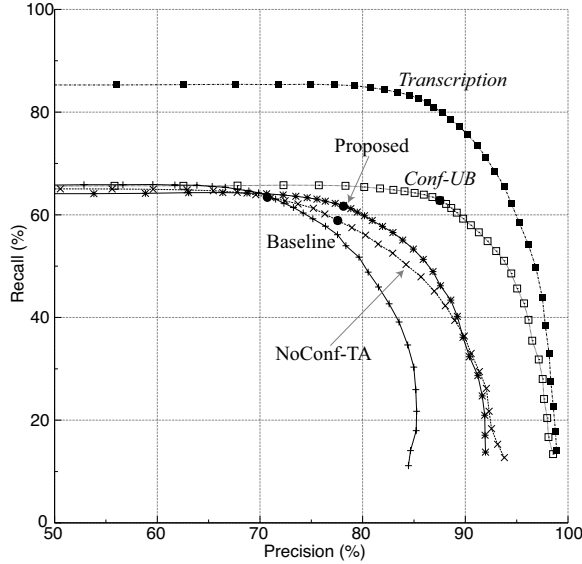


Figure 3: NER precision and recall with NER-level rejection by t_o

obtained higher precision with $t_o > 0$. Proposed achieved more than 5% higher precision than Baseline on most recall ranges and showed higher precision than NoConf-TA on recall ranges higher than about 35%.

5 Discussion

The proposed method effectively improves NER performance, as shown by the difference between Proposed and Baseline in Tables 4 and 5. Improvement comes from two factors: using both text-based and ASR-based training data and incorporating ASR confidence feature. As shown by the difference between Baseline and the methods using ASR-based training data (NoConf-A, NoConf-TA, Conf-A, Proposed, Conf-Reject), ASR-based training data increases precision and decreases recall. In ASR-based training data, all words constituting NEs that contain ASR errors are regarded as non-NE words, and those NE examples are lost in training, which emphasizes NER precision. When text-based training data are also available, they compensate for the loss of NE examples and recover NER recall, as shown by the difference between the methods without text-based training data (NoConf-A, Conf-A) and those with (NoConf-TA, Proposed). The ASR confidence feature also increases NER recall, as shown by the difference between the methods without it (NoConf-A, NoConf-TA) and with it (Conf-A, Proposed). This suggests that the ASR confidence

feature helps distinguish whether ASR error influences NER and suppresses excessive rejection of NEs around ASR errors.

With respect to the ASR confidence feature, the small difference between Conf-Reject and Proposed suggests that ASR confidence is a more dominant feature in misrecognized words than the other features: the word itself, its part-of-speech tag, and its character type. In addition, the difference between Conf-UB and Proposed indicated that there is room to improve NER performance with better ASR confidence scoring.

NER-level rejection also increased precision, as shown in Figure 3. We can control the trade-off between precision and recall with t_o according to the task requirements, even in text-based NER. In the NER of speech data, we can obtain much higher precision using both ASR-based training data and NER-level rejection than using either one.

6 Related work

Recent studies on the NER of speech data consider more than 1-best ASR results in the form of N-best lists and word lattices. Using many ASR hypotheses helps recover the ASR errors of NE words in 1-best ASR results and improves NER accuracy. Our method can be extended to multiple ASR hypotheses.

Generative NER models were used for multi-pass ASR and NER searches using word lattices (Horlock and King, 2003b; Béchet et al., 2004; Favre et al., 2005). Horlock and King (2003a) also proposed discriminative training of their NER models. These studies showed the advantage of using multiple ASR hypotheses, but they do not use overlapping features.

Discriminative NER models were also applied to multiple ASR hypotheses. Zhai et al. (2004) applied text-based NER to N-best ASR results, and merged the N-best NER results by weighted voting based on several sentence-level results such as ASR and NER scores. Using the ASR confidence feature does not depend on SVMs and can be used with their method and other discriminative models.

7 Conclusion

We proposed a method for NER of speech data that incorporates ASR confidence as a feature of discriminative NER, where the NER model

is trained using both text-based and ASR-based training data. In experiments using SVMs, the proposed method showed a higher NER F-measure, especially in terms of improving precision, than simply applying text-based NER to ASR results. The method effectively rejected erroneous NEs due to ASR errors with a small drop of recall, thanks to both the ASR confidence feature and ASR-based training data. NER-level rejection also effectively increased precision.

Our approach can also be used in other tasks in spoken language processing, and we expect it to be effective. Since confidence itself is not limited to speech, our approach can also be applied to other noisy inputs, such as optical character recognition (OCR). For further improvement, we will consider N-best ASR results or word lattices as inputs and introduce more speech-specific features such as word durations and prosodic features.

Acknowledgments We would like to thank anonymous reviewers for their helpful comments.

References

- Frédéric Béchet, Allen L. Gorin, Jeremy H. Wright, and Dilek Hakkani-Tür. 2004. Detecting and extracting named entities from spontaneous speech in a mixed-initiative spoken dialogue context: How May I Help You? *Speech Communication*, 42(2):207–225.
- Andrew Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. thesis, New York University.
- Hai Leong Chieu and Hwee Tou Ng. 2003. Named entity recognition with a maximum entropy approach. In *Proc. CoNLL*, pages 160–163.
- Benoît Favre, Frédéric Béchet, and Pascal Nocéra. 2005. Robust named entity extraction from large spoken archives. In *Proc. HLT-EMNLP*, pages 491–498.
- Takaaki Hori, Chiori Hori, and Yasuhiro Minami. 2004. Fast on-the-fly composition for weighted finite-state transducers in 1.8 million-word vocabulary continuous-speech recognition. In *Proc. IC-SLP*, volume 1, pages 289–292.
- James Horlock and Simon King. 2003a. Discriminative methods for improving named entity extraction on speech data. In *Proc. EUROSPEECH*, pages 2765–2768.
- James Horlock and Simon King. 2003b. Named entity extraction from word lattices. In *Proc. EUROSPEECH*, pages 1265–1268.
- Hideki Isozaki and Hideto Kazawa. 2002. Efficient support vector classifiers for named entity recognition. In *Proc. COLING*, pages 390–396.
- Simo O. Kamppari and Timothy J. Hazen. 2000. Word and phone level acoustic confidence scoring. In *Proc. ICASSP*, volume 3, pages 1799–1802.
- Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proc. CoNLL*, pages 188–191.
- David Miller, Richard Schwartz, Ralph Weischedel, and Rebecca Stone. 1999. Named entity extraction from broadcast news. In *Proceedings of the DARPA Broadcast News Workshop*, pages 37–40.
- David D. Palmer and Mari Ostendorf. 2001. Improving information extraction by modeling errors in speech recognizer output. In *Proc. HLT*, pages 156–160.
- Thomas Schaaf and Thomas Kemp. 1997. Confidence measures for spontaneous speech recognition. In *Proc. ICASSP*, volume II, pages 875–878.
- Satoshi Sekine and Yoshio Eriguchi. 2000. Japanese named entity extraction evaluation - analysis of results. In *Proc. COLING*, pages 25–30.
- Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinou. 1998. A decision tree method for finding and classifying names in Japanese texts. In *Proc. the Sixth Workshop on Very Large Corpora*, pages 171–178.
- Frank Wessel, Ralf Schlüter, Klaus Macherey, and Hermann Ney. 2001. Confidence measures for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, 9(3):288–298.
- Lufeng Zhai, Pascale Fung, Richard Schwartz, Marine Carpuat, and Dekai Wu. 2004. Using N-best lists for named entity recognition from chinese speech. In *Proc. HLT-NAACL*, pages 37–40.
- Rong Zhang and Alexander I. Rudnicky. 2001. Word level confidence annotation using combinations of features. In *Proc. EUROSPEECH*, pages 2105–2108.

Exploiting Syntactic Patterns as Clues in Zero-Anaphora Resolution

Ryu Iida, Kentaro Inui and Yuji Matsumoto

Graduate School of Information Science,
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
{ryu-i,inui,matsu}@is.naist.jp

Abstract

We approach the zero-anaphora resolution problem by decomposing it into intra-sentential and inter-sentential zero-anaphora resolution. For the former problem, syntactic patterns of the appearance of zero-pronouns and their antecedents are useful clues. Taking Japanese as a target language, we empirically demonstrate that incorporating rich syntactic pattern features in a state-of-the-art learning-based anaphora resolution model dramatically improves the accuracy of intra-sentential zero-anaphora, which consequently improves the overall performance of zero-anaphora resolution.

1 Introduction

Zero-anaphora is a gap in a sentence that has an anaphoric function similar to a pro-form (e.g. pronoun) and is often described as “referring back” to an expression that supplies the information necessary for interpreting the sentence. For example, in the sentence “*There are two roads to eternity, a straight and narrow, and a broad and crooked,*” the gaps in “*a straight and narrow (gap)*” and “*a broad and crooked (gap)*” have a zero-anaphoric relationship to “*two roads to eternity.*”

The task of identifying zero-anaphoric relations in a given discourse, *zero-anaphora resolution*, is essential in a wide range of NLP applications. This is the case particularly in such a language as Japanese, where even obligatory arguments of a predicate are often omitted when they are inferable from the context. In fact, in our Japanese newspaper corpus, for example, 45.5% of the nominative arguments of verbs are omitted. Since such gaps

can not be interpreted only by shallow syntactic parsing, a model specialized for zero-anaphora resolution needs to be devised on the top of shallow syntactic and semantic processing.

Recent work on zero-anaphora resolution can be located in two different research contexts. First, zero-anaphora resolution is studied in the context of anaphora resolution (AR), in which zero-anaphora is regarded as a subclass of anaphora. In AR, the research trend has been shifting from rule-based approaches (Baldwin, 1995; Lappin and Leass, 1994; Mitkov, 1997, etc.) to empirical, or corpus-based, approaches (McCarthy and Lehnert, 1995; Ng and Cardie, 2002a; Soon et al., 2001; Strube and Müller, 2003; Yang et al., 2003) because the latter are shown to be a cost-efficient solution achieving a performance that is comparable to best performing rule-based systems (see the Coreference task in MUC¹ and the Entity Detection and Tracking task in the ACE program²). The same trend is observed also in Japanese zero-anaphora resolution, where the findings made in rule-based or theory-oriented work (Kameyama, 1986; Nakaiwa and Shirai, 1996; Okumura and Tamura, 1996, etc.) have been successfully incorporated in machine learning-based frameworks (Seki et al., 2002; Iida et al., 2003).

Second, the task of zero-anaphora resolution has some overlap with Propbank³-style semantic role labeling (SRL), which has been intensively studied, for example, in the context of the CoNLL SRL task⁴. In this task, given a sentence “*To attract younger listeners, Radio Free Europe intersperses the latest in Western rock groups*”, an SRL

¹http://www-nlpir.nist.gov/related_projects/muc/

²<http://projects ldc.upenn.edu/ace/>

³http://www.cis.upenn.edu/~mpalmer/project_pages/ACE.htm

⁴<http://www.lsi.upc.edu/~srlconll/>

model is asked to identify the NP *Radio Free Europe* as the A0 (Agent) argument of the verb *attract*. This can be seen as the task of finding the zero-anaphoric relationship between a nominal gap (the A0 argument of *attract*) and its antecedent (*Radio Free Europe*) under the condition that the gap and its antecedent appear in the same sentence.

In spite of this overlap between AR and SRL, there are some important findings that are yet to be exchanged between them, partly because the two fields have been evolving somewhat independently. The AR community has recently made two important findings:

- A model that identifies the antecedent of an anaphor by a series of comparisons between candidate antecedents has a remarkable advantage over a model that estimates the absolute likelihood of each candidate independently of other candidates (Iida et al., 2003; Yang et al., 2003).
- An AR model that carries out antecedent identification *before* anaphoricity determination, the decision whether a given NP is anaphoric or not (i.e. discourse-new), significantly outperforms a model that executes those subtasks in the reverse order or simultaneously (Poesio et al., 2004; Iida et al., 2005).

To our best knowledge, however, existing SRL models do not exploit these advantages. In SRL, on the other hand, it is common to use syntactic features derived from the parse tree of a given input sentence for argument identification. A typical syntactic feature is the path on a parse tree from a target predicate to a noun phrase in question (Gildea and Jurafsky, 2002; Carreras and Marquez, 2005). However, existing AR models deal with intra- and inter-sentential anaphoric relations in a uniform manner; that is, they do not use as rich syntactic features as state-of-the-art SRL models do, even in finding intra-sentential anaphoric relations. We believe that the AR and SRL communities can learn more from each other.

Given this background, in this paper, we show that combining the aforementioned techniques derived from each research trend makes significant impact on zero-anaphora resolution, taking Japanese as a target language. More specifically, we demonstrate the following:

- Incorporating rich syntactic features in a state-of-the-art AR model dramatically im-

proves the accuracy of intra-sentential zero-anaphora resolution, which consequently improves the overall performance of zero-anaphora resolution. This is to be considered as a contribution to AR research.

- Analogously to inter-sentential anaphora, decomposing the antecedent identification task into a series of comparisons between candidate antecedents works remarkably well also in intra-sentential zero-anaphora resolution. We hope this finding to be adopted in SRL.

The rest of the paper is organized as follows. Section 2 describes the task definition of zero-anaphora resolution in Japanese. In Section 3, we review previous approaches to AR. Section 4 described how the proposed model incorporates effectively syntactic features into the machine learning-based approach. We then report the results of our experiments on Japanese zero-anaphora resolution in Section 5 and conclude in Section 6.

2 Zero-anaphora resolution

In this paper, we consider only zero-pronouns that function as an obligatory argument of a predicate for two reasons:

- Providing a clear definition of zero-pronouns appearing in adjunctive argument positions involves awkward problems, which we believe should be postponed until obligatory zero-anaphora is well studied.
- Resolving obligatory zero-anaphora tends to be more important than adjunctive zero-pronouns in actual applications.

A zero-pronoun may have its antecedent in the discourse; in this case, we say the zero-pronoun is *anaphoric*. On the other hand, a zero-pronoun whose referent does not explicitly appear in the discourse is called a *non-anaphoric* zero-pronoun. A zero-pronoun may be non-anaphoric typically when it refers to an extralinguistic entity (e.g. the first or second person) or its referent is unspecified in the context.

The following are Japanese examples. In sentence (1), zero-pronoun ϕ_i is anaphoric as its antecedent, ‘*shusho* (prime minister)’, appears in the same sentence. In sentence (2), on the other hand, ϕ_j is considered non-anaphoric if its referent (i.e. the first person) does not appear in the discourse.

- (1) *shusho_i-wa houbesei-te ,*
 prime minister_i-TOP visit-U.S.-CONJ PUNC

ryoukoku-no gaikou-o
both countries-BETWEEN diplomacy-OBJ

(ϕ_i -ga) suishinsuru
(ϕ_i -NOM) promote-ADNOM

houshin-o akirakanisi-ta .
plan-OBJ unveil-PAST PUNC

The prime minister visited the united states
and unveiled the plan to push diplomacy
between the two countries.

- (2) (ϕ_j -ga) ie-ni kaeri-tai .
(ϕ_j -NOM) home-DAT want to go back PUNC
(I) want to go home.

Given this distinction, we consider the task of zero-anaphora resolution as the combination of two sub-problems, antecedent identification and anaphoricity determination, which is analogous to NP-anaphora resolution:

For each zero-pronoun in a given discourse, find its antecedent if it is anaphoric; otherwise, conclude it to be non-anaphoric.

3 Previous work

3.1 Antecedent identification

Previous machine learning-based approaches to antecedent identification can be classified as either the *candidate-wise classification* approach or the *preference-based* approach. In the former approach (Soon et al., 2001; Ng and Cardie, 2002a, etc.), given a target anaphor, *TA*, the model estimates the absolute likelihood of each of the candidate antecedents (i.e. the NPs preceding *TA*), and selects the best-scored candidate. If all the candidates are classified negative, *TA* is judged non-anaphoric.

In contrast, the preference-based approach (Yang et al., 2003; Iida et al., 2003) decomposes the task into comparisons of the preference between candidates and selects the most preferred one as the antecedent. For example, Iida et al. (2003) proposes a method called the *tournament model*. This model conducts a tournament consisting of a series of matches in which candidate antecedents compete with each other for a given anaphor.

While the candidate-wise classification model computes the score of each single candidate independently of others, the tournament model learns the relative preference between candidates, which is empirically proved to be a significant advantage over candidate-wise classification (Iida et al., 2003).

3.2 Anaphoricity determination

There are two alternative ways for anaphoricity determination: the *single-step model* and the *two-step model*. The single-step model (Soon et al., 2001; Ng and Cardie, 2002a) determines the anaphoricity of a given anaphor indirectly as a by-product of the search for its antecedent. If an appropriate candidate antecedent is found, the anaphor is classified as anaphoric; otherwise, it is classified as non-anaphoric. One disadvantage of this model is that it cannot employ the preference-based model because the preference-based model is not capable of identifying non-anaphoric cases.

The two-step model (Ng, 2004; Poesio et al., 2004; Iida et al., 2005), on the other hand, carries out anaphoricity determination in a separate step from antecedent identification. Poesio et al. (2004) and Iida et al. (2005) claim that the latter subtask should be done before the former. For example, given a target anaphor (*TA*), Iida et al.'s *selection-then-classification* model:

1. selects the most likely candidate antecedent (*CA*) of *TA* using the tournament model,
2. classifies *TA* paired with *CA* as either *anaphoric* or *non-anaphoric* using an anaphoricity determination model. If the *CA-TA* pair is classified as *anaphoric*, *CA* is identified as the antecedent of *TA*; otherwise, *TA* is conclude to be *non-anaphoric*.

The anaphoricity determination model learns the non-anaphoric class directly from non-anaphoric training instances whereas the single-step model cannot not use non-anaphoric cases in training.

4 Proposal

4.1 Task decomposition

We approach the zero-anaphora resolution problem by decomposing it into two subtasks: intra-sentential and inter-sentential zero-anaphora resolution. For the former problem, syntactic patterns in which zero-pronouns and their antecedents appear may well be useful clues, which, however, does not apply to the latter problem. We therefore build a separate component for each subtask, adopting Iida et al. (2005)'s selection-then-classification model for each component:

1. *Intra-sentential antecedent identification*: For a given zero-pronoun *ZP* in a given sentence *S*, select the most-likely candidate antecedent C_1^* from the candidates appearing in *S* by the intra-sentential tournament model

2. *Intra-sentential anaphoricity determination*: Estimate plausibility p_1 that C_1^* is the true antecedent, and return C_1^* if $p_1 \geq \theta_{intra}$ (θ_{intra} is a preselected threshold) or go to 3 otherwise
3. *Inter-sentential antecedent identification*: Select the most-likely candidate antecedent C_2^* from the candidates appearing outside of S by the inter-sentential tournament model.
4. *Inter-sentential anaphoricity determination*: Estimate plausibility p_2 that C_2^* is the true antecedent, and return C_2^* if $p_2 \geq \theta_{inter}$ (θ_{inter} is a preselected threshold) or return non-anaphoric otherwise.

4.2 Representation of syntactic patterns

In the first two of the above four steps, we use syntactic pattern features. Analogously to SRL, we extract the parse path between a zero-pronoun to its antecedent to capture the syntactic pattern of their occurrence. Among many alternative ways of representing a path, in the experiments reported in the next section, we adopted a method as we describe below, leaving the exploration of other alternatives as future work.

Given a sentence, we first use a standard dependency parser to obtain the dependency parse tree, in which words are structured according to the dependency relation between them. Figure 1(a), for example, shows the dependency tree of sentence (1) given in Section 2. We then extract the path between a zero-pronoun and its antecedent as in Figure 1(b). Finally, to encode the order of siblings and reduce data sparseness, we further transform the extracted path as in Figure 1(c):

- A path is represented by a subtree consisting of backbone nodes: ϕ (zero-pronoun), Ant (antecedent), Node (the lowest common ancestor), LeftNode (left-branch node) and RightNode.
- Each backbone node has daughter nodes, each corresponding to a function word associated with it.
- Content words are deleted.

This way of encoding syntactic patterns is used in intra-sentential anaphoricity determination. In antecedent identification, on the other hand, the tournament model allows us to incorporate three paths, a path for each pair of a zero-pronoun and left and right candidate antecedents, as shown in

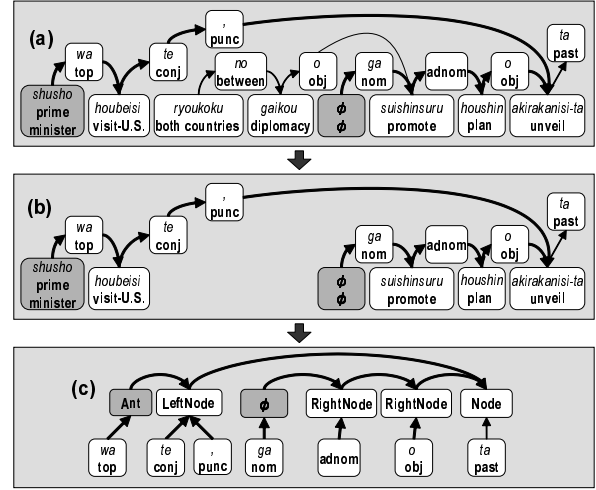


Figure 1: Representation of the path between a zero-pronoun to its antecedent

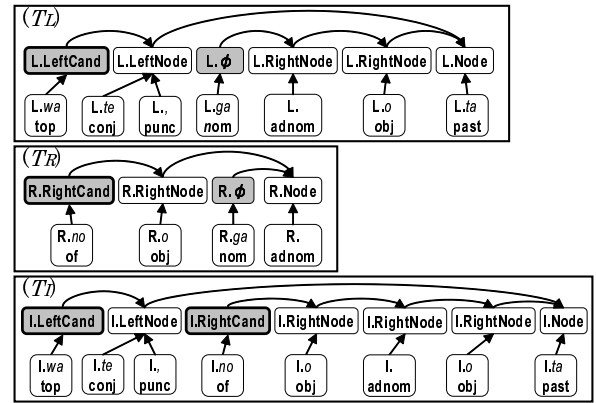


Figure 2: Paths used in the tournament model

Figure 2⁵.

4.3 Learning algorithm

As noted in Section 1, the use of zero-pronouns in Japanese is relatively less constrained by syntax compared, for example, with English. This forces the above way of encoding path information to produce an explosive number of different paths, which inevitably leads to serious data sparseness.

This issue can be addressed in several ways. The SRL community has devised a range of variants of the standard path representation to reduce the complexity (Carreras and Marquez, 2005). Applying Kernel methods such as Tree kernels (Collins and Duffy, 2001) and Hierarchical DAG kernels (Suzuki et al., 2003) is another strong option. The Boosting-based algorithm pro-

⁵To indicate which node belongs to which subtree, the label of each node is prefixed either with L, R or I.

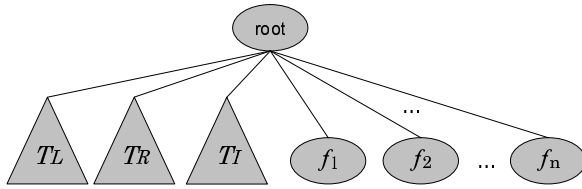


Figure 4: Tree representation of features for the tournament model.

posed by Kudo and Matsumoto (2004) is designed to learn subtrees useful for classification.

Leaving the question of selecting learning algorithms open, in our experiments, we have so far examined Kudo and Matsumoto (2004)’s algorithm, which is implemented as the BACT system⁶. Given a set of training instances, each of which is represented as a tree labeled either positive or negative, the BACT system learns a list of weighted decision stumps with a Boosting algorithm. Each decision stump is associated with tuple $\langle t, l, w \rangle$, where t is a subtree appearing in the training set, l a label, and w a weight, indicating that if a given input includes t , it gives w votes to l . The strength of this algorithm is that it deals with structured feature and allows us to analyze the utility of features.

In antecedent identification, we train the tournament model by providing a set of labeled trees as a training set, where a label is either left or right. Each labeled tree has (i) path trees T_L , T_R and T_I (as given in Figure 2) and (ii) a set nodes corresponding to the binary features summarized in Table 3, each of which is linked to the root node as illustrated in Figure 4. This way of organizing a labeled tree allows the model to learn, for example, the combination of a subtree of T_L and some of the binary features. Analogously, for anaphoricity determination, we use trees $(T_C, f_1, \dots, f_n)$, where T_C denotes a path subtree as in Figure 1(c).

5 Experiments

We conducted an evaluation of our method using Japanese newspaper articles. The following four models were compared:

1. BM: Ng and Cardie (2002a)’s model, which identify antecedents by the candidate-wise classification model, and determine anaphoricity using the one-step model.

2. BM_STR: BM with the syntactic features such as those in Figure 1(c).
3. SCM: The selection-then-classification model explained in Section 3.
4. SCM_STR: SCM with all types of syntactic features shown in Figure 2.

5.1 Setting

We created an anaphoric relation-tagged corpus consisting of 197 newspaper articles (1,803 sentences), 137 articles annotated by two annotators and 60 by one. The agreement ratio between two annotators on the 197 articles was 84.6%, which indicated that the annotation was sufficiently reliable.

In the experiments, we removed from the above data set the zero-pronouns to which the two annotators did not agree. Consequently, the data set contained 995 intra-sentential anaphoric zero-pronouns, 754 inter-sentential anaphoric zero-pronouns, and 603 non-anaphoric zero-pronouns (2,352 zero-pronouns in total), with each anaphoric zero-pronoun annotated to be linked to its antecedent. For each of the following experiments, we conducted five-fold cross-validation over 2,352 zero-pronouns so that the set of the zero-pronouns from a single text was not divided into the training and test sets.

In the experiments, all the features were automatically acquired with the help of the following NLP tools: the Japanese morphological analyzer *ChaSen*⁷ and the Japanese dependency structure analyzer *CaboCha*⁸, which also carried out named-entity chunking.

5.2 Results on intra-sentential zero-anaphora resolution

In both intra-anaphoricity determination and antecedent identification, we investigated the effect of introducing the syntactic features for improving the performance. First, the results of antecedent identification are shown in Table 1. The comparison between BM (SCM) with BM_STR (SCM_STR) indicates that introducing the structural information effectively contributes to this task. In addition, the large improvement from BM_STR to SCM_STR indicates that the use of the preference-based model has significant impact on intra-sentential antecedent identification. This

⁶<http://chasen.org/~taku/software/bact/>

⁷<http://chasen.naist.jp/hiki/ChaSen/>

⁸<http://chasen.org/~taku/software/cabocha/>

Figure 3: Feature set.

Feature Type	Feature	Description
Lexical	HEAD_BF	characters of right-most morpheme in <i>NP</i> (<i>PRED</i>).
Grammatical	PRED_IN_MATRIX	1 if <i>PRED</i> exists in the matrix clause; otherwise 0.
	PRED_IN_EMBEDDED	1 if <i>PRED</i> exists in the relative clause; otherwise 0.
	PRED_VOICE	1 if <i>PRED</i> contains auxiliaries such as ‘(ra)reru’; otherwise 0.
	PRED_AUX	1 if <i>PRED</i> contains auxiliaries such as ‘(sa)seru’, ‘hosii’, ‘morau’, ‘itadaku’, ‘kudasaru’, ‘yaru’ and ‘ageru’.
	PRED_ALT	1 if PRED_VOICE is 1 or PRED_AUX is 1; otherwise 0.
	POS	Part-of-speech of <i>NP</i> followed by IPADIC (Asahara and Matsumoto, 2003).
	DEFINITE	1 if <i>NP</i> contains the article corresponding to DEFINITE ‘the’, such as ‘sore’ or ‘sono’; otherwise 0.
	DEMONSTRATIVE	1 if <i>NP</i> contains the article corresponding to DEMONSTRATIVE ‘that’ or ‘this’, such as ‘kono’, ‘ano’; otherwise 0.
Semantic	PARTICLE	Particle followed by <i>NP</i> , such as ‘wa (topic)’, ‘ga (subject)’, ‘o (object)’.
	NE	Named entity of <i>NP</i> : PERSON, ORGANIZATION, LOCATION, ARTIFACT, DATE, TIME, MONEY, PERCENT or N/A.
	EDR_HUMAN	1 if <i>NP</i> is included among the concept ‘a human being’ or ‘attribute of a human being’ in EDR dictionary (Jap, 1995); otherwise 0.
	PRONOUN_TYPE	Pronoun type of <i>NP</i> . (e.g. ‘kare (he)’ → PERSON, ‘koko (here)’ → LOCATION, ‘sore (this)’ → OTHERS)
	SELECT_REST	1 if <i>NP</i> satisfies selectional restrictions in Nihongo Goi Taikei (Japanese Lexicon) (Ikehara et al., 1997); otherwise 0.
Positional	COOC	the score of well-formedness model estimated from a large number of triplets $\langle \text{Noun}, \text{Case}, \text{Predicate} \rangle$ proposed by Fujita et al. (2004)
	SENTNUM	Distance between <i>NP</i> and <i>PRED</i> .
	BEGINNING	1 if <i>NP</i> is located in the beginning of sentence; otherwise 0.
	END	1 if <i>NP</i> is located in the end of sentence; otherwise 0.
	PRED_NP	1 if <i>PRED</i> precedes <i>NP</i> ; otherwise 0.
	NP_PRED	1 if <i>NP</i> precedes <i>PRED</i> ; otherwise 0.
	DEP_PRED	1 if NP_i depends on <i>PRED</i> ; otherwise 0.
	DEP_NP	1 if <i>PRED</i> depends on NP_i ; otherwise 0.
Heuristic	IN_QUOTE	1 if <i>NP</i> exists in the quoted text; otherwise 0.
	CL_RANK	a rank of <i>NP</i> in forward looking-center list based on Centering Theory (Grosz et al., 1995)
	CL_ORDER	a order of <i>NP</i> in forward looking-center list based on Centering Theory (Grosz et al., 1995)

NP and *PRED* stand for a bunsetsu-chunk of a candidate antecedent and a bunsetsu-chunk of a predicate which has a target zero-pronoun respectively.

finding may well contribute to semantic role labeling because these two tasks have a large overlap as discussed in Section 1.

Second, to evaluate the performance of intra-sentential zero-anaphora resolution, we plotted recall-precision curves altering threshold parameter and θ_{inter} for intra-anaphoricity determination as shown in Figure 5, where recall R and precision P were calculated by:

$$R = \frac{\# \text{ of detected antecedents correctly}}{\# \text{ of anaphoric zero-pronouns}},$$

$$P = \frac{\# \text{ of detected antecedents correctly}}{\# \text{ of zero-pronouns classified as anaphoric}}.$$

The curves indicate the upperbound of the performance of these models; in practical settings, the parameters have to be trained beforehand.

Figure 5 shows that BM_STR (SCM_STR) outperforms BM (SCM), which indicates that incorporating syntactic pattern features works remarkably well for intra-sentential zero-anaphora

Table 1: Accuracy of antecedent identification.

BM	BM_STR	SCM	SCM_STR
48.0%	63.5%	65.1%	70.5%
(478/995)	(632/995)	(648/995)	(701/995)

resolution. Furthermore, SCM_STR is significantly better than BM_STR. This result supports that the former has an advantage of learning non-anaphoric zero-pronouns (181 instances) as negative training instances in intra-sentential anaphoricity determination, which enables it to reject non-anaphoric zero-pronouns more accurately than the others.

5.3 Discussion

Our error analysis reveals that a majority of errors can be attributed to the current way of handling quoted phrases and sentences. Figure 6 shows the difference in resolution accuracy between zero-pronouns appearing in a quotation

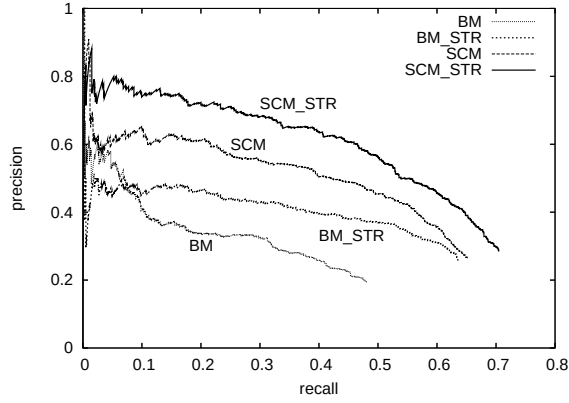


Figure 5: Recall-precision curves of intra-sentential zero-anaphora resolution.

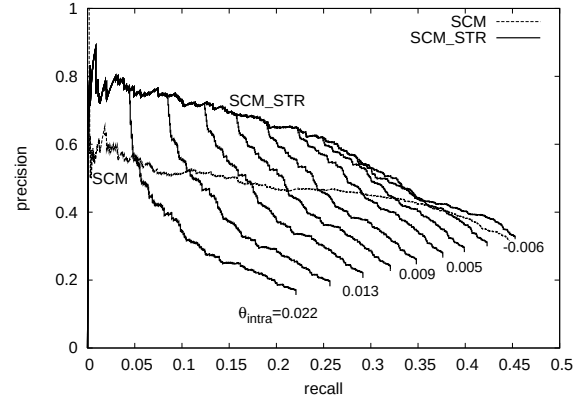


Figure 7: Recall-precision curves of overall zero-anaphora resolution.

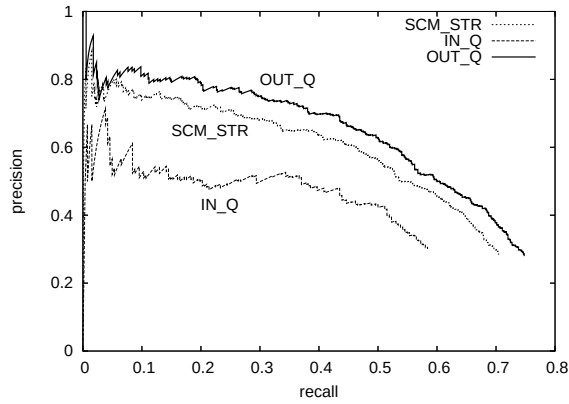


Figure 6: Recall-precision curves of resolving in-quote and out-quote zero-pronouns.

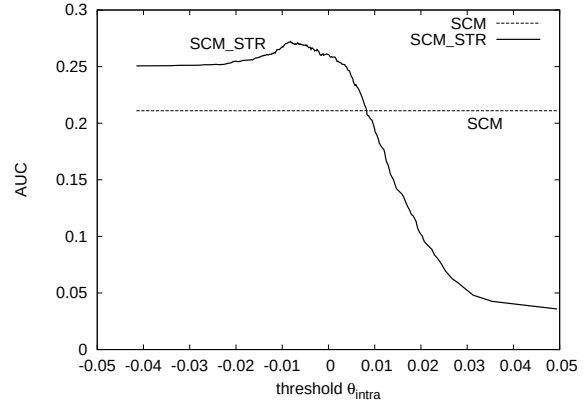


Figure 8: AUC curves plotted by altering θ_{intra} .

(262 zero-pronouns) and the rest (733 zero-pronouns), where “IN_Q” denotes the former (in-quote zero-pronouns) and “OUT_Q” the latter. The accuracy on the IN_Q problems is considerably lower than that on the OUT_Q cases, which indicates that we should deal with in-quote cases with a separate model so that it can take into account the nested structure of discourse segments introduced by quotations.

5.4 Impact on overall zero-anaphora resolution

We next evaluated the effects of introducing the proposed model on overall zero-anaphora resolution including inter-sentential cases.

As a baseline model, we implemented the original SCM, designed to resolve intra-sentential zero-anaphora and inter-sentential zero-anaphora simultaneously with no syntactic pattern features. Here, we adopted Support Vector Machines (Vapnik, 1998) to train the classifier on the baseline

model and the inter-sentential zero-anaphora resolution in the SCM using structural information.

For the proposed model, we plotted several recall-precision curves by selecting different value for threshold parameters θ_{intra} and θ_{inter} . The results are shown in Figure 7, which indicates that the proposed model significantly outperforms the original SCM if θ_{intra} is appropriately chosen.

We then investigated the feasibility of parameter selection for θ_{intra} by plotting the AUC values for different θ_{intra} values. Here, each AUC value is the area under a recall-precision curve. The results are shown in Figure 8. Since the original SCM does not use θ_{intra} , the AUC value of it is constant, depicted by the SCM. As shown in the Figure 8, the AUC-value curve of the proposed model is not peaky, which indicates the selection of parameter θ_{intra} is not difficult.

6 Conclusion

In intra-sentential zero-anaphora resolution, syntactic patterns of the appearance of zero-pronouns and their antecedents are useful clues. Taking Japanese as a target language, we have empirically demonstrated that incorporating rich syntactic pattern features in a state-of-the-art learning-based anaphora resolution model dramatically improves the accuracy of intra-sentential zero-anaphora, which consequently improves the overall performance of zero-anaphora resolution.

In our next step, we are going to address the issue of how to find zero-pronouns, which requires us to design a broader framework that allows zero-anaphora resolution to interact with predicate-argument structure analysis. Another important issue is how to find a globally optimal solution to the set of zero-anaphora resolution problems in a given discourse, which leads us to explore methods as discussed by McCallum and Wellner (2003).

References

- M. Asahara and Y. Matsumoto. 2003. *IPADIC User Manual*. Nara Institute of Science and Technology, Japan.
- B. Baldwin. 1995. *CogNIAC: A Discourse Processing Engine*. Ph.D. thesis, Department of Computer and Information Sciences, University of Pennsylvania.
- X. Carreras and L. Marquez. 2005. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth CoNLL*, pages 152–164.
- M. Collins and N.I. Duffy. 2001. Convolution kernels for natural language. In *Proceedings of the NIPS*, pages 625–632.
- A. Fujita, K. Inui, and Y. Matsumoto. 2004. Detection of incorrect case assignments in automatically generated paraphrases of japanese sentences. In *Proceeding of the first IJCNLP*, pages 14–21.
- D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. In *Computational Linguistics*, pages 245–288.
- B. J. Grosz, A. K. Joshi, and S. Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226.
- R. Iida, K. Inui, H. Takamura, and Y. Matsumoto. 2003. Incorporating contextual cues in trainable models for coreference resolution. In *Proceedings of the 10th EACL Workshop on The Computational Treatment of Anaphora*, pages 23–30.
- R. Iida, K. Inui, and Y. Matsumoto. 2005. Anaphora resolution by antecedent identification followed by anaphoricity determination. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4:417–434.
- S. Ikehara, M. Miyazaki, S. Shirai, A. Yokoo, H. Nakaiwa, K. Ogura, Y. Ooyama, and Y. Hayashi. 1997. *Nihongo Goi Taikēi (in Japanese)*. Iwanami Shoten.
- Japan Electronic Dictionary Research Institute, Ltd. Japan, 1995. *EDR Electronic Dictionary Technical Guide*.
- M. Kameyama. 1986. A property-sharing constraint in centering. In *Proceedings of the 24th ACL*, pages 200–206.
- T. Kudo and Y. Matsumoto. 2004. A boosting algorithm for classification of semi-structured text. In *Proceedings of the 2004 EMNLP*, pages 301–308.
- S. Lappin and H. J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.
- A. McCallum and B. Wellner. 2003. Object consolidation by graph partitioning with a conditionally trained distance metric. In *Proceedings of the KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, pages 19–24.
- J. F. McCarthy and W. G. Lehnert. 1995. Using decision trees for coreference resolution. In *Proceedings of the 14th IJCAI*, pages 1050–1055.
- R. Mitkov. 1997. Factors in anaphora resolution: they are not the only things that matter. a case study based on two different approaches. In *Proceedings of the ACL’97/EACL’97 Workshop on Operational Factors in Practical, Robust Anaphora Resolution*.
- H. Nakaiwa and S. Shirai. 1996. Anaphora resolution of japanese zero pronouns with deictic reference. In *Proceedings of the 16th COLING*, pages 812–817.
- V. Ng. 2004. Learning noun phrase anaphoricity to improve coreference resolution: Issues in representation and optimization. In *Proceedings of the 42nd ACL*, pages 152–159.
- V. Ng and C. Cardie. 2002a. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th ACL*, pages 104–111.
- M. Okumura and K. Tamura. 1996. Zero pronoun resolution in japanese discourse based on centering theory. In *Proceedings of the 16th COLING*, pages 871–876.
- M. Poesio, O. Uryupina, R. Vieira, M. Alexandrov-Kabadjov, and R. Goulart. 2004. Discourse-new detectors for definite description resolution: A survey and a preliminary proposal. In *Proceedings of the 42nd ACL Workshop on Reference Resolution and its Applications*, pages 47–54.
- K. Seki, A. Fujii, and T. Ishikawa. 2002. A probabilistic method for analyzing japanese anaphora integrating zero pronoun detection and resolution. In *Proceedings of the 19th COLING*, pages 911–917.
- W. M. Soon, H. T. Ng, and D. C. Y. Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.
- M. Strube and C. Müller. 2003. A machine learning approach to pronoun resolution in spoken dialogue. In *Proceedings of the 41st ACL*, pages 168–175.
- J. Suzuki, T. Hirao, Y. Sasaki, and E. Maeda. 2003. Hierarchical directed acyclic graph kernel: Methods for structured natural language data. In *Proceeding of the 41st ACL*, pages 32–39.
- V. N. Vapnik. 1998. *Statistical Learning Theory*. Adaptive and Learning Systems for Signal Processing Communications, and control. John Wiley & Sons.
- X. Yang, G. Zhou, J. Su, and C. L. Tan. 2003. Coreference resolution using competition learning approach. In *Proceedings of the 41st ACL*, pages 176–183.

Self-Organizing n -gram Model for Automatic Word Spacing

Seong-Bae Park Yoon-Shik Tae Se-Young Park

Department of Computer Engineering

Kyungpook National University

Daegu 702-701, Korea

{sbpark, ystae, sympark}@sejong.knu.ac.kr

Abstract

An automatic word spacing is one of the important tasks in Korean language processing and information retrieval. Since there are a number of confusing cases in word spacing of Korean, there are some mistakes in many texts including news articles. This paper presents a high-accurate method for automatic word spacing based on self-organizing n -gram model. This method is basically a variant of n -gram model, but achieves high accuracy by automatically adapting context size.

In order to find the optimal context size, the proposed method automatically increases the context size when the contextual distribution after increasing it does not agree with that of the current context. It also decreases the context size when the distribution of reduced context is similar to that of the current context. This approach achieves high accuracy by considering higher dimensional data in case of necessity, and the increased computational cost are compensated by the reduced context size. The experimental results show that the self-organizing structure of n -gram model enhances the basic model.

1 Introduction

Even though Korean widely uses Chinese characters, the ideograms, it has a word spacing model unlike Chinese and Japanese. The word spacing of Korean, however, is not a simple task, though the basic rule for it is simple. The basic rule asserts that all content words should be spaced. However,

there are a number of exceptions due to various *postpositions* and *endings*. For instance, it is difficult to distinguish some postpositions from incomplete nouns. Such exceptions induce many mistakes of word spacing even in news articles.

The problem of the inaccurate word spacing is that they are fatal in language processing and information retrieval. The incorrect word spacing would result in the incorrect morphological analysis. For instance, let us consider a famous Korean sentence: “아버지가방에들어가셨다.” The true word spacing for this sentence is “아버지가#방에#들어가셨다.” whose meaning is that my father entered the room. If the sentence is written as “어버지#가방에#들어가셨다.”, it means that my father entered the bag, which is totally different from the original meaning. That is, since the morphological analysis is the first-step in most NLP applications, the sentences with incorrect word spacing must be corrected for their further processing. In addition, the wrong word spacing would result in the incorrect index for terms in information retrieval. Thus, correcting the sentences with incorrect word spacing is a critical task in Korean information processing.

One of the most simple and strong models for automatic word spacing is n -gram model. In spite of the advantages of the n -gram model, its problem should be also considered for achieving high performance. The main problem of the model is that it is usually modeled with fixed window size, n . The small value for n represents the narrow context in modeling, which results in poor performance in general. However, it is also difficult to increase n for better performance due to data sparseness. Since the corpus size is physically limited, it is highly possible that many n -grams which do not appear in the corpus exist in the real world.

The goal of this paper is to provide a new method for processing automatic word spacing with an n -gram model. The proposed method automatically adapts the window size n . That is, this method begins with a bigram model, and it shrinks to an unigram model when data sparseness occurs. It also grows up to a trigram, fourgram, and so on when it requires more specific information in determining word spacing. In a word, the proposed model organizes the windows size n online, and achieves high accuracy by removing both data sparseness and information lack.

The rest of the paper is organized as follows. Section 2 surveys the previous work on automatic word spacing and the smoothing methods for n -gram models. Section 3 describes the general way to automatic word spacing by an n -gram model, and Section 4 proposes a self-organizing n -gram model to overcome some drawbacks of n -gram models. Section 5 presents the experimental results. Finally, Section 6 draws conclusions.

2 Previous Work

Many previous work has explored the possibility of automatic word spacing. While most of them reported high accuracy, they can be categorized into two parts in methodology: *analytic approach* and *statistical approach*. The analytic approach is based on the results of morphological analysis. Kang used the fundamental morphological analysis techniques (Kang, 2000), and Kim et al. distinguished each word by the morphemic information of postpositions and endings (Kim et al., 1998). The main drawbacks of this approach are that (i) the analytic step is very complex, and (ii) it is expensive to construct and maintain the analytic knowledge.

In the other hand, the statistical approach extracts from corpora the probability that a space is put between two syllables. Since this approach can obtain the necessary information automatically, it does require neither the linguistic knowledge on syllable composition nor the costs for knowledge construction and maintenance. In addition, the fact that it does not use a morphological analyzer produces solid results even for unknown words. Many previous studies using corpora are based on *bigram* information. According to (Kang, 2004), the number of syllables used in modern Korean is about 10^4 , which implies that the number of bigrams reaches 10^8 . In order to obtain stable statis-

tics for all bigrams, a great large volume of corpora will be required. If higher order n -gram is adopted for better accuracy, the volume of corpora required will be increased exponentially.

The main drawback of n -gram model is that it suffers from data sparseness however large the corpus is. That is, there are many n -grams of which frequency is zero. To avoid this problem, many smoothing techniques have been proposed for construction of n -gram models (Chen and Goodman, 1996). Most of them belongs to one of two categories. One is to pretend each n -gram occurs once more than it actually did (Mitchell, 1996). The other is to interpolate n -grams with lower dimensional data (Jelinek and Mercer, 1980; Katz, 1987). However, these methods artificially modify the original distribution of corpus. Thus, the final probabilities used in learning with n -grams are the ones distorted by a smoothing technique.

A maximum entropy model can be considered as another way to avoid zero probability in n -gram models (Rosenfeld, 1996). Instead of constructing separate models and then interpolate them, it builds a single, combined model to capture all the information provided by various knowledge sources. Even though a maximum entropy approach is simple, general, and strong, it is computationally very expensive. In addition, its performance is mainly dependent on the relevance of knowledge sources, since the prior knowledge on the target problem is very important (Park and Zhang, 2002). Thus, when prior knowledge is not clear and computational cost is an important factor, n -gram models are more suitable than a maximum entropy model.

Adapting features or contexts has been an important issue in language modeling (Siu and Ostendorf, 2000). In order to incorporate long-distance features into a language model, (Rosenfeld, 1996) adopted triggers, and (Mochihashi and Mastumoto, 2006) used a particle filter. However, these methods are restricted to a specific language model. Instead of long-distance features, some other researchers tried local context extension. For this purpose, (Schütze and Singer, 1994) adopted a variable memory Markov model proposed by (Ron et al., 1996), (Kim et al., 2003) applied selective extension of features to POS tagging, and (Dickinson and Meurers, 2005) expanded context of n -gram models to find errors in syntactic anno-

tation. In these methods, only neighbor words or features of the target n -grams became candidates to be added into the context. Since they required more information for better performance or detecting errors, only the context extension was considered.

3 Automatic Word Spacing by n -gram Model

The problem of automatic word spacing can be regarded as a binary classification task. Let a sentence be given as $S = w_1 w_2 \dots w_N$. If i.i.d. sampling is assumed, the data from this sentence are given as $D = \langle (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N) \rangle$ where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{true, false\}$. In this representation, $\mathbf{x}_i$ is a contextual representation of a syllable w_i . If a space should be put after w_i , then y_i , the class of $\mathbf{x}_i$, is *true*. It is *false* otherwise. Therefore, the automatic word spacing is to estimate a function $f : \mathbb{R}^n \rightarrow \{true, false\}$. That is, our task is to determine whether a space should be put after a syllable w_i expressed as $\mathbf{x}_i$ with its context.

The probabilistic method is one of the strong and most widely used methods for estimating f . That is, for each w_i ,

$$f(\mathbf{x}_i) = \arg \max_{y_i \in \{true, false\}} P(y_i | \mathbf{x}_i),$$

where $P(y_i | \mathbf{x}_i)$ is rewritten as

$$P(y_i | \mathbf{x}_i) = \frac{P(\mathbf{x}_i | y_i) P(y_i)}{P(\mathbf{x}_i)}.$$

Since $P(\mathbf{x}_i)$ is independent of finding the class of $\mathbf{x}_i$, $f(\mathbf{x}_i)$ is determined by multiplying $P(\mathbf{x}_i | y_i)$ and $P(y_i)$. That is,

$$f(\mathbf{x}_i) = \arg \max_{y_i \in \{true, false\}} P(\mathbf{x}_i | y_i) P(y_i).$$

In n -gram model, $\mathbf{x}_i$ is expressed with n neighbor syllables around w_i . Typically, n is taken to be two or three, corresponding to a bigram or trigram respectively. $\mathbf{x}_i$ corresponds to $w_{i-1} w_i$ when $n = 2$. In the same way, it is $w_{i-2} w_{i-1} w_i$ when $n = 3$. The simple and easy way to estimate $P(\mathbf{x}_i | y_i)$ is to use maximum likelihood estimate with a large corpus. For instance, consider the case $n = 2$. Then, the probability $P(\mathbf{x}_i | y_i)$ is represented as $P(w_{i-1} w_i | y_i)$, and is computed by

$$\begin{aligned} P(w_{i-1} w_i | y_i) &= \frac{P(w_{i-1} w_i \& y_i)}{P(y_i)} \\ &= \frac{C(w_{i-1} w_i \& y_i)}{C(y_i)}, \end{aligned} \quad (1)$$

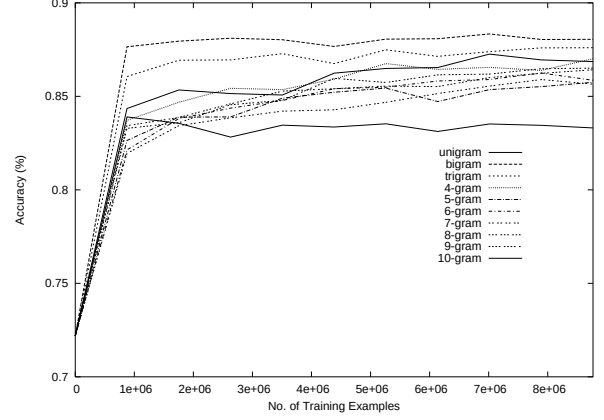


Figure 1: The performance of n -gram models according to the values of n in automatic word spacing.

where C is a counting function.

Determining the context size, the value of n , in n -gram models is closely related with the corpus size. The larger is n , the larger corpus is required to avoid data sparseness. In contrast, though low-order n -grams do not suffer from data sparseness severely, they do not reflect the language characteristics well, either. Typically researchers have used $n = 2$ or $n = 3$, and achieved high performance in many tasks (Bengio et al., 2003). Figure 1 supports that bigram and trigram outperform low-order ($n = 1$) and high-order ($n \geq 4$) n -grams in automatic word spacing. All the experimental settings for this figure follows those in Section 5. In this figure, bigram model shows the best accuracy and trigram achieves the second best, whereas unigram model results in the worst accuracy. Since the bigram model is best, a self-organizing n -gram model explained below starts from bigram.

4 Self-Organizing n -gram Model

To tackle the problem of fixed window size in n -gram models, we propose a self-organizing structure for them.

4.1 Expanding n -grams

When n -grams are compared with $(n + 1)$ -grams, their performance in many tasks is lower than that of $(n + 1)$ -grams (Charniak, 1993). Simultaneously the computational cost for $(n + 1)$ -grams is far higher than that for n -grams. Thus, it can be justified to use $(n + 1)$ -grams instead of n -

Function *HowLargeExpand*($\mathbf{x}_n$)
Input: $\mathbf{x}_n$: n -grams
Output: an integer for expanding size

1. Retrieve $(n + 1)$ -grams $\mathbf{x}_{n+1}$ for $\mathbf{x}_n$.
2. Compute
$$D = KL(P(y|\mathbf{x}_n)||P(y|\mathbf{x}_{n+1})).$$
3. **If** $D < \theta_{\text{EXP}}$ **Then return** 0.
4. **return** *HowLargeExpand*($\mathbf{x}_{n+1}$) + 1.

Figure 2: A function that determines how large a window size should be.

grams only when higher performance is expected. In other words, $(n + 1)$ -grams should be different from n -grams. Otherwise, the performance would not be different. Since our task is attempted with a probabilistic method, the difference can be measured with conditional distributions. If the conditional distributions of n -grams and $(n + 1)$ -grams are similar each other, there is no reason to adopt $(n + 1)$ -grams.

Let $P(y_i|\mathbf{x}_{ni})$ be a class-conditional probability by n -grams and $P(y_i|\mathbf{x}_{(n+1)i})$ that by $(n + 1)$ -grams. Then, the difference $D(n, n + 1)$ between them is measured by Kullback-Leibler divergence. That is,

$$D(n, n + 1) = KL(P(y_i|\mathbf{x}_{ni})||P(y_i|\mathbf{x}_{(n+1)i})),$$

which is computed by

$$\sum_{z \in \{true, false\}} p(z|\mathbf{x}_{ni}) \log \frac{p(z|\mathbf{x}_{ni})}{p(z|\mathbf{x}_{(n+1)i})}. \quad (2)$$

$D(n, n + 1)$ that is larger than a predefined threshold θ_{EXP} implies that $P(y_i|\mathbf{x}_{(n+1)i})$ is different from $P(y_i|\mathbf{x}_{ni})$. In this case, $(n + 1)$ -grams is used instead of n -grams.

Figure 2 depicts an algorithm that determines how large n -grams should be used. It recursively finds the optimal expanding window size. For instance, let bigrams ($n = 2$) be used at first. When the difference between bigrams and trigrams ($n = 3$) is larger than θ_{EXP} , that between trigrams and fourgrams ($n = 4$) is checked again. If it is less than θ_{EXP} , then this function returns 1 and trigrams are used instead of bigrams. Otherwise, it considers higher n -grams again.

Function *HowSmallShrink*($\mathbf{x}_n$)
Input: $\mathbf{x}_n$: n -grams
Output: an integer for shrinking size

1. **If** $n < 1$ **Then return** 0.
2. Retrieve $(n - 1)$ -grams $\mathbf{x}_{n-1}$ for $\mathbf{x}_n$.
3. Compute
$$D = KL(P(y|\mathbf{x}_n)||P(y|\mathbf{x}_{n-1})).$$
4. **If** $D > \theta_{\text{SHR}}$ **Then return** 0.
5. **return** *HowSmallShrink*($\mathbf{x}_{n-1}$) - 1.

Figure 3: A function that determines how small a window size should be used.

4.2 Shrinking n -grams

Shrinking n -grams is accomplished in the direction opposite to expanding n -grams. After comparing n -grams with $(n - 1)$ -grams, $(n - 1)$ -grams are used instead of n -grams only when they are similar enough. The difference $D(n, n - 1)$ between n -grams and $(n - 1)$ -grams is, once again, measured by Kullback-Leibler divergence. That is,

$$D(n, n - 1) = KL(P(y_i|\mathbf{x}_{ni})||P(y_i|\mathbf{x}_{(n-1)i})).$$

If $D(n, n - 1)$ is smaller than another predefined threshold θ_{SHR} , then $(n - 1)$ -grams are used instead of n -grams.

Figure 3 shows an algorithm which determines how deeply the shrinking is occurred. The main stream of this algorithm is equivalent to that in Figure 2. It also recursively finds the optimal shrinking window size, but can not be further reduced when the current model is a unigram.

The merit of shrinking n -grams is that it can construct a model with a lower dimensionality. Since the maximum likelihood estimate is used in calculating probabilities, this helps obtaining stable probabilities. According to the well-known *curse of dimensionality*, the data density required is reduced exponentially by reducing dimensions. Thus, if the lower dimensional model is not different so much from the higher dimensional one, it is highly possible that the probabilities from lower dimensional space are more stable than those from higher dimensional space.

Function *ChangingWindowSize*($\mathbf{x}_n$)
Input: $\mathbf{x}_n$: n -grams
Output: an integer for changing window size

1. **Set** $\text{exp} := \text{HowLargeExpand}(\mathbf{x}_n)$.
2. **If** $\text{exp} > 0$ **Then return** exp .
3. **Set** $\text{shr} := \text{HowSmallShrink}(\mathbf{x}_n)$.
4. **If** $\text{shr} < 0$ **Then return** shr .
5. **return** 0.

Figure 4: A function that determines the changing window size of n -grams.

4.3 Overall Self-Organizing Structure

For a given i.i.d. sample $\mathbf{x}_i$, there are three possibilities on changing n -grams. First one is not to change n -grams. It is obvious when n -grams are not changed. This occurs when both $D(n, n+1) \geq \theta_{\text{EXP}}$ and $D(n, n-1) \leq \theta_{\text{SHR}}$ are met. This is when the expanding results in too similar distribution to that of the current n -grams and the distribution after shrinking is too different from that of the current n -grams.

The remaining possibilities are then *expanding* and *shrinking*. The application order between them can affect the performance of the proposed method. In this paper, an expanding is checked prior to a shrinking as shown in Figure 4. The function *ChangingWindowSize* first calls *HowLargeExpand*. The non-zero return value of *HowLargeExpand* implies that the window size of the current n -grams should be enlarged. Otherwise, *ChangingWindowSize* checks if the window size should be shrunk by calling *HowSmallShrink*. If *HowSmallShrink* returns a negative integer, the window size should be shrunk to $(n + \text{shr})$. If both functions return zero, the window size should not be changed.

The reason why *HowLargeExpand* is called prior to *HowSmallShrink* is that the expanded n -grams handle more specific data. $(n+1)$ -grams, in general, help obtaining higher accuracy than n -grams, since $(n+1)$ -gram data are more specific than n -gram ones. However, it is time-consuming to consider higher-order data, since the number of kinds of data increases. The time increased due to expanding is compensated by shrinking. After shrinking, only lower-order data are considered, and then processing time for them decreases.

4.4 Sequence Tagging

Since natural language sentences are sequential as their nature, the word spacing can be considered as a special POS tagging task (Lee et al., 2002) for which a hidden Markov model is usually adopted. The best sequence of word spacing for the sentence is defined as

$$\begin{aligned} y_{1,N}^* &= \arg \max_{y_{1,N}} P(y_{1,N} | \mathbf{x}_{1,N}) \\ &= \arg \max_{y_{1,N}} \frac{P(\mathbf{x}_{1,N} | y_{1,N}) P(y_{1,N})}{P(\mathbf{x}_{1,N})} \\ &= \arg \max_{y_{1,N}} P(\mathbf{x}_{1,N} | y_{1,N}) P(y_{1,N}) \end{aligned}$$

by where N is a sentence length.

If we assume that the syllables are independent of each other, $P(\mathbf{x}_{1,N} | y_{1,N})$ is given by

$$P(\mathbf{x}_{1,N} | y_{1,N}) = \prod_{i=1}^N P(\mathbf{x}_i | y_i),$$

which can be computed using Equation (1). In addition, by Markov assumption, the probability of a current tag y_i conditionally depends on only the previous k tags. That is,

$$P(y_{1,N}) = \prod_{i=1}^N P(y_i | y_{i-k}, y_{i-1}).$$

Thus, the best sequence is determined by

$$y_{1,N}^* = \arg \max_{y_{1,N}} \prod_{i=1}^N P(\mathbf{x}_i | y_i) \cdot P(y_i | y_{i-k}, y_{i-1}). \quad (3)$$

Since this equation follows Markov assumption, the best sequence is found by applying the Viterbi algorithm.

5 Experiments

5.1 Data Set

The data set used in this paper is the HANTEC corpora version 2.0 distributed by KISTI¹. From this corpus, we extracted only the HKIB94 part which consists of 22,000 news articles in 1994 from Hankook Ilbo. The reason why HKIB94 is chosen is that the word spacing of news articles is relatively more accurate than other texts. Even though this data set is composed of totally 12,523,688 Korean syllables, the number of unique syllables is just

¹<http://www.kisti.re.kr>

Methods	Accuracy (%)
baseline	72.19
bigram	88.34
trigram	87.59
self-organizing bigram	91.31
decision tree	88.68
support vector machine	89.10

Table 1: The experimental results of various methods for automatic word spacing.

2,037 after removing all special symbols, digits, and English alphabets.

The data set is divided into three parts: *training* (70%), *held-out* (20%), and *test* (10%). The held-out set is used only to estimate θ_{EXP} and θ_{SHR} . The number of instances in the training set is 8,766,578, that in the held-out set is 2,504,739, and that in test set is 1,252,371. Among the 1,252,371 test cases, the number of positive instances is 348,278, and that of negative instances is 904,093. Since about 72% of test cases are negative, this is the baseline of the automatic word spacing.

5.2 Experimental Results

To evaluate the performance of the proposed method, two well-known machine learning algorithms are compared together. The tested machine learning algorithms are (i) decision tree and (ii) support vector machines. We use C4.5 release 8 (Quinlan, 1993) for decision tree induction and *SVM^{light}* (Joachims, 1998) for support vector machines. For all experiments with decision trees and support vector machines, the context size is set to two since the bigram shows the best performance in Figure 1.

Table 1 gives the experimental results of various methods including machine learning algorithms and self-organizing n -gram model. The ‘self-organizing bigram’ in this table is the one proposed in this paper. The normal n -grams achieve an accuracy of around 88%, while decision tree and support vector machine produce that of around 89%. The self-organizing n -gram model achieves 91.31%. The accuracy improvement by the self-organizing n -gram model is about 19% over the baseline, about 3% over the normal n -gram model, and 2% over decision trees and support vector machines.

In order to organize the context size for n -grams

Order	No. of Errors
Expanding then Shrinking	108,831
Shrinking then Expanding	114,343

Table 2: The number of errors caused by the application order of context expanding and shrinking.

online, two operations of *expanding* and *shrinking* were proposed. Table 2 shows how much the number of errors is affected by their application order. The number of errors made by expanding first is 108,831 while that by shrinking first is 114,343. That is, if shrinking is applied ahead of expanding, 5,512 additional errors are made. Thus, it is clear that expanding should be considered first.

The errors by expanding can be explained with two reasons: (i) the expression power of the model and (ii) data sparseness. Since Korean is a partially-free word order language and the omission of words are very frequent, n -gram model that captures local information could not express the target task sufficiently. In addition, the class-conditional distribution after expanding could be very different from that before expanding due to data sparseness. In such cases, the expanding should not be applied since the distribution after expanding is not trustworthy. However, only the difference between two distributions is considered in the proposed method, and the errors could be made by data sparseness.

Figure 5 shows that the number of training instances does not matter in computing probabilities of n -grams. Even though the accuracy increases slightly, the accuracy difference after 900,000 instances is not significant. It implies that the errors made by the proposed method is not from the lack of training instance but from the lack of its expression power for the target task. This result also complies with Figure 1.

5.3 Effect of Right Context

All the experiments above considered left context only. However, Kang reported that the probabilistic model using both left and right context outperforms the one that uses left context only (Kang, 2004). In his work, the word spacing probability $P(w_i, w_{i+1})$ between two adjacent syllables w_i and w_{i+1} is given as

$$\begin{aligned}
 P(w_i, w_{i+1}) &= 0.25 \times P_L(w_{i-1}, w_i) \\
 &\quad 0.5 \times P_M(w_i, w_{i+1}) \quad (4) \\
 &\quad 0.25 \times P_R(w_{i+1}, w_{i+2}),
 \end{aligned}$$

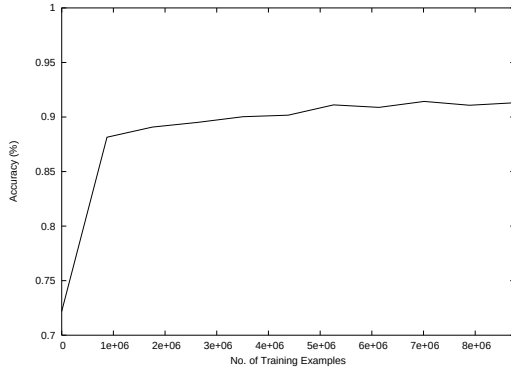


Figure 5: The effect of the number of training examples in the self-organizing n -gram model.

Context	Accuracy (%)
Left Context Only	91.31
Right Context Only	88.26
Both Contexts	92.54

Table 3: The effect of using both left and right context.

where P_R, P_M, P_L are computed respectively based on the syllable frequency.

In order to reflect the idea of bidirectional context in the proposed model, the model is enhanced by modifying $P(w_{i-1}w_i|y_i)$ in Equation (1). That is, the likelihood of $P(w_{i-1}w_i|y_i)$ is expanded to be

$$\begin{aligned} \tilde{P}(w_{i-1}w_i|y_i) &= \alpha_1 \times P(w_{i-2}w_{i-1}|y_i) \\ &\quad \alpha_2 \times P(w_{i-1}w_i|y_i) \\ &\quad \alpha_3 \times P(w_i, w_{i+1}|y_i). \end{aligned}$$

Since the coefficients of Equation (4) were determined arbitrarily (Kang, 2004), they are replaced with parameters α_i of which values are determined using a held-out data.

The change of accuracy by the context is shown in Table 3. When only the right context is used, the accuracy gets 88.26% which is worse than the left context only. That is, the original n -gram is a relatively good model. However, when both left and right context are used, the accuracy becomes 92.54%. The accuracy improvement by using additional right context is 1.23%. This results coincide with the previous report (Lee et al., 2002). The α_i 's to achieve this accuracy are $\alpha_1 = 0.5$, $\alpha_2 = 0.3$, and $\alpha_3 = 0.2$.

Method	Accuracy(%)
Normal HMM	92.37
Self-Organizing HMM	94.71

Table 4: The effect of considering a tag sequence.

5.4 Effect of Considering Tag Sequence

The state-of-the-art performance on Korean word spacing is to use the hidden Markov model. According to the previous work (Lee et al., 2002), the hidden Markov model shows the best performance when it sees two previous tags and two previous syllables.

For the simplicity in the experiments, the value for k in Equation (3) is set to be one. The performance comparison between normal HMM and the proposed method is given in Table 4. The proposed method considers the various number of previous syllables, whereas the normal HMM has the fixed context. Thus, the proposed method in Table 4 is specified as ‘self-organizing HMM.’ The accuracy of the self-organizing HMM is 94.71%, while that of the normal HMM is just 92.37%. Even though the normal HMM considers more previous tags ($k = 2$), the accuracy of the self-organizing model is 2.34% higher than that of the normal HMM. Therefore, the proposed method that considers the sequence of word spacing tags achieves higher accuracy than any other methods reported ever.

6 Conclusions

In this paper we have proposed a new method to learn word spacing in Korean by adaptively organizing context size. Our method is based on the simple n -gram model, but the context size n is changed as needed. When the increased context is much different from the current one, the context size is increased. In the same way, the context is decreased, if the decreased context is not so much different from the current one. The benefits of this method are that it can consider wider context by increasing context size as required, and save the computational cost due to the reduced context.

The experiments on HANTEC corpora showed that the proposed method improves the accuracy of the trigram model by 3.72%. Even compared with some well-known machine learning algorithms, it achieved the improvement of 2.63% over decision trees and 2.21% over support vector machines. In addition, we showed two ways for improving the

proposed method: considering *right context* and *word spacing sequence*. By considering left and right context at the same time, the accuracy is improved by 1.23%, and the consideration of word spacing sequence gives the accuracy improvement of 2.34%.

The n -gram model is one of the most widely used methods in natural language processing and information retrieval. Especially, it is one of the successful language models, which is a key technique in language and speech processing. Therefore, the proposed method can be applied to not only word spacing but also many other tasks. Even though word spacing is one of the important tasks in Korean information processing, it is just a simple task in many other languages such as English, German, and French. However, due to its generality, the importance of the proposed method yet does hold in such languages.

Acknowledgements

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (KRF-2005-202-D00465).

References

- Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, Vol. 3, pp. 1137–1155.
- E. Charniak. 1993. *Statistical Language Learning*. MIT Press.
- S. Chen and J. Goodman. 1996. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 310–318.
- M. Dickinson and W. Meurers. 2005. Detecting Errors in Discontinuous Structural Annotation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pp. 322–329.
- F. Jelinek and R. Mercer. 1980. Interpolated Estimation of Markov Source Parameters from Sparse Data. In *Proceedings of the Workshop on Pattern Recognition in Practice*.
- T. Joachims. 1998. *Making Large-Scale SVM Learning Practical*. LS8, Universität Dortmund.
- S.-S. Kang. 2000. Eojeol-Block Bidirectional Algorithm for Automatic Word Spacing of Hangul Sentences. *Journal of KISS*, Vol. 27, No. 4, pp. 441–447. (in Korean)
- S.-S. Kang. 2004. Improvement of Automatic Word Segmentation of Korean by Simplifying Syllable Bigram. In *Proceedings of the 15th Conference on Korean Language and Information Processing*, pp. 227–231. (in Korean)
- S. 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*. Vol. 35, No. 3, pp. 400–401.
- K.-S. Kim, H.-J. Lee, and S.-J. Lee. 1998. Three-Stage Spacing System for Korean in Sentence with No Word Boundaries. *Journal of KISS*, Vol. 25, No. 12, pp. 1838–1844. (in Korean)
- J.-D. Kim, H.-C. Rim, and J. Tsujii. 2003. Self-Organizing Markov Models and Their Application to Part-of-Speech Tagging. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pp. 296–302.
- D.-G. Lee, S.-Z. Lee, H.-C. Rim, and H.-S. Lim. 2002. Automatic Word Spacing Using Hidden Markov Model for Refining Korean Text Corpora. In *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization*, pp. 51–57.
- T. Mitchell. 1997. *Machine Learning*. McGraw Hill.
- D. Mochihashi and Y. Matsumoto. 2006. Context as Filtering. *Advances in Neural Information Processing Systems 18*, pp. 907–914.
- S.-B. Park and B.-T. Zhang. 2002. A Boosted Maximum Entropy Model for Learning Text Chunking. In *Proceedings of the 19th International Conference on Machine Learning*, pp. 482–489.
- R. Quinlan. 1993. *C4.5: Program for Machine Learning*. Morgan Kaufmann Publishers.
- D. Ron, Y. Singer, and N. Tishby. 1996. The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length. *Machine Learning*, Vol. 25, No. 2, pp. 117–149.
- R. Rosenfeld. 1996. A Maximum Entropy Approach to Adaptive Statistical Language Modeling. *Computer, Speech and Language*, Vol. 10, pp. 187–228.
- H. Schütze and Y. Singer. 1994. Part-of-Speech Tagging Using a Variable Memory Markov Model. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 181–187.
- M. Siu and M. Ostendorf. 2000. Variable N-Grams and Extensions for Conversational Speech Language Modeling. *IEEE Transactions on Speech and Audio Processing*, Vol. 8, No. 1, pp. 63–75.

Concept Unification of Terms in Different Languages for IR

Qing Li, Sung-Hyon Myaeng
Information & Communications
University, Korea
{liqing, myaeng}@icu.ac.kr

Yun Jin
Chungnam National
University, Korea
wkim@cnu.ac.kr

Bo-yeong Kang
Seoul National University,
Korea
comeng99@snu.ac.kr

Abstract

Due to the historical and cultural reasons, English phrases, especially the proper nouns and new words, frequently appear in Web pages written primarily in Asian languages such as Chinese and Korean. Although these English terms and their equivalences in the Asian languages refer to the same concept, they are erroneously treated as independent index units in traditional Information Retrieval (IR). This paper describes the degree to which the problem arises in IR and suggests a novel technique to solve it. Our method firstly extracts an English phrase from Asian language Web pages, and then unifies the extracted phrase and its equivalence(s) in the language as one index unit. Experimental results show that the high precision of our conceptual unification approach greatly improves the IR performance.

1 Introduction

The mixed use of English and local languages presents a classical problem of vocabulary mismatch in monolingual information retrieval (MIR). The problem is significant especially in Asian language because words in the local languages are often mixed with English words. Although English terms and their equivalences in a local language refer to the same concept, they are erroneously treated as independent index units in traditional MIR. Such separation of semantically identical words in different languages may limit retrieval performance. For instance, as shown in Figure 1, there are three kinds of Chinese Web pages containing information related with “Viterbi Algorithm (韦特比算法)”. The first case contains “Viterbi Algorithm” but not its Chinese equivalence “韦特比算法”. The second

[PDF](#) 基于SRN/HMM的非特定人连续手语识别系统
as the states of HMM (hidden Markov models) in which the lattice [Viterbi algorithm](#)
is employed for searching the best word sequence. ...
[www.jdl.ac.cn/doc/2002/JournalSoft.pdf](#) - [Similar pages](#) - [Remove result](#)
[PDF](#) 概率句法分析(Statistical Parsing)
File Format: Microsoft Powerpoint 97 - [View as HTML](#)
[韦特比算法](#) 剪枝策略: 计算相同规约、相同起点、相同后继的边的概率, 保留最大概率边。
[icl.pku.edu.cn/WebData_http-dir-listable/ICLseminars/2002spring/张浩——概率句法分析的研究.ppt](#) - [Similar pages](#) - [Remove result](#)
[北京大学汉语语言学研究中心](#) - [[Translate this page](#)]
22. [韦特比算法](#), [Viterbi Algorithm](#), 计算语言学, 85. 中文术语, 韦特比算法. 英文术语, [Viterbi Algorithm](#). 拼音, [wei2 te4 bi3 suan4 fa3](#) ...
[ccl.pku.edu.cn/term.asp?item=1&spell=V&page=3&expand=118-20k](#) - [Supplemental Result](#) - [Cached](#) - [Similar pages](#) - [Remove result](#)

Figure 1. Three Kinds of Web Pages

contains “韦特比算法” but not “Viterbi Algorithm”. The third has both of them. A user would expect that a query with either “Viterbi Algorithm” or “韦特比算法” would retrieve all of these three groups of Chinese Web pages. Otherwise some potentially useful information will be ignored.

Furthermore, one English term may have several corresponding terms in a different language. For instance, Korean words “디지털”, “디지틀”, and “디지털” are found in local Web pages, which all correspond to the English word “digital” but are in different forms because of different phonetic interpretations. Establishing an equivalence class among the three Korean words and the English counterpart is indispensable. By doing so, although the query is “디지털”, the Web pages containing “디지틀”, “디지털” or “digital” can be all retrieved. The same goes to Chinese terms. For example, two same semantic Chinese terms “维特比” and “韦特比” correspond to one English term “Viterbi”. There should be a semantic equivalence relation between them.

Although tracing the original English term from a term in a native language by back transliteration (Jeong et al., 1999) is a good way to build such mapping, it is only applicable to the words that are amenable for transliteration based on the phoneme. It is difficult to expand the method to abbreviations and compound words.

Since English abbreviations frequently appear in Korean and Chinese texts, such as “세계무역기구 (WTO)” in Korean, “世界贸易组织 (WTO)” in Chinese, it is essential in IR to have a mapping between these English abbreviations and the corresponding words. The same applies to the compound words like “서울대 (Seoul National University)” in Korean, “疯牛病 (mad cow disease)” in Chinese. Realizing the limitation of the transliteration, we present a way to extract the key English phrases in local Web pages and conceptually unify them with their semantically identical terms in the local language.

2 Concept Unification

The essence of the concept unification of terms in different languages is similar to that of the query translation for cross-language information retrieval (CLIR) which has been widely explored (Cheng et al., 2004; Cao and Li, 2002; Fung et al., 1998; Lee, 2004; Nagata et al., 2001; Rapp, 1999; Zhang et al., 2005; Zhang and Vine, 2004). For concept unification in index, firstly key English phrases should be extracted from local Web pages. After translating them into the local language, the English phrase and their translation(s) are treated as the same index units for IR. Different from previous work on query term translation that aims at finding relevant terms in another language for the target term in source language, conceptual unification requires a high translation precision. Although the fuzzy Chinese translations (e.g. “病毒 (virus), 陈盈豪 (designer’s name), 电脑病毒 (computer virus)”) of English term “CIH” can enhance the CLIR performance by the “query expansion” gain (Cheng et al., 2004), it does not work in the conceptual unification of terms in different languages for IR. While there are lots of additional sources to be utilized for phrase translation (e.g., anchor text, parallel or comparable corpus), we resort to the mixed language Web pages which are the local Web pages with some English words, because they are easily obtainable and frequently self-refresh.

Observing the fact that English words sometimes appear together with their equivalence in a local language in Web texts as shown in Figure 1, it is possible to mine the mixed language search-result pages obtained from Web search engines and extract proper translations for these English words that are treated as queries. Due to the language nature of Chinese and Korean, we integrate the phoneme and semanteme instead of

statistical information alone to pick out the right translation from the search-result pages.

3 Key Phrase Extraction

Since our intention is to unify the semantically identical words in different languages and index them together, the primary task is to decide what kinds of key English phrases in local Web pages are necessary to be conceptually unified.

In (Jeong et al., 1999), it extracts the Korean foreign words for concept unification based on statistical information. Some of the English equivalences of these Korean foreign words, however, may not exist in the Korean Web pages. Therefore, it is meaningless to do the cross-language concept unification for these words. The English equivalence would not benefit any retrieval performance since no local Web pages contain it, even if the search system builds a semantic class among both local language and English for these words. In addition, the method for detecting Korean foreign words may bring some noise. The Korean terms detected as foreign words sometimes are not meaningful. Therefore, we do it the other way around by choosing the English phrases from the local Web pages based on a certain selection criteria.

Instead of extracting all the English phrases in the local Web pages, we only select the English phrases that occurred within the special marks including quotation marks and parenthesis. Because English phrases within these markers reveal their significance in information searching to some extent. In addition, if the phrase starts with some stemming words (e.g., for, as) or includes some special sign, it is excluded as the phrases to be translated.

4 Translation of English Phrases

In order to translate the English phrases extracted, we query the search engine with English phrases to retrieve the local Web pages containing them. For each document returned, only the title and the query-biased summary are kept for further analysis. We dig out the translation(s) for the English phrases from these collected documents.

4.1 Extraction of Candidates for Selection

After querying the search engine with the English phrase, we can get the snippets (title and summary) of Web texts in the returned search-result pages as shown in Figure 1. The next step then is to extract translation candidates within a window of a limited size, which includes the

English phrase, in the snippets of Web texts in the returned search-result pages. Because of the agglutinative nature of the Chinese and Korean languages, we should group the words in the local language into proper units as translation candidates, instead of treating each individual word as candidates. There are two typical ways: one is to group the words based on their co-occurrence information in the corpus (Cheng et al., 2004), and the other is to employ all sequential combinations of the words as the candidates (Zhang and Vine, 2004). Although the first reduces the number of candidates, it risks losing the right combination of words as candidates. We adopt the second in our approach, so that, return to the aforementioned example in Figure 1, if there are three Chinese characters (韦特比) within the predefined window, the translation candidates for English phrases “Viterbi” are “韦”, “特”, “比”, “韦特”, “特比”, and “韦特比”. The number of candidates in the second method, however, is greatly increased by enlarging the window size k . Realizing that the number of words, n , available in the window size, k , is generally larger than the predefined maximum length of candidate, m , it is unreasonable to use all adjacent sequential combinations of available words within the window size k . Therefore, we tune the method as follows:

1. If $n \leq m$, all adjacent sequential combinations of words within the window are treated as candidates

2. If $n > m$, only adjacent sequential combinations of which the word number is less than m are regarded as candidates. For example, if we set n to 4 and m to 2, the window “ $w_1 w_2 w_3 w_4$ ” consists of four words. Therefore, only “ $w_1 w_2$ ”, “ $w_2 w_3$ ”, “ $w_3 w_4$ ”, “ w_1 ”, “ w_2 ”, “ w_3 ”, “ w_4 ” are employed as the candidates for final translation selection.

Based on our experiments, this tuning method achieves the same performance while reducing the candidate size greatly.

4.2 Selection of candidates

The final step is to select the proper candidate(s) as the translation(s) of the key English phrase. We present a method that considers the statistical, phonetic and semantic features of the English candidates for selection.

Statistical information such as co-occurrence, Chi-square, mutual information between the English term and candidates helps distinguish the right translation(s). Using Cheng’s Chi-square

method (Cheng et al., 2004), the probability to find the right translation for English specific term is around 30% in the top-1 case and 70% in the top-5 case. Since our goal is to find the corresponding counterpart(s) of the English phrase to treat them as one index unit in IR, the accuracy level is not satisfactory. Since it seems difficult to improve the precision solely through variant statistical methods, we also consider semantic and phonetic information of candidates besides the statistical information. For example, given the English Key phrase “Attack of the clones”, the right Korean translation “클론의 습격” is far away from the top-10 selected by Chi-square method (Cheng et al., 2004). However, based on the semantic match of “습격” and “Attack”, and the phonetic match of “클론” and “clones”, we can safely infer they are the right translation. The same rule applies to the Chinese translation “克隆人的进攻”, where “克隆人” is phonetically match for “clones” and “进攻” semantically corresponds to “attack”.

In selection step, we first remove most of the noise candidates based on the statistical method and re-rank the candidates based on the semantic and phonetic similarity.

4.3 Statistical model

There are several statistical models to rank the candidates. Nagata (2001) and Huang (2005) use the frequency of co-occurrence and the textual distance, the number of words between the Key phrase and candidates in texts to rank the candidates, respectively. Although the details of the methods are quite different, both of them share the same assumption that the higher co-occurrence between candidates and the Key phrase, the more possible they are the right translations for each other. In addition, they observed that most of the right translations for the Key phrase are close to it in the text, especially, right after or before the key phrase (e.g. “... 연방수사국(FBI)이...”). Zhang (2004) suggested a statistical model based on the frequency of co-occurrence and the length of the candidates. In the model, since the distance between the key phrase and a candidate is not considered, the right translation located far away from the key phrase also has a chance to be selected. We observe, however, that such case is very rare in our study, and most of right translations are located within 5~8 words. The distance information is a valuable factor to be considered.

In our statistical model, we consider the frequency, length and location of candidates together. The intuition is that if the candidate is the right translation, it tends to co-occur with the key phrase frequently; its location tends to be close to the key phrase; and the longer the candidates' length, the higher the chance to be the right translation. The formula to calculate the ranking score for a candidate is as follows:

$$w_{FL}(q, c_i) = \alpha \times \frac{\text{len}(c_i)}{\max_{\text{len}}} + (1 - \alpha) \times \frac{\sum_k \frac{1}{d_k(q, c_i)}}{\max_{\text{Freq-len}}}$$

where $d_k(q, c_i)$ is the word distance between the English phrase q and the candidate c_i in the k -th occurrence of candidate in the search-result pages. If q is adjacent to c_i , the word distance is one. If there is one word between them, it is counted as two and so forth. α is the coefficient constant, and $\max_{\text{Freq-len}}$ is the max reciprocal of $d_k(q, c_i)$ among all the candidates. $\text{len}(c_i)$ is the number of characters in the candidate c_i .

4.4 Phonetic and semantic model

Phonetic and semantic match: There has been some related work on extracting term translation based on the transliteration model (Kang and Choi, 2002; Kang and Kim, 2000). Different from transliteration that attempts to generate English transliteration given a foreign word in local language, our approach is a kind a match problem since we already have the candidates and aim at selecting the right candidates as the final translation(s) for the English key phrase.

While the transliteration method is partially successful, it suffers from the problem that transliteration rules are not applied consistently. The English key phrase for which we are looking for the translation sometimes contains several words that may appear in a dictionary as an independent unit. Therefore, it can only be partially matched based on the phonetic similarity, and the rest part may be matched by the semantic similarity in such situation. Returning to the above example, “clone” is matched with “클론” by phonetic similarity. “of” and “attack” are matched with “의” and “습격” respectively by semantic similarity. The objective is to find a set of mappings between the English word(s) in the key phrase and the local language word(s) in candidates, which maximize the sum of the semantic and phonetic mapping weights. We call the sum as SSP (Score of semanteme and phoneme). The

higher SSP value is, the higher the probability of the candidate to be the right translation.

The solution for a maximization problem can be found using an exhaustive search method. However, the complexity is very high in practice for a large number of pairs to be processed. As shown in Figure 2, the problem can be represented as a bipartite weighted graph matching problem. Let the English key phrase, E , be represented as a sequence of tokens $\langle ew_1, \dots, ew_m \rangle$, and the candidate in local language, C , be represented as a sequence of tokens $\langle cw_1, \dots, cw_n \rangle$. Each English and candidate token is represented as a graph vertex. An edge (ew_i, cw_j) is formed with the weight $\omega(ew_i, cw_j)$ calculated as the average of normalized semantic and phonetic values, whose calculation details are explained below. In order to balance the number of vertices on both sides, we add the virtual vertex (vertices) with zero weight on the side with less number of vertices. The SSP is calculated:

$$\text{SSP} = \arg\max \sum_{i=1}^n \omega(kw_i, ew_{\pi(i)})$$

where π is a permutation of $\{1, 2, 3, \dots, n\}$. It can be solved by the Kuhn-Munkres algorithm (also known as Hungarian algorithm) with polynomial time complexity (Munkres, 1957).

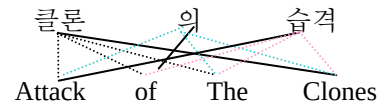


Figure 2. Matching based on the semanteme and phoneme

Phonetic & Semantic Weights: If two languages have a close linguistic relationship such as English and French, cognate matching (Davis, 1997) is typically employed to translate the untranslatable terms. Interestingly, Buckley et al., (2000) points out that “English query words are treated as potentially misspelled French words” and attempts to treat English words as variations of French words according to lexicographical rules. However, when two languages are very distinct, e.g., English–Korean, English–Chinese, transliteration from English words is utilized for cognate matching.

Phonetic weight is the transliteration probability between English and candidates in local language. We adopt the method in (Jeong et al., 1999) with some adjustments. In essence, we compute the probabilities of particular English

key phrase EW given a candidate in the local language CW .

$$\begin{aligned}\omega_{\text{phoneme}}(EW, CW) &= \omega_{\text{phoneme}}(e_1, \dots, e_m, c_1, \dots, c_k) \\ &= \omega_{\text{phoneme}}(g_1, \dots, g_n, c_1, \dots, c_k) = \frac{1}{n} \sum_j \log P(g_j | g_{j-1}) P(c_j | g_j)\end{aligned}$$

where the English phrase consists of a string of English alphabets $e_1, \dots, e_m$, and the candidate in the local language is comprised of a string of phonetic elements. $c_1, \dots, c_k$. For Korean language, the phonetic element is the Korean alphabets such as “ㄱ”, “ ㄴ”, “ㄷ”, “ㅎ” and etc. For Chinese language, the phonetic elements mean the elements of “pinyin”. g_i is a pronunciation unit comprised of one or more English alphabets (e.g., ‘ss’ for ‘ㅅ’, a Korean alphabet).

The first term in the product corresponds to the transition probability between two states in HMM and the second term to the output probability for each possible output that could correspond to the state, where the states are all possible distinct English pronunciation units for the given Korean or Chinese word. Because the difference between Korean/Chinese and English phonetic systems makes the above uni-gram model almost impractical in terms of output quality, bi-grams are applied to substitute the single alphabet in the above equation. Therefore, the phonetic weight should be calculated as:

$$\omega_{\text{phoneme}}(E, C) = \frac{1}{n} \sum_j \log P(g_j g_{j+1} | g_{j-1} g_j) P(c_j c_{j+1} | g_j g_{j+1})$$

where $P(c_j c_{j+1} | g_j g_{j+1})$ is computed from the training corpus as the ratio between the frequency of $c_j c_{j+1}$ in the candidates, which were originated from $g_j g_{j+1}$ in English words, to the frequency of $g_j g_{j+1}$. If $j=1$ or $j=n$, g_{j-1} or g_{j+1} , c_{j+1} is substituted with a space marker.

The semantic weight is calculated from the bilingual dictionary. The current bilingual dictionary we employed for the local languages are Korean-English WorldNet and LDC Chinese-English dictionary with additional entries inserted manually. The weight relies on the degree of overlaps between an English translation and the candidate

$$w_{\text{semanteme}}(E, C) = \text{argmax} \frac{\text{No. of overlapping units}}{\text{total No. of units}}$$

For example, given the English phrase “Inha University” and its candidate “인하대 (Inha

University), “University” is translated into “대학교”, therefore, the semantic weight between “University” and “대” is about 0.33 because only one third of the full translation is available in the candidate.

Due to the range difference between phonetic and semantic weights, we normalized them by dividing the maximum phonetic and semantic weights in each pair of the English phrase and a candidate if the maximum is larger than zero.

The strategy for us to pick up the final translation(s) is distinct on two different aspects from the others. If the SSP values of all candidates are less than the threshold, the top one obtained by statistical model is selected as the final translation. Otherwise, we re-rank the candidates according to the SSP value. Then we look down through the new rank list and draw a “virtual” line if there is a big jump of SSP value. If there is no big jump of SSP values, the “virtual” line is drawn at the bottom of the new rank list. Instead of the top-1 candidate, the candidates above the “virtual” line are all selected as the final translations. It is because that an English phrase may have more than one correct translation in the local language. Return to the previous example, the English term “Viterbi” corresponds to two Chinese translations “维特比” and “韦特比”. The candidate list based on the statistical information is “编码, 算法, 译码, 维特比, ..., 韦特比”. We then calculate the SSP value of these candidates and re-rank the candidates whose SSP values are larger than the threshold which we set to 0.3. Since the SSP value of “维特比(0.91)” and “韦特比(0.91)” are both larger than the threshold and there is no big jump, both of them are selected as the final translation.

5 Experimental Evaluation

Although the technique we developed has values in their own right and can be applied for other language engineering fields such as query translation for CLIR, we intend to understand to what extent monolingual information retrieval effectiveness can be increased when relevant terms in different language are treated as one unit while indexing. We first examine the translation precision and then study the impact of our approach for monolingual IR.

We crawls the web pages of a specific domain (university & research) by WIRE crawler provided by center of Web Research, university of Chile (<http://www.cwr.cl/projects/WIRE/>). Currently, we have downloaded 32 sites with 5,847

Korean Web pages and 74 sites with 13,765 Chinese Web pages. 232 and 746 English terms were extracted from Korean Web pages and Chinese Web pages, respectively. The accuracy of unifying semantically identical words in different languages is dependant on the translation performance. The translation results are shown in table 1. As it can be observed, 77% of English terms from Korean web pages and 83% of English terms from Chinese Web pages can be strictly translated into accurate Korean and Chinese, respectively. However, additional 15% and 14% translations contained at least one Korean and Chinese translations, respectively. The errors were brought in by containing additional related information or incomplete translation. For instance, the English term “blue chip” is translated into “蓝芯(blue chip)”, “蓝筹股 (a kind of stock)”. However, another acceptable translation “绩优股 (a kind of stock)” is ignored. An example for incomplete translation is English phrase “SIGIR 2005” which only can be translate into “国际计算机检索年会 (international conference of computer information retrieval)” ignoring the year.

	Korean		Chinese	
	No.	%	No.	%
Exactly correct	179	77%	618	83%
At least one is correct but not all	35	15%	103	14%
Wrong translation	18	8%	25	3%
Total	232	100%	746	100%

Table 1. Translation performance

We also compare our approach with two well-known translation systems. We selected 200 English words and translate them into Chinese and Korean by these systems. Table 2 and Table 3 show the results in terms of the top 1, 3, 5 inclusion rates for Korean and Chinese translation, respectively. “Exactly and incomplete” translations are all regarded as the right translations. “LiveTrans” and “Google” represent the systems against which we compared the translation ability. Google provides a machine translation function to translate text such as Web pages. Although it works pretty well to translate sentences, it is ineligible for short terms where only a little contextual information is available for translation. LiveTrans (Cheng et al., 2004) provided by the WKD lab in Academia Sinica is the first unknown word translation system based on web-mining. There are two ways in this system to

translate words: the fast one with lower precision is based on the “chi-square” method (χ^2) and the smart one with higher precision is based on “context-vector” method (CV) and “chi-square” method (χ^2) together. “ST” and “ST+PS” represent our approaches based on statistic model and statistic model plus phonetic and semantic model, respectively.

		Top -1	Top-3	Top -5
Google		56%	NA	NA
Live Trans	“Fast” χ^2	37%	43%	53.5%
	“Smart” χ^2 +CV	42%	49%	60%
Our Methods	ST($d_k=1$)	28.5 %	41%	47%
	ST	39 %	46.5%	55.5%
	ST+PS	93%	93%	93%

Table 2. Comparison (Chinese case)

		Top -1	Top-3	Top -5
Google		44%	NA	NA
Live Trans	“Fast” χ^2	28%	37.5%	45%
	“Smart” χ^2 +CV	24.5%	44%	50%
Our Methods	ST($d_k=1$)	26.5 %	35.5%	41.5%
	ST	32 %	40%	46.5%
	ST+PS	89%	89.5%	89.5%

Table 3. Comparison (Korean case)

Even though the overall performance of LiveTrans’ combined method (χ^2 +CV) is better than the simple method (χ^2) in both Table 2 and 3, the same doesn’t hold for each individual. For instance, “Jordan” is the English translation of Korean term “요르단”, which ranks 2nd and 5th in (χ^2) and (χ^2 +CV), respectively. The context-vector sometimes misguides the selection.

In our two-step selection approach, the final selection would not be diverted by the false statistic information. In addition, in order to examine the contribution of distance information in the statistical method, we ran our experiments based on statistical method (ST) with two different conditions. In the first case, we set $d_k(q, c_i)$ to 1, that is, the location information of all candidates is ignored. In the second case, $d_k(q, c_i)$ is calculated based on the real textual distance of the candidates. As in both Table 2 and Table 3, the later case shows better performance.

As shown in both Table 2 and Table 3, it can be observed that “ST+PS” shows the best performance, then followed by “LiveTrans (smart)”, “ST”, “LiveTrans(fast)”, and “Google”. The sta-

tistical methods seem to be able to give a rough estimate for potential translations without giving high precision. Considering the contextual words surrounding the candidates and the English phrase can further improve the precision but still less than the improvement made by the phonetic and semantic information in our approach. High precision is very important to the practical application of the translation results. The wrong translation sometimes leads to more damage to its later application than without any translation available. For instance, the Chinese translation of “viterbi” is “算法(algorithm)” by LiveTrans (fast). Obviously, treating “Viterbi” and “算法(algorithm)” as one index unit is not acceptable.

We ran monolingual retrieval experiment to examine the impact of our concept unification on IR. The retrieval system is based on the vector space model with our own indexing scheme to which the concept unification part was added. We employed the standard $tf \times idf$ scheme for index term weighting and idf for query term weighting. Our experiment is based on KT-SET test collection (Kim et al., 1994). It contains 934 documents and 30 queries together with relevance judgments for them.

In our index scheme, we extracted the key English phrases in the Korean texts, and translated them. Each English phrases and its equivalence(s) in Korean is treated as one index unit. The baseline against which we compared our approach applied a relatively simple indexing technique. It uses a dictionary that is Korean-English WordNet, to identify index terms. The effectiveness of the baseline scheme is comparable with other indexing methods (Lee and Ahn, 1999). While there is a possibility that an indexing method with a full morphological analysis may perform better than our rather simple method, it would also suffer from the same problem, which can be alleviated by concept unification approach. As shown in Figure 3, we obtained 14.9 % improvement based on mean average 11-pt precision. It should be also noted that this result was obtained even with the errors made by the unification of semantically identical terms in different languages.

6 Conclusion

In this paper, we showed the importance of the unification of semantically identical terms in different languages for Asian monolingual information retrieval, especially Chinese and Korean. Taking the utilization of the high translation ac-

curacy of our previous work, we successfully unified the most semantically identical terms in the corpus. This is along the line of work where researchers attempt to index documents with concepts rather than words. We would extend our work along this road in the future.

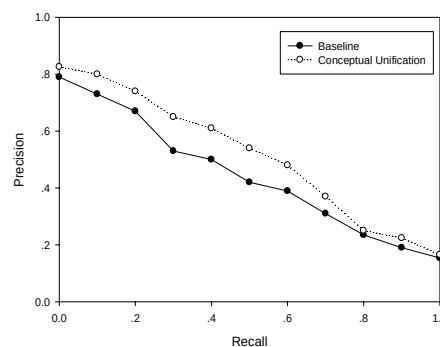


Figure 3. Korean Monolingual IR

Reference

- Buckley, C., Mitra, M., Janet, A. and Walz, C.C.. 2000. Using Clustering and Super Concepts within SMART: TREC 6. Information Processing & Management. 36(1): 109-131.
- Cao, Y. and Li., H.. 2002. Base Noun Phrase Translation Using Web Data and the EM Algorithm. In Proc. of. the 19th COLING.
- Cheng, P., Teng, J., Chen, R., Wang, J., Liu, W., Chen, L.. 2004. Translating Specific Queries with Web Corpora for Cross-language Information Retrieval. In Proc. of ACM SIGIR.
- Davis, M.. 1997. New Experiments in Cross-language Text Retrieval at NMSU's Computing Research Lab. In Proc. Of TREC-5.
- Fung, P. and Yee., L.Y.. 1998. An IR Approach for Translating New Words from Nonparallel, Comparable Texts. In Proc. of COLING/ACL-98.
- Huang, F., Zhang, Y. and Vogel, S.. 2005. Mining Key Phrase Translations from Web Corpora, In Proc. of the Human Language Technologies Conference (HLT-EMNLP).
- Jeong, K. S., Myaeng, S. H., Lee, J. S., Choi, K. S.. 1999. Automatic identification and back-transliteration of foreign words for information retrieval. Information Processing & Management. 35(4): 523-540.
- Kang, B. J., and Choi, K. S. 2002. Effective Foreign Word Extraction for Korean Information Retrieval. Information Processing & Management, 38(1): 91-109.

- Kang, I. H. and Kim, G. C.. 2000. English-to-Korean Transliteration using Multiple Unbounded Overlapping Phoneme Chunks. In Proc. of COLING .
- Kim, S.-H. et al.. 1994. Development of the Test Set for Testing Automatic Indexing. In Proc. of the 22nd KISS Spring Conference. (in Korean).
- Lee, J. H. and Ahn, J. S.. 1996. Using N-grams for Korean Test Retrieval. In Proc. of SIGIR.
- Lee, J. S.. 2004. Automatic Extraction of Translation Phrase Enclosed within Parentheses using Bilingual Alignment Method. In Proc. of the 5th China-Korea Joint Symposium on Oriental Language Processing and Pattern Recognition.
- Munkres, J.. 1957. Algorithms for the Assignment and Transportation Problems. J. Soc. Indust. Appl. Math., 5 (1957).
- Nagata, M., Saito, T., and Suzuki, K.. 2001. Using the Web as a Bilingual Dictionary. In Proc. of ACL '2001 DD-MT Workshop.
- Rapp, R.. 1999. Automatic Identification of Word Translations from Unrelated English and German corpora. In Proc. of ACL.
- Zhang, Y., Huang, F. and Vogel, S.. 2005. Mining Translations of OOV Terms from the Web through Cross-lingual Query Expansion, In Proc. of ACM SIGIR-05.
- Zhang, Y. and Vines, P.. 2004. Using the Web for Automated Translation Extraction in Cross-Language Information Retrieval. In Proc. of ACM SIGIR-04.

Word Alignment in English-Hindi Parallel Corpus Using Recency-Vector Approach: Some Studies

Niladri Chatterjee

Department of Mathematics
Indian Institute of Technology Delhi
Hauz Khas, New Delhi
INDIA - 110016
niladri_iitd@yahoo.com

Saumya Agrawal

Department of Mathematics
Indian Institute of Technology
Kharagpur, West Bengal
INDIA - 721302
saumya-agrawal2000@yahoo.co.in

Abstract

Word alignment using recency-vector based approach has recently become popular. One major advantage of these techniques is that unlike other approaches they perform well even if the size of the parallel corpora is small. This makes these algorithms worth-studying for languages where resources are scarce. In this work we studied the performance of two very popular recency-vector based approaches, proposed in (Fung and McKeown, 1994) and (Somers, 1998), respectively, for word alignment in English-Hindi parallel corpus. But performance of the above algorithms was not found to be satisfactory. However, subsequent addition of some new constraints improved the performance of the recency-vector based alignment technique significantly for the said corpus. The present paper discusses the new version of the algorithm and its performance in detail.

1 Introduction

Several approaches including statistical techniques (Gale and Church, 1991; Brown et al., 1993), lexical techniques (Huang and Choi, 2000; Tiedemann, 2003) and hybrid techniques (Ahrenberg et al., 2000), have been pursued to design schemes for word alignment which aims at establishing links between words of a source language and a target language in a parallel corpus. All these schemes rely heavily on rich linguistic resources, either in the form of huge data of parallel texts or various language/grammar related tools, such as parser, tagger, morphological analyser etc.

Recency vector based approach has been proposed as an alternative strategy for word alignment. Approaches based on recency vectors typically consider the positions of the word in the corresponding texts rather than sentence boundaries. Two algorithms of this type can be found in (Fung and McKeown, 1994) and (Somers, 1998). The algorithms first compute the *position vector* V_w for the word w in the text. Typically, V_w is of the form $\langle p_1 p_2 \dots p_k \rangle$, where the p_i s indicate the positions of the word w in a text T . A new vector R_w , called the *recency vector*, is computed using the position vector V_w , and is defined as $\langle p_2 - p_1, p_3 - p_2, \dots, p_k - p_{k-1} \rangle$. In order to compute the alignment of a given word in the source language text, the recency vector of the word is compared with the recency vector of each target language word and the similarity between them is measured by computing a matching cost associated with the recency vectors using dynamic programming. The target language word having the least cost is selected as the aligned word.

The results given in the above references show that the algorithms worked quite well in aligning words in parallel corpora of language pairs consisting of various European languages and Chinese, Japanese, taken pair-wise. Precision of about 70% could be achieved using these algorithms. The major advantage of this approach is that it can work even on a relatively small dataset and it does not rely on rich language resources.

The above advantage motivated us to study the effectiveness of these algorithms for aligning words in English-Hindi parallel texts. The corpus used for this work is described in Table 1. It has been made manually from three different sources: children's storybooks, English to Hindi translation book material, and advertisements. We shall call

the three corpora as Storybook corpus, Sentence corpus and Advertisement corpus, respectively.

2 Word Alignment Algorithm: Recency Vector Based Approach

DK-vec algorithm given in (Fung and McKeown, 1994) uses the following dynamic programming based approach to compute the matching cost $C(n, m)$ of two vectors v_1 and v_2 of lengths n and m , respectively. The cost is calculated recursively using the following formula,

$$C(i, j) = |v_1(i) - v_2(j)| + \min\{C(i-1, j), C(i-1, j-1), C(i, j-1)\}$$

where i and j have values from 2 to n and 2 to m respectively, n and m being the number of distinct words in source and target language corpus respectively. Note that $v_l(k)$ denotes the k^{th} entry of the vector v_l , for $l = 1$ and 2 . The costs are initialised as follows.

$$\begin{aligned} C(1, 1) &= |v_1(1) - v_2(1)|; \\ C(i, 1) &= |v_1(i) - v_2(1)| + C(i-1, 1); \\ C(1, j) &= |v_1(1) - v_2(j)| + C(1, j-1); \end{aligned}$$

The word in the target language that has the minimum normalized cost ($C(n, m)/(n + m)$) is taken as the translation of the word considered in the source text.

One major shortcoming of the above scheme is its high computational complexity i.e. $O(mn)$. A variation of the above scheme has been proposed in (Somers, 1998) which has a much lower computational complexity $O(\min(m, n))$. In this new scheme, a distance called Levenshtein distance(S) is successively measured using :

$$S = S + \min\{|v_1(i+1) - v_2(j)|, |v_1(i+1) - v_2(j+1)|, |v_1(i) - v_2(j+1)|\}$$

The word in the target text having the minimum value of S (Levenshtein difference) is considered to be the translation of the word in the source text.

2.1 Constraints Used in the Dynamic Programming Algorithms

In order to reduce the complexity of the dynamic programming algorithm certain constraints have been proposed in (Fung and McKeown, 1994).

1. *Starting Point Constraint*: The constraint imposed is: |first-occurrence of source language word (w_1) - first-occurrence of target language word w_2 | $< \frac{1}{2} * (\text{length of the text})$.

2. *Euclidean distance constraint*: The constraint imposed is:

$$\sqrt{(m_1 - m_2)^2 + (s_1 - s_2)^2} < T, \text{ where } m_j \text{ and } s_j \text{ are the mean and standard deviation, respectively, of the recency vector of } w_j, j = 1 \text{ or } 2. \text{ Here, } T \text{ is some predefined threshold:}$$

3. *Length Constraint*: The constraint imposed is: $\frac{1}{2} * f_2 < f_1 < 2 * f_2$, where f_1 and f_2 are the frequencies of occurrence of w_1 and w_2 , in their respective texts.

2.2 Experiments with DK-vec Algorithm

The results of the application of this algorithm have been very poor when applied on the three English to Hindi parallel corpora mentioned above without imposing any constraints.

We then experimented by varying the values of the parameters in the constraints in order to observe their effects on the accuracy of alignment. As was suggested in (Somers, 1998), we also observed that the Euclidean distance constraint is not very beneficial when the corpus size is small. So this constraint has not been considered in our subsequent experiments. Starting point constraint imposes a range within which the search for the matching word is restricted. Although Fung and McKeown suggested the range to be half of the length of the text, we felt that the optimum value of this range will vary from text to text depending on the type of corpus, length ratio of the two texts etc. Table 2 shows the results obtained on applying the DK vec algorithm on Sentence corpus for different lower values of range. Similar results were obtained for the other two corpora. The maximum increase observed in the F-score is around 0.062 for the Sentence corpus, 0.03 for the Story book corpus and 0.05 for the Advertisement corpus. None of these improvements can be considered to be significant.

2.3 Experiments with Somers' Algorithm

The algorithm provided by Somers works by first finding all the minimum score word pairs using dynamic programming, and then applying three filters *Multiple Alignment Selection filter*, *Best Alignment Score Selection filter* and *Frequency Range constraint* to the raw results to increase the accuracy of alignment.

The *Multiple Alignment Selection*(MAS) filter takes care of situations where a single target language word is aligned with the number of source

Corpora	English corpus		Hindi corpus	
	Total words	Distinct words	Total words	Distinct words
Storybook corpus	6545	1079	7381	1587
Sentence corpus	8541	1186	9070	1461
Advertisement corpus	3709	1307	4009	1410

Table 1: Details of English-Hindi Parallel Corpora

Range	Available	Proposed	Correct	P%	R%	F-score
50	516	430	34	7.91	6.59	0.077
150	516	481	51	10.60	09.88	0.102
250	516	506	98	19.37	18.99	0.192
500	516	514	100	19.46	19.38	0.194
700	516	515	94	18.25	18.22	0.182
800	516	515	108	20.97	20.93	0.209
900	516	515	88	17.09	17.05	0.171
1000	516	516	100	19.38	19.38	0.194
2000	516	516	81	15.70	15.70	0.157
4535	516	516	76	14.73	14.73	0.147

Table 2: Results of DK-vec Algorithm on Sentence Corpus for different range

language words. Somers has suggested that in such cases only the word pair that has the minimum alignment score should be considered. Table 3 provides results (see column F-score old) when the raw output is passed through the MAS filters for the three corpora. Note that for all the three corpora a variety of frequency ranges have been considered, and we have observed that the results obtained are slightly better when the MAS filter has been used.

The best F-score is obtained when frequency range is high i.e. 100-150, 100-200. But here the words are very few in number and are primarily pronoun, determiner or conjunction which are not significant from alignment perspective. Also, it was observed that when medium frequency ranges, such as 30-50, are used the best result, in terms of precision, is around 20-28% for the three corpora. However, since the corpus size is small, here too the available and proposed aligned word pairs are very few (below 25). Lower frequency ranges (viz. 2-20 and its sub-ranges) result in the highest number of aligned pairs. We noticed that these aligned word pairs are typically verb, adjective, noun and adverb. But here too the performance of the algorithm may be considered to be unsatisfactory. Although Somers has recommended words in the frequency ranges 10-30 to be considered for alignment, we have con-

sidered lower frequency words too in our experiments. This is because the corpus size being small we would otherwise have effectively overlooked many small-frequency words (e.g. noun, verb, adjective) that are significant from the alignment point of view.

Somers has further observed that if the Best Alignment Score Selection (BASS) filter is applied to yield the first few best results of alignment the overall quality of the result improves. Figure 1 shows the results of the experiments done for different alignment score cut-off without considering the Frequency Range constraint on the three corpora. However, it was observed that the performance of the algorithm reduced slightly on introducing this BASS filter.

The above experiments suggest that the performance of the two algorithms is rather poor in the context of English-Hindi parallel texts as compared to other language pairs as shown by Fung and Somers. In the following section we discuss the reasons for the low recall and precision values.

2.4 Why Recall and Precision are Low

We observed that the primary reason for the poor performance of the above algorithms in English - Hindi context is the presence of multiple Hindi equivalents for the same English word. This can happen primarily due to three reasons:

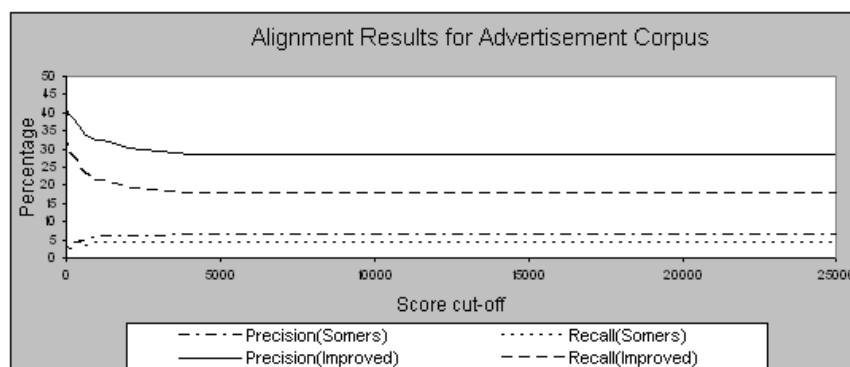


Figure 1: Results of Somers' Algorithm and Improved approach for different score cut-off

Declension of Adjective: Declensions of adjectives are not present in English grammar. No morphological variation in adjectives takes place along with the number and gender of the noun. But, in Hindi, adjectives may have such declensions. For example, the Hindi for "black" is *kaalaa* when the noun is masculine singular number (e.g. *black horse* ~ *kaalaa ghodaa*). But the Hindi translation of "black horses" is *kaale ghode*; whereas "black mare" is translated as *kaalii ghodii*. Thus the same English word "black" may have three Hindi equivalents *kaalaa*, *kaalii*, and *kale* which are to be used judiciously by considering the number and gender of the noun concerned.

Declensions of Pronouns and Nouns: Nouns or pronouns may also have different declensions depending upon the case endings and/or the gender and number of the object. For example, the same English word "my" may have different forms (e.g. *meraa*, *merii*, *mere*) when translated in Hindi. For illustration, while "my book" is translated as ~ *merii kitaab*, the translation of "my name" is *meraa naam*. This happens because *naam* is masculine in Hindi, while *kitaab* is feminine. (Note that in Hindi there is no concept of Neuter gender). Similar declensions may be found with respect to nouns too. For example, the Hindi equivalent of the word "hour" is *ghantaa*. In plural form it becomes *ghante* (e.g. "two hours" ~ *do ghante*). But when used in a prepositional phrase, it becomes *ghanto*. Thus the Hindi translation for "in two hours" is *do ghanto mein*.

Verb Morphology: Morphology of verbs in Hindi depends upon the gender, number and person of the subject. There are 11 possible suffixes (e.g. *taa*, *te*, *tii*, *egaa*) in Hindi that may be at-

tached to the root Verb to render morphological variations. For illustration,

I read. → *main padtaa hoon* (Masculine) but
main padtii hoon (Feminine)

You read. → *tum padte ho* (Masculine) or
tum padtii ho (Feminine)

He will read. → *wah padegaa*.

Due to the presence of multiple Hindi equivalents, the frequencies of word occurrences differ significantly, and thereby jeopardize the calculations. As a consequence, many English words are wrongly aligned.

In the following section we describe certain measures that we propose for improving the efficiency of the recency vector based algorithms for word alignment in English - Hindi parallel texts.

3 Improvements in Word Alignment

In order to take care of morphological variations, we propose to use root words instead of various declensions of the word. For the present work this has been done manually for Hindi. However, algorithms similar to Porter's algorithm may be developed for Hindi too for cleaning a Hindi text of morphological inflections (Ramanathan and Rao, 2003). The modified text, for both English and Hindi, are then subjected to word alignment.

Table 4 gives the details about the root word corpus used to improve the result of word alignment. Here the total number of words for the three types of corpora is greater than the total number of words in the original corpus (Table 1). This is because of the presence of words like "I'll" in the English corpus which have been taken as "I shall" in the root word corpus. Also words like *Unkaa* have been taken as *Un kaa* in the Hindi root word corpus, leading to an increase in the corpus size.

Since we observed (see Section 2.2) that Euclidean distance constraint does not add significantly to the performance, we propose not to use this constraint for English-Hindi word alignment. However, we propose to impose both *frequency range constraint* and *length constraint* (see Section 2.1 and Section 2.3). Instead of the starting point constraint, we have introduced a new constraint, viz. *segment constraint*, to localise the search for the matching words. The starting point constraint expresses range in terms of number of words. However, it has been observed (see section 2.2) that the optimum value of the range varies with the nature of text. Hence no value for range may be identified that applies uniformly on different corpora. Also for noisy corpora the segment constraint is expected to yield better results as the search here is localised better. The proposed segment constraint expresses range in terms of segments. In order to impose this constraint, first the parallel texts are aligned at sentence level. The search for a target language word is then restricted to few segments above and below the current one.

Use of sentimentally aligned corpora for word alignment has already been recommended in (Brown et al., 1993). However, the requirement there is quite stringent – all the sentences are to be correctly aligned. The segment constraint proposed herein works well even if the text alignment is not perfect. Use of roughly aligned corpora has also been proposed in (Dagan and Gale, 1993) for word alignment in bilingual corpora, where statistical techniques have been used as the underlying alignment scheme. In this work, the sentence level alignment algorithm given in (Gale and Church, 1991) has been used for applying segment constraint. As shown in Table 5, the alignment obtained using this algorithm is not very good (only 70% precision for Storybook corpus). The three aligned root word corpora are then subjected to segment constraint in our experiments.

Next important decision we need to take which dynamic programming algorithm should be used. Results shown in Section 2.2 and 2.3 demonstrate that the performance of DK-vec algorithm and Somers' algorithm are almost at par. Hence keeping in view the improved computational complexity, we choose to use Levenshtein distance as used in Somers' algorithm for comparing recency vectors. In the following subsection we discuss the experimental results of the proposed approach.

3.1 Experimental Results and Comparison with Existing algorithms

We have conducted experiments to determine the number of segments above and below the current segment that should be considered for searching the match of a word for each corpus. In this respect we define *i-segment constraint* in which the search is restricted to the segments $k - i$ to $k + i$ of the target language corpus when the word under consideration is in the segment k of the source language corpus.

Evidently, the value of i depends on the accuracy of sentence alignment. Table 5 suggests that the quality of alignment is different for the three corpora that we considered. Due to the very high precision and recall for Sentence corpus we have restricted our search to the k^{th} segment only, i.e. the value of i is 0. However, since the results are not so good for the Storybook and Advertisement corpora we found after experimenting that the best results were obtained when i was 1. During the experiments it was observed that as the number of segments was lowered or increased from the optimum segment the accuracy of alignment decreased continuously by around 10% for low frequency ranges for the three corpora and remained almost same for high frequency ranges.

Table 3 shows the results obtained when segment constraint is applied on the three corpora at optimum segment range for various frequency ranges. A comparison between the F-score given by algorithm in (Somers, 1998) (the column F-score old in the table) and the F-score obtained by applying the improved scheme (the column F-score new in the table) indicate that the results have improved significantly for low frequency ranges.

It is observed that the accuracy of alignment for almost 95% of the available words has increased significantly. This accounts for words within low frequency range of 2–40 for Sentence corpus, 2–30 for Storybook corpus, and 2–20 for Advertisement corpus. Also, most of the correct word pairs given by the modified approach are verbs, adjectives or nouns. Also it was observed that as the noise in the corpus increased the results became poorer. This accounts for the lowest F-score values for advertisement corpus. The Sentence corpus, however, has been found to be the least noisy, and highest precision and recall values were obtained with this corpus.

Using Somers' second filter on each corpus for the optimum segment we found that the results at low scores were better as shown in Figure 1. The word pairs obtained after applying the modified approach can be used as anchor points for further alignment as well as for vocabulary extraction. In case of the Sentence corpus, best result for anchor points for further alignment lies at the score cut off 1000 where precision and recall are 86.88% and 80.35% respectively. Hence F-score is 0.835 which is very high as compared to 0.173 obtained by Somers' approach and indicates an improvement of 382.65%. Also, here the number of correct word pairs is 198, whereas the algorithms in (Fung and McKeown, 1994) and (Somers, 1998) gave only 62 and 61 correct word pairs, respectively. Hence the results are very useful for vocabulary extraction as well. Similarly, Figure 2 and Figure 3 show significant improvements for the other two corpora. At any score cut-off, the modified approach gives better results than the algorithms proposed in (Somers, 1998).

4 Conclusion

This paper focuses on developing suitable word alignment schemes in parallel texts where the size of the corpus is not large. In languages, where rich linguistic tools are yet to be developed, or available freely, such an algorithm may prove to be beneficial for various NLP activities, such as, vocabulary extraction, alignment etc. This work considers word alignment in English - Hindi parallel corpus, where the size of the corpus used is about 18 thousand words for English and 20 thousand words for Hindi.

The paucity of the resources suggests that statistical techniques are not suitable for the task. On the other hand, Lexicon-based approaches are highly resource-dependent. As a consequence, they could not be considered as suitable schemes. Recency vector based approaches provide a suitable alternative. Variations of this approach have already been used for word alignment in parallel texts involving European languages and Chinese, Japanese. However, our initial experiments with these algorithms on English-Hindi did not produce good results. In order to improve their performances certain measures have been taken. The proposed algorithm improved the performance manifold. This approach can be used for word alignment in language pairs like English-Hindi.

Since the available corpus size is rather small we could not compare the results obtained with various other word alignment algorithms proposed in the literature. In particular we like to compare the proposed scheme with the famous IBM models. We hope that with a much larger corpus size we shall be able to make the necessary comparisons in near future.

References

- L. Ahrenberg, M. Merkel, A. Sagvall Hein, and J. Tiedemann. 2000. Evaluation of word alignment systems. In *Proc. 2nd International conference on Linguistic resources and Evaluation (LREC-2000)*, volume 3, pages 1255–1261, Athens, Greece.
- P. 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.
- K. W. Church Dagan, I. and W. A. Gale. 1993. Robust bilingual word alignment for machine aided translation. In *Proc. Workshop on Very Large Corpora: Academic and Industrial Perspectives*, pages 1–8, Columbus, Ohio.
- P. Fung and K. McKeown. 1994. Aligning noisy parallel corpora across language groups: Word pair feature matching by dynamic time warping. In *Technology Partnerships for Crossing the Language Barrier: Proc. First conference of the Association for Machine Translation in the Americas*, pages 81–88, Columbia, Maryland.
- W. A. Gale and K. W. Church. 1991. Identifying word correspondences in parallel texts. In *Proc. Fourth DARPA Workshop on Speech and Natural Language*, pages 152–157. Morgan Kaufmann Publishers, Inc.
- Jin-Xia Huang and Key-Sun Choi. 2000. Chinese Korean word alignment based on linguistic comparison. In *Proc. 38th annual meeting of the association of computational linguistic*, pages 392–399, Hong Kong.
- Ananthakrishnan Ramanathan and Durgesh D. Rao. 2003. A lightweight stemmer for hindi. In *Proc. Workshop of Computational Linguistics for South Asian Languages -Expanding Synergies with Europe, EACL-2003*, pages 42–48, Budapest, Hungary.
- H Somers. 1998. Further experiments in bilingual text alignment. *International Journal of Corpus Linguistics*, 3:115–150.
- Jörg Tiedemann. 2003. Combining clues word alignment. In *Proc. 10th Conference of The European Chapter of the Association for Computational Linguistics*, pages 339–346, Budapest, Hungary.

Segment Constraint: 0-segment (Sentence Corpus)								
Frequency range	a	p	c	P%	R%	F-score (new)	F-score (old)	% increase
2-5	285	181	141	77.90	49.74	0.61	0.118	416.90
3-5	147	108	81	75.00	55.10	0.64	0.169	278.69
3-10	211	152	119	78.29	56.40	0.61	0.168	263.10
5-20	146	103	79	76.70	54.12	0.64	0.216	196.29
10-20	49	35	29	82.86	59.18	0.69	0.233	196.14
20-30	19	12	9	75.00	47.37	0.58	0.270	114.62
30-50	14	8	6	75.00	42.86	0.55	0.229	140.17
40-50	4	2	2	100.00	50.00	0.67	0.222	201.80
50-100	15	12	8	66.67	53.33	0.59	0.392	50.51
100-200	6	5	5	100.00	83.33	0.91	0.91	-
200-300	3	3	3	100.00	100.00	1.00	1.00	-
Segment Constraint: 1-segment (Story book Corpus)								
2-5	281	184	89	48.37	31.67	0.38	0.039	874.35
3-5	143	108	52	48.15	36.36	0.41	0.042	876.19
5-10	125	89	35	39.39	28.00	0.33	0.090	266.67
10-20	75	50	25	50.00	33.33	0.40	0.115	247.83
10-30	117	76	39	51.32	33.33	0.41	0.114	259.65
20-30	32	23	11	47.83	34.38	0.37	0.041	802.43
30-40	14	8	2	25.00	14.29	0.18	0.100	80
40-50	7	7	2	28.57	28.57	0.29	0.200	45.00
50-100	11	10	2	20.00	18.18	0.19	0.110	72.72
100-200	5	5	2	40.00	40.00	0.40	0.444	-
Segment Constraint: 1-segment (Advertisement Corpus)								
2-5	411	250	67	26.80	16.30	0.20	0.035	471.43
3-5	189	145	41	28.28	21.69	0.25	0.073	242.47
3-10	237	172	48	27.91	20.03	0.23	0.075	206.67
5-20	107	73	27	36.99	25.23	0.30	0.141	112.77
10-20	31	22	6	27.27	19.35	0.23	0.229	4.37
10-30	40	28	8	32.14	22.50	0.26	0.247	5.26
30-40	3	2	1	50.00	33.33	0.40	0.222	80.18
30-50	3	2	1	50.00	33.33	0.40	0.222	80.18
50-100	4	3	1	33.33	25.00	0.29	0.178	60.60
100-200	2	2	0	0	0	-	1.000	-

Table 3: Comparison of experimental results with Segment Constraint on the three English-Hindi parallel corpora

Corpora	English corpus		Hindi corpus	
	Total words	Distinct words	Total words	Distinct words
Storybook corpus	6609	895	7606	1100
Advertisement corpus	3795	1213	4057	1198
Sentence corpus	8540	1012	9159	1152

Table 4: Experimental root word parallel corpora of English -Hindi

Different Corpora	Actual alignment in text	Alignment given by system	Correct alignment given by system	R%	P%
Advertisement corpus	323	358	253	78.32	70.68
Storybook corpus	609	546	476	78.16	87.18
Sentence corpus	4548	4548	4458	98.02	98.02

Table 5: Results of Church and Gale Algorithm for Sentence level Alignment

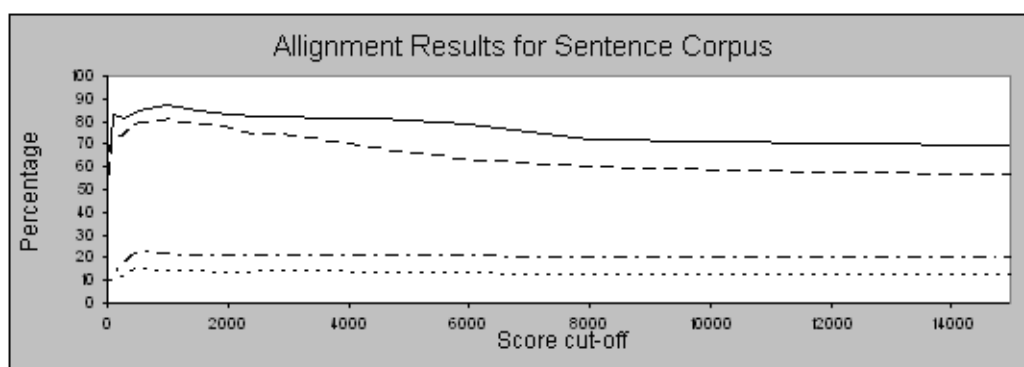


Figure 2: Alignment Results for Sentence Corpus

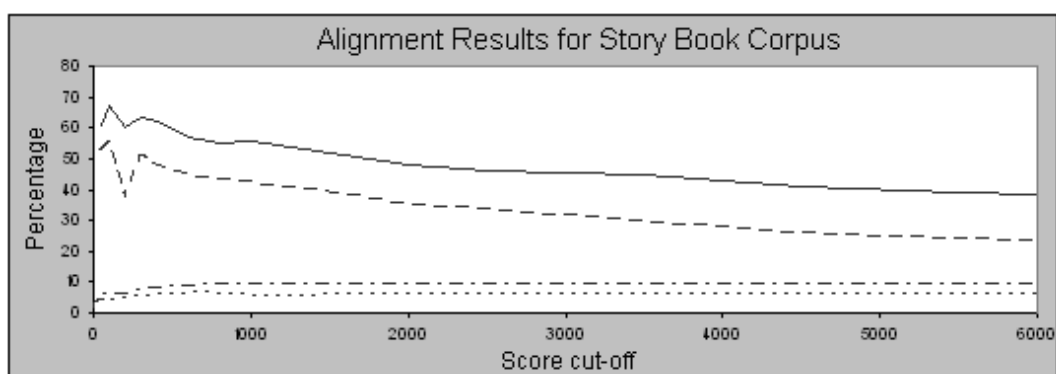


Figure 3: Alignment Results for Story Book Corpus

Extracting loanwords from Mongolian corpora and producing a Japanese-Mongolian bilingual dictionary

Badam-Osor Khaltar
Graduate School of Library,
Information and Media Studies
University of Tsukuba
1-2 Kasuga Tsukuba, 305-8550
Japan
khab23@slis.tsukuba.ac.jp

Atsushi Fujii
Graduate School of Library,
Information and Media Studies
University of Tsukuba
1-2 Kasuga Tsukuba, 305-8550
Japan
fujii@slis.tsukuba.ac.jp

Tetsuya Ishikawa
The Historiographical Institute
The University of Tokyo
3-1 Hongo 7-chome, Bunkyo-ku
Tokyo, 133-0033
Japan
ishikawa@hi.u-tokyo.ac.jp

Abstract

This paper proposes methods for extracting loanwords from Cyrillic Mongolian corpora and producing a Japanese–Mongolian bilingual dictionary. We extract loanwords from Mongolian corpora using our own handcrafted rules. To complement the rule-based extraction, we also extract words in Mongolian corpora that are phonetically similar to Japanese Katakana words as loanwords. In addition, we correspond the extracted loanwords to Japanese words and produce a bilingual dictionary. We propose a stemming method for Mongolian to extract loanwords correctly. We verify the effectiveness of our methods experimentally.

1 Introduction

Reflecting the rapid growth in science and technology, new words and technical terms are being progressively created, and these words and terms are often transliterated when imported as loanwords in another language.

Loanwords are often not included in dictionaries, and decrease the quality of natural language processing, information retrieval, machine translation, and speech recognition. At the same time, compiling dictionaries is expensive, because it relies on human introspection and supervision. Thus, a number of automatic methods have been proposed to extract loanwords and their translations from corpora,

targeting various languages.

In this paper, we focus on extracting loanwords in Mongolian. The Mongolian language is divided into Traditional Mongolian, written using the Mongolian alphabet, and Modern Mongolian, written using the Cyrillic alphabet. We focused solely on Modern Mongolian, and use the word “Mongolian” to refer to Modern Mongolian in this paper.

There are two major problems in extracting loanwords from Mongolian corpora.

The first problem is that Mongolian uses the Cyrillic alphabet to represent both conventional words and loanwords, and so the automatic extraction of loanwords is difficult. This feature provides a salient contrast to Japanese, where the Katakana alphabet is mainly used for loanwords and proper nouns, but not used for conventional words.

The second problem is that content words, such as nouns and verbs, are inflected in sentences in Mongolian. Each sentence in Mongolian is segmented on a phrase-by-phrase basis. A phrase consists of a content word and one or more suffixes, such as postpositional particles. Because loanwords are content words, then to extract loanwords correctly, we have to identify the original form using stemming.

In this paper, we propose methods for extracting loanwords from Cyrillic Mongolian and producing a Japanese–Mongolian bilingual dictionary. We also propose a stemming method to identify the original forms of content words in Mongolian phrases.

2 Related work

To the best of our knowledge, no attempt has been made to extract loanwords and their translations targeting Mongolian. Thus, we will discuss existing methods targeting other languages.

In Korean, both loanwords and conventional words are spelled out using the Korean alphabet, called *Hangul*. Thus, the automatic extraction of loanwords in Korean is difficult, as it is in Mongolian. Existing methods that are used to extract loanwords from Korean corpora (Myaeng and Jeong, 1999; Oh and Choi, 2001) use the phonetic differences between conventional Korean words and loanwords. However, these methods require manually tagged training corpora, and are expensive.

A number of corpus-based methods are used to extract bilingual lexicons (Fung and McKeown, 1996; Smadja, 1996). These methods use statistics obtained from a parallel or comparable bilingual corpus, and extract word or phrase pairs that are strongly associated with each other. However, these methods cannot be applied to a language pair where a large parallel or comparable corpus is not available, such as Mongolian and Japanese.

Fujii et al. (2004) proposed a method that does not require tagged corpora or parallel corpora to extract loanwords and their translations. They used a monolingual corpus in Korean and a dictionary consisting of Japanese Katakana words. They assumed that loanwords in multiple countries corresponding to the same source word are phonetically similar. For example, the English word “system” has been imported into Korean, Mongolian, and Japanese. In these languages, the romanized words are “siseutem”, “sistem”, and “shisutemu”, respectively.

It is often the case that new terms have been imported into multiple languages simultaneously, because the source words are usually influential across cultures. It is feasible that a large number of loanwords in Korean can also be loanwords in Japanese. Additionally, Katakana words can be extracted from Japanese corpora with a high accuracy. Thus, Fujii et al. (2004) extracted the loanwords in Korean corpora that were phonetically similar to Japanese Katakana words. Because each

of the extracted loanwords also corresponded to a Japanese word during the extraction process, a Japanese–Korean bilingual dictionary was produced in a single framework.

However, a number of open questions remain from Fujii et al.’s research. First, their stemming method can only be used for Korean. Second, their accuracy in extracting loanwords was low, and thus, an additional extraction method was required. Third, they did not report on the accuracy of extracting translations, and finally, because they used Dynamic Programming (DP) matching for computing the phonetic similarities between Korean and Japanese words, the computational cost was prohibitive.

In an attempt to extract Chinese–English translations from corpora, Lam et al. (2004) proposed a similar method to Fujii et al. (2004). However, they searched the Web for Chinese–English bilingual comparable corpora, and matched named entities in each language corpus if they were similar to each other. Thus, Lam et al.’s method cannot be used for a language pair where comparable corpora do not exist. In contrast, using Fujii et al.’s (2004) method, the Katakana dictionary and a Korean corpus can be independent.

In addition, Lam et al.’s method requires Chinese–English named entity pairs to train the similarity computation. Because the accuracy of extracting named entities was not reported, it is not clear to what extent this method is effective in extracting loanwords from corpora.

3 Methodology

3.1 Overview

In view of the discussion outlined in Section 2, we enhanced the method proposed by Fujii et al. (2004) for our purpose. Figure 1 shows the method that we used to extract loanwords from a Mongolian corpus and to produce a Japanese–Mongolian bilingual dictionary. Although the basis of our method is similar to that used by Fujii et al. (2004), “Stemming”, “Extracting loanwords based on rules”, and “N-gram retrieval” are introduced in this paper.

First, we perform stemming on a Mongolian corpus to segment phrases into a content word and one or more suffixes.

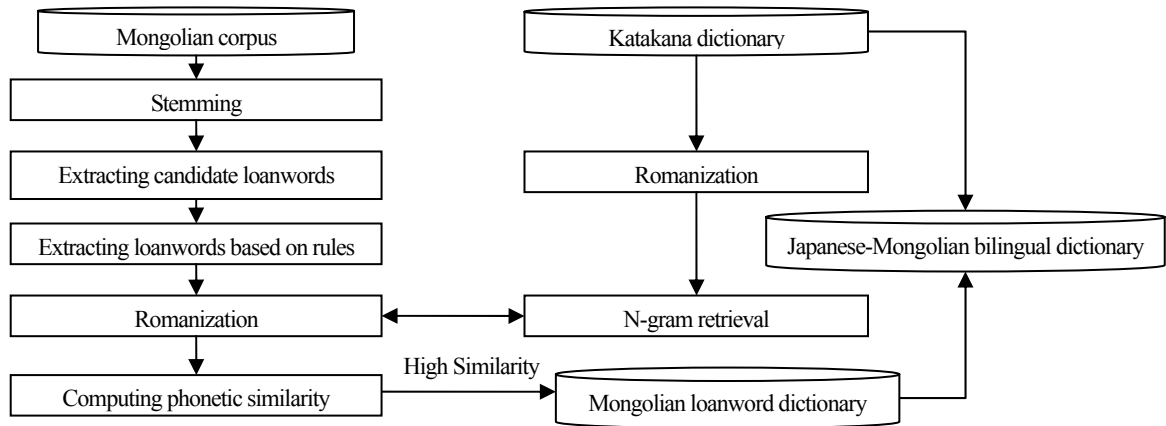


Figure 1: Overview of our extraction method.

Second, we discard segmented content words if they are in an existing dictionary, and extract the remaining words as candidate loanwords.

Third, we use our own handcrafted rules to extract loanwords from the candidate loanwords. While the rule-based method can extract loanwords with a high accuracy, a number of loanwords cannot be extracted using predefined rules.

Fourth, as performed by Fujii et al. (2004), we use a Japanese Katakana dictionary and extract a candidate loanword that is phonetically similar to a Katakana word as a loanword. We romanize the candidate loanwords that were not extracted using the rules. We also romanize all words in the Katakana dictionary.

However, unlike Fujii et al. (2004), we use N-gram retrieval to limit the number of Katakana words that are similar to the candidate loanwords. Then, we compute the phonetic similarities between each candidate loanword and each retrieved Katakana word using DP matching, and select a pair whose score is above a predefined threshold. As a result, we can extract loanwords in Mongolian and their translations in Japanese simultaneously.

Finally, to identify Japanese translations for the loanwords extracted using the rules defined in the third step above, we perform N-gram retrieval and DP matching.

We will elaborate further on each step in Sections 3.2–3.7.

3.2 Stemming

A phrase in Mongolian consists of a content word and one or more suffixes. A content word can potentially be inflected in a phrase. Figure 2 shows

Type	Example
(a) No inflection.	ном + ын → номын Book + Genitive Case
(b) Vowel elimination.	ажил + аас + аа → ажлаасаа Work + Ablative Case + Reflexive
(c) Vowel insertion.	ах + д → ахад Brother + Dative Case
(d) Consonant insertion.	байшин + ийн → байшингийн Building + Genitive Case
(e) The letter “ь” is converted to “и”, and the vowel is eliminated.	сургууль + аас → сургуулиас School + Ablative Case

Figure 2: Inflection types of nouns in Mongolian.

the inflection types of content words in phrases. In phrase (a), there is *no inflection* in the content word “ном (book)” concatenated with the suffix “ын (genitive case)”.

However, in phrases (b)–(e) in Figure 2, the content words are inflected. Loanwords are also inflected in all of these types, except for phrase (b). Thus, we have to identify the original form of a content word using stemming. While most loanwords are nouns, a number of loanwords can also be verbs. In this paper, we propose a stemming method for nouns. Figure 3 shows our stemming method. We will explain our stemming method further, based on Figure 3.

First, we consult a “Suffix dictionary” and perform backward partial matching to determine whether or not one or more suffixes are concatenated at the end of a target phrase.

Second, if a suffix is detected, we use a “Suffix segmentation rule” to segment the suffix and extract

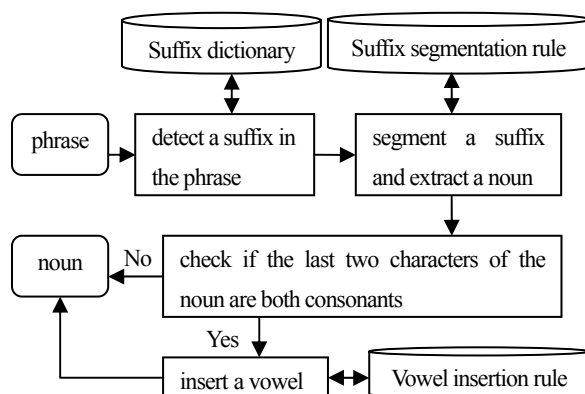


Figure 3: Overview of our noun stemming method.

the noun. The inflection type in phrases (c)–(e) in Figure 2 is also determined.

Third, we investigate whether or not the *vowel elimination* in phrase (b) in Figure 2 occurred in the extracted noun. Because the vowel elimination occurs only in the last vowel of a noun, we check the last two characters of the extracted noun. If both of the characters are consonants, the eliminated vowel is inserted using a “Vowel insertion rule” and the noun is converted into its original form.

Existing Mongolian stemming methods (Ehara et al., 2004; Sanduijav et al., 2005) use noun dictionaries. Because we intend to extract loanwords that are not in existing dictionaries, the above methods cannot be used. Noun dictionaries have to be updated as new words are created.

Our stemming method does not require a noun dictionary. Instead, we manually produced a suffix dictionary, suffix segmentation rule, and vowel insertion rule. However, once these resources are produced, almost no further compilation is required.

The suffix dictionary consists of 37 suffixes that can concatenate with nouns. These suffixes are postpositional particles. Table 1 shows the dictionary entries, in which the inflection forms of the postpositional particles are shown in parentheses.

The suffix segmentation rule consists of 173 rules. We show examples of these rules in Figure 4. Even if suffixes are identical in their phrases, the segmentation rules can be different, depending on the counterpart noun.

In Figure 4, the suffix “ийн” matches both the noun phrases (a) and (b) by backward partial matching. However, each phrase is segmented by a

Table 1: Entries of the suffix dictionary.

Case	Suffix
Genitive	н, ы, ын, ны, ий, ийн, ний
Accusative	ыг, ийг, г
Dative	д, т
Ablative	аас (иас), оос (иос), ээс, өөс
Instrumental	аар (иар), оор (иор), ээр, өөр
Cooperative	тай, той, тэй
Reflexive	аа (иа), оо (ио), ээ, өө
Plural	ууд (иуд), үүд (иүд)

Suffix	Noun phrase	Noun
ийн Genitive	(a) <u>Ээжийн</u> mother’s	Ээж mother
	(b) <u>Хараагийн</u> Haraa’(river name)s	Хараа Haraa

Figure 4: Examples of the suffix segmentation rule.

deferent rule independently. The underlined suffixes are segmented in each phrase, respectively. In phrase (a), there is *no inflection*, and the suffix is easily segmented. However, in phrase (b), a *consonant insertion* has occurred. Thus, both the inserted consonant, “г”, and the suffix have to be removed.

The vowel insertion rule consists of 12 rules. To insert an eliminated vowel and extract the original form of the noun, we check the last two characters of a target noun. If both of these are consonants, we determine that a vowel was eliminated.

However, a number of nouns end with two consonants inherently, and therefore, we referred to a textbook on Mongolian grammar (Bayarmaa, 2002) to produce 12 rules to determine when to insert a vowel between two consecutive consonants.

For example, if any of “м”, “г”, “л”, “б”, “в”, or “п” are at the end of a noun, a vowel is inserted. However, if any of “ц”, “ж”, “з”, “с”, “д”, “т”, “ш”, “ч”, or “х” are the second to last consonant in a noun, a vowel is not inserted.

The Mongolian vowel harmony rule is a phonological rule in which female vowels and male vowels are prohibited from occurring in a single word together (with the exception of proper nouns). We used this rule to determine which vowel should be inserted. The appropriate vowel is determined by the first vowel of the first syllable in the target noun.

For example, if there are “a” and “y” in the first syllable, the vowel “a” is inserted between the last two consonants.

3.3 Extracting candidate loanwords

After collecting nouns using our stemming method, we discard the conventional Mongolian nouns. We discard nouns defined in a noun dictionary (Sanduijav et al., 2005), which includes 1,926 nouns. We also discard proper nouns and abbreviations. The first characters of proper nouns, such as “Эрдэнэбат (Erdenebat)”, and all the characters of abbreviations, such as “ЦИИИИ (Nuclear research centre)”, are written using capital letters in Mongolian. Thus, we discard words that are written using capital characters, except those occurring at the beginning of sentences. In addition, because “ө” and “ү” are not used to spell out Western languages, words including those characters are also discarded.

3.4 Extracting loanwords based on rules

We manually produced seven rules to identify loanwords in Mongolian. Words that match with one of the following rules are extracted as loanwords.

- (a) A word including the consonants “к”, “п”, “ф”, or “ш”.

These consonants are usually used to spell out foreign words.

- (b) A word that violated the Mongolian vowel harmony rule.

Because of the vowel harmony rule, a word that includes female and male vowels, which is not based on the Mongolian phonetic system, is probably a loanword.

- (c) A word beginning with two consonants.

A conventional Mongolian word does not begin with two consonants.

- (d) A word ending with two particular consonants.

A word whose penultimate character is any of: “п”, “б”, “т”, “ц”, “ч”, “з”, or “ш” and whose last character is a consonant violates Mongolian grammar, and is probably a loanword.

- (e) A word beginning with the consonant “b”.

In a modern Mongolian dictionary (Ozawa, 2000), there are 54 words beginning with “b”, of which 31 are loanwords. Therefore, a word beginning with “b” is probably a loanword.

- (f) A word beginning with the consonant “p”.

In a modern Mongolian dictionary (Ozawa, 2000), there are 49 words beginning with “p”, of which only four words are conventional Mongolian words. Therefore, a word beginning with “p” is probably a loanword.

- (g) A word ending with “<consonant> + и”.

We discovered this rule empirically.

3.5 Romanization

We manually aligned each Mongolian Cyrillic alphabet to its Roman representation¹.

In Japanese, the Hepburn and *Kunrei* systems are commonly used for romanization proposes. We used the Hepburn system, because its representation is similar to that used in Mongolian, compared to the *Kunrei* system.

However, we adapted 11 Mongolian romanization expressions to the Japanese Hepburn romanization. For example, the sound of the letter “L” does not exist in Japanese, and thus, we converted “L” to “R” in Mongolian.

3.6 N-gram retrieval

By using a document retrieval method, we efficiently identify Katakana words that are phonetically similar to a candidate loanword. In other words, we use a candidate loanword, and each Katakana word as a query and a document, respectively. We call this method “N-gram retrieval”.

Because the N-gram retrieval method does not consider the order of the characters in a target word, the accuracy of matching two words is low, but the computation time is fast. On the other hand, because DP matching considers the order of the characters in a target word, the accuracy of matching two words is high, but the computation time is slow. We combined these two methods to achieve a high matching accuracy with a reasonable computation time.

First, we extract Katakana words that are phonetically similar to a candidate loanword using N-gram retrieval. Second, we compute the similarity between the candidate loanword and each of the retrieved Katakana words using DP matching to improve the accuracy.

We romanize all the Katakana words in the dictionary and index them using consecutive N

¹ http://badaa.mngl.net/docs.php?p=trans_table (May, 2006)

characters. We also romanize each candidate loanword when use as a query. We experimentally set $N = 2$, and use the Okapi BM25 (Robertson et al., 1995) for the retrieval model.

3.7 Computing phonetic similarity

Given the romanized Katakana words and the romanized candidate loanwords, we compute the similarity between the two strings, and select the pairs associated with a score above a predefined threshold as translations. We use DP matching to identify the number of differences (i.e., insertion, deletion, and substitution) between two strings on an alphabet-by-alphabet basis.

While consonants in transliteration are usually the same across languages, vowels can vary depending on the language. The difference in consonants between two strings should be penalized more than the difference in vowels. We compute the similarity between two romanized words using Equation (1).

$$1 - \frac{2 \times (\alpha \times dc + dv)}{\alpha \times c + v} \quad (1)$$

Here, dc and dv denote the number of differences in consonants and vowels, respectively, and α is a parametric consonant used to control the importance of the consonants. We experimentally set $\alpha = 2$. Additionally, c and v denote the number of all the consonants and vowels in the two strings, respectively. The similarity ranges from 0 to 1.

4 Experiments

4.1 Method

We collected 1,118 technical reports published in Mongolian from the “Mongolian IT Park”² and used them as a Mongolian corpus. The number of phrase types and phrase tokens in our corpus were 110,458 and 263,512, respectively.

We collected 111,116 Katakana words from multiple Japanese dictionaries, most of which were technical term dictionaries.

We evaluated our method from four perspectives: “stemming”, “loanword extraction”, “translation extraction”, and “computational cost.” We will discuss these further in Sections 4.2–4.5, respectively.

4.2 Evaluating stemming

We randomly selected 50 Mongolian technical

reports from our corpus, and used them to evaluate the accuracy of our stemming method. These technical reports were related to: *medical science* (17), *geology* (10), *light industry* (14), *agriculture* (6), and *sociology* (3). In these 50 reports, the number of phrase types including conventional Mongolian nouns and loanword nouns was 961 and 206, respectively. We also found six phrases including loanword verbs, which were not used in the evaluation.

Table 2 shows the results of our stemming experiment, in which the accuracy for conventional Mongolian nouns was 98.7% and the accuracy for loanwords was 94.6%. Our stemming method is practical, and can also be used for morphological analysis of Mongolian corpora.

We analyzed the reasons for any failures, and found that for 12 conventional nouns and 11 loanwords, the suffixes were incorrectly segmented.

4.3 Evaluating loanword extraction

We used our stemming method on our corpus and selected the most frequently used 1,300 words. We used these words to evaluate the accuracy of our loanword extraction method. Of these 1,300 words, 165 were loanwords. We varied the threshold for the similarity, and investigated the relationship between precision and recall. Recall is the ratio of the number of correct loanwords extracted by our method to the total number of correct loanwords. Precision is the ratio of the number of correct loanwords extracted by our method to the total number of words extracted by our method. We extracted loanwords using rules (a)–(g) defined in Section 3.4. As a result, 139 words were extracted.

Table 3 shows the precision and recall of each rule. The precision and recall showed high values using “All rules”, which combined the words extracted by rules (a)–(g) independently.

We also extracted loanwords using the phonetic similarity, as discussed in Sections 3.6 and 3.7.

Table 2: Results of our noun stemming method.

	No. of each phrase type	Accuracy (%)
Conventional nouns	961	98.7
Loanwords	206	94.6

² <http://www.itpark.mn/> (May, 2006)

Table 3: Precision and recall for rule-based loanword extraction.

Rules	(a)	(b)	(c)	(d)	(e)	(f)	(g)	All rules
Words extracted automatically	102	63	21	6	4	5	24	150
Extracted correct loanwords	101	60	20	5	4	5	19	139
Precision (%)	99.0	95.2	95.2	83.3	100	100	79.2	92.7
Recall (%)	61.2	36.4	12.1	3.0	2.4	3.03	11.5	84.2

We used the N-gram retrieval method to obtain up to the top 500 Katakana words that were similar to each candidate loanword. Then, we selected up to the top five pairs of a loanword and a Katakana word whose similarity computed using Equation (1) was greater than 0.6. Table 4 shows the results of our similarity-based extraction.

Both the precision and the recall for the similarity-based loanword extraction were lower than those for the “All rules” data listed in Table 3.

Table 4: Precision and recall for our similarity-based loanword extraction.

Words extracted automatically	Extracted correct loanwords	Precision (%)	Recall (%)
3,479	109	3.1	66.1

We also evaluated the effectiveness of a combination of the N-gram and DP matching methods. We performed similarity-based extraction after rule-based extraction. Table 5 shows the results, in which the data of the “Rule” are identical to those of the “All rules” data listed in Table 3. However, the “Similarity” data are not identical to those listed in Table 4, because we performed similarity-based extraction using only the words that were not extracted by rule-based extraction.

When we combined the rule-based and similarity-based methods, the recall improved from 84.2% to 91.5%. The recall value should be high when a human expert modifies or verifies the resultant dictionary.

Figure 5 shows example of extracted loanwords in Mongolian and their English glosses.

4.4 Evaluating Translation extraction

In the row “Both” shown in Table 5, 151 loanwords were extracted, for each of which we selected up to the top five Katakana words whose similarity computed using Equation (1) was greater than 0.6 as

Table 5: Precision and recall of different loanword extraction methods.

	No. of words	No. that were correct	Precision (%)	Recall (%)
Rule	150	139	92.7	84.2
Similarity	60	12	20.0	46.2
Both	210	151	71.2	91.5

Mongolian	English gloss
альбумин	albumin
лаборатор	laboratory
механизм	mechanism
МИТОХОНДР	mitochondria

Figure 5: Example of extracted loanwords.

translations. As a result, Japanese translations were extracted for 109 loanwords. Table 6 shows the results, in which the precision and recall of extracting Japanese–Mongolian translations were 56.2% and 72.2%, respectively.

We analyzed the data and identified the reasons for any failures. For five loanwords, the N-gram retrieval failed to search for the similar Katakana words. For three loanwords, the phonetic similarity computed using Equation (1) was not high enough for a correct translation. For 27 loanwords, the Japanese translations did not exist inherently. For seven loanwords, the Japanese translations existed, but were not included in our Katakana dictionary.

Figure 6 shows the Japanese translations extracted for the loanwords shown in Figure 5.

Table 6: Precision and recall for translation extraction.

No. of translations extracted automatically	No. of extracted correct translations	Precision (%)	Recall (%)
194	109	56.2	72.2

Japanese	Mongolian	English gloss
アルブミン	альбумин	albumin
ラボラトリー	лаборатор	laboratory
メカニズム	механизм	mechanism
ミトコンドリア	митохондър	mitochondria

Figure 6: Japanese translations extracted for the loanwords shown in Figure 5.

4.5 Evaluating computational cost

We randomly selected 100 loanwords from our corpus, and used them to evaluate the computational cost of the different extraction methods. We compared the computation time and the accuracy of “N-gram”, “DP matching”, and “N-gram + DP matching” methods. The experiments were performed using the same PC (CPU = Pentium III 1 GHz dual, Memory = 2 GB).

Table 7 shows the improvement in computation time by “N-gram + DP matching” on “DP matching”, and the average rank of the correct translations for “N-gram”. We improved the efficiency, while maintaining the sorting accuracy of the translations.

Table 7: Evaluation of the computational cost.

Method	N-gram	DP	N-gram + DP
Loanwords	100		
Computation time (sec.)	95	136,815	293
Extracted correct translations	66	66	66
Average rank of correct translations	44.8	2.7	2.7

5 Conclusion

We proposed methods for extracting loanwords from Cyrillic Mongolian corpora and producing a Japanese–Mongolian bilingual dictionary. Our research is the first serious effort in producing dictionaries of loanwords and their translations targeting Mongolian. We devised our own rules to extract loanwords from Mongolian corpora. We also extracted words in Mongolian corpora that are phonetically similar to Japanese Katakana words as loanwords. We also corresponded the extracted loanwords to Japanese words, and produced a Japanese–Mongolian bilingual dictionary. A noun stemming method that does not require noun

dictionaries was also proposed. Finally, we evaluated the effectiveness of the components experimentally.

References

- Terumasa Ehara, Suzushi Hayata, and Nobuyuki Kimura. 2004. Mongolian morphological analysis using ChaSen. *Proceedings of the 10th Annual Meeting of the Association for Natural Language Processing*, pp. 709-712. (In Japanese).
- Atsushi Fujii, Tetsuya Ishikawa, and Jong-Hyeok Lee. 2004. Term extraction from Korean corpora via Japanese. *Proceedings of the 3rd International Workshop on Computational Terminology*, pp. 71-74.
- Pascal Fung and Kathleen McKeown. 1996. Finding terminology translations from non-parallel corpora. *Proceedings of the 5th Annual Workshop on Very Large Corpora*, pp. 53-87.
- Wai Lam, Ruizhang Huang, and Pik-Shan Cheung. 2004. Learning phonetic similarity for matching named entity translations and mining new translations. *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 289-296.
- Sung Hyun Myaeng and Kil-Soon Jeong. 1999. Back-Transliteration of foreign words for information retrieval. *Information Processing and Management*, Vol. 35, No. 4, pp. 523 -540.
- Jong-Hooh Oh and Key-Sun Choi. 2001. Automatic extraction of transliterated foreign words using hidden markov model. *Proceedings of the International Conference on Computer Processing of Oriental Languages*, 2001, pp. 433-438.
- Shigeo Ozawa. Modern Mongolian Dictionary. Daigakushorin. 2000.
- Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1995. Okapi at TREC-3, *Proceedings of the Third Text REtrieval Conference (TREC-3)*, NIST Special Publication 500-226. pp. 109-126.
- Enkhbayar Sanduijav, Takehito Utsuro, and Satoshi Sato. 2005. Mongolian phrase generation and morphological analysis based on phonological and morphological constraints. *Journal of Natural Language Processing*, Vol. 12, No. 5, pp. 185-205. (In Japanese).
- Frank Smadja, Vasileios Hatzivassiloglou, Kathleen R. McKeown. 1996. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, Vol. 22, No. 1, pp. 1-38.
- Bayarmaa Ts. 2002. Mongolian grammar in I-IV grades. (In Mongolian).

An Unsupervised Morpheme-Based HMM for Hebrew Morphological Disambiguation

Meni Adler

Department of Computer Science
Ben Gurion University of the Negev
84105 Beer Sheva, Israel
adlerm@cs.bgu.ac.il

Michael Elhadad

Department of Computer Science
Ben Gurion University of the Negev
84105 Beer Sheva, Israel
elhadad@cs.bgu.ac.il

Abstract

Morphological disambiguation is the process of assigning one set of morphological features to each individual word in a text. When the word is ambiguous (there are several possible analyses for the word), a disambiguation procedure based on the word context must be applied. This paper deals with morphological disambiguation of the Hebrew language, which combines morphemes into a word in both agglutinative and fusional ways. We present an unsupervised stochastic model – the only resource we use is a morphological analyzer – which deals with the data sparseness problem caused by the affixational morphology of the Hebrew language.

We present a text encoding method for languages with affixational morphology in which the knowledge of word formation rules (which are quite restricted in Hebrew) helps in the disambiguation. We adapt HMM algorithms for learning and searching this text representation, in such a way that segmentation and tagging can be learned in parallel in one step. Results on a large scale evaluation indicate that this learning improves disambiguation for complex tag sets. Our method is applicable to other languages with affix morphology.

1 Introduction

Morphological disambiguation is the process of assigning one set of morphological features to each individual word in a text, according to the word context.

In this work, we investigate morphological disambiguation in Modern Hebrew. We explore unsupervised learning method, which is more challenging than the supervised case. The main motivation for this approach is that despite the development

of annotated corpora in Hebrew¹, there is still not enough data available for supervised training. The other reason, is that unsupervised methods can handle the dynamic nature of Modern Hebrew, as it evolves over time.

In the case of English, because morphology is simpler, morphological disambiguation is generally covered under the task of part-of-speech tagging. The main morphological variations are embedded in the tag name (for example, Ns and Np for noun singular or plural). The tagging accuracy of supervised stochastic taggers is around 96%-97% (Manning and Schütze, 1999, 10.6.1). Meraldo (1994) reports an accuracy of 86.6% for an unsupervised word-based HMM, trained on a corpus of 42,186 sentences (about 1M words), over a tag set of 159 different tags. Elworthy (1994), in contrast, reports an accuracy of 75.49%, 80.87% and 79.12% for unsupervised word-based HMM trained on parts of the LOB corpora, with a tagset of 134 tags. With good initial conditions, such as good approximation of the tag distribution for each word, Elworthy reports an improvement to 94.6%, 92.27% and 94.51% on the same data sets. Meraldo, on the other hand, reports an improvement to 92.6% and 94.4% for the case where 100 and 2000 sentences of the training corpus are manually tagged.

Modern Hebrew is characterized by rich morphology, with a high level of ambiguity. On average, in our corpus, the number of possible analyses per word reached 2.4 (in contrast to 1.4 for English). In Hebrew, several morphemes combine into a single word in both agglutinative and fusional ways. This results in a potentially high number of tags for each word.

In contrast to English tag sets whose sizes range from 48 to 195, the number of tags for Hebrew, based on all combinations of the morphological attributes (part-of-speech, gender, number, person, tense, status, and the affixes' properties²),

*This work is supported by the Lynn and William Frankel Center for Computer Sciences, and by the Knowledge Center for Hebrew Processing, Israel Science Ministry.

¹The Knowledge Center for Hebrew processing is developing such corpora: <http://mila.cs.technion.ac.il/>

²The list of morphological attributes is described in (Yona and Wintner, 2005). An in-depth discussion of the Hebrew word form is provided in (Allon, 1995, pp.

can grow theoretically to about 300,000 tags. In practice, we found only 1,934 tags in a corpus of news stories we gathered, which contains about 6M words.

The large size of such a tag set (about 10 times larger than the most comprehensive English tag set) is problematic in term of data sparseness. Each morphological combination appears rarely, and more samples are required in order to learn the probabilistic model.

In this paper, we hypothesize that the large set of morphological features of Hebrew words, should be modeled by a compact morpheme model, based on the segmented words (into prefix, baseform, and suffix). Our main result is that best performance is obtained when learning segmentation and morpheme tagging in one step, which is made possible by an appropriate text representation.

2 Hebrew and Arabic Tagging - Previous Work

Several works have dealt with Hebrew tagging in the past decade. In Hebrew, morphological analysis requires complex processing according to the rules of Hebrew word formation. The task of a morphological analyzer is to produce all possible analyses for a given word. Recent analyzers provide good performance and documentation of this process (Yona and Wintner, 2005; Segal, 2000). Morphological analyzers rely on a dictionary, and their performance is, therefore, impacted by the occurrence of unknown words. The task of a morphological disambiguation system is to pick the most likely analysis produced by an analyzer in the context of a full sentence.

Levinger et al. (1995) developed a context-free method in order to acquire the morpho-lexical probabilities, from an untagged corpus. Their method handles the data sparseness problem by using a set of similar words for each word, built according to a set of rules. The rules produce variations of the morphological properties of the word analyses. Their tests indicate an accuracy of about 88% for context-free analysis selection based on the approximated analysis distribution. In tests we reproduced on a larger data set (30K tagged words), the accuracy is only 78.2%. In order to improve the results, the authors recommend merging their method together with other morphological disambiguation methods – which is the approach we pursue in this work.

Levinger's morphological disambiguation system (Levinger, 1992) combines the above approximated probabilities with an expert system, based on a manual set of 16 syntactic constraints. In the first phase, the expert system is applied, dis-

ambiguating 35% of the ambiguous words with an accuracy of 99.6%. In order to increase the applicability of the disambiguation, approximated probabilities are used for words that were not disambiguated in the first stage. Finally, the expert system is used again over the new probabilities that were set in the previous stage. Levinger reports an accuracy of about 94% for disambiguation of 85% of the words in the text (overall 80% disambiguation). The system was also applied to prune out the least likely analyses in a corpus but without, necessarily, selecting a single analysis for each word. For this task, an accuracy of 94% was reported while reducing 92% of the ambiguous analyses.

Carmel and Maarek (1999) use the fact that on average 45% of the Hebrew words are unambiguous, to rank analyses, based on the number of disambiguated occurrences in the text, normalized by the total number of occurrences for each word. Their application – indexing for an information retrieval system – does not require all of the morphological attributes but only the lemma and the PoS of each word. As a result, for this case, 75% of the words remain with one analysis with 95% accuracy, 20% with two analyses and 5% with three analyses.

Segal (2000) built a transformation-based tagger in the spirit of Brill (1995). In the first phase, the analyses of each word are ranked according to the frequencies of the possible lemmas and tags in a training corpus of about 5,000 words. Selection of the highest ranked analysis for each word gives an accuracy of 83% of the test text – which consists of about 1,000 words. In the second stage, a transformation learning algorithm is applied (in contrast to Brill, the observed transformations are not applied, but used for re-estimation of the word couples probabilities). After this stage, the accuracy is about 93%. The last stage uses a bottom-up parser over a hand-crafted grammar with 150 rules, in order to select the analysis which causes the parsing to be more accurate. Segal reports an accuracy of 95%. Testing his system over a larger test corpus, gives poorer results: Lembersky (2001) reports an accuracy of about 85%.

Bar-Haim et al. (2005) developed a word segmenter and PoS tagger for Hebrew. In their architecture, words are first segmented into morphemes, and then, as a second stage, these morphemes are tagged with PoS. The method proceeds in two sequential steps: segmentation into morphemes, then tagging over morphemes. The segmentation is based on an HMM and trained over a set of 30K annotated words. The segmentation step reaches an accuracy of 96.74%. PoS tagging, based on unsupervised estimation which combines a small annotated corpus with an untagged corpus of 340K

Word	Segmentation	Tag	Translation
<i>bclm</i>	<i>bclm</i>	PNN	name of a human rights association (Betsalem)
<i>bclm</i>	<i>bclm</i>	VB	while taking a picture
<i>bclm</i>	<i>bcl-m</i>	cons-NNM-suf	their onion
<i>bclm</i>	<i>b-cl-m</i>	P1-NNM-suf	under their shadow
<i>bclm</i>	<i>b-clm</i>	P1-NNM	in a photographer
<i>bclm</i>	<i>b-clm</i>	P1-cons-NNM	in a photographer
<i>bclm</i>	<i>b-clm</i>	P1-h-NNM	in the photographer
<i>hn'im</i>	<i>h-n'im</i>	P1-VBR	that are moving
<i>hn'im</i>	<i>hn'im</i>	P1-h-JJM	the lovely
<i>hn'im</i>	<i>hn'im</i>	VBP	made pleasant

Table 1: Possible analyses for the words *bclm hn'im*

words by using smoothing technique, gives an accuracy of 90.51%.

As noted earlier, there is as yet no large scale Hebrew annotated corpus. We are in the process of developing such a corpus, and we have developed tagging guidelines (Elhadad et al., 2005) to define a comprehensive tag set, and assist human taggers achieve high agreement. The results discussed above should be taken as rough approximations of the real performance of the systems, until they can be re-evaluated on such a large scale corpus with a standard tag set.

Arabic is a language with morphology quite similar to Hebrew. Theoretically, there might be 330,000 possible morphological tags, but in practice, Habash and Rambow (2005) extracted 2,200 different tags from their corpus, with an average number of 2 possible tags per word. As reported by Habash and Rambow, the first work on Arabic tagging which used a corpus for training and evaluation was the work of Diab et al. (2004). Habash and Rambow were the first to use a morphological analyzer as part of their tagger. They developed a supervised morphological disambiguator, based on training corpora of two sets of 120K words, which combines several classifiers of individual morphological features. The accuracy of their analyzer is 94.8% – 96.2% (depending on the test corpus). An unsupervised HMM model for dialectal Arabic (which is harder to be tagged than written Arabic), with accuracy of 69.83%, was presented by Duh and Kirchhoff (2005). Their supervised model, trained on a manually annotated corpus, reached an accuracy of 92.53%.

Arabic morphology seems to be similar to Hebrew morphology, in term of complexity and data sparseness, but comparison of the performances of the baseline tagger used by Habash and Rambow – which selects the most frequent tag for a given word in the training corpus – for Hebrew and Arabic, shows some intriguing differences: 92.53% for Arabic and 71.85% for Hebrew. Furthermore, as mentioned above, even the use of a sophisticated context-free tagger, based on (Levinger et al., 1995), gives low accuracy of 78.2%. This might

imply that, despite the similarities, morphological disambiguation in Hebrew might be harder than in Arabic. It could also mean that the tag set used for the Arabic corpora has not been adapted to the specific nature of Arabic morphology (a comment also made in (Habash and Rambow, 2005)).

We propose an unsupervised morpheme-based HMM to address the data sparseness problem. In contrast to Bar-Haim et al., our model combines segmentation and morphological disambiguation, in parallel. The only resource we use in this work is a morphological analyzer. The analyzer itself can be generated from a word list and a morphological generation module, such as the HSpell wordlist (Har'el and Kenigsberg, 2004).

3 Morpheme-Based Model for Hebrew

3.1 Morpheme-Based HMM

The lexical items of word-based models are the words of the language. The implication of this decision is that both lexical and syntagmatic relations of the model, are based on a *word-oriented tagset*. With such a tagset, it must be possible to tag any word of the language with at least one tag.

Let us consider, for instance, the Hebrew phrase *bclm hn'im*³, which contains two words. The word *bclm* has several possible morpheme segmentations and analyses⁴ as described in Table 1. In word-based HMM, we consider such a phrase to be generated by a Markov process, based on the word-oriented tagset of $N = 1934$ tags/states and about $M = 175K$ word types. Line W of Table 2 describes the size of a first-order word-based HMM, built over our corpus. In this model, we found 834 entries for the Π vector (which models the distribution of tags in first position in sentences) out of possibly $N = 1934$, about 250K entries for the A matrix (which models the transition probabilities from tag to tag) out of possibly $N^2 \approx 3.7M$, and about 300K entries for the B matrix (which models

³Transcription according to Ornan (2002).

⁴The tagset we use for the annotation follows the guidelines we have developed (Elhadad et al., 2005).

	States	PI	A	A2	B	B2
W	1934	834	250K	7M	300K	5M
M	202	145	20K	700K	130K	1.7M

Table 2: Model Sizes

the emission probabilities from tag to word) out of possibly $M \cdot N \approx 350M$. For the case of a second-order HMM, the size of the $A2$ matrix (which models the transition probabilities from two tags to the third one), grows to about 7M entries, where the size of the $B2$ matrix (which models the emission probabilities from two tags to a word) is about 5M. Despite the sparseness of these matrices, the number of their entries is still high, since we model the whole set of features of the complex word forms.

Let us assume, that the right segmentation for the sentence is provided to us – for example: *bclm hn'im* – as is the case for English text. In such a way, the observation is composed of morphemes, generated by a Markov process, based on a morpheme-based tagset. The size of such a tagset for Hebrew is about 200, where the size of the $\Pi, A, B, A2$ and $B2$ matrices is reduced to 145, 16K, 140K, 700K, and 1.7M correspondingly, as described in line M of Table 2 – a reduction of 90% when compared with the size of a word-based model.

The problem in this approach, is that "someone" along the way, agglutinates the morphemes of each word leaving the observed morphemes uncertain. For example, the word *bclm* can be segmented in four different ways in Table 1, as indicated by the placement of the '-' in the Segmentation column, while the word *hn'im* can be segmented in two different ways. In the next section, we adapt the parameter estimation and the searching algorithms for such uncertain output observation.

3.2 Learning and Searching Algorithms for Uncertain Output Observation

In contrast to standard HMM, the output observations of the above morpheme-based HMM are ambiguous. We adapted Baum-Welch (Baum, 1972) and Viterbi (Manning and Schutze, 1999, 9.3.2) algorithms for such uncertain observation. We first formalize the output representation and then describe the algorithms.

Output Representation The learning and searching algorithms of HMM are based on the output sequence of the underlying Markov process. For the case of a morpheme-based model, the output sequence is uncertain – we don't see the emitted morphemes but the words they form. If, for instance, the Markov process emitted the morphemes *bclm hn'im*, we would see two words (*bclm hn'im*) instead. In order to handle the output ambiguity, we use static knowledge of how morphemes are combined into a word, such as the four known combinations of the word *bclm*, the two possible

combinations of the word *hn'im*, and their possible tags within the original words. Based on this information, we encode the sentence into a structure that represents all the possible "readings" of the sentence, according to the possible morpheme combinations of the words, and their possible tags.

The representation consists of a set of vectors, each vector containing the possible morphemes and their tags for each specific "time" (sequential position within the morpheme expansion of the words of the sentence). A morpheme is represented by a tuple (symbol, state, prev, next), where *symbol* denotes a morpheme, *state* is one possible tag for this morpheme, *prev* and *next* are sets of indexes, denoting the indexes of the morphemes (of the previous and the next vectors) that precede and follow the current morpheme in the overall lattice, representing the sentence. Fig. 2 describes the representation of the sentence *bclm hn'im*. An emission is denoted in this figure by its symbol, its state index, directed edges from its previous emissions, and directed edges to its next emissions.

In order to meet the condition of Baum-Eagon inequality (Baum, 1972) that the polynomial $P(O|\mu)$ – which represents the probability of an observed sequence O given a model μ – be homogeneous, we must add a sequence of special EOS (end of sentence) symbols at the end of each path up to the last vector, so that all the paths reach the same length.

The above text representation can be used to model multi-word expressions (MWEs). Consider the Hebrew sentence: *hw' 'wrk dyn gdwl*, which can be interpreted as composed of 3 units (he lawyer great / he is a great lawyer) or as 4 units (he edits law big / he is editing an important legal decision). In order to select the correct interpretation, we must determine whether *'wrk dyn* is an MWE. This is another case of uncertain output observation, which can be represented by our text encoding, as done in Fig. 1.



Figure 1: The sentence *hw' 'wrk dyn gdwl*

This representation seems to be expensive in term of the number of emissions per sentence. However, we observe in our data that most of the words have only one or two possible segmentations, and most of the segmentations consist of at most one affix. In practice, we found the average number of emissions per sentence in our corpus (where each symbol is counted as the number of its predecessor emissions) to be 455, where the average number of words per sentence is about 18. That is, the

cost of operating over an ambiguous sentence representation increases the size of the sentence (from 18 to 455), but on the other hand, it reduces the probabilistic model by a factor of 10 (as discussed above).

Morphological disambiguation over such a sequence of vectors of uncertain morphemes is similar to words extraction in automatic speech recognition (ASR) (Jurafsky and Martin, 2000, chp. 5,7). The states of the ASR model are phones, where each observation is a vector of spectral features. Given a sequence of observations for a sentence, the encoding – based on the lattice formed by the phones distribution of the observations, and the language model – searches for the set of words, made of phones, which maximizes the acoustic likelihood and the language model probabilities. In a similar manner, the supervised training of a speech recognizer combines a training corpus of speech wave files, together with word-transcription, and language model probabilities, in order to learn the phones model.

There are two main differences between the typical ASR model and ours: (1) an ASR decoder deals with one aspect - segmentation of the observations into a set of words, where this segmentation can be modeled at several levels: subphones, phones and words. These levels can be trained individually (such as training a language model from a written corpus, and training the phones model for each word type, given transcribed wave file), and then combined together (in a hierarchical model). Morphological disambiguation over uncertain morphemes, on the other hand, deals with both morpheme segmentation and the tagging of each morpheme with its morphological features. Modeling morpheme segmentation, within a given word, without its morphology features would be insufficient. (2) The supervised resources of ASR are not available for morphological disambiguation: we don't have a model of morphological features sequences (equivalent to the language model of ASR) nor a tagged corpus (equivalent to the transcribed wave files of ASR).

These two differences require a design which combines the two dimensions of the problem, in order to support unsupervised learning (and searching) of morpheme sequences and their morphological features, simultaneously.

Parameter Estimation We present a variation of the Baum-Welch algorithm (Baum, 1972) which operates over the lattice representation we have defined above. The algorithm starts with a probabilistic model μ (which can be chosen randomly or obtained from good initial conditions), and at each iteration, a new model $\bar{\mu}$ is derived in order to better explain the given output observations. For a given sentence, we define T as the number of words

in the sentence, and $\bar{T}$ as the number of vectors of the output representation $O = \{o_t\}, 1 \leq t \leq \bar{T}$, where each item in the output is denoted by $o_t^l = (sym, state, prev, next), 1 \leq t \leq \bar{T}, 1 \leq l \leq |o_t|$. We define $\alpha(t, l)$ as the probability to reach o_t^l at time t , and $\beta(t, l)$ as the probability to end the sequence from o_t^l . Fig. 3 describes the expectation and the maximization steps of the learning algorithm for a first-order HMM. The algorithm works in $O(\dot{T})$ time complexity, where $\dot{T}$ is the total number of symbols in the output sequence encoding, where each symbol is counted as the size of its *prev* set.

Searching for best state sequence The searching algorithm gets an observation sequence O and a probabilistic model μ , and looks for the best state sequence that generates the observation. We define $\delta(t, l)$ as the probability of the best state sequence that leads to emission o_t^l , and $\psi(t, l)$ as the index of the emission at time $t-1$ that precedes o_t^l in the best state sequence that leads to it. Fig. 4 describes the adaptation of the Viterbi (Manning and Schutze, 1999, 9.3.2) algorithm to our text representation for first-order HMM, which works in $O(\dot{T})$ time.

4 Experimental Results

We ran a series of experiments on a Hebrew corpus to compare various approaches to the full morphological disambiguation and PoS tagging tasks. The training corpus is obtained from various newspaper sources and is characterized by the following statistics: 6M word occurrences, 178,580 distinct words, 64,541 distinct lemmas. Overall, the ambiguity level is 2.4 (average number of analyses per word).

We tested the results on a test corpus, manually annotated by 2 taggers according to the guidelines we published and checked for agreement. The test corpus contains about 30K words. We compared two unsupervised models over this data set: Word model [W], and Morpheme model [M]. We also tested two different sets of initial conditions. Uniform distribution [Uniform]: For each word, each analysis provided by the analyzer is estimated with an equal likelihood. Context Free approximation [CF]: We applied the CF algorithm of Levinger et al.(1995) to estimate the likelihood of each analysis.

Table 3 reports the results of full morphological disambiguation. For each morpheme and word models, three types of models were tested: [1] First-order HMM, [2-] Partial second-order HMM - only state transitions were modeled (excluding B2 matrix), [2] Second-order HMM (including the B2 matrix).

Analysis If we consider the tagger which selects the most probable morphological analysis for each

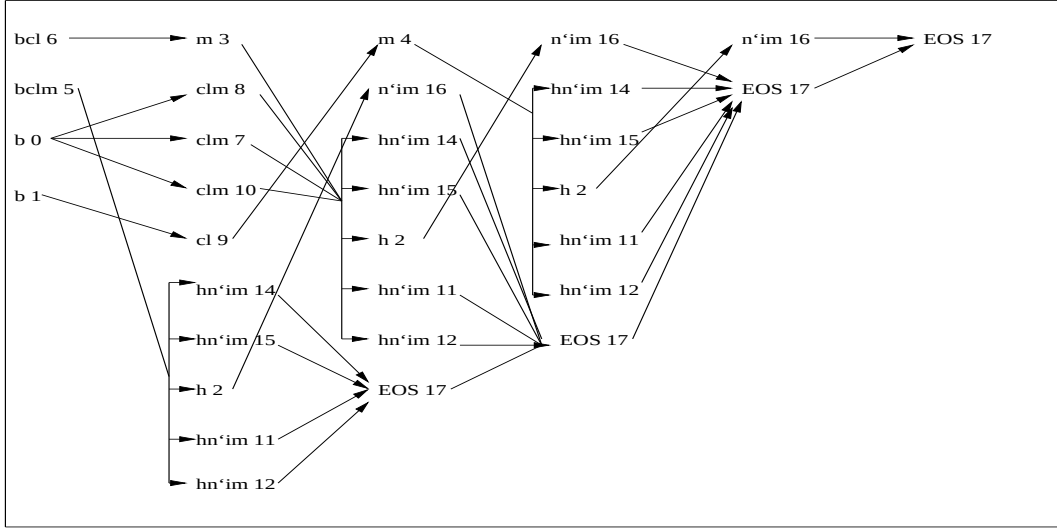


Figure 2: Representation of the sentence *bclm hn'im*

Expectation

$$\alpha(1, l) = \pi_{o_1^l.state} b_{o_1^l.state, o_1^l.sym} \quad (1)$$

$$\alpha(t, l) = b_{o_t^l.state, o_t^l.sym} \sum_{l' \in o_t^l.prev} \alpha(t-1, l') a_{o_{t-1}^{l'}.state, o_t^l.state} \quad (2)$$

$$\beta(\bar{T}, l) = 1 \quad (2)$$

$$\beta(t, l) = \sum_{l' \in o_t^l.next} a_{o_t^l.state, o_{t+1}^{l'}.state} b_{o_{t+1}^{l'}.state, o_{t+1}^{l'}.sym} \beta(t+1, l')$$

Maximization

$$\bar{\pi}_i = \frac{\sum_{l: o_1^l.state=i} \alpha(1, l) \beta(1, l)}{\sum_l \alpha(1, l) \beta(1, l)} \quad (3)$$

$$\bar{a}_{i,j} = \frac{\sum_{t=2}^{\bar{T}} \sum_{l: o_t^l.state=j} \sum_{l' \in o_t^l.prev: o_{t-1}^{l'}.state=i} \alpha(t-1, l') a_{i,j, o_t^l.sym} \beta(t, l)}{\sum_{t=1}^{\bar{T}-1} \sum_{l: o_t^l.state=i} \alpha(t, l) \beta(t, l)} \quad (4)$$

$$\bar{b}_{i,k} = \frac{\sum_{t=1}^{\bar{T}} \sum_{l: o_t^l.sym=k, o_t^l.state=i} \alpha(t, l) \beta(t, l)}{\sum_{t=1}^{\bar{T}} \sum_{l: o_t^l.state=i} \alpha(t, l) \beta(t, l)} \quad (5)$$

Figure 3: The learning algorithm for first-order model

Initialization

$$\delta(1, l) = \pi_{o_1^l.state} b_{o_1^l.state, o_1^l.sym} \quad (6)$$

Induction

$$\delta(t, l) = \max_{l' \in o_t^l.prev} \delta(t-1, l') a_{o_{t-1}^{l'}.state, o_t^l.state} b_{o_t^l.state, o_t^l.sym} \quad (7)$$

$$\psi(t, l) = \operatorname{argmax}_{l' \in o_t^l.prev} \delta(t-1, l') a_{o_{t-1}^{l'}.state, o_t^l.state} b_{o_t^l.state, o_t^l.sym} \quad (8)$$

Termination and path readout

$$\bar{X}_{\bar{T}} = \operatorname{argmax}_{1 \leq l \leq |\bar{T}|} \delta(\bar{T}, l) \quad (9)$$

$$\begin{aligned} \bar{X}_t &= \psi(t+1, \bar{X}_{t+1}) \\ P(\bar{X}) &= \max_{1 \leq l \leq |O_{\bar{T}}|} \delta(\bar{T}, l) \end{aligned} \quad (10)$$

Figure 4: The searching algorithm for first-order model

	Order	Uniform	CF
W	1	82.01	84.08
W	2-	80.44	85.75
W	2	79.88	85.78
M	1	81.08	84.54
M	2-	81.53	88.5
M	2	83.39	85.83

Table 3: Morphological Disambiguation

word in the text, according to Levinger et al. (1995) approximations, with accuracy of 78.2%, as the baseline tagger, four steps of error reduction can be identified. **(1)** Contextual information: The simplest first-order word-based HMM with uniform initial conditions, achieves error reduction of 17.5% (78.2 – 82.01). **(2)** Initial conditions: Error reductions in the range: 11.5% – 37.8% (82.01 – 84.08 for word model 1, and 81.53 – 88.5 for morpheme model 2-) were achieved by initializing the various models with context-free approximations. While this observation confirms Elworthy (1994), the impact of error reduction is much less than reported there for English - about 70% (79 – 94). The key difference (beside the unclear characteristic of Elworthy initial condition - since he made use of an annotated corpus) is the much higher quality of the uniform distribution for Hebrew. **(3)** Model order: The partial second-order HMM [2-] produced the best results for both word (85.75%) and morpheme (88.5%) models over the initial condition. The full second-order HMM [2] didn't upgrade the accuracy of the partial second-order, but achieved the best results for the uniform distribution morpheme model. This is because the context-free approximation does not take into account the tag of the previous word, which is part of model 2. We believe that initializing the morpheme model over a small set of annotated corpus will set much stronger initial condition for this model. **(4)** Model type: The main result of this paper is the error reduction of the morpheme model with respect to the word model: about 19.3% (85.75 – 88.5).

In addition, we apply the above models for the simpler task of segmentation and PoS tagging, as reported in Table 4. The task requires picking the correct morphemes of each word with their correct PoS (excluding all other morphological features). The best result for this task is obtained with the morpheme model 2: 92.32%. For this simpler task, the improvement brought by the morpheme model over the word model is less significant, but still consists of a 5% error reduction.

Unknown words account for a significant chunk of the errors. Table 5 shows the distribution of errors contributed by unknown words (words that cannot be analyzed by the morphological analyzer). 7.5% of the words in the test corpus are unknown: 4% are not recognized at all by the morphological analyzer (marked as [None] in the ta-

	Order	Uniform	CF
W	1	91.07	91.47
W	2-	90.45	91.93
W	2	90.21	91.84
M	1	89.23	91.42
M	2-	89.77	91.76
M	2	91.42	92.32

Table 4: Segmentation and PoS Tagging

ble), and for 3.5%, the set of analyses proposed by the analyzer does not contain the correct analysis [Missing]. We extended the lexicon to include *missing* and *none* lexemes of the closed sets. In addition, we modified the analyzer to extract all possible segmentations of unknown words, with all the possible tags for the segmented affixes, where the remaining unknown baseforms are tagged as UK. The model was trained over this set. In the next phase, the corpus was automatically tagged, according to the trained model, in order to form a tag distribution for each unknown word, according to its context and its form. Finally, the tag for each unknown word were selected according to its tag distribution. This strategy accounts for about half of the 7.5% unknown words.

	None	Missing	%
Proper name	26	36	62
Closed Set	8	5.6	13.6
Other	16.5	5.4	21.9
Junk	2.5	0	2.5
	53	47	100

Table 5: Unknown Word Distribution

Table 6 shows the confusion matrix for known words (5% and up). The key confusions can be attributed to linguistic properties of Modern Hebrew: most Hebrew proper names are also nouns (and they are not marked by capitalization) – which explains the PN/N confusion. The verb/noun and verb/adjective confusions are explained by the nature of the participle form in Hebrew (beinoni) – participles behave syntactically almost in an identical manner as nouns.

Correct	Error	%
proper name	noun	17.9
noun	verb	15.3
noun	proper name	6.6
verb	noun	6.3
adjective	noun	5.4
adjective	verb	5.0

Table 6: Confusion Matrix for Known Words

5 Conclusions and Future Work

In this work, we have introduced a new text encoding method that captures rules of word formation in a language with affixational morphology such as Hebrew. This text encoding method allows us to

learn in parallel segmentation and tagging rules in an unsupervised manner, despite the high ambiguity level of the morphological data (average number of 2.4 analyses per word). Reported results on a large scale corpus (6M words) with fully unsupervised learning are 92.32% for PoS tagging and 88.5% for full morphological disambiguation.

In this work, we used the backoff smoothing method, suggested by Thede and Harper (1999), with an extension of additive smoothing (Chen, 1996, 2.2.1) for the lexical probabilities (B and B2 matrices). To complete this study, we are currently investigating several smoothing techniques (Chen, 1996), in order to check whether the morpheme model is critical for the data sparseness problem, or whether it can be handled with smoothing over a word model.

We are currently investigating two major methods to improve our results: first, we have started gathering a larger corpus of manually tagged text and plan to perform semi-supervised learning on a corpus of 100K manually tagged words. Second, we plan to improve the unknown word model, such as integrating it with named entity recognition system (Ben-Mordechai, 2005).

References

- Emmanuel Allon. 1995. *Unvocalized Hebrew Writing*. Ben Gurion University Press. (in Hebrew).
- Roy Bar-Haim, Khalil Sima'an, and Yoad Winter. 2005. Choosing an optimal architecture for segmentation and pos-tagging of modern Hebrew. In *Proceedings of ACL-05 Workshop on Computational Approaches to Semitic Languages*.
- Leonard E. Baum. 1972. An inequality and associated maximization technique in statistical estimation for probabilistic functions of a Markov process. *Inequalities*, 3:1–8.
- Na'ama Ben-Mordechai. 2005. Named entities recognition in Hebrew. Master's thesis, Ben Gurion University of the Negev, Beer Sheva, Israel. (in Hebrew).
- Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21:543–565.
- David Carmel and Yoelle S. Maarek. 1999. Morphological disambiguation for Hebrew search systems. In *Proceeding of NGITS-99*.
- Stanley F. Chen. 1996. *Building Probabilistic Models for Natural Language*. Ph.D. thesis, Harvard University, Cambridge, MA.
- Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic tagging of Arabic text: From raw text to base phrase chunks. In *Proceeding of HLT-NAACL-04*.
- Kevin Duh and Katrin Kirchhoff. 2005. Pos tagging of dialectal Arabic: A minimally supervised approach. In *Proceedings of ACL-05 Workshop on Computational Approaches to Semitic Languages*.
- Michael Elhadad, Yael Netzer, David Gabay, and Meni Adler. 2005. Hebrew morphological tagging guidelines. Technical report, Ben Gurion University, Dept. of Computer Science.
- David Elworthy. 1994. Does Baum-Welch re-estimation help taggers? In *Proceeding of ANLP-94*.
- Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceeding of ACL-05*.
- Nadav Har'el and Dan Kenigsberg. 2004. HSpell - the free Hebrew spell checker and morphological analyzer. *Israeli Seminar on Computational Linguistics, December 2004*.
- Daniel Jurafsky and James H. Martin. 2000. *Speech and language processing*. Prentice-Hall.
- Gennady Lembersky. 2001. Named entities recognition; compounds: approaches and recognitions methods. Master's thesis, Ben Gurion University of the Negev, Beer Sheva, Israel. (in Hebrew).
- Moshe Levinger, Uzi Ornan, and Alon Itai. 1995. Learning morpholexical probabilities from an untagged corpus with an application to Hebrew. *Computational Linguistics*, 21:383–404.
- Moshe Levinger. 1992. Morphological disambiguation in hebrew. Master's thesis, Technion, Haifa, Israel. (in Hebrew).
- Christopher D. Manning and Hinrich Schutze. 1999. *Foundation of Statistical Language Processing*. MIT Press.
- Bernard Merialdo. 1994. Tagging English text with probabilistic model. *Computational Linguistics*, 20:155–171.
- Uzi Ornan. 2002. Hebrew in latin script. *Lěšonénu*, LXIV:137–151. (in Hebrew).
- Erel Segal. 2000. Hebrew morphological analyzer for Hebrew undotted texts. Master's thesis, Technion, Haifa, Israel. (in Hebrew).
- Scott M. Thede and Mary P. Harper. 1999. A second-order hidden Markov model for part-of-speech tagging. In *Proceeding of ACL-99*.
- Shlomo Yona and Shuly Wintner. 2005. A finite-state morphological grammar of Hebrew. In *Proceedings of ACL-05 Workshop on Computational Approaches to Semitic Languages*.

Contextual Dependencies in Unsupervised Word Segmentation*

Sharon Goldwater and Thomas L. Griffiths and Mark Johnson

Department of Cognitive and Linguistic Sciences

Brown University

Providence, RI 02912

{Sharon_Goldwater, Tom_Griffiths, Mark_Johnson}@brown.edu

Abstract

Developing better methods for segmenting continuous text into words is important for improving the processing of Asian languages, and may shed light on how humans learn to segment speech. We propose two new Bayesian word segmentation methods that assume unigram and bigram models of word dependencies respectively. The bigram model greatly outperforms the unigram model (and previous probabilistic models), demonstrating the importance of such dependencies for word segmentation. We also show that previous probabilistic models rely crucially on sub-optimal search procedures.

1 Introduction

Word segmentation, i.e., discovering word boundaries in continuous text or speech, is of interest for both practical and theoretical reasons. It is the first step of processing orthographies without explicit word boundaries, such as Chinese. It is also one of the key problems that human language learners must solve as they are learning language.

Many previous methods for unsupervised word segmentation are based on the observation that transitions between units (characters, phonemes, or syllables) within words are generally more predictable than transitions across word boundaries. Statistics that have been proposed for measuring these differences include “successor frequency” (Harris, 1954), “transitional probabilities” (Saffran et al., 1996), mutual information (Sun et al.,

1998), “accessor variety” (Feng et al., 2004), and boundary entropy (Cohen and Adams, 2001).

While methods based on local statistics are quite successful, here we focus on approaches based on explicit probabilistic models. Formulating an explicit probabilistic model permits us to cleanly separate assumptions about the input and properties of likely segmentations from details of algorithms used to find such solutions. Specifically, this paper demonstrates the importance of contextual dependencies for word segmentation by comparing two probabilistic models that differ only in that the first assumes that the probability of a word is independent of its local context, while the second incorporates bigram dependencies between adjacent words. The algorithms we use to search for likely segmentations do differ, but so long as the segmentations they produce are close to optimal we can be confident that any differences in the segmentations reflect differences in the probabilistic models, i.e., in the kinds of dependencies between words.

We are not the first to propose explicit probabilistic models of word segmentation. Two successful word segmentation systems based on explicit probabilistic models are those of Brent (1999) and Venkataraman (2001). Brent’s Model-Based Dynamic Programming (MBDP) system assumes a unigram word distribution. Venkataraman uses standard unigram, bigram, and trigram language models in three versions of his system, which we refer to as n -gram Segmentation (NGS). Despite their rather different generative structure, the MBDP and NGS segmentation accuracies are very similar. Moreover, the segmentation accuracy of the NGS unigram, bigram, and trigram models hardly differ, suggesting that contextual dependencies are irrelevant to word segmentation. How-

*This work was partially supported by the following grants: NIH 1R01-MH60922, NIH RO1-DC000314, NSF IGERT-DGE-9870676, and the DARPA CALO project.

ever, the segmentations produced by both these methods depend crucially on properties of the search procedures they employ. We show this by exhibiting for each model a segmentation that is less accurate but more probable under that model.

In this paper, we present an alternative framework for word segmentation based on the Dirichlet process, a distribution used in nonparametric Bayesian statistics. This framework allows us to develop extensible models that are amenable to standard inference procedures. We present two such models incorporating unigram and bigram word dependencies, respectively. We use Gibbs sampling to sample from the posterior distribution of possible segmentations under these models.

The plan of the paper is as follows. In the next section, we describe MBDP and NGS in detail. In Section 3 we present the unigram version of our own model, the Gibbs sampling procedure we use for inference, and experimental results. Section 4 extends that model to incorporate bigram dependencies, and Section 5 concludes the paper.

2 NGS and MBDP

The NGS and MBDP systems are similar in some ways: both are designed to find utterance boundaries in a corpus of phonemically transcribed utterances, with known utterance boundaries. Both also use approximate online search procedures, choosing and fixing a segmentation for each utterance before moving onto the next. In this section, we focus on the very different probabilistic models underlying the two systems. We show that the optimal solution under the NGS model is the unsegmented corpus, and suggest that this problem stems from the fact that the model assumes a uniform prior over hypotheses. We then present the MBDP model, which uses a non-uniform prior but is difficult to extend beyond the unigram case.

2.1 NGS

NGS assumes that each utterance is generated independently via a standard n -gram model. For simplicity, we will discuss the unigram version of the model here, although our argument is equally applicable to the bigram and trigram versions. The unigram model generates an utterance u according to the grammar in Figure 1, so

$$P(u) = p_{\$}(1 - p_{\$})^{n-1} \prod_{j=1}^n P(w_j) \quad (1)$$

$$\begin{array}{ll} 1 - p_{\$} & U \rightarrow W U \\ p_{\$} & U \rightarrow W \\ P(w) & W \rightarrow w \end{array} \quad \forall w \in \Sigma^*$$

Figure 1: The unigram NGS grammar.

where u consists of the words $w_1 \dots w_n$ and $p_{\$}$ is the probability of the utterance boundary marker $\$$. This model can be used to find the highest probability segmentation hypothesis h given the data d by using Bayes' rule:

$$P(h|d) \propto P(d|h)P(h)$$

NGS assumes a uniform prior $P(h)$ over hypotheses, so its goal is to find the solution that maximizes the likelihood $P(d|h)$.

Using this model, NGS's approximate search technique delivers competitive results. However, the true maximum likelihood solution is not competitive, since it contains no utterance-internal word boundaries. To see why not, consider the solution in which $p_{\$} = 1$ and each utterance is a single 'word', with probability equal to the empirical probability of that utterance. Any other solution will match the empirical distribution of the data less well. In particular, a solution with additional word boundaries must have $1 - p_{\$} > 0$, which means it wastes probability mass modeling unseen data (which can now be generated by concatenating observed utterances together).

Intuitively, the NGS model considers the unsegmented solution to be optimal because it ranks all hypotheses equally probable *a priori*. We know, however, that hypotheses that memorize the input data are unlikely to generalize to unseen data, and are therefore poor solutions. To prevent memorization, we could restrict our hypothesis space to models with fewer parameters than the number of utterances in the data. A more general and mathematically satisfactory solution is to assume a non-uniform prior, assigning higher probability to hypotheses with fewer parameters. This is in fact the route taken by Brent in his MBDP model, as we shall see in the following section.

2.2 MBDP

MBDP assumes a corpus of utterances is generated as a single probabilistic event with four steps:

1. Generate L , the number of lexical types.
2. Generate a phonemic representation for each type (except the utterance boundary type, $\$$).

3. Generate a token frequency for each type.
4. Generate an ordering for the set of tokens.

In a final deterministic step, the ordered tokens are concatenated to create an unsegmented corpus. This means that certain segmented corpora will produce the observed data with probability 1, and all others will produce it with probability 0. The posterior probability of a segmentation given the data is thus proportional to its prior probability under the generative model, and the best segmentation is that with the highest prior probability.

There are two important points to note about the MBDP model. First, the distribution over L assigns higher probability to models with fewer lexical items. We have argued that this is necessary to avoid memorization, and indeed the unsegmented corpus is not the optimal solution under this model, as we will show in Section 3. Second, the factorization into four separate steps makes it theoretically possible to modify each step independently in order to investigate the effects of the various modeling assumptions. However, the mathematical statement of the model and the approximations necessary for the search procedure make it unclear how to modify the model in any interesting way. In particular, the fourth step uses a uniform distribution, which creates a unigram constraint that cannot easily be changed. Since our research aims to investigate the effects of different modeling assumptions on lexical acquisition, we develop in the following sections a far more flexible model that also incorporates a preference for sparse solutions.

3 Unigram Model

3.1 The Dirichlet Process Model

Our goal is a model of language that prefers sparse solutions, allows independent modification of components, and is amenable to standard search procedures. We achieve this goal by basing our model on the *Dirichlet process* (DP), a distribution used in nonparametric Bayesian statistics. Our unigram model of word frequencies is defined as

$$\begin{aligned} w_i | G &\sim G \\ G | \alpha_0, P_0 &\sim \text{DP}(\alpha_0, P_0) \end{aligned}$$

where the *concentration parameter* α_0 and the *base distribution* P_0 are parameters of the model. Each word w_i in the corpus is drawn from a

distribution G , which consists of a set of possible words (the lexicon) and probabilities associated with those words. G is generated from a $\text{DP}(\alpha_0, P_0)$ distribution, with the items in the lexicon being sampled from P_0 and their probabilities being determined by α_0 , which acts like the parameter of an infinite-dimensional symmetric Dirichlet distribution. We provide some intuition for the roles of α_0 and P_0 below.

Although the DP model makes the distribution G explicit, we never deal with G directly. We take a Bayesian approach and integrate over all possible values of G . The conditional probability of choosing to generate a word from a particular lexical entry is then given by a simple stochastic process known as the Chinese restaurant process (CRP) (Aldous, 1985). Imagine a restaurant with an infinite number of tables, each with infinite seating capacity. Customers enter the restaurant and seat themselves. Let z_i be the table chosen by the i th customer. Then

$$P(z_i | \mathbf{z}_{-i}) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{i-1+\alpha_0} & 0 \leq k < K(\mathbf{z}_{-i}) \\ \frac{\alpha_0}{i-1+\alpha_0} & k = K(\mathbf{z}_{-i}) \end{cases} \quad (2)$$

where $\mathbf{z}_{-i} = z_1 \dots z_{i-1}$, $n_k^{(\mathbf{z}_{-i})}$ is the number of customers already sitting at table k , and $K(\mathbf{z}_{-i})$ is the total number of occupied tables. In our model, the tables correspond to (possibly repeated) lexical entries, having labels generated from the distribution P_0 . The seating arrangement thus specifies a distribution over word tokens, with each customer representing one token. This model is an instance of the two-stage modeling framework described by Goldwater et al. (2006), with P_0 as the generator and the CRP as the adaptor.

Our model can be viewed intuitively as a *cache model*: each word in the corpus is either retrieved from a cache or generated anew. Summing over all the tables labeled with the same word yields the probability distribution for the i th word given previously observed words $\mathbf{w}_{-i}$:

$$P(w_i | \mathbf{w}_{-i}) = \frac{n_{w_i}^{(\mathbf{w}_{-i})}}{i-1+\alpha_0} + \frac{\alpha_0 P_0(w_i)}{i-1+\alpha_0} \quad (3)$$

where $n_w^{(\mathbf{w}_{-i})}$ is the number of instances of w observed in $\mathbf{w}_{-i}$. The first term is the probability of generating w from the cache (i.e., sitting at an occupied table), and the second term is the probability of generating it anew (sitting at an unoccupied table). The actual table assignments $\mathbf{z}_{-i}$ only become important later, in the bigram model.

There are several important points to note about this model. First, the probability of generating a particular word from the cache increases as more instances of that word are observed. This rich-get-richer process creates a power-law distribution on word frequencies (Goldwater et al., 2006), the same sort of distribution found empirically in natural language. Second, the parameter α_0 can be used to control how sparse the solutions found by the model are. This parameter determines the total probability of generating *any* novel word, a probability that decreases as more data is observed, but never disappears. Finally, the parameter P_0 can be used to encode expectations about the nature of the lexicon, since it defines a probability distribution across different novel words. The fact that this distribution is defined separately from the distribution on word frequencies gives the model additional flexibility, since either distribution can be modified independently of the other.

Since the goal of this paper is to investigate the role of context in word segmentation, we chose the simplest possible model for P_0 , i.e. a unigram phoneme distribution:

$$P_0(w) = p_{\#}(1 - p_{\#})^{n-1} \prod_{i=1}^n P(m_i) \quad (4)$$

where word w consists of the phonemes $m_1 \dots m_n$, and $p_{\#}$ is the probability of the word boundary $\#$. For simplicity we used a uniform distribution over phonemes, and experimented with different fixed values of $p_{\#}$.¹

A final detail of our model is the distribution on utterance lengths, which is geometric. That is, we assume a grammar similar to the one shown in Figure 1, with the addition of a symmetric Beta($\frac{\tau}{2}$) prior over the probability of the U productions,² and the substitution of the DP for the standard multinomial distribution over the W productions.

3.2 Gibbs Sampling

Having defined our generative model, we are left with the problem of inference: we must determine the posterior distribution of hypotheses given our input corpus. To do so, we use Gibbs sampling, a standard Markov chain Monte Carlo method (Gilks et al., 1996). Gibbs sampling is an iterative procedure in which variables are repeatedly

¹Note, however, that our model could be extended to learn both $p_{\#}$ and the distribution over phonemes.

²The Beta distribution is a Dirichlet distribution over two outcomes.

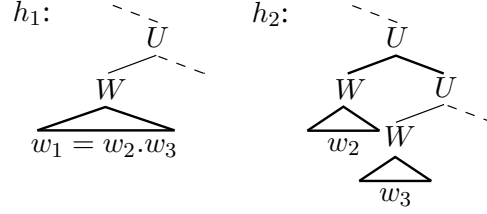


Figure 2: The two hypotheses considered by the unigram sampler. Dashed lines indicate possible additional structure. All rules except those in bold are part of h^- .

sampled from their conditional posterior distribution given the current values of all other variables in the model. The sampler defines a Markov chain whose stationary distribution is $P(h|d)$, so after convergence samples are from this distribution.

Our Gibbs sampler considers a single possible boundary point at a time, so each sample is from a set of two hypotheses, h_1 and h_2 . These hypotheses contain all the same boundaries except at the one position under consideration, where h_2 has a boundary and h_1 does not. The structures are shown in Figure 2. In order to sample a hypothesis, we need only calculate the relative probabilities of h_1 and h_2 . Since h_1 and h_2 are the same except for a few rules, this is straightforward. Let h^- be all of the structure shared by the two hypotheses, including n^- words, and let d be the observed data. Then

$$\begin{aligned} P(h_1|h^-, d) &= P(w_1|h^-, d) \\ &= \frac{n_{w_1}^{(h^-)} + \alpha_0 P_0(w_1)}{n^- + \alpha_0} \end{aligned} \quad (5)$$

where the second line follows from Equation 3 and the properties of the CRP (in particular, that it is *exchangeable*, with the probability of a seating configuration not depending on the order in which customers arrive (Aldous, 1985)). Also,

$$\begin{aligned} P(h_2|h^-, d) &= P(r, w_2, w_3|h^-, d) \\ &= P(r|h^-, d)P(w_2|h^-, d)P(w_3|w_2, h^-, d) \\ &= \frac{n_r + \frac{\tau}{2}}{n^- + 1 + \tau} \cdot \frac{n_{w_2}^{(h^-)} + \alpha_0 P_0(w_2)}{n^- + \alpha_0} \\ &\quad \cdot \frac{n_{w_3}^{(h^-)} + I(w_2 = w_3) + \alpha_0 P_0(w_3)}{n^- + 1 + \alpha_0} \end{aligned} \quad (6)$$

where n_r is the number of branching rules $r = U \rightarrow W U$ in h^- , and $I(\cdot)$ is an indicator function taking on the value 1 when its argument is

true, and 0 otherwise. The n_r term is derived by integrating over all possible values of $p_\$,$ and noting that the total number of U productions in h^- is $n^- + 1$.

Using these equations we can simply proceed through the data, sampling each potential boundary point in turn. Once the Gibbs sampler converges, these samples will be drawn from the posterior distribution $P(h|d)$.

3.3 Experiments

In our experiments, we used the same corpus that NGS and MBDP were tested on. The corpus, supplied to us by Brent, consists of 9790 transcribed utterances (33399 words) of child-directed speech from the Bernstein-Ratner corpus (Bernstein-Ratner, 1987) in the CHILDES database (MacWhinney and Snow, 1985). The utterances have been converted to a phonemic representation using a phonemic dictionary, so that each occurrence of a word has the same phonemic transcription. Utterance boundaries are given in the input to the system; other word boundaries are not.

Because our Gibbs sampler is slow to converge, we used annealing to speed inference. We began with a temperature of $\gamma = 10$ and decreased γ in 10 increments to a final value of 1. A temperature of γ corresponds to raising the probabilities of h_1 and h_2 to the power of $\frac{1}{\gamma}$ prior to sampling.

We ran our Gibbs sampler for 20,000 iterations through the corpus (with $\gamma = 1$ for the final 2000) and evaluated our results on a single sample at that point. We calculated precision (P), recall (R), and F-score (F) on the word tokens in the corpus, where both boundaries of a word must be correct to count the word as correct. The induced lexicon was also scored for accuracy using these metrics (LP, LR, LF).

Recall that our DP model has three parameters: $\tau, p_\#,$ and α_0 . Given the large number of known utterance boundaries, we expect the value of τ to have little effect on our results, so we simply fixed $\tau = 2$ for all experiments. Figure 3 shows the effects of varying of $p_\#$ and α_0 .³ Lower values of $p_\#$ cause longer words, which tends to improve recall (and thus F-score) in the lexicon, but decrease token accuracy. Higher values of α_0 allow more novel words, which also improves lexicon recall,

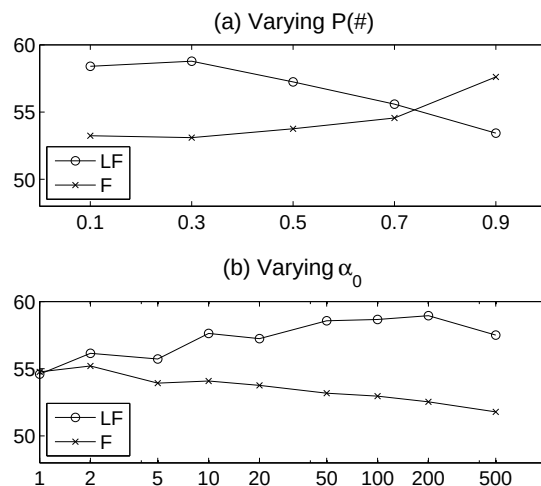


Figure 3: Word (F) and lexicon (LF) F-score (a) as a function of $p_\#$, with $\alpha_0 = 20$ and (b) as a function of α_0 , with $p_\# = .5$.

but begins to degrade precision after a point. Due to the negative correlation between token accuracy and lexicon accuracy, there is no single best value for either $p_\#$ or α_0 ; further discussion refers to the solution for $p_\# = .5, \alpha_0 = 20$ (though others are qualitatively similar).

In Table 1(a), we compare the results of our system to those of MBDP and NGS.⁴ Although our system has higher lexicon accuracy than the others, its token accuracy is much worse. This result occurs because our system often mis-analyzes frequently occurring words. In particular, many of these words occur in common collocations such as *what's that* and *do you*, which the system interprets as a single words. It turns out that a full 31% of the proposed lexicon and nearly 30% of tokens consist of these kinds of errors.

Upon reflection, it is not surprising that a unigram language model would segment words in this way. Collocations violate the unigram assumption in the model, since they exhibit strong word-to-word dependencies. The only way the model can capture these dependencies is by assuming that these collocations are in fact words themselves.

Why don't the MBDP and NGS unigram models exhibit these problems? We have already shown that NGS's results are due to its search procedure rather than its model. The same turns out to be true for MBDP. Table 2 shows the probabili-

³It is worth noting that all these parameters could be inferred. We leave this for future work.

⁴We used the implementations of MBDP and NGS available at <http://www.speech.sri.com/people/anand/> to obtain results for those systems.

(a)	P	R	F	LP	LR	LF
NGS	67.7	70.2	68.9	52.9	51.3	52.0
MBDP	67.0	69.4	68.2	53.6	51.3	52.4
DP	61.9	47.6	53.8	57.0	57.5	57.2

(b)	P	R	F	LP	LR	LF
NGS	76.6	85.8	81.0	60.0	52.4	55.9
MBDP	77.0	86.1	81.3	60.8	53.0	56.6
DP	94.2	97.1	95.6	86.5	62.2	72.4

Table 1: Accuracy of the various systems, with best scores in bold. The unigram version of NGS is shown. DP results are with $p_{\#} = .5$ and $\alpha_0 = 20$. (a) Results on the true corpus. (b) Results on the permuted corpus.

Seg:	True	None	MBDP	NGS	DP
NGS	204.5	90.9	210.7	210.8	183.0
MBDP	208.2	321.7	217.0	218.0	189.8
DP	222.4	393.6	231.2	231.6	200.6

Table 2: Negative log probabilities (x 1000) under each model of the true solution, the solution with no utterance-internal boundaries, and the solutions found by each algorithm. Best solutions under each model are bold.

ties under each model of various segmentations of the corpus. From these figures, we can see that the MBDP model assigns higher probability to the solution found by our Gibbs sampler than to the solution found by Brent’s own incremental search algorithm. In other words, Brent’s model *does* prefer the lower-accuracy collocation solution, but his search algorithm instead finds a higher-accuracy but lower-probability solution.

We performed two experiments suggesting that our own inference procedure does not suffer from similar problems. First, we initialized our Gibbs sampler in three different ways: with no utterance-internal boundaries, with a boundary after every character, and with random boundaries. Our results were virtually the same regardless of initialization. Second, we created an artificial corpus by randomly permuting the words in the true corpus, leaving the utterance lengths the same. The artificial corpus adheres to the unigram assumption of our model, so if our inference procedure works correctly, we should be able to correctly identify the words in the permuted corpus. This is exactly what we found, as shown in Table 1(b). While all three models perform better on the artificial cor-

pus, the improvements of the DP model are by far the most striking.

4 Bigram Model

4.1 The Hierarchical Dirichlet Process Model

The results of our unigram experiments suggested that word segmentation could be improved by taking into account dependencies between words. To test this hypothesis, we extended our model to incorporate bigram dependencies using a *hierarchical Dirichlet process* (HDP) (Teh et al., 2005). Our approach is similar to previous n -gram models using hierarchical Pitman-Yor processes (Goldwater et al., 2006; Teh, 2006). The HDP is appropriate for situations in which there are multiple distributions over similar sets of outcomes, and the distributions are believed to be similar. In our case, we define a bigram model by assuming each word has a different distribution over the words that follow it, but all these distributions are linked. The definition of our bigram language model as an HDP is

$$\begin{aligned}
 w_i | w_{i-1} = w, H_w &\sim H_w & \forall w \\
 H_w | \alpha_1, G &\sim \text{DP}(\alpha_1, G) & \forall w \\
 G | \alpha_0, P_0 &\sim \text{DP}(\alpha_0, P_0)
 \end{aligned}$$

That is, $P(w_i | w_{i-1} = w)$ is distributed according to H_w , a DP specific to word w . H_w is linked to the DPs for all other words by the fact that they share a common base distribution G , which is generated from another DP.⁵

As in the unigram model, we never deal with H_w or G directly. By integrating over them, we get a distribution over bigram frequencies that can be understood in terms of the CRP. Now, each word type w is associated with its own restaurant, which represents the distribution over words that follow w . Different restaurants are not completely independent, however: the labels on the tables in the restaurants are all chosen from a common base distribution, which is another CRP.

To understand the HDP model in terms of a grammar, we consider \$ as a special word type, so that w_i ranges over $\Sigma^* \cup \{\$\}$. After observing $\mathbf{w}_{-i}$, the HDP grammar is as shown in Figure 4,

⁵This HDP formulation is an oversimplification, since it does not account for utterance boundaries properly. The grammar formulation (see below) does.

$$\begin{aligned}
P(h_1|h^-, d) &= \frac{n_{(w_l, w_1)} + \alpha_1 P_1(w_1|h^-, d)}{n_{w_l} + \alpha_1} \cdot \frac{n_{(w_1, w_r)} + I(w_l = w_1 = w_r) + \alpha_1 P_1(w_r|h^-, d)}{n_{w_1} + 1 + \alpha_1} \\
P(h_2|h^-, d) &= \frac{n_{(w_l, w_2)} + \alpha_1 P_1(w_2|h^-, d)}{n_{w_l} + \alpha_1} \cdot \frac{n_{(w_2, w_3)} + I(w_l = w_2 = w_3) + \alpha_1 P_1(w_3|h^-, d)}{n_{w_2} + 1 + \alpha_1} \cdot \\
&\quad \frac{n_{(w_3, w_r)} + I(w_l = w_3, w_2 = w_r) + I(w_2 = w_3 = w_r) + \alpha_1 P_1(w_r|h^-, d)}{n_{w_3} + 1 + I(w_2 = w_4) + \alpha_1}
\end{aligned}$$

Figure 6: Gibbs sampling equations for the bigram model. All counts are with respect to h^- .

jectives, which has led to misleading results. In particular, previous work suggested that the use of word-to-word dependencies has little effect on word segmentation. Our experiments indicate instead that bigram dependencies can be crucial for avoiding under-segmentation of frequent collocations. Incorporating these dependencies into our model greatly improved segmentation accuracy, and led to better performance than previous approaches on all measures.

References

- D. Aldous. 1985. Exchangeability and related topics. In *École d'été de probabilités de Saint-Flour, XIII—1983*, pages 1–198. Springer, Berlin.
- C. Antoniuk. 1974. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2:1152–1174.
- N. Bernstein-Ratner. 1987. The phonology of parent-child speech. In K. Nelson and A. van Kleeck, editors, *Children's Language*, volume 6. Erlbaum, Hillsdale, NJ.
- M. Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105.
- P. Cohen and N. Adams. 2001. An algorithm for segmenting categorical timeseries into meaningful episodes. In *Proceedings of the Fourth Symposium on Intelligent Data Analysis*.
- H. Feng, K. Chen, X. Deng, and W. Zheng. 2004. Accessor variety criteria for chinese word extraction. *Computational Linguistics*, 30(1).
- W.R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors. 1996. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, Suffolk.
- S. Goldwater, T. Griffiths, and M. Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, Cambridge, MA. MIT Press.
- Z. Harris. 1954. Distributional structure. *Word*, 10:146–162.
- B. MacWhinney and C. Snow. 1985. The child language data exchange system. *Journal of Child Language*, 12:271–296.
- J. Saffran, E. Newport, and R. Aslin. 1996. Word segmentation: The role of distributional cues. *Journal of Memory and Language*, 35:606–621.
- M. Sun, D. Shen, and B. Tsou. 1998. Chinese word segmentation without using lexicon and hand-crafted training data. In *Proceedings of COLING-ACL*.
- Y. Teh, M. Jordan, M. Beal, and D. Blei. 2005. Hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA.
- Y. Teh. 2006. A Bayesian interpretation of interpolated kneser-ney. Technical Report TRA2/06, National University of Singapore, School of Computing.
- A. Venkataraman. 2001. A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27(3):351–372.

Appendix

To sample from the posterior distribution over segmentations in the bigram model, we define h_1 and h_2 as we did in the unigram sampler so that for the corpus substring s , h_1 has a single word ($s = w_1$) where h_2 has two ($s = w_2.w_3$). Let w_l and w_r be the words (or \$) preceding and following s . Then the posterior probabilities of h_1 and h_2 are given in Figure 6. $P_1(\cdot)$ can be calculated exactly using the equation in Section 4.1, but this requires explicitly tracking and sampling the assignment of words to tables. For easier and more efficient implementation, we use an approximation, replacing each table count t_{w_i} by its expected value $E[t_{w_i}]$. In a $DP(\alpha, P)$, the expected number of CRP tables for an item occurring n times is $\alpha \log \frac{n+\alpha}{\alpha}$ (Antoniuk, 1974), so

$$E[t_{w_i}] = \alpha_1 \sum_j \log \frac{n_{(w_j, w_i)} + \alpha_1}{\alpha_1}$$

This approximation requires only the bigram counts, which we must track anyway.

MAGEAD: A Morphological Analyzer and Generator for the Arabic Dialects

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 MAGEAD, a morphological analyzer and generator for the Arabic language family. Our work is novel in that it explicitly addresses the need for processing the morphology of the dialects. MAGEAD performs an on-line analysis to or generation from a root+pattern+features representation, it has separate phonological and orthographic representations, and it allows for combining morphemes from different dialects. We present a detailed evaluation of MAGEAD.

1 Introduction

In this paper we present MAGEAD, a morphological analyzer and generator for the Arabic language family, by which we mean both Modern Standard Arabic (MSA) and the spoken dialects.¹ Our work is novel in that it explicitly addresses the need for processing the morphology of the dialects as well.

The principal theoretical contribution of this paper is an organization of morphological knowledge for processing multiple variants of one language family. The principal practical contribution is the first morphological analyzer and generator for an Arabic dialect that includes a root-and-pattern analysis (which is also the first wide-coverage implementation of root-and-pattern morphology for any language using a multitape finite-state machine). We also provide a novel type of detailed evaluation in which we investigate how

different sources of lexical information affect performance of morphological analysis.

This paper is organized as follows. In Section 2, we present the relevant facts about morphology in the Arabic language family. Previous work is summarized in Section 3. We present our design goals in Section 4, and then discuss our approach to representing linguistic knowledge for morphological analysis in Section 5. The implementation is sketched in Section 6. We outline the steps involved in creating a Levantine analyzer in Section 7. We evaluate our system in Section 8, and then conclude.

2 Arabic Morphology

2.1 Variants of Arabic

The Arabic-speaking world is characterized by diglossia (Ferguson, 1959). Modern Standard Arabic (MSA) is the shared written language from Morocco to the Gulf, but it is not a native language of anyone. It is spoken only in formal, scripted contexts (news, speeches). In addition, there is a continuum of spoken dialects (varying geographically, but also by social class, gender, etc.) which are native languages, but rarely written (except in very informal contexts: collections of folk tales, newsgroups, email, etc). We will refer to MSA and the dialects as **variants** of Arabic. Variants differ phonologically, lexically, morphologically, and syntactically from one another; many pairs of variants are mutually unintelligible. In unscripted situations where spoken MSA would normally be required (such as talk shows on TV), speakers usually resort to repeated code-switching between their dialect and MSA, as nearly all native speakers of Arabic are unable to produce sustained spontaneous discourse in MSA.

¹We would like to thank several anonymous reviewers for comments that helped us improve this paper. The work reported in this paper was supported by NSF Award 0329163, with additional work performed under the DARPA GALE program, contract HR0011-06-C-0023. The authors are listed in alphabetical order.

In this paper, we discuss MSA and Levantine, the dialect spoken (roughly) in Syria, Lebanon, Jordan, Palestine, and Israel. Our Levantine data comes from Jordan. The discussion in this section uses only examples from MSA, but all variants show a combination of root-and-pattern and affixational morphology and similar examples could be found for Levantine.

2.2 Roots, Patterns and Vocalism

Arabic morphemes fall into three categories: templatic morphemes, affixational morphemes, and non-templatic word stems (NTWSs). NTWSs are word stems that are not constructed from a root/pattern/vocalism combination. Verbs are never NTWSs.

Templatic morphemes come in three types that are equally needed to create a word stem: roots, patterns and vocalisms. The root morpheme is a sequence of three, four, or five consonants (termed *radicals*) that signifies some abstract meaning shared by all its derivations. For example, the words *كتب* *katab* ‘to write’, *كاتب* *kaAtib* ‘writer’, and *مكتوب* *maktuwb* ‘written’ all share the root morpheme *ktb* (ك ت ب) ‘writing-related’. The pattern morpheme is an abstract template in which roots and vocalisms are inserted. The vocalism morpheme specifies which short vowels to use with a pattern. We will represent the pattern as a string made up of numbers to indicate radical position, of the symbol *V* to indicate the position of the vocalism, and of pattern consonants (if needed).

A word stem is constructed by interleaving the three types of templatic morphemes. For example, the word stem *كتب* *katab* ‘to write’ is constructed from the root *ktb* (ك ت ب), the pattern *1V2V3* and the vocalism *aa*.

2.3 Affixational Morphemes

Arabic affixes can be prefixes such as *sa+* (سـ) ‘will/[future]’, suffixes such as *+uwna* (ونـ) ‘[masculine plural]’ or circumfixes such as *ta++na* (تـ++نـ) ‘[imperfective subject 2nd person fem. plural]’. Multiple affixes can appear in a word. For example, the word *وسيكتبونها* *wasayaktubuwnahA* ‘and they will write it’ has two prefixes, one circumfix and one suffix:²

- (1) *wasayaktubuwnahA*
wa+ sa+ y+ aktub +uwna
 and will 3person write masculine-plural
+hA
 it

2.4 Morphological Rewrite Rules

An Arabic word is constructed by first creating a word stem from templatic morphemes or by using a NTWS. Affixational morphemes are then added to this stem. The process of combining morphemes involves a number of phonological, morphemic and orthographic rules that modify the form of the created word so it is not a simple interleaving or concatenation of its morphemic components.

An example of a phonological rewrite rule is the voicing of the /t/ of the verbal pattern *V1tV2V3* (Form VIII) when the first root radical is /z/, /d/, or /*/ (ز, د, or ذ): the verbal stem *zhr+V1tV2V3+iaa* is realized phonologically as /izdahar/ (orthographically: ازدهر) ‘flourish’ not /iztahar/ (orthographically: ازتهر). An example of an orthographic rewrite rule is the deletion of the Alif (l) of the definite article morpheme *Al+* (+ال) in nouns when preceded by the preposition *l+* (+ل).

3 Previous Work

There has been a considerable amount of work on Arabic morphological analysis; for an overview, see (Al-Sughaiyer and Al-Kharashi, 2004). We summarize some of the most relevant work here.

Kataja and Koskeniemi (1988) present a system for handling Akkadian root-and-pattern morphology by adding an additional lexicon component to Koskeniemi’s two-level morphology (1983). The first large scale implementation of Arabic morphology within the constraints of finite-state methods is that of Beesley et al. (1989) with a ‘detouring’ mechanism for access to multiple lexica, which gives rise to other works by Beesley (Beesley, 1998) and, independently, by Buckwalter (2004).

The approach of McCarthy (1981) to describing root-and-pattern morphology in the framework of autosegmental phonology has given rise to a number of computational proposals. Kay (1987) proposes a framework with which each of the autosegmental tiers is assigned a tape in a multi-tape finite state machine, with an additional tape for the surface form. Kiraz (2000,2001) extends Kay’s

²We analyze the imperfective word stem as including an initial short vowel, and leave a discussion of this analysis to future publications.

approach and implements a small working multi-tape system for MSA and Syriac. Other autosegmental approaches (described in more details in Kiraz 2001 (Chapter 4)) include those of Kornai (1995), Bird and Ellison (1994), Pulman and Hople (1993), whose formalism Kiraz adopts, and others.

4 Design Goals for MAGEAD

This work is aimed at a unified processing architecture for the morphology of all variants of Arabic, including the dialects. Three design goals follow from this overall goal:

- First, we want to be able to use the analyzer when we do not have a lexicon, or only a partial lexicon. This is because, despite the similarities between dialects at the morphological and lexical levels, we do cannot assume we have a complete lexicon for every dialect we wish to morphologically analyze. As a result, we want an **on-line analyzer** which performs full morphological analysis at run time.

- Second, we want to be able to exploit the existing regularities among the variants, in particular systematic sound changes which operate at the level of the radicals, and pattern changes. This requires an **explicit analysis into root and pattern**.

- Third, the dialects are mainly used in spoken communication and in the rare cases when they are written they do not have standard orthographies, and different (inconsistent) orthographies may be used even within a single written text. We thus need a representation of morphology that incorporates **models of both phonology and orthography**.

In addition, we add two general requirements for morphological analyzers. First, we want both a morphological analyzer and a morphological generator. Second, we want to use a representation that is defined in terms of a lexeme and attribute-value pairs for morphological features such as **aspect** or **person**. This is because we want our component to be usable in natural language processing (NLP) applications such as natural language generation and machine translation, and the lexeme provides a usable lexicographic abstraction. Note that the second general requirement (an analysis to a lexemic representation) appears to clash with the first design desideratum (we may not have a lexicon).

We tackle these requirements by doing a full

analysis of templatic morphology, rather than “precompiling” the templatic morphology into stems and only analyzing affixational morphology on-line (as is done in (Buckwalter, 2004)). Our implementation uses the multitape approach of Kiraz (2000). This is the first large-scale implementation of that approach. We extend it by adding an additional tape for independently modeling phonology and orthography. The use of finite state technology makes MAGEAD usable as a generator as well as an analyzer, unlike some morphological analyzers which cannot be converted to generators in a straightforward manner (Buckwalter, 2004; Habash, 2004).

5 The MAGEAD System: Representation of Linguistic Knowledge

MAGEAD relates (bidirectionally) a lexeme and a set of linguistic features to a surface word form through a sequence of transformations. In a generation perspective, the features are translated to abstract morphemes which are then ordered, and expressed as concrete morphemes. The concrete templatic morphemes are interdigitated and affixes added, and finally morphological and phonological rewrite rules are applied. In this section, we discuss our organization of linguistic knowledge, and give some examples; a more complete discussion of the organization of linguistic knowledge in MAGEAD can be found in (Habash et al., 2006).

5.1 Morphological Behavior Classes

Morphological analyses are represented in terms of a lexeme and features. We define the **lexeme** to be a triple consisting of a root (or an NTWS), a meaning index, and a **morphological behavior class** (MBC). We do not deal with issues relating to word sense here and therefore do not further discuss the meaning index. It is through this view of the lexeme (which incorporates productive derivational morphology without making claims about semantic predictability) that we can both have a lexeme-based representation, and operate without a lexicon. In fact, because lexemes have internal structure, we can hypothesize lexemes on the fly without having to make wild guesses (we know the pattern, it is only the root that we are guessing). We will see in Section 8 that this approach does not wildly overgenerate.

We use as our example the surface form **أزدهرات** *Azdaharat* (*Azdhrt* without diacritics)

‘she/it flourished’. The lexeme-and-features representation of this word form is as follows:

- (2) Root:zhr MBC:verb-VIII POS:V PER:3
GEN:F NUM:SG ASPECT:PERF

An MBC maps sets of linguistic feature-value pairs to sets of abstract morphemes. For example, MBC verb-VIII maps the feature-value pair ASPECT:PERF to the abstract root morpheme [PAT_PV:VIII], which in MSA corresponds to the concrete root morpheme AV1tV2V3, while the MBC verb-I maps ASPECT:PERF to the abstract root morpheme [PAT_PV:I], which in MSA corresponds to the concrete root morpheme 1V2V3. We define MBCs using a hierarchical representation with non-monotonic inheritance. The hierarchy allows us to specify only once those feature-to-morpheme mappings for all MBCs which share them. For example, the root node of our MBC hierarchy is a word, and all Arabic words share certain mappings, such as that from the linguistic feature conj:w to the clitic w+. This means that all Arabic words can take a cliticized conjunction. Similarly, the object pronominal clitics are the same for all transitive verbs, no matter what their templatic pattern is. We have developed a specification language for expressing MBC hierarchies in a concise manner. Our hypothesis is that the MBC hierarchy is variant-independent, though as more variants are added, some modifications may be needed. Our current MBC hierarchy specification for both MSA and Levantine, which covers only the verbs, comprises 66 classes, of which 25 are abstract, i.e., only used for organizing the inheritance hierarchy and never instantiated in a lexeme.

5.2 Ordering and Mapping Abstract and Concrete Morphemes

To keep the MBC hierarchy variant-independent, we have also chosen a variant-independent representation of the morphemes that the MBC hierarchy maps to. We refer to these morphemes as **abstract morphemes** (AMs). The AMs are then ordered into the surface order of the corresponding concrete morphemes. The ordering of AMs is specified in a variant-independent context-free grammar. At this point, our example (2) looks like this:

- (3) [Root:zhr][PAT_PV:VIII]
[VOC_PV:VIII-act] + [SUBJSUF_PV:3FS]

Note that as the root, pattern, and vocalism are not ordered with respect to each other, they are simply juxtaposed. The ‘+’ sign indicates the ordering of affixational morphemes. Only now are the AMs translated to **concrete morphemes** (CMs), which are concatenated in the specified order. Our example becomes:

- (4) <zhr,AV1tV2V3,iaa> +at

The interdigitation of root, pattern and vocalism then yields the form *Aiztahar+at*.

5.3 Morphological, Phonological, and Orthographic Rules

We have two types of rules. **Morphophonemic/phonological rules** map from the morphemic representation to the phonological and orthographic representations. This includes default rules which copy roots and vocalisms to the phonological and orthographic tiers, and specialized rules to handle hollow verbs (verbs with a glide as their middle radical), or more specialized rules for cases such as the pattern consonant change in Form VIII (the /t/ of the pattern changes to a /d/ if the first radical is /z/, /d/, or /*/; this rule operates in our example). For MSA, we have 69 rules of this type.

Orthographic rules rewrite only the orthographic representation. These include, for examples, rules for using the *shadda* (consonant doubling diacritic). For MSA, we have 53 such rules.

For our example, we get /izdaharat/ at the phonological level. Using standard MSA diacritized orthography, our example becomes *Aizdaharat* (in transliteration). Removing the diacritics turns this into the more familiar *Azdhrt* از دهرت. Note that in analysis mode, we hypothesize all possible diacritics (a finite number, even in combination) and perform the analysis on the resulting multi-path automaton.

6 The MAGEAD System: Implementation

We follow (Kiraz, 2000) in using a multitape representation. We extend the analysis of Kiraz by introducing a fifth tier. The five tiers are used as follows: Tier 1: pattern and affixational morphemes; Tier 2: root; Tier 3: vocalism; Tier 4: phonological representation; Tier 5: orthographic representation. In the generation direction, tiers 1 through 3 are always input tiers. Tier 4 is first an output tier, and subsequently an input tier. Tier 5 is always an output tier.

We have implemented multi-tape finite state automata as a layer on top of the AT&T two-tape finite state transducers (Mohri et al., 1998). We have defined a specification language for the higher multitape level, the new **Morphtools format**. Specification in the Morphtools format of different types of information such as rules or context-free grammars for morpheme ordering are compiled to the appropriate Lextools format (an NLP-oriented extension of the AT&T toolkit for finite-state machines, (Sproat, 1995)). For reasons of space, we omit a further discussion of Morphtools. For details, see (Habash et al., 2005).

7 From MSA to Levantine

We modified MAGEAD so that it accepts Levantine rather than MSA verbs. Our effort concentrated on the orthographic representation; to simplify our task, we used a diacritic-free orthography for Levantine developed at the Linguistic Data Consortium (Maamouri et al., 2006). Changes were done only to the representations of linguistic knowledge at the four levels discussed in Section 5, not to the processing engine.

Morphological Behavior Classes: The MBCs are variant-independent, so in theory no changes needed to be implemented. However, as Levantine is our first dialect, we expand the MBCs to include two AMs not found in MSA: the aspectual particle and the postfix negation marker.

Abstract Morpheme Ordering: The context-free grammar representing the ordering of AMs needed to be extended to order the two new AMs, which was straightforward.

Mapping Abstract to Concrete Morphemes: This step requires four types of changes to a table representing this mapping. In the first category, the new AMs require mapping to CMs. Second, those AMs which do not exist in Levantine need to be mapped to zero (or to an error value). These are dual number, and subjunctive and jussive moods. Third, in Levantine some AMs allow additional CMs in allomorphic variation with the same CMs as seen in MSA. This affects three object clitics; for example, the second person masculine plural, in addition to كُمْ +*kum* (also found in MSA), also can be كُوا +*kuwA*. Fourth, in five cases, the subject suffix in the imperfective is simply different for Levantine. For example, the second person feminine singular indicative imperfective suffix changes from يِن +*iyina* in MSA to يِي +*iy* in

Levantine. Note that more changes in CMs would be required were we completely modeling Levantine phonology (i.e., including the short vowels).

Morphological, Phonological, and Orthographic Rules. We needed to change one rule, and add one. In MSA, the vowel between the second and third radical is deleted when they are identical (“gemination”) only if the third radical is followed by a suffix starting with a vowel. In Levantine, in contrast, gemination always happens, independently of the suffix. If the suffix starts with a consonant, a long /e/ is inserted *after* the third radical. The new rule deletes the first person singular subject prefix for the imperfective, +¹A+, when it is preceded by the aspectual marker +₂b+.

We summarize now the expertise required to convert MSA resources to Levantine, and we comment on the amount of work needed for adding a further dialect. We modified the MBC hierarchy, but only minor changes were needed. We expect only one major further change to the MBCs, namely the addition of an indirect object clitic (since the indirect object in some dialects is sometimes represented as an orthographic clitic). The AM ordering can be read off from examples in a fairly straightforward manner; the introduction of an indirect object AM would, for example, require an extension of the ordering specification. The mapping from AMs to CMs, which is variant-specific, can be obtained easily from a linguistically trained (near-)native speaker or from a grammar handbook, and with a little more effort from an informant. Finally, the rules, which again can be variant-specific, require either a good morphophonological treatise for the dialect, a linguistically trained (near-)native speaker, or extensive access to an informant. In our case, the entire conversion from MSA to Levantine was performed by a native speaker linguist in about six hours.

8 Evaluation

The goal of the evaluation is primarily to investigate how reduced lexical resources affect the performance of morphological analysis, as we will not have complete lexicons for the dialects. A second goal is to validate MAGEAD in analysis mode by comparing it to the Buckwalter analyzer (Buckwalter, 2004) when MAGEAD has a full lexicon at its disposal. Because of the lack of resources for the dialects, we use primarily MSA for both goals, but we also discuss a more modest evaluation on a

Levantine corpus.

We first discuss the different sources of lexical knowledge, and then present our evaluation metrics. We then separately evaluate MSA and Levantine morphological analysis.

8.1 Lexical Knowledge Sources

We evaluate the following sources of lexical knowledge on what roots, i.e., combinations of radicals, are possible. Except for **all**, these are lists of attested verbal roots. It is not a trivial task to compile a list of verbal roots for MSA, and we compare different sources for these lists.

- **all**: All radical combinations are allowed, we use no lexical knowledge at all.
- **dar**: List of roots extracted by (Darwish, 2003) from *Lisan Al'arab*, a large Arabic dictionary.
- **bwl**: A list of roots appearing as comments in the Buckwalter lexicon (Buckwalter, 2004).
- **lex**: Roots extracted by us from the list of lexeme citation forms in the Buckwalter lexicon using surfacy heuristics for quick-and-dirty morphological analysis.
- **mbc**: This is the same list as **lex**, except that we pair each root with the MBCs with which it was seen in the Buckwalter lexicon (recall that for us, a lexeme is a root with an MBC). Note that **mbc** represents a full lexicon, though it was converted automatically from the Buckwalter lexicon and it has not been hand-checked.

8.2 Test Corpora and Metrics

For development and testing purposes, we use MSA and Levantine. For MSA, we use the Penn Arabic Treebank (ATB) (Maamouri et al., 2004). The morphological annotation we use is the “before-file”, which lists the untokenized words (as they appear in the Arabic original text) and all possible analyses according to the Buckwalter analyzer (Buckwalter, 2004). The analysis which is correct for the given token in its context is marked; sometimes, it is also hand-corrected (or added by hand), while the contextually incorrect analyses are never hand-corrected. For development, we use ATB1 section 20000715, and for testing, Sections 20001015 and 20001115 (13,885 distinct verbal types).

For Levantine, we use a similarly annotated corpus, the Levantine Arabic Treebank (LATB) from the Linguistic Data Consortium. However, there are three major differences: the text is transcribed

speech, the corpus is much smaller, and, since, there is no morphological analyzer for Levantine currently, the before-files are the result of running the MSA Buckwalter analyzer on the Levantine token, with many of the analyses incorrect, and only the analysis chosen for the token in context usually hand-corrected. We use LATB files fsa_16* for development, and for testing, files fsa_17*, fsa_18* (14 conversations, 3,175 distinct verbal types).

We evaluate using three different metrics. The token-based metrics are the corresponding type-based metric weighted by the number of occurrences of the type in the test corpus.

- **Recall (TyR for type recall, ToR for token recall)**: what proportion of the analyses in the gold standard does MAGEAD get?
- **Precision (TyP for type precision, ToP for token precision)**: what proportion of the analyses that MAGEAD gets are also in the gold standard?
- **Context token recall (CToR)**: how often does MAGEAD get the contextually correct analysis for that token?

We do not give context precision figures, as MAGEAD does not determine the contextually correct analysis (this is a tagging problem). Rather, we interpret the context recall figures as a measure of how often MAGEAD gets the most important of the analyses (i.e., the correct one) for each token.

	Roots	TyR	TyP	ToR	ToP	CToR
all	21952	98.5	44.8	98.6	36.9	97.9
dar	10377	98.1	50.5	98.3	43.3	97.7
bwl	6450	96.7	52.2	97.2	42.9	96.7
lex	3658	97.3	55.6	97.3	49.2	97.5
mbc	3658	96.1	63.5	95.8	59.4	96.4

Figure 1: Results comparing MAGEAD to the Buckwalter Analyzer on MSA for different root restrictions, and for different metrics; “Roots” indicates the number of possible roots for that restriction; all numbers are percent figures

8.3 Quantitative Analysis: MSA

The results are summarized in Figure 1. We see that we get a (rough) recall-precision trade-off, both for types and for tokens: the more restrictive we are, the higher our precision, but recall declines. For **all**, we get excellent recall, and an overgeneration by a factor of only 2. This performance, assuming it is roughly indicative of dialect performance, allows us to conclude that we can use MAGEAD as a dialect morphological analyzer without a lexicon.

For the root lists, we see that precision is al-

ways higher than for **all**, as many false analyses are eliminated. At the same time, some correct analyses are also eliminated. Furthermore, **bwl** under performs somewhat. The change from **lex** to **mbc** is interesting, as **mbc** is a true lexicon (since it does not only state which roots are possible, but also what their MBC is). Precision increases substantially, but not as much as we had hoped. We investigate the errors of **mbc** in the next subsection in more detail.

8.4 Qualitative Analysis: MSA

The gold standard we are using has been generated automatically using the Buckwalter analyzer. Only the contextually correct analysis has been hand-checked. As a result, our quantitative analysis in Section 8.3 leaves open the question of how good the gold standard is in the first place. We analyzed all of the 2,536 false positives (types) produced by MAGEAD on our development set (analyses it suggested, but which the Test corpus did not have). In 75% of the errors, the Buckwalter analyzer does not provide a passive voice analysis which differs from the active voice one only in diacritics which are not written. 7% are cases where Buckwalter does not make distinctions that MAGEAD makes (e.g. mood variations that are not phonologically realized); in 4.4% of the errors a correct analysis was created but it was not produced by Buckwalter for various reasons. If we count these cases as true positives rather than as false positives (as in the case in Figure 1) and take type frequency into account, we obtain a token precision rate of 94.9% on the development set.

The remaining cases are MAGEAD errors. 3.3% are missing rules to handle special cases such as jussive mood interaction with weak radicals; 5.4% are incorrect combinations of morphemes such as passive voice and object pronouns; 2.6% of the errors are cases of pragmatic overgeneration such as second person masculine subjects with a second person feminine plural object. 1.5% of the errors are errors of the mbc-root list and 1.2% are other errors. A large number of these errors are fixable errors.

There were 162 false negatives (gold standard analyses MAGEAD did not get). 65.4% of these errors were a result of the use of the **mbc** list restriction. The rest of the errors are all a result of unhandled phenomena in MAGEAD: quadrilat-

eral roots (13.6%), imperatives (8%), and specific missing rules/ rule failures (13%) (e.g., for handling some weak radicals/hamza cases, pattern IX gemination-like behavior, etc.).

We conclude that we can claim that our precision numbers are actually much higher, and that we can further improve them by adding more rules and knowledge to MAGEAD.

8.5 Quantitative and Qualitative Analysis: Levantine

For the Levantine, we do not have a list of all possible analyses for each word in the gold standard: only the contextually appropriate analysis is hand-checked. We therefore only report context recall in Figure 2. As a baseline, we report the MSA MAGEAD with the **all** restriction applied to the same Levantine test corpus. As we can see, the MSA system performs poorly on Levantine input. The Levantine system we use is the one described in Section 7. We use the resulting analyzer with the **all** option as we have no information on roots in Levantine. MAGEAD with Levantine knowledge does well, missing only one in 20 contextually correct analyses. We take this to mean that the architecture of MAGEAD allows us to port MAGEAD fairly rapidly to a new dialect and to perform adequately well on the most important analysis for each token, the contextually relevant one.

System	CTyR	CToR
MSA-all	52.9	60.4
LEV-all	95.4	94.2

Figure 2: Results on Levantine; MSA-all is a baseline

For the Levantine MAGEAD, there were 25 errors, cases of contextually selected analyses that MAGEAD did not get (false negatives). Most of these are related to phenomena that MAGEAD doesn't currently handle: imperatives (48%) (which are much more common in speech corpora) and quadrilateral roots (8%). There were four cases (16%) of an unhandled variant spelling of an object pronoun and 7 cases (28%) of hamza/weak radical rule errors.

9 Outlook

We have described a morphological analyzer for Arabic and its dialects which decomposes word forms into the templatic morphemes and relates

morphemes to strings. We have evaluated the current state of the implementation both for MSA and for Levantine, both quantitatively and in a detailed error analysis, and have shown that we have met our design objectives of having a flexible analyzer which can be used on a new dialect in the absence of a lexicon and with a restrained amount of manual knowledge engineering needed.

In ongoing work, we are populating MAGEAD with more knowledge (morphemes and rules) for MSA nouns and other parts of speech, for more of Levantine, and for more dialects. We intend to include a full phonological representation for Levantine (including short vowels). In future work, we will investigate the derivation of words with morphemes from more than one variant (code switching). We will also investigate ways of using morphologically tagged corpora to assign weights to the arcs in the transducer so that the analyses returned by MAGEAD are ranked.

References

- Imad A. Al-Sughaiyer and Ibrahim A. Al-Kharashi. 2004. Arabic morphological analysis techniques: A comprehensive survey. *Journal of the American Society for Information Science and Technology*, 55(3):189–213.
- K. Beesley, T. Buckwalter, and S. Newton. 1989. Two-level finite-state analysis of Arabic morphology. In *Proceedings of the Seminar on Bilingual Computing in Arabic and English*, page n.p.
- K. Beesley. 1998. Arabic morphology using only finite-state operations. In M. Rosner, editor, *Proceedings of the Workshop on Computational Approaches to Semitic Languages*, pages 50–7, Montetereal.
- S. Bird and T. Ellison. 1994. One-level phonology. *Computational Linguistics*, 20(1):55–90.
- Tim Buckwalter. 2004. Buckwalter Arabic morphological analyzer version 2.0.
- Kareem Darwish. 2003. Building a shallow Arabic morphological analyser in one day. In *ACL02 Workshop on Computational Approaches to Semitic Languages*, Philadelphia, PA. Association for Computational Linguistics.
- Charles F Ferguson. 1959. Diglossia. *Word*, 15(2):325–340.
- Nizar Habash, Owen Rambow, and Gerooge Kiraz. 2005. Morphological analysis and generation for arabic dialects. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, Ann Arbor, MI.
- Nizar Habash, Owen Rabmow, and Richard Sproat. 2006. The representation of linguistic knowledge in a pan-Arabic morphological analyzer. Paper under preparation, Columbia University and UIUC.
- Nizar Habash. 2004. Large scale lexeme based arabic morphological generation. In *Proceedings of Traitement Automatique du Langage Naturel (TALN-04)*. Fez, Morocco.
- L. Kataja and K. Koskeniemi. 1988. Finite state description of Semitic morphology. In *COLING-88: Papers Presented to the 12th International Conference on Computational Linguistics*, volume 1, pages 313–15.
- Martin Kay. 1987. Nonconcatenative finite-state morphology. In *Proceedings of the Third Conference of the European Chapter of the Association for Computational Linguistics*, pages 2–10.
- George Anton Kiraz. 2000. Multi-tiered nonlinear morphology using multi-tape finite automata: A case study on Syriac and Arabic. *Computational Linguistics*, 26(1):77–105.
- George Kiraz. 2001. *Computational Nonlinear Morphology: With Emphasis on Semitic Languages*. Cambridge University Press.
- A. Kornai. 1995. *Formal Phonology*. Garland Publishing.
- K. Koskeniemi. 1983. *Two-Level Morphology*. Ph.D. thesis, University of Helsinki.
- 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, Tim Buckwalter, Mona Diab, Nizar Habash, Owen Rambow, and Dalila Tabessi. 2006. Developing and using a pilot dialectal arabic treebank. In *Proceedings of LREC*, Genoa, Italy.
- John McCarthy. 1981. A prosodic theory of nonconcatenative morphology. *Linguistic Inquiry*, 12(3):373–418.
- M. Mohri, F. Pereira, and M. Riley. 1998. A rational design for a weighted finite-state transducer library. In D. Wood and S. Yu, editors, *Automata Implementation*, Lecture Notes in Computer Science 1436, pages 144–58. Springer.
- S. Pulman and M. Hepple. 1993. A feature-based formalism for two-level phonology: a description and implementation. *Computer Speech and Language*, 7:333–58.
- Richard Sproat. 1995. Lextools: Tools for finite-state linguistic analysis. Technical Report 11522-951108-10TM, Bell Laboratories.

Noun Phrase Chunking in Hebrew

Influence of Lexical and Morphological Features

Yoav Goldberg and Meni Adler and Michael Elhadad

Computer Science Department

Ben Gurion University of the Negev

P.O.B 653 Be'er Sheva 84105, Israel

{yoavg, adlerm, elhadad}@cs.bgu.ac.il

Abstract

We present a method for Noun Phrase chunking in Hebrew. We show that the traditional definition of base-NPs as non-recursive noun phrases does not apply in Hebrew, and propose an alternative definition of Simple NPs. We review syntactic properties of Hebrew related to noun phrases, which indicate that the task of Hebrew SimpleNP chunking is harder than base-NP chunking in English. As a confirmation, we apply methods known to work well for English to Hebrew data. These methods give low results (F from 76 to 86) in Hebrew. We then discuss our method, which applies SVM induction over lexical and morphological features. Morphological features improve the average precision by ~0.5%, recall by ~1%, and F-measure by ~0.75, resulting in a system with average performance of 93% precision, 93.4% recall and 93.2 F-measure.*

1 Introduction

Modern Hebrew is an agglutinative Semitic language, with rich morphology. Like most other non-European languages, it lacks NLP resources and tools, and specifically there are currently no available syntactic parsers for Hebrew. We address the task of NP chunking in Hebrew as a

first step to fulfill the need for such tools. We also illustrate how this task can successfully be approached with little resource requirements, and indicate how the method is applicable to other resource-scarce languages.

NP chunking is the task of labelling noun phrases in natural language text. The input to this task is free text with part-of-speech tags. The output is the same text with brackets around base noun phrases. A base noun phrase is an NP which does not contain another NP (it is not recursive). NP chunking is the basis for many other NLP tasks such as shallow parsing, argument structure identification, and information extraction

We first realize that the definition of base-NPs must be adapted to the case of Hebrew (and probably other Semitic languages as well) to correctly handle its syntactic nature. We propose such a definition, which we call *simple NPs* and assess the difficulty of chunking such NPs by applying methods that perform well in English to Hebrew data. While the syntactic problem in Hebrew is indeed more difficult than in English, morphological clues do provide additional hints, which we exploit using an SVM learning method. The resulting method reaches performance in Hebrew comparable to the best results published in English.

2 Previous Work

Text chunking (and NP chunking in particular), first proposed by Abney (1991), is a well studied problem for English. The CoNLL2000 shared task (Tjong Kim Sang et al., 2000) was general chunking. The best result achieved for the shared task data was by Zhang et al (2002), who achieved NP chunking results of 94.39% precision, 94.37% recall and 94.38 F-measure using a

* This work was funded by the Israel Ministry of Science and Technology under the auspices of the Knowledge Center for Processing Hebrew. Additional funding was provided by the Lynn and William Frankel Center for Computer Sciences.

generalized Winnow algorithm, and enhancing the feature set with the output of a dependency parser. Kudo and Matsumoto (2000) used an SVM based algorithm, and achieved NP chunking results of 93.72% precision, 94.02% recall and 93.87 F-measure for the same shared task data, using only the words and their PoS tags. Similar results were obtained using Conditional Random Fields on similar features (Sha and Pereira, 2003).

The NP chunks in the shared task data are base-NP chunks – which are non-recursive NPs, a definition first proposed by Ramshaw and Marcus (1995). This definition yields good NP chunks for English, but results in very short and uninformative chunks for Hebrew (and probably other Semitic languages).

Recently, Diab et al (2004) used SVM based approach for Arabic text chunking. Their chunks data was derived from the LDC Arabic TreeBank using the same program that extracted the chunks for the shared task. They used the same features as Kudo and Matsumoto (2000), and achieved over-all chunking performance of 92.06% precision, 92.09% recall and 92.08 F-measure (The results for NP chunks alone were not reported). Since Arabic syntax is quite similar to Hebrew, we expect that the issues reported below apply to Arabic results as well.

3 Hebrew Simple NP Chunks

The standard definition of English base-NPs is any noun phrase that does not contain another noun phrase, with possessives treated as a special case, viewing the possessive marker as the first word of a new base-NP (Ramshaw and Marcus, 1995). To evaluate the applicability of this definition to Hebrew, we tested this definition on the Hebrew TreeBank (Sima'an *et al*, 2001) published by the Hebrew Knowledge Center. We extracted all base-NPs from this TreeBank, which is similar in genre and contents to the English one. This results in extremely simple chunks.

	English BaseNPs	Hebrew BaseNPs	Hebrew SimpleNPs
Avg # of words	2.17	1.39	2.49
% length 1	30.95	63.32	32.83
% length 2	39.35	35.48	32.12
% length 3	18.68	0.83	14.78
% length 4	6.65	0.16	9.47
% length 5	2.70	0.16	4.56
% length > 5	1.67	0.05	6.22

Table 1. Size of Hebrew and English NPs

Table 1 shows the average number of words in a base-NP for English and Hebrew. The Hebrew chunks are basically one-word groups around Nouns, which is not useful for any practical purpose, and so we propose a new definition for Hebrew NP chunks, which allows for some nestedness. We call our chunks Simple NP chunks.

3.1 Syntax of NPs in Hebrew

One of the reasons the traditional base-NP definition fails for the Hebrew TreeBank is related to syntactic features of Hebrew – specifically, *smixut* (construct state – used to express noun compounds), definite marker and the expression of possessives. These differences are reflected to some extent by the tagging guidelines used to annotate the Hebrew Treebank and they result in trees which are in general less flat than the Penn TreeBank ones.

Consider the example base noun phrase [*The homeless people*]. The Hebrew equivalent is

(1) האנשים מחוסרי הבית

which by the non-recursive NP definition will be bracketed as:

[האנשים] [מחוסרי הבית], or, loosely translating back to English: [*the home*]*less* [*people*].

In this case, the fact that the bound-morpheme *less* appears as a separate construct state word with its own definite marker (*ha-*) in Hebrew would lead the chunker to create two separate NPs for a simple expression. We present below syntactic properties of Hebrew which are relevant to NP chunking. We then present our definition of Simple NP Chunks.

Construct State: The Hebrew genitive case is achieved by placing two nouns next to each other. This is called “noun construct”, or *smixut*. The semantic interpretation of this construct is varied (Netzer and Elhadad, 1998), but it specifically covers possession. The second noun can be treated as an adjective modifying the next noun. The first noun is morphologically marked in a form known as the *construct form* (denoted by *const*). The definite article marker is placed on the second word of the construction:

(2) בית ספר

beit sefer / house-[const] book

School

(3) בית הספר

beit ha-sefer / house-[const] the-book

The school

The construct form can also be embedded:

(4) משרד ראש הממשלה

misrad ro\$ ha-mem\$ala
Office-[const poss] head-[const] the-government
The prime-minister's office

Possessive: the *smixut* form can be used to indicate possession. Other ways to express possession include the possessive marker של - '\$el' / 'of' - (5), or adding a possessive suffix on the noun (6). The various forms can be mixed together, as in (7):

(5) הבית שלי
ha-bait \$el-i / the-house of-[poss 1st person]
My house
(6) ביתי
beit-i / house-[poss 1st person]
My house
(7) משרדו של ראש הממשלה
misrad-o \$el ro\$ ha-mem\$ala
Office-[poss 3rd] of head-[const] the-government
The prime minister office

Adjective: Hebrew adjectives come after the noun, and agree with it in number, gender and definite marker:

(8) התפוח הירוק
ha-tapu'ah ha-yarok / the-Apple the-green
The green apple

Some aspects of the predicate structure in Hebrew directly affect the task of NP chunking, as they make the decision to “split” NPs more or less difficult than in English.

Word order and the preposition 'et': Hebrew sentences can be either in SVO or VSO form. In order to keep the object separate from the subject, definite direct objects are marked with the special preposition 'et', which has no analog in English.

Possible null equative: The equative form in Hebrew can be null. Sentence (9) is a non-null equative, (10) a null equative, while (11) and (12) are predicative NPs, which look very similar to the null-equative form:

(9) הבית הוא גדול
ha-bait hu gadol
The-house is big
The house is big

(10) הבית גדול
ha-bait gadol
The-house big
The house is big

(11) בית גדול
bait gadol
House big
A big house

(12) הבית הגדול
ha-bait ha-gadol
The-house the-big
The big house

Morphological Issues: In Hebrew morphology, several lexical units can be concatenated into a single textual unit. Most prepositions, the definite article marker and some conjunctions are concatenated as prefixes, and possessive pronouns and some adverbs are concatenated as suffixes. The Hebrew Treebank is annotated over a segmented version of the text, in which prefixes and suffixes appear as separate lexical units. On the other hand, many bound morphemes in English appear as separate lexical units in Hebrew. For example, the English morphemes *re-*, *ex-*, *un-*, *-less*, *-like*, *-able*, appear in Hebrew as separate lexical units – לחסר, לשעבר, מחדש, דמוי, חסר, לא/אי/בלתי, לסר, בלתי, לא, בר

In our experiment, we use as input to the chunker the text after it has been morphologically disambiguated and segmented. Our analyzer provides segmentation and PoS tags with 92.5% accuracy and full morphology with 88.5% accuracy (Adler and Elhadad, 2006).

3.2 Defining Simple NPs

Our definition of Simple NPs is pragmatic. We want to tag phrases that are complete in their syntactic structure, avoid the requirement of tagging recursive structures that include full clauses (relative clauses for example) and in general, tag phrases that have a simple denotation. To establish our definition, we start with the most complex NPs, and break them into smaller parts by stating what should **not** appear inside a Simple NP. This can be summarized by the following table:

Outside SimpleNP	Exceptions
Prepositional Phrases	% related PPs are allowed: 5% מ המכירות 5% of the sales
Relative Clauses	
Verb Phrases	
Apposition ¹	
Some conjunctions (Conjunctions are marked according to the TreeBank guidelines) ² .	Possessive של - '\$el' / 'of' - is not considered a PP

Table 2. Definition of Simple NP chunks

Examples for some Simple NP chunks resulting from that definition:

¹ Apposition structure is not annotated in the TreeBank. As a heuristic, we consider every comma inside a non conjunctive NP which is not followed by an adjective or an adjective phrase to be marking the beginning of an apposition.

² As a special case, Adjectival Phrases and possessive conjunctions are considered to be inside the Simple NP.

[תופעה זו] התבררה אתמול ב[וועדת העבודה והרווחה של הכנסת] שדנה ב[נושא העסקת עובדים זרים]
[This phenomenon] was highlighted yesterday at [the labor and welfare committee-const of the Knesset] that dealt with [the topic-const of foreign workers employment-const].

[המעסיקים] אינם מצפים שיצליחו למשוך [מספר ניכר של עובדים ישראליים] ל[קטיף] בגלל [השכר הנמוך] המשולם³ ל[עבודה זו].
[The employers] do not expect to succeed in attracting [a significant number of Israeli workers] for [the fruit-picking] because of [the low salaries] paid for [this work].

This definition can also yield some rather long and complex chunks, such as:

[כיבושיהם של גינגיס חאן והצבא המונגולי הטטרי שלי]
[The conquests of Genghis Khan and his Mongol Tartar army]

ל [דברי פקידי ממשלה מקומיים] , [מפעלים] על [שטח טטרי] הרוויחו ב [ה שנה] ש עברה [סך של 3 . 7 מיליארד רובל (2 . 2 מיליארד דולר)] , ש [את כולם כמעט] לקחה [מוסקבה]

According to [reports of local government officials], [factories] on [Tartar territory] earned in [the year] that passed [a sum of 3.7 billion Rb (2.2 billion dollars)], which [Moscow] took [almost all].

Note that Simple NPs are split, for example, by the preposition ‘on’ ([factories] on [Tartar territory]), and by a relative clause ([a sum of 3.7Bn Rb] which [Moscow] took [almost all]).

3.3 Hebrew Simple NPs are harder than English base NPs

The Simple NPs derived from our definition are highly coherent units, but are also more complex than the non-recursive English base NPs. As can be seen in Table 1, our definition of Simple NP yields chunks which are on average considerably longer than the English chunks, with about 20% of the chunks with 4 or more words (as opposed to about 10% in English) and a significant portion (6.22%) of chunks with 6 or more words (1.67% in English).

Moreover, the baseline used at the CoNLL shared task⁴ (selecting the chunk tag which was most frequently associated with the current PoS)

³ For readers familiar with Hebrew and feel that המשולם is an adjective and should be inside the NP, we note that this is not the case – המשולם here is actually a Verb in the Beinoni form and the definite marker is actually used as relative marker.

⁴ <http://www.cnts.ua.ac.be/conll2000/chunking/>

gives far inferior results for Hebrew SimpleNPs (see Table 3).

4 Chunking Methods

4.1 Baseline Approaches

We have experimented with different known methods for English NP chunking, which resulted in poor results for Hebrew. We describe here our experiment settings, and provide the best scores obtained for each method, in comparison to the reported scores for English.

All tests were done on the corpus derived from the Hebrew Tree Bank. The corpus contains 5,000 sentences, for a total of 120K tokens (agglutinated words) and 27K NP chunks (more details on the corpus appear below). The last 500 sentences were used as the test set, and all the other sentences were used for training. The results were evaluated using the CoNLL shared task evaluation tools⁵. The approaches tested were Error Driven Pruning (EDP) (Cardie and Pierce, 1998) and Transformational Based Learning of IOB tagging (TBL) (Ramshaw and Marcus, 1995).

The Error Driven Pruning method does not take into account lexical information and uses only the PoS tags. For the Transformation Based method, we have used both the PoS tag and the word itself, with the same templates as described in (Ramshaw and Marcus, 1995). We tried the Transformational Based method with more features than just the PoS and the word, but obtained lower performance. Our best results for these methods, as well as the CoNLL baseline (BASE), are presented in Table 3. These results confirm that the task of Simple NP chunking is harder in Hebrew than in English.

4.2 Support Vector Machines

We chose to adopt a tagging perspective for the Simple NP chunking task, in which each word is to be tagged as either B, I or O depending on whether it is in the Beginning, Inside, or Outside of the given chunk, an approach first taken by Ramshaw and Marcus (1995), and which has become the *de-facto* standard for this task. Using this tagging method, chunking becomes a classification problem – each token is predicted as being either I, O or B, given features from a predefined linguistic context (such as the

⁵ http://www.cnts.ua.ac.be/conll2000/chunking/conllev_al.txt

words surrounding the given word, and their PoS tags).

One model that allows for this prediction is Support Vector Machines - SVM (Vapnik, 1995). SVM is a supervised machine learning algorithm which can handle gracefully a large set of overlapping features. SVMs learn binary classifiers, but the method can be extended to multi-class classification (Allwein et al., 2000; Kudo and Matsumoto, 2000).

SVMs have been successfully applied to many NLP tasks since (Joachims, 1998), and specifically for base phrase chunking (Kudo and Matsumoto, 2000; 2003). It was also successfully used in Arabic (Diab et al., 2004).

The traditional setting of SVM for chunking uses for the context of the token to be classified a window of two tokens around the word, and the features are the PoS tags and lexical items (word forms) of all the tokens in the context. Some settings (Kudo and Matsumoto, 2000) also include the IOB tags of the two “previously tagged” tokens as features (see Fig. 1).

This setting (including the last 2 IOB tags) performs nicely for the case of Hebrew Simple NPs chunking as well.

Linguistic features are mapped to SVM feature vectors by translating each feature such as “PoS at location n-2 is NOUN” or “word at location n+1 is DOG” to a unique vector entry, and setting this entry to 1 if the feature occurs, and 0 otherwise. This results in extremely large yet extremely sparse feature vectors.

Method	English BaseNPs		Hebrew SimpleNPs		
	Prec	Rec	Prec	Rec	F
BASE	72.58	82.14	64.7	75.4	69.78
EDP	92.7	93.7	74.6	78.1	76.3
TBL	91.3	91.8	84.7	87.7	86.2

Table 3. Baseline results for Simple NP chunking SVM Chunking in Hebrew

WORD	POS	CHUNK	Feature Set
ה	NA	B-NP	
מאבק	NOUN	I-NP	Estimated Tag
בין	PREP	O	
רוסיה	NAME	B-NP	
ל	PREP	O	
ה	NA	B-NP	
טורים	NOUN	I-NP	

Figure 1. Linguistic features considered in the basic SVM setting for Hebrew

4.3 Augmentation of Morphological Features

Hebrew is a morphologically rich language. Recent PoS taggers and morphological analyzers for Hebrew (Adler and Elhadad, 2006) address this issue and provide for each word not only the PoS, but also full morphological features, such as Gender, Number, Person, Construct, Tense, and the affixes' properties. Our system, currently, computes these features with an accuracy of 88.5%.

Our original intuition is that the difficulty of Simple NP chunking can be overcome by relying on morphological features in a small context. These features would help the classifier decide on agreement, and split NPs more accurately.

Since SVMs can handle large feature sets, we utilize additional morphological features. In particular, we found the combination of the Number and the Construct features to be most effective in improving chunking results. Indeed, our experiments show that introducing morphological features improves chunking quality by as much as 3-point in F-measure when compared with lexical and PoS features only.

5 Experiment

5.1 The Corpus

The Hebrew TreeBank⁶ consists of 4,995 hand annotated sentences from the *Ha'aretz* newspaper. Besides the syntactic structure, every word is PoS annotated, and also includes morphological features. The words in the TreeBank are segmented: *בית של אנחנו* (instead of *בבית שלנו*). Our morphological analyzer also provides such segmentation.

We derived the Simple NPs structure from the TreeBank using the definition given in Section 3.2. We then converted the original Hebrew TreeBank tagset to the tagset of our PoS tagger. For each token, we specify its word form, its PoS, its morphological features, and its correct IOB tag. The result is the Hebrew Simple NP chunks corpus⁷. The corpus consists of 4,995 sentences, 27,226 chunks and 120,396 segmented tokens. 67,919 of these tokens are covered by NP chunks. A sample annotated sentence is given in Fig. 2.

⁶<http://mila.cs.technion.ac.il/website/english/resources/corpora/treebank/index.html>

⁷<http://www.cs.bgu.ac.il/~nlpproj/chunking>

ב	PREPOSITION	NA	NA	N	NA	N	NA	N	NA	NA	O
ה	DEF_ART	NA	NA	N	NA	N	NA	N	NA	NA	B-NP
עבר	NOUN	M	S	N	NA	N	NA	N	NA	NA	I-NP
היה	AUXVERB	M	S	N	3	N	PAST	N	NA	NA	O
קל	ADJECTIVE	M	S	N	NA	N	NA	N	NA	NA	O
יותר	ADVERB	NA	NA	N	NA	N	NA	N	NA	NA	O
לקרוא	VERB	NA	NA	N	NA	Y	TOINF	N	NA	NA	O
את	ET_PREP	NA	NA	N	NA	N	NA	N	NA	NA	B-NP
ה	DEF_ART	NA	NA	N	NA	N	NA	N	NA	NA	I-NP
מפה	NOUN	F	S	N	NA	N	NA	N	NA	NA	I-NP
.	PUNCTUATION	NA	NA	N	NA	N	NA	N	NA	NA	O

Figure 2. A Sample annotated sentence

5.2 Morphological Features:

The PoS tagset we use consists of 22 tags:

ADJECTIVE	ADVERB	ET_PREP
AUXVERB	CONJUNCTION	DEF_ART
DETERMINER	EXISTENTIAL	INTERJECTION
INTERROGATIVE	MODAL	NEGATION
PARTICLE	NOUN	NUMBER
PRONOUN	PREFIX	PREPOSITION
UNKNOWN	PROPERNAME	PUNCTUATION
VERB		

For each token, we also supply the following morphological features (in that order):

Feature	Possible Values
Gender	(M)ale, (F)emale, (B)oth (unmarked case), (NA)
Number	(S)ingle, (P)lurar, (D)ual, can be (ALL), (NA)
Construct	(Y)es, (N)o
Person	(1)st, (2)nd, (3)rd, (123)all, (NA)
To-Infinitive	(Y)es, (N)o
Tense	Past, Present, Future, Beinoni, Imperative, ToInf, BareInf
(has) Suffix	(Y)es, (N)o
Suffix-Num	(M)ale, (F)emale, (B)oth, (NA)
Suffix-Gen	(S)ingle, (P)lurar, (D)ual, (DP)-dual plural, can be (ALL), (NA)

As noted in (Rambow and Habash 2005), one cannot use the same tagset for a Semitic language as for English. The tagset we have derived has been extensively validated through manual tagging by several testers and cross-checked for agreement.

5.3 Setup and Evaluation

For all the SVM chunking experiments, we use the YamCha⁸ toolkit (Kudo and Matsumoto, 2003). We use forward moving tagging, using standard SVM with polynomial kernel of degree 2, and C=1. For the multiclass classification, we

use pairwise voting. For all the reported experiments, we chose the context to be a -2/+2 tokens windows, centered at the current token.

We use the standard metrics of *accuracy* (% of correctly tagged tokens), *precision*, *recall* and *F-measure*, with the only exception of normalizing all punctuation tokens from the data prior to evaluation, as the TreeBank is highly inconsistent regarding the bracketing of punctuations, and we don't consider the exclusions/inclusions of punctuations from our chunks to be errors (*i.e.*, “[a book ,] [an apple]” “[a book] , [an apple]” and “[a book] [, an apple]” are all equivalent chunkings in our view).

All our development work was done with the first 500 sentences allocated for testing, and the rest for training. For evaluation, we used a 10-fold cross-validation scheme, each time with different consecutive 500 sentences serving for testing and the rest for training.

5.4 Features Used

We run several SVM experiments, each with the settings described in section 5.3, but with a different feature set. In all of the experiments the two previously tagged IOB tags were included in the feature set. In the first experiment (denoted WP) we considered the word and PoS tags of the context tokens to be part of the feature set.

In the other experiments, we used different subsets of the morphological features of the tokens to enhance the features set. We found that good results were achieved by using the *Number* and *Construct* features together with the word and PoS tags (we denote this WPNC). Bad results were achieved when using all the morphological features together. The usefulness of feature sets was stable across all tests in the ten-fold cross validation scheme.

5.5 Results

We discuss the results of the WP and WPNC experiments in details, and also provide the results for the WPG (using the *Gender* feature), and ALL (using *all* available morphological features) experiments, and P (using only PoS tags).

As can be seen in Table 4, lexical information is very important: augmenting the PoS tag with lexical information boosted the F-measure from 77.88 to 92.44. The addition of the extra morphological features of *Construct* and *Number* yields another increase in performance, resulting in a final F-measure of 93.2%. Note that the effect of these morphological features on the overall accuracy (the number of BIO tagged cor-

⁸ <http://chasen.org/~taku/software/yamcha/>

rectly) is minimal (Table 5), yet the effect on the precision and recall is much more significant. It is also interesting to note that the Gender feature hurts performance, even though Hebrew has agreement on both Number and Gender. We do not have a good explanation for this observation – but we are currently verifying the consistency of the gender annotation in the corpus (in particular, the effect of the unmarked gender tag).

We performed the WP and WPNC experiment on two forms of the corpus: (1) WP, WPNC using the manually tagged morphological features included in the TreeBank and (2) WPE, WPNC using the results of our automatic morphological analyzer, which includes about 10% errors (both in PoS and morphological features). With the manual morphology tags, the final F-measure is 93.20, while it is 91.40 with noise. Interestingly, the improvement brought by adding morphological features to chunking in the noisy case (WPNC) is almost 3.0 F-measure points (as opposed to 0.758 for the "clean" morphology case WPNC).

Features	Acc	Prec	Rec	F
P	91.77	77.03	78.79	77.88
WP	97.49	92.54	92.35	92.44
WPE	94.87	89.14	87.69	88.41
WPG	97.41	92.41	92.22	92.32
ALL	96.68	90.21	90.60	90.40
WPNC	97.61	92.99	93.41	93.20
WPNC	96.99	91.49	91.32	91.40

Table 4. SVM results for Hebrew

Features	Prec	Rec	F
WPNC	0.456	1.058	0.758
WPNC	2.35	3.60	2.99

Table 5. Improvement over WP

5.6 Error Analysis and the Effect of Morphological Features

We performed detailed error analysis on the WPNC results for the entire corpus. At the individual token level, Nouns and Conjunctions caused the most confusion, followed by Adverbs and Adjectives. Table 6 presents the confusion matrix for all POSs with a substantial amount of errors. I→O means that the correct chunk tag was I, but the system classified it as O. By examining the errors on the chunks level, we identified 7 common classes of errors:

Conjunction related errors: bracketing “[a] and [b]” instead of “[a and b]” and *vice versa*.

Split errors: bracketing [a][b] instead of [a b]

Merge errors: bracketing [a b] instead of [a][b]

Short errors: bracketing “a [b]” or “[a b]” instead of [a b]

Long errors: bracketing “[a b]” instead of “[a b]” or “a [b]”

Whole Chunk errors: either missing a whole chunk, or bracketing something which doesn’t overlap with a chunk at all (extra chunk).

Missing/ExtraToken errors: this is a generalized form of conjunction errors: either “[a T [b]” instead of “[a T b]” or vice versa, where T is a single token. The most frequent of such words (other than the conjuncts) was 'של' - the possessive 'של'.

POS	All	O→I	O→B	I→O	I→B	B→O	B→I
NOUN	602	7	27	0	342	4	222
CONJ	405	146	5	232	1	21	0
ADVERB	306	87	32	104	10	68	5
DEF_ART	247	18	10	9	104	5	101
ADJECTIVE	215	140	0	58	0	16	1
PROPN	168	4	8	0	82	0	74
NUMBER	158	14	6	4	64	8	62
PREP	152	81	0	62	0	9	0
PRONOUN	99	2	27	3	24	12	31

Table 6. WPNC Confusion Matrix

The data in Table 6 suggests that Adverbs and Adjectives related errors are mostly of the “short” or “long” types, while the Noun (including proper names and pronouns) related errors are of the “split” or “merge” types.

The most frequent error type was conjunction related, closely followed by split and merge. Much less significant errors were cases of extra Adverbs or Adjectives at the end of the chunk, and missing adverbs before or after the chunk.

Conjunctions are a major source of errors for English chunking as well (Ramshaw and Marcus, 1995, Cardie and Pierce, 1998)⁹, and we plan to address them in future work. The split and merge errors are related to argument structure, which can be more complicated in Hebrew than in English, because of possible null equatives. The too-long and too-short errors were mostly attachment related. Most of the errors are related to linguistic phenomena that cannot be inferred by the localized context used in our SVM encoding. We examine the types of errors that the addition of

⁹ Although base-NPs are by definition non-recursive, they may still contain CCs when the coordinators are ‘trapped’: “[securities and exchange commission]” or conjunctions of adjectives.

Number and Construct features fixed. Table 7 summarizes this information.

ERROR	WP	WPNC	# Fixed	% Fixed
CONJUNCTION	256	251	5	1.95
SPLIT	198	225	-27	-13.64
MERGE	366	222	144	39.34
LONG (ADJ AFTER)	120	117	3	2.50
EXTRA CHUNK	89	88	1	1.12
LONG (ADV AFTER)	77	81	-4	-5.19
SHORT (ADV AFTER)	67	65	2	2.99
MISSING CHUNK	50	54	-4	-8.00
SHORT (ADV BEFORE)	53	48	5	9.43
EXTRA לֹא TOK	47	47	0	0.00

Table 7. Effect of Number and Construct information on most frequent error classes

The error classes most affected by the number and construct information were split and merge – WPNC has a tendency of splitting chunks, which resulted in some unjustified splits, but compensates this by fixing over a third of the merging mistakes. This result makes sense – construct and local agreement information can aid in the identification of predicate boundaries. This confirms our original intuition that morphological features do help in identifying boundaries of NP chunks.

6 Conclusion and Future work

We have noted that due to syntactic features such as *smixut*, the traditional definition of base NP chunks does not translate well to Hebrew and probably to other Semitic languages. We defined the notion of Simple NP chunks instead. We have presented a method for identifying Hebrew Simple NPs by supervised learning using SVM, providing another evidence for the suitability of SVM to chunk identification.

We have also shown that using morphological features enhances chunking accuracy. However, the set of morphological features used should be chosen with care, as some features actually hurt performance.

Like in the case of English, a large part of the errors were caused by conjunctions – this problem clearly requires more than local knowledge. We plan to address this issue in future work.

References

Meni Adler and Michael Elhadad, 2006. Unsupervised Morpheme-based HMM for Hebrew Morphological Disambiguation. In *Proc. of COLING/ACL 2006*, Sidney.

Steven P. Abney. 1991. Parsing by Chunks. In Robert C. Berwick, Steven P. Abney, and Carol Tenny

editors, *Principle Based Parsing*. Kluwer Academic Publishers.

- Erin L. Allwein, Robert E. Schapire, and Yoram Singer. 2000. Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. *Journal of Machine Learning Research*, 1:113-141.
- Claire Cardie and David Pierce. 1998. Error-Driven Pruning of Treebank Grammars for Base Noun Phrase Identification. In *Proc. of COLING-98*, Montréal.
- Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks, In *Proc. of HLT/NAACL 2004*, Boston.
- Nizar Habash and Owen Rambow, 2005. Arabic Tokenization, Part-of-speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proc. of ACL 2005*, Ann Arbor.
- Thorsten Joachims. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proc. of ECML-98*, Chemnitz.
- Taku Kudo and Yuji Matsumoto. 2000. Use of Support Vector Learning for Chunk Identification. In *Proc. of CoNLL-2000 and LLL-2000*, Lisbon.
- Taku Kudo and Yuji Matsumoto. 2003. Fast Methods for Kernel-Based Text Analysis. In *Proc. of ACL 2003*, Sapporo.
- Yael Netzer-Dahan and Michael Elhadad, 1998. Generation of Noun Compounds in Hebrew: Can Syntactic Knowledge be Fully Encapsulated? In *Proc. of INLG-98*, Ontario.
- Lance A. Ramshaw and Mitchel P. Marcus. 1995. Text Chunking Using Transformation-based Learning. In *Proc. of the 3rd ACL Workshop on Very Large Corpora*. Cambridge.
- Khalil Sima'an, Alon Itai, Yoad Winter, Alon Altman and N. Nativ, 2001. Building a Tree-bank of Modern Hebrew Text, in *Traitement Automatique des Langues* 42(2).
- Fei Sha and Fernando Pereira. 2003. Shallow Parsing with Conditional Random Fields. *Technical Report CIS TR MS-CIS-02-35*, University of Pennsylvania.
- Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 Shared Task: Chunking. In *Proc. of CoNLL-2000 and LLL-2000*, Lisbon.
- Vladimir Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, NY.
- Tong Zhang, Fred Damerau and David Johnson. 2002. Text Chunking based on a Generalization of Winnow. *Journal of Machine Learning Research*, 2: 615-637.

Multi-Tagging for Lexicalized-Grammar Parsing

James R. Curran
School of IT
University of Sydney
NSW 2006, Australia
james@it.usyd.edu.au

Stephen Clark
Computing Laboratory
Oxford University
Wolfson Building
Parks Road
Oxford, OX1 3QD, UK
sclark@comlab.ox.ac.uk

David Vadas
School of IT
University of Sydney
NSW 2006, Australia
dvadas1@it.usyd.edu.au

Abstract

With performance above 97% accuracy for newspaper text, part of speech (POS) tagging might be considered a solved problem. Previous studies have shown that allowing the parser to resolve POS tag ambiguity does not improve performance. However, for grammar formalisms which use more fine-grained grammatical categories, for example TAG and CCG, tagging accuracy is much lower. In fact, for these formalisms, premature ambiguity resolution makes parsing infeasible.

We describe a multi-tagging approach which maintains a suitable level of lexical category ambiguity for accurate and efficient CCG parsing. We extend this multi-tagging approach to the POS level to overcome errors introduced by automatically assigned POS tags. Although POS tagging accuracy seems high, maintaining some POS tag ambiguity in the language processing pipeline results in more accurate CCG supertagging.

1 Introduction

State-of-the-art part of speech (POS) tagging accuracy is now above 97% for newspaper text (Collins, 2002; Toutanova et al., 2003). One possible conclusion from the POS tagging literature is that accuracy is approaching the limit, and any remaining improvement is within the noise of the Penn Treebank training data (Ratnaparkhi, 1996; Toutanova et al., 2003).

So why should we continue to work on the POS tagging problem? Here we give two reasons. First, for lexicalized grammar formalisms such as TAG

and CCG, the tagging problem is much harder. Second, any errors in POS tagger output, even at 97% accuracy, can have a significant impact on components further down the language processing pipeline. In previous work we have shown that using automatically assigned, rather than gold standard, POS tags reduces the accuracy of our CCG parser by almost 2% in dependency F-score (Clark and Curran, 2004b).

CCG *supertagging* is much harder than POS tagging because the CCG tag set consists of fine-grained lexical categories, resulting in a larger tag set – over 400 CCG lexical categories compared with 45 Penn Treebank POS tags. In fact, using a state-of-the-art tagger as a front end to a CCG parser makes accurate parsing infeasible because of the high supertagging error rate.

Our solution is to use *multi-tagging*, in which a CCG supertagger can potentially assign more than one lexical category to a word. In this paper we significantly improve our earlier approach (Clark and Curran, 2004a) by adapting the forward-backward algorithm to a Maximum Entropy tagger, which is used to calculate a probability distribution over lexical categories for each word. This distribution is used to assign one or more categories to each word (Charniak et al., 1996). We report large increases in accuracy over single-tagging at only a small cost in increased ambiguity.

A further contribution of the paper is to also use multi-tagging for the POS tags, and to maintain some POS ambiguity in the language processing pipeline. In particular, since POS tags are important features for the supertagger, we investigate how supertagging accuracy can be improved by not prematurely committing to a POS tag decision. Our results first demonstrate that a surprising in-

crease in POS tagging accuracy can be achieved with only a tiny increase in ambiguity; and second that maintaining some POS ambiguity can significantly improve the accuracy of the supertagger.

The parser uses the CCG lexical categories to build syntactic structure, and the POS tags are used by the supertagger and parser as part of their statistical models. We show that using a multi-tagger for supertagging results in an effective pre-processor for CCG parsing, and that using a multi-tagger for POS tagging results in more accurate CCG supertagging.

2 Maximum Entropy Tagging

The tagger uses conditional probabilities of the form $P(y|x)$ where y is a tag and x is a local context containing y . The conditional probabilities have the following log-linear form:

$$P(y|x) = \frac{1}{Z(x)} e^{\sum_i \lambda_i f_i(x,y)} \quad (1)$$

where $Z(x)$ is a normalisation constant which ensures a proper probability distribution for each context x .

The feature functions $f_i(x,y)$ are binary-valued, returning either 0 or 1 depending on the tag y and the value of a particular contextual predicate given the context x . Contextual predicates identify elements of the context which might be useful for predicting the tag. For example, the following feature returns 1 if the current word is *the* and the tag is DT; otherwise it returns 0:

$$f_i(x,y) = \begin{cases} 1 & \text{if word}(x) = \text{the} \ \& \ y = \text{DT} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

$\text{word}(x) = \text{the}$ is an example of a *contextual predicate*. The POS tagger uses the same contextual predicates as Ratnaparkhi (1996); the supertagger adds contextual predicates corresponding to POS tags and bigram combinations of POS tags (Curran and Clark, 2003).

Each feature f_i has an associated weight λ_i which is determined during training. The training process aims to maximise the entropy of the model subject to the constraints that the expectation of each feature according to the model matches the empirical expectation from the training data. This can be also thought of in terms of maximum likelihood estimation (MLE) for a log-linear model (Della Pietra et al., 1997). We use the L-BFGS op-

timisation algorithm (Nocedal and Wright, 1999; Malouf, 2002) to perform the estimation.

MLE has a tendency to overfit the training data. We adopt the standard approach of Chen and Rosenfeld (1999) by introducing a Gaussian prior term to the objective function which penalises feature weights with large absolute values. A parameter defined in terms of the standard deviation of the Gaussian determines the degree of smoothing.

The conditional probability of a sequence of tags, $y_1, \dots, y_n$, given a sentence, $w_1, \dots, w_n$, is defined as the product of the individual probabilities for each tag:

$$P(y_1, \dots, y_n | w_1, \dots, w_n) = \prod_{i=1}^n P(y_i | x_i) \quad (3)$$

where x_i is the context for word w_i . We use the standard approach of Viterbi decoding to find the highest probability sequence.

2.1 Multi-tagging

Multi-tagging — assigning one or more tags to a word — is used here in two ways: first, to retain ambiguity in the CCG lexical category sequence for the purpose of building parse structure; and second, to retain ambiguity in the POS tag sequence. We retain ambiguity in the lexical category sequence since a single-tagger is not accurate enough to serve as a front-end to a CCG parser, and we retain some POS ambiguity since POS tags are used as features in the statistical models of the supertagger and parser.

Charniak et al. (1996) investigated multi-POS tagging in the context of PCFG parsing. It was found that multi-tagging provides only a minor improvement in accuracy, with a significant loss in efficiency; hence it was concluded that, given the particular parser and tagger used, a single-tag POS tagger is preferable to a multi-tagger. More recently, Watson (2006) has revisited this question in the context of the RASP parser (Briscoe and Carroll, 2002) and found that, similar to Charniak et al. (1996), multi-tagging at the POS level results in a small increase in parsing accuracy but at some cost in efficiency.

For lexicalized grammars, such as CCG and TAG, the motivation for using a multi-tagger to assign the elementary structures (supertags) is more compelling. Since the set of supertags is typically much larger than a standard POS tag set, the tagging problem becomes much harder. In

fact, when using a state-of-the-art single-tagger, the per-word accuracy for CCG supertagging is so low (around 92%) that wide coverage, high accuracy parsing becomes infeasible (Clark, 2002; Clark and Curran, 2004a). Similar results have been found for a highly lexicalized HPSG grammar (Prins and van Noord, 2003), and also for TAG. As far as we are aware, the only approach to successfully integrate a TAG supertagger and parser is the Lightweight Dependency Analyser of Bangalore (2000). Hence, in order to perform effective full parsing with these lexicalized grammars, the tagger front-end must be a multi-tagger (given the current state-of-the-art).

The simplest approach to CCG supertagging is to assign all categories to a word which the word was seen with in the data. This leaves the parser the task of managing the very large parse space resulting from the high degree of lexical category ambiguity (Hockenmaier and Steedman, 2002; Hockenmaier, 2003). However, one of the original motivations for supertagging was to significantly reduce the syntactic ambiguity before full parsing begins (Bangalore and Joshi, 1999). Clark and Curran (2004a) found that performing CCG supertagging prior to parsing can significantly increase parsing efficiency with no loss in accuracy.

Our multi-tagging approach follows that of Clark and Curran (2004a) and Charniak et al. (1996): assign all categories to a word whose probabilities are within a factor, β , of the probability of the most probable category for that word:

$$\mathcal{C}_i = \{c \mid P(C_i = c|S) > \beta P(C_i = c_{\max}|S)\}$$

$\mathcal{C}_i$ is the set of categories assigned to the i th word; C_i is the random variable corresponding to the category of the i th word; $c_{\max}$ is the category with the highest probability of being the category of the i th word; and S is the sentence. One advantage of this adaptive approach is that, when the probability of the highest scoring category is much greater than the rest, no extra categories will be added.

Clark and Curran (2004a) propose a simple method for calculating $P(C_i = c|S)$: use the word and POS features in the local context to calculate the probability and ignore the previously assigned categories (the *history*). However, it is possible to incorporate the history in the calculation of the tag probabilities. A greedy approach is to use the locally highest probability history as a feature, which avoids any summing over alternative histories. Alternatively, there is a well-known

dynamic programming algorithm — the forward backward algorithm — which efficiently calculates $P(C_i = c|S)$ (Charniak et al., 1996).

The multitagger uses the following conditional probabilities:

$$P(y_i|w_{1,n}) = \sum_{y_{1,i-1}, y_{i+1,n}} P(y_i, y_{1,i-1}, y_{i+1,n}|w_{1,n})$$

where $x_{i,j} = x_i, \dots, x_j$. Here y_i is to be thought of as a fixed category, whereas y_j ($j \neq i$) varies over the possible categories for word j . In words, the probability of category y_i , given the sentence, is the sum of the probabilities of all sequences containing y_i . This sum is calculated efficiently using the forward-backward algorithm:

$$P(C_i = c|S) = \alpha_i(c)\beta_i(c) \quad (4)$$

where $\alpha_i(c)$ is the total probability of all the category sub-sequences that end at position i with category c ; and $\beta_i(c)$ is the total probability of all the category sub-sequences through to the end which start at position i with category c .

The standard description of the forward-backward algorithm, for example Manning and Schutze (1999), is usually given for an HMM-style tagger. However, it is straightforward to adapt the algorithm to the Maximum Entropy models used here. The forward-backward algorithm we use is similar to that for a Maximum Entropy Markov Model (Lafferty et al., 2001).

POS tags are very informative features for the supertagger, which suggests that using a multi-POS tagger may benefit the supertagger (and ultimately the parser). However, it is unclear whether multi-POS tagging will be useful in this context, since our single-tagger POS tagger is highly accurate: over 97% for WSJ text (Curran and Clark, 2003). In fact, in Clark and Curran (2004b) we report that using automatically assigned, as opposed to gold-standard, POS tags as features results in a 2% loss in parsing accuracy. This suggests that retaining some ambiguity in the POS sequence may be beneficial for supertagging and parsing accuracy. In Section 4 we show this is the case for supertagging.

3 CCG Supertagging and Parsing

Parsing using CCG can be viewed as a two-stage process: first assign lexical categories to the words in the sentence, and then combine the categories

The WSJ is a paper that I enjoy reading
 $\overline{NP/N}$ $\overline{N}$ $\overline{(S[dcl]\backslash NP)/NP}$ $\overline{NP/N}$ $\overline{N}$ $\overline{(NP\backslash NP)/(S[dcl]/NP)}$ $\overline{NP}$ $\overline{(S[dcl]\backslash NP)/(S[ng]\backslash NP)}$ $\overline{(S[ng]\backslash NP)/NP}$

Figure 1: Example sentence with CCG lexical categories.

together using CCG’s combinatory rules.¹ We perform stage one using a supertagger.

The set of lexical categories used by the supertagger is obtained from CCGbank (Hockenmaier, 2003), a corpus of CCG normal-form derivations derived semi-automatically from the Penn Treebank. Following our earlier work, we apply a frequency cutoff to the training set, only using those categories which appear at least 10 times in sections 02-21, which results in a set of 425 categories. We have shown that the resulting set has very high coverage on unseen data (Clark and Curran, 2004a). Figure 1 gives an example sentence with the CCG lexical categories.

The parser is described in Clark and Curran (2004b). It takes POS tagged sentences as input with each word assigned a set of lexical categories. A packed chart is used to efficiently represent all the possible analyses for a sentence, and the CKY chart parsing algorithm described in Steedman (2000) is used to build the chart. A log-linear model is used to score the alternative analyses.

In Clark and Curran (2004a) we described a novel approach to integrating the supertagger and parser: start with a very restrictive supertagger setting, so that only a small number of lexical categories is assigned to each word, and only assign more categories if the parser cannot find a spanning analysis. This strategy results in an efficient and accurate parser, with speeds up to 35 sentences per second. Accurate supertagging at low levels of lexical category ambiguity is therefore particularly important when using this strategy.

We found in Clark and Curran (2004b) that a large drop in parsing accuracy occurs if automatically assigned POS tags are used throughout the parsing process, rather than gold standard POS tags (almost 2% F-score over labelled dependencies). This is due to the drop in accuracy of the supertagger (see Table 3) and also the fact that the log-linear parsing model uses POS tags as features. The large drop in parsing accuracy demonstrates that improving the performance of POS tag-

¹See Steedman (2000) for an introduction to CCG, and see Hockenmaier (2003) for an introduction to wide-coverage parsing using CCG.

TAGS/WORD	β	WORD ACC	SENT ACC
1.00	1	96.7	51.8
1.01	0.8125	97.1	55.4
1.05	0.2969	98.3	70.7
1.10	0.1172	99.0	80.9
1.20	0.0293	99.5	89.3
1.30	0.0111	99.6	91.7
1.40	0.0053	99.7	93.2
4.23	0	99.8	94.8

Table 1: POS tagging accuracy on Section 00 for different levels of ambiguity.

gers is still an important research problem. In this paper we aim to reduce the performance drop of the supertagger by maintaining some POS ambiguity through to the supertagging phase. Future work will investigate maintaining some POS ambiguity through to the parsing phase also.

4 Multi-tagging Experiments

We performed several sets of experiments for POS tagging and CCG supertagging to explore the trade-off between ambiguity and tagging accuracy. For both POS tagging and supertagging we varied the average number of tags assigned to each word, to see whether it is possible to significantly increase tagging accuracy with only a small increase in ambiguity. For CCG supertagging, we also compared multi-tagging approaches, with a fixed category ambiguity of 1.4 categories per word.

All of the experiments used Section 02-21 of CCGbank as training data, Section 00 as development data and Section 23 as final test data. We evaluate both per-word tag accuracy and sentence accuracy, which is the percentage of sentences for which every word is tagged correctly. For the multi-tagging results we consider the word to be tagged correctly if the correct tag appears in the set of tags assigned to the word.

4.1 Results

Table 1 shows the results for multi-POS tagging for different levels of ambiguity. The row corresponding to 1.01 tags per word shows that adding

METHOD	GOLD POS		AUTO POS	
	WORD	SENT	WORD	SENT
single	92.6	36.8	91.5	32.7
noseq	96.2	51.9	95.2	46.1
best hist	97.2	63.8	96.3	57.2
fwdbwd	97.9	72.1	96.9	64.8

Table 2: Supertagging accuracy on Section 00 using different approaches with multi-tagger ambiguity fixed at 1.4 categories per word.

TAGS/ WORD	β	GOLD POS		AUTO POS	
		WORD	SENT	WORD	SENT
1.0	1	92.6	36.8	91.5	32.7
1.2	0.1201	96.8	63.4	95.8	56.5
1.4	0.0337	97.9	72.1	96.9	64.8
1.6	0.0142	98.3	76.4	97.5	69.3
1.8	0.0074	98.4	78.3	97.7	71.0
2.0	0.0048	98.5	79.4	97.9	72.5
2.5	0.0019	98.7	80.6	98.1	74.3
3.0	0.0009	98.7	81.4	98.3	75.6
12.5	0	98.9	82.3	98.8	80.1

Table 3: Supertagging accuracy on Section 00 for different levels of ambiguity.

even a tiny amount of ambiguity (1 extra tag in every 100 words) gives a reasonable improvement, whilst adding 1 tag in 20 words, or approximately one extra tag per sentence on the WSJ, gives a significant boost of 1.6% word accuracy and almost 20% sentence accuracy.

The bottom row of Table 1 gives an upper bound on accuracy if the maximum ambiguity is allowed. This involves setting the β value to 0, so all feasible tags are assigned. Note that the performance gain is only 1.6% in sentence accuracy, compared with the previous row, at the cost of a large increase in ambiguity.

Our first set of CCG supertagging experiments compared the performance of several approaches. In Table 2 we give the accuracies when using gold standard POS tags, and also POS tags automatically assigned by our POS tagger described above. Since POS tags are important features for the supertagger maximum entropy model, erroneous tags have a significant impact on supertagging accuracy.

The *single* method is the single-tagger supertagger, which at 91.5% per-word accuracy is too inaccurate for use with the CCG parser. The remaining rows in the table give multi-tagger results for a cat-

egory ambiguity of 1.4 categories per word. The *noseq* method, which performs significantly better than *single*, does not take into account the previously assigned categories. The *best hist* method gains roughly another 1% in accuracy over *noseq* by taking the greedy approach of using only the two most probable previously assigned categories. Finally, the full forward-backward approach described in Section 2.1 gains roughly another 0.6% by considering all possible category histories. We see the largest jump in accuracy just by returning multiple categories. The other more modest gains come from producing progressively better models of the category sequence.

The final set of supertagging experiments in Table 3 demonstrates the trade-off between ambiguity and accuracy. Note that the ambiguity levels need to be much higher to produce similar performance to the POS tagger and that the upper bound case ($\beta = 0$) has a very high average ambiguity. This is to be expected given the much larger CCG tag set.

5 Tag uncertainty throughout the pipeline

Tables 2 and 3 show that supertagger accuracy when using gold-standard POS tags is typically 1% higher than when using automatically assigned POS tags. Clearly, correct POS tags are important features for the supertagger.

Errors made by the supertagger can multiply out when incorrect lexical categories are passed to the parser, so a 1% increase in lexical category error can become much more significant in the parser evaluation. For example, when using the dependency-based evaluation in Clark and Curran (2004b), getting the lexical category wrong for a ditransitive verb automatically leads to three dependencies in the output being incorrect.

We have shown that multi-tagging can significantly increase the accuracy of the POS tagger with only a small increase in ambiguity. What we would like to do is maintain some degree of POS tag ambiguity and pass multiple POS tags through to the supertagging stage (and eventually the parser). There are several ways to encode multiple POS tags as features. The simplest approach is to treat all of the POS tags as binary features, but this does not take into account the uncertainty in each of the alternative tags. What we need is a way of incorporating probability information into the Maximum Entropy supertagger.

6 Real-values in ME models

Maximum Entropy (ME) models, in the NLP literature, are typically defined with binary features, although they do allow real-valued features. The only constraint comes from the optimisation algorithm; for example, GIS only allows non-negative values. Real-valued features are commonly used with other machine learning algorithms.

Binary features suffer from certain limitations of the representation, which make them unsuitable for modelling some properties. For example, POS taggers have difficulty determining if capitalised, sentence initial words are proper nouns. A useful way to model this property is to determine the ratio of capitalised and non-capitalised instances of a particular word in a large corpus and use a real-valued feature which encodes this ratio (Vadas and Curran, 2005). The only way to include this feature in a binary representation is to discretize (or bin) the feature values. For this type of feature, choosing appropriate bins is difficult and it may be hard to find a discretization scheme that performs optimally.

Another problem with discretizing feature values is that it imposes artificial boundaries to define the bins. For the example above, we may choose the bins $0 \leq x < 1$ and $1 \leq x < 2$, which separate the values 0.99 and 1.01 even though they are close in value. At the same time, the model does not distinguish between 0.01 and 0.99 even though they are much further apart.

Further, if we have not seen cases for the bin $2 \leq x < 3$, then the discretized model has no evidence to determine the contribution of this feature. But for the real-valued model, evidence supporting $1 \leq x < 2$ and $3 \leq x < 4$ provides evidence for the missing bin. Thus the real-valued model generalises more effectively.

One issue that is not addressed here is the interaction between the Gaussian smoothing parameter and real-valued features. Using the same smoothing parameter for real-valued features with vastly different distributions is unlikely to be optimal. However, for these experiments we have used the same value for the smoothing parameter on all real-valued features. This is the same value we have used for the binary features.

7 Multi-POS Supertagging Experiments

We have experimented with four different approaches to passing multiple POS tags as features

through to the supertagger. For the later experiments, this required the existing binary-valued framework to be extended to support real values. The level of POS tag ambiguity was varied between 1.05 and 1.3 POS tags per word on average. These results are shown in Table 4.

The first approach is to treat the multiple POS tags as binary features (*bin*). This simply involves adding the multiple POS tags for each word in both the training and test data. Every assigned POS tag is treated as a separate feature and considered equally important regardless of its uncertainty. Here we see a minor increase in performance over the original supertagger at the lower levels of POS ambiguity. However, as the POS ambiguity is increased, the performance of the binary-valued features decreases and is eventually worse than the original supertagger. This is because at the lowest levels of ambiguity the extra POS tags can be treated as being of similar reliability. However, at higher levels of ambiguity many POS tags are added which are unreliable and should not be trusted equally.

The second approach (*split*) uses real-valued features to model some degree of uncertainty in the POS tags, dividing the POS tag probability mass evenly among the alternatives. This has the effect of giving smaller feature values to tags where many alternative tags have been assigned. This produces similar results to the binary-valued features, again performing best at low levels of ambiguity.

The third approach (*invrank*) is to use the inverse rank of each POS tag as a real-valued feature. The inverse rank is the reciprocal of the tag's rank ordered by decreasing probability. This method assumes the POS tagger correctly orders the alternative tags, but does not rely on the probability assigned to each tag. Overall, *invrank* performs worse than *split*.

The final and best approach is to use the probabilities assigned to each alternative tag as real-valued features:

$$f_i(x, y) = \begin{cases} p(\text{POS}(x) = \text{NN}) & \text{if } y = \text{NP} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

This model gives the best performance at 1.1 POS tags per-word average ambiguity. Note that, even when using the probabilities as features, only a small amount of additional POS ambiguity is required to significantly improve performance.

METHOD	POS AMB	WORD	SENT
orig	1.00	96.9	64.8
bin	1.05	97.3	67.7
	1.10	97.3	66.3
	1.20	97.0	63.5
	1.30	96.8	62.1
split	1.05	97.4	68.5
	1.10	97.4	67.9
	1.20	97.3	67.0
	1.30	97.2	65.1
prob	1.05	97.5	68.7
	1.10	97.5	69.1
	1.20	97.5	68.7
	1.30	97.5	68.7
invrank	1.05	97.3	68.0
	1.10	97.4	68.0
	1.20	97.3	67.1
	1.30	97.3	67.1
gold	-	97.9	72.1

Table 4: Multi-POS supertagging on Section 00 with different levels of POS ambiguity and using different approaches to POS feature encoding.

Table 5 shows our best performance figures for the multi-POS supertagger, against the previously described method using both gold standard and automatically assigned POS tags.

Table 6 uses the Section 23 test data to demonstrate the improvement in supertagging when moving from single-tagging (*single*) to simple multi-tagging (*noseq*); from simple multi-tagging to the full forward-backward algorithm (*fwdbwd*); and finally when using the probabilities of multiply-assigned POS tags as features (MULTI-POS column). All of these multi-tagging experiments use an ambiguity level of 1.4 categories per word and the last result uses POS tag ambiguity of 1.1 tags per word.

8 Conclusion

The NLP community may consider POS tagging to be a solved problem. In this paper, we have suggested two reasons why this is not the case. First, tagging for lexicalized-grammar formalisms, such as CCG and TAG, is far from solved. Second, even modest improvements in POS tagging accuracy can have a large impact on the performance of downstream components in a language processing pipeline.

TAGS/ WORD	AUTO POS		MULTI POS		GOLD POS	
	WORD	SENT	WORD	SENT	WORD	SENT
1.0	91.5	32.7	91.9	34.3	92.6	36.8
1.2	95.8	56.5	96.3	59.2	96.8	63.4
1.4	96.9	64.8	97.5	67.0	97.9	72.1
1.6	97.5	69.3	97.9	73.3	98.3	76.4
1.8	97.7	71.0	98.2	76.1	98.4	78.3
2.0	97.9	72.5	98.4	77.4	98.5	79.4
2.5	98.1	74.3	98.5	78.7	98.7	80.6
3.0	98.3	75.6	98.6	79.7	98.7	81.4

Table 5: Best multi-POS supertagging accuracy on Section 00 using POS ambiguity of 1.1 and the probability real-valued features.

METHOD	AUTO POS	MULTI POS	GOLD POS
single	92.0	-	93.3
noseq	95.4	-	96.6
fwdbwd	97.1	97.7	98.2

Table 6: Final supertagging results on Section 23.

We have developed a novel approach to maintaining tag ambiguity in language processing pipelines which avoids premature ambiguity resolution. The tag ambiguity is maintained by using the forward-backward algorithm to calculate individual tag probabilities. These probabilities can then be used to select multiple tags and can also be encoded as real-valued features in subsequent statistical models.

With this new approach we have increased POS tagging accuracy significantly with only a tiny ambiguity penalty and also significantly improved on previous CCG supertagging results. Finally, using POS tag probabilities as real-valued features in the supertagging model, we demonstrated performance close to that obtained with gold-standard POS tags. This will significantly improve the robustness of the parser on unseen text.

In future work we will investigate maintaining tag ambiguity further down the language processing pipeline and exploiting the uncertainty from previous stages. In particular, we will incorporate real-valued POS tag and lexical category features in the statistical parsing model. Another possibility is to investigate whether similar techniques can improve other tagging tasks, such as Named Entity Recognition.

This work can be seen as part of the larger goal of maintaining ambiguity and exploiting un-

certainty throughout language processing systems (Roth and Yih, 2004), which is important for coping with the compounding of errors that is a significant problem in language processing pipelines.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful feedback. This work has been supported by the Australian Research Council under Discovery Project DP0453131.

References

- Srinivas Bangalore and Aravind Joshi. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics*, 25(2):237–265.
- Srinivas Bangalore. 2000. A lightweight dependency analyser for partial parsing. *Natural Language Engineering*, 6(2):113–138.
- Ted Briscoe and John Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd LREC Conference*, pages 1499–1504, Las Palmas, Gran Canaria.
- Eugene Charniak, Glenn Carroll, John Adcock, Anthony Cassandra, Yoshihiko Gotoh, Jeremy Katz, Michael Littman, and John McCann. 1996. Taggers for parsers. *Artificial Intelligence*, 85:45–57.
- Stanley Chen and Ronald Rosenfeld. 1999. A Gaussian prior for smoothing maximum entropy models. Technical report, Carnegie Mellon University, Pittsburgh, PA.
- Stephen Clark and James R. Curran. 2004a. The importance of supertagging for wide-coverage CCG parsing. In *Proceedings of COLING-04*, pages 282–288, Geneva, Switzerland.
- Stephen Clark and James R. Curran. 2004b. Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Meeting of the ACL*, pages 104–111, Barcelona, Spain.
- Stephen Clark. 2002. A supertagger for Combinatory Categorical Grammar. In *Proceedings of the TAG+ Workshop*, pages 19–24, Venice, Italy.
- Michael Collins. 2002. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proceedings of the EMNLP Conference*, pages 1–8, Philadelphia, PA.
- James R. Curran and Stephen Clark. 2003. Investigating GIS and smoothing for maximum entropy taggers. In *Proceedings of the 10th Meeting of the EACL*, pages 91–98, Budapest, Hungary.
- Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. 1997. Inducing features of random fields. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 19(4):380–393.
- Julia Hockenmaier and Mark Steedman. 2002. Generative models for statistical parsing with Combinatory Categorical Grammar. In *Proceedings of the 40th Meeting of the ACL*, pages 335–342, Philadelphia, PA.
- Julia Hockenmaier. 2003. *Data and Models for Statistical Parsing with Combinatory Categorical Grammar*. Ph.D. thesis, University of Edinburgh.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, Williams College, MA.
- Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Workshop on Natural Language Learning*, pages 49–55, Taipei, Taiwan.
- Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.
- Jorge Nocedal and Stephen J. Wright. 1999. *Numerical Optimization*. Springer, New York, USA.
- Robbert Prins and Gertjan van Noord. 2003. Reinforcing parser preferences through tagging. *Traitement Automatique des Langues*, 44(3):121–139.
- Adwait Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of the EMNLP Conference*, pages 133–142, Philadelphia, PA.
- D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 1–8. Association for Computational Linguistics.
- Mark Steedman. 2000. *The Syntactic Process*. The MIT Press, Cambridge, MA.
- Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the HLT/NAACL conference*, pages 252–259, Edmonton, Canada.
- David Vadas and James R. Curran. 2005. Tagging unknown words with raw text features. In *Proceedings of the Australasian Language Technology Workshop 2005*, pages 32–39, Sydney, Australia.
- Rebecca Watson. 2006. Part-of-speech tagging models for parsing. In *Proceedings of the Computational Linguistics in the UK Conference (CLUK-06)*, Open University, Milton Keynes, UK.

Guessing Parts-of-Speech of Unknown Words Using Global Information

Tetsuji Nakagawa

Corporate R&D Center
Oki Electric Industry Co., Ltd.
2-5-7 Honmachi, Chuo-ku
Osaka 541-0053, Japan
nakagawa378@oki.com

Yuji Matsumoto

Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma
Nara 630-0101, Japan
matsu@is.naist.jp

Abstract

In this paper, we present a method for guessing POS tags of unknown words using local and global information. Although many existing methods use only local information (i.e. limited window size or intra-sentential features), global information (extra-sentential features) provides valuable clues for predicting POS tags of unknown words. We propose a probabilistic model for POS guessing of unknown words using global information as well as local information, and estimate its parameters using Gibbs sampling. We also attempt to apply the model to semi-supervised learning, and conduct experiments on multiple corpora.

1 Introduction

Part-of-speech (POS) tagging is a fundamental language analysis task. In POS tagging, we frequently encounter words that do not exist in training data. Such words are called unknown words. They are usually handled by an exceptional process in POS tagging, because the tagging system does not have information about the words. Guessing the POS tags of such unknown words is a difficult task. But it is an important issue both for conducting POS tagging accurately and for creating word dictionaries automatically or semi-automatically. There have been many studies on POS guessing of unknown words (Mori and Nagao, 1996; Mikheev, 1997; Chen et al., 1997; Nagata, 1999; Orphanos and Christodoulakis, 1999). In most of these previous works, POS tags of unknown words were predicted using only local information, such as lexical forms and POS tags of surrounding words or word-internal features (e.g. suffixes and character types) of the unknown words. However, this approach has limitations in available information. For example, common nouns and proper nouns are sometimes difficult to distinguish with only the information of a single occurrence because their syntactic functions are almost identical. In English, proper nouns are capitalized and there is generally little ambiguity between common nouns and proper nouns. In Chinese and Japanese, no such convention exists and the problem of the ambiguity is serious. However, if an unknown word with the same lex-

ical form appears in another part with informative local features (e.g. titles of persons), this will give useful clues for guessing the part-of-speech of the ambiguous one, because unknown words with the same lexical form usually have the same part-of-speech. For another example, there is a part-of-speech named *sahen-noun* (verbal noun) in Japanese. Verbal nouns behave as common nouns, except that they are used as verbs when they are followed by a verb “*suru*”; e.g., a verbal noun “*dokusho*” means “reading” and “*dokusho-suru*” is a verb meaning to “read books”. It is difficult to distinguish a verbal noun from a common noun if it is used as a noun. However, it will be easy if we know that the word is followed by “*suru*” in another part in the document. This issue was mentioned by Asahara (2003) as a problem of *possibility-based POS tags*. A possibility-based POS tag is a POS tag that represents all the possible properties of the word (e.g., a verbal noun is used as a noun or a verb), rather than a property of each instance of the word. For example, a *sahen-noun* is actually a noun that can be used as a verb when it is followed by “*suru*”. This property cannot be confirmed without observing real usage of the word appearing with “*suru*”. Such POS tags may not be identified with only local information of one instance, because the property that each instance has is only one among all the possible properties.

To cope with these issues, we propose a method that uses global information as well as local information for guessing the parts-of-speech of unknown words. With this method, all the occurrences of the unknown words in a document¹ are taken into consideration at once, rather than that each occurrence of the words is processed separately. Thus, the method models the whole document and finds a set of parts-of-speech by maximizing its conditional joint probability given the document, rather than independently maximizing the probability of each part-of-speech given each sentence. Global information is known to be useful in other NLP tasks, especially in the named entity recognition task, and several studies successfully used global features (Chieu and Ng, 2002; Finkel et al., 2005).

One potential advantage of our method is its

¹In this paper, we use the word *document* to denote the whole data consisting of multiple sentences (training corpus or test corpus).

ability to incorporate unlabeled data. Global features can be increased by simply adding unlabeled data into the test data.

Models in which the whole document is taken into consideration need a lot of computation compared to models with only local features. They also cannot process input data one-by-one. Instead, the entire document has to be read before processing. We adopt Gibbs sampling in order to compute the models efficiently, and these models are suitable for offline use such as creating dictionaries from raw text where real-time processing is not necessary but high-accuracy is needed to reduce human labor required for revising automatically analyzed data.

The rest of this paper is organized as follows: Section 2 describes a method for POS guessing of unknown words which utilizes global information. Section 3 shows experimental results on multiple corpora. Section 4 discusses related work, and Section 5 gives conclusions.

2 POS Guessing of Unknown Words with Global Information

We handle POS guessing of unknown words as a sub-task of POS tagging, in this paper. We assume that POS tags of known words are already determined beforehand, and positions in the document where unknown words appear are also identified. Thus, we focus only on prediction of the POS tags of unknown words.

In the rest of this section, we first present a model for POS guessing of unknown words with global information. Next, we show how the test data is analyzed and how the parameters of the model are estimated. A method for incorporating unlabeled data with the model is also discussed.

2.1 Probabilistic Model Using Global Information

We attempt to model the probability distribution of the parts-of-speech of all occurrences of the unknown words in a document which have the same lexical form. We suppose that such parts-of-speech have correlation, and the part-of-speech of each occurrence is also affected by its local context. Similar situations to this are handled in physics. For example, let us consider a case where a number of electrons with spins exist in a system. The spins interact with each other, and each spin is also affected by the external magnetic field. In the physical model, if the state of the system is $\mathbf{s}$ and the energy of the system is $E(\mathbf{s})$, the probability distribution of $\mathbf{s}$ is known to be represented by the following Boltzmann distribution:

$$P(\mathbf{s}) = \frac{1}{Z} \exp\{-\beta E(\mathbf{s})\}, \quad (1)$$

where β is inverse temperature and Z is a normalizing constant defined as follows:

$$Z = \sum_{\mathbf{s}} \exp\{-\beta E(\mathbf{s})\}. \quad (2)$$

Takamura et al. (2005) applied this model to an NLP task, semantic orientation extraction, and we apply it to POS guessing of unknown words here.

Suppose that unknown words with the same lexical form appear K times in a document. Assume that the number of possible POS tags for unknown words is N , and they are represented by integers from 1 to N . Let t_k denote the POS tag of the k th occurrence of the unknown words, let w_k denote the local context (e.g. the lexical forms and the POS tags of the surrounding words) of the k th occurrence of the unknown words, and let $\mathbf{w}$ and $\mathbf{t}$ denote the sets of w_k and t_k respectively:

$$\mathbf{w} = \{w_1, \dots, w_K\}, \quad \mathbf{t} = \{t_1, \dots, t_K\}, \quad t_k \in \{1, \dots, N\}.$$

$\lambda_{i,j}$ is a weight which denotes strength of the interaction between parts-of-speech i and j , and is symmetric ($\lambda_{i,j} = \lambda_{j,i}$). We define the energy where POS tags of unknown words given $\mathbf{w}$ are $\mathbf{t}$ as follows:

$$E(\mathbf{t}|\mathbf{w}) = -\left\{ \frac{1}{2} \sum_{k=1}^K \sum_{\substack{k'=1 \\ k' \neq k}}^K \lambda_{t_k, t_{k'}} + \sum_{k=1}^K \log p_0(t_k|w_k) \right\}, \quad (3)$$

where $p_0(t|w)$ is an initial distribution (local model) of the part-of-speech t which is calculated with only the local context w , using arbitrary statistical models such as maximum entropy models. The right hand side of the above equation consists of two components; one represents global interactions between each pair of parts-of-speech, and the other represents the effects of local information.

In this study, we fix the inverse temperature $\beta = 1$. The distribution of $\mathbf{t}$ is then obtained from Equation (1), (2) and (3) as follows:

$$P(\mathbf{t}|\mathbf{w}) = \frac{1}{Z(\mathbf{w})} p_0(\mathbf{t}|\mathbf{w}) \exp\left\{ \frac{1}{2} \sum_{k=1}^K \sum_{\substack{k'=1 \\ k' \neq k}}^K \lambda_{t_k, t_{k'}} \right\}, \quad (4)$$

$$Z(\mathbf{w}) = \sum_{\mathbf{t} \in \mathcal{T}(\mathbf{w})} p_0(\mathbf{t}|\mathbf{w}) \exp\left\{ \frac{1}{2} \sum_{k=1}^K \sum_{\substack{k'=1 \\ k' \neq k}}^K \lambda_{t_k, t_{k'}} \right\}, \quad (5)$$

$$p_0(\mathbf{t}|\mathbf{w}) \equiv \prod_{k=1}^K p_0(t_k|w_k), \quad (6)$$

where $\mathcal{T}(\mathbf{w})$ is the set of possible configurations of POS tags given $\mathbf{w}$. The size of $\mathcal{T}(\mathbf{w})$ is N^K , because there are K occurrences of the unknown words and each unknown word can have one of N POS tags. The above equations can be rewritten as follows by defining a function $f_{i,j}(\mathbf{t})$:

$$f_{i,j}(\mathbf{t}) \equiv \frac{1}{2} \sum_{k=1}^K \sum_{\substack{k'=1 \\ k' \neq k}}^K \delta(t_k, i) \delta(t_{k'}, j), \quad (7)$$

$$P(\mathbf{t}|\mathbf{w}) = \frac{1}{Z(\mathbf{w})} p_0(\mathbf{t}|\mathbf{w}) \exp\left\{ \sum_{i=1}^N \sum_{j=1}^N \lambda_{i,j} f_{i,j}(\mathbf{t}) \right\}, \quad (8)$$

$$Z(\mathbf{w}) = \sum_{\mathbf{t} \in \mathcal{T}(\mathbf{w})} p_0(\mathbf{t}|\mathbf{w}) \exp\left\{ \sum_{i=1}^N \sum_{j=1}^N \lambda_{i,j} f_{i,j}(\mathbf{t}) \right\}, \quad (9)$$

where $\delta(i, j)$ is the Kronecker delta:

$$\delta(i, j) = \begin{cases} 1 & (i = j), \\ 0 & (i \neq j). \end{cases} \quad (10)$$

$f_{i,j}(\mathbf{t})$ represents the number of occurrences of the POS tag pair i and j in the whole document (divided by 2), and the model in Equation (8) is essentially a maximum entropy model with the document level features.

As shown above, we consider the conditional joint probability of all the occurrences of the unknown words with the same lexical form in the document given their local contexts, $P(\mathbf{t}|\mathbf{w})$, in contrast to conventional approaches which assume independence of the sentences in the document and use the probabilities of all the words only in a sentence. Note that we assume independence between the unknown words with different lexical forms, and each set of the unknown words with the same lexical form is processed separately from the sets of other unknown words.

2.2 Decoding

Let us consider how to find the optimal POS tags $\mathbf{t}$ basing on the model, given K local contexts of the unknown words with the same lexical form (test data) $\mathbf{w}$, an initial distribution $p_0(t|w)$ and a set of model parameters $\Lambda = \{\lambda_{1,1}, \dots, \lambda_{N,N}\}$. One way to do this is to find a set of POS tags which maximizes $P(\mathbf{t}|\mathbf{w})$ among all possible candidates of $\mathbf{t}$. However, the number of all possible candidates of the POS tags is N^K and the calculation is generally intractable. Although HMMs, MEMMs, and CRFs use dynamic programming and some studies with probabilistic models which have specific structures use efficient algorithms (Wang et al., 2005), such methods cannot be applied here because we are considering interactions (dependencies) between all POS tags, and their joint distribution cannot be decomposed. Therefore, we use a sampling technique and approximate the solution using samples obtained from the probability distribution.

We can obtain a solution $\hat{\mathbf{t}} = \{\hat{t}_1, \dots, \hat{t}_K\}$ as follows:

$$\hat{t}_k = \underset{t}{\operatorname{argmax}} P_k(t|\mathbf{w}), \quad (11)$$

where $P_k(t|\mathbf{w})$ is the marginal distribution of the part-of-speech of the k th occurrence of the unknown words given a set of local contexts $\mathbf{w}$, and is calculated as an expected value over the distribution of the unknown words as follows:

$$\begin{aligned} P_k(t|\mathbf{w}) &= \sum_{\substack{t_1, \dots, t_{k-1}, t_{k+1}, \dots, t_K \\ t_k = t}} P(\mathbf{t}|\mathbf{w}), \\ &= \sum_{\mathbf{t} \in \mathcal{T}(\mathbf{w})} \delta(t_k, t) P(\mathbf{t}|\mathbf{w}). \end{aligned} \quad (12)$$

Expected values can be approximately calculated using enough number of samples generated from the distribution (MacKay, 2003). Suppose that $A(\mathbf{x})$ is a function of a random variable $\mathbf{x}$, $P(\mathbf{x})$

```

initialize  $\mathbf{t}^{(1)}$ 
for  $m := 2$  to  $M$ 
  for  $k := 1$  to  $K$ 
     $t_k^{(m)} \sim P(t_k|\mathbf{w}, t_1^{(m)}, \dots, t_{k-1}^{(m)}, t_{k+1}^{(m-1)}, \dots, t_K^{(m-1)})$ 

```

Figure 1: Gibbs Sampling

is a distribution of $\mathbf{x}$, and $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(M)}\}$ are M samples generated from $P(\mathbf{x})$. Then, the expectation of $A(\mathbf{x})$ over $P(\mathbf{x})$ is approximated by the samples:

$$\sum_{\mathbf{x}} A(\mathbf{x}) P(\mathbf{x}) \simeq \frac{1}{M} \sum_{m=1}^M A(\mathbf{x}^{(m)}). \quad (13)$$

Thus, if we have M samples $\{\mathbf{t}^{(1)}, \dots, \mathbf{t}^{(M)}\}$ generated from the conditional joint distribution $P(\mathbf{t}|\mathbf{w})$, the marginal distribution of each POS tag is approximated as follows:

$$P_k(t|\mathbf{w}) \simeq \frac{1}{M} \sum_{m=1}^M \delta(t_k^{(m)}, t). \quad (14)$$

Next, we describe how to generate samples from the distribution. We use Gibbs sampling for this purpose. Gibbs sampling is one of the Markov chain Monte Carlo (MCMC) methods, which can generate samples efficiently from high-dimensional probability distributions (Andrieu et al., 2003). The algorithm is shown in Figure 1. The algorithm firstly set the initial state $\mathbf{t}^{(1)}$, then one new random variable is sampled at a time from the conditional distribution in which all other variables are fixed, and new samples are created by repeating the process. Gibbs sampling is easy to implement and is guaranteed to converge to the true distribution. The conditional distribution $P(t_k|\mathbf{w}, t_1, \dots, t_{k-1}, t_{k+1}, \dots, t_K)$ in Figure 1 can be calculated simply as follows:

$$\begin{aligned} &P(t_k|\mathbf{w}, t_1, \dots, t_{k-1}, t_{k+1}, \dots, t_K) \\ &= \frac{P(\mathbf{t}|\mathbf{w})}{P(t_1, \dots, t_{k-1}, t_{k+1}, \dots, t_K|P(\mathbf{w}))}, \\ &= \frac{\frac{1}{Z(\mathbf{w})} p_0(\mathbf{t}|\mathbf{w}) \exp\{\frac{1}{2} \sum_{k'=1}^K \sum_{\substack{k''=1 \\ k'' \neq k'}}^K \lambda_{t_{k'}, t_{k''}}\}}{\sum_{t_k^*=1}^N P(t_1, \dots, t_{k-1}, t_k^*, t_{k+1}, \dots, t_K|\mathbf{w})}, \\ &= \frac{p_0(t_k|w_k) \exp\{\sum_{\substack{k'=1 \\ k' \neq k}}^K \lambda_{t_{k'}, t_k}\}}{\sum_{t_k^*=1}^N p_0(t_k^*|w_k) \exp\{\sum_{\substack{k'=1 \\ k' \neq k}}^K \lambda_{t_{k'}, t_k^*}\}}, \end{aligned} \quad (15)$$

where the last equation is obtained using the following relation:

$$\frac{1}{2} \sum_{k'=1}^K \sum_{\substack{k''=1 \\ k'' \neq k'}}^K \lambda_{t_{k'}, t_{k''}} = \frac{1}{2} \sum_{\substack{k'=1 \\ k' \neq k}}^K \sum_{\substack{k''=1 \\ k'' \neq k, k'' \neq k'}}^K \lambda_{t_{k'}, t_{k''}} + \sum_{\substack{k'=1 \\ k' \neq k}}^K \lambda_{t_{k'}, t_k}.$$

In later experiments, the number of samples M is set to 100, and the initial state $\mathbf{t}^{(1)}$ is set to the POS tags which maximize $p_0(\mathbf{t}|\mathbf{w})$.

The optimal solution obtained by Equation (11) maximizes the probability of each POS tag given $\mathbf{w}$, and this kind of approach is known as the maximum posterior marginal (MPM) estimate (Marroquin, 1985). Finkel et al. (2005) used simulated annealing with Gibbs sampling to find a solution in a similar situation. Unlike simulated annealing, this approach does not need to define a cooling

schedule. Furthermore, this approach can obtain not only the best solution but also the second best or the other solutions according to $P_k(t|\mathbf{w})$, which are useful when this method is applied to semi-automatic construction of dictionaries because human annotators can check the ranked lists of candidates.

2.3 Parameter Estimation

Let us consider how to estimate the parameter $\Lambda = \{\lambda_{1,1}, \dots, \lambda_{N,N}\}$ in Equation (8) from training data consisting of L examples; $\{\langle \mathbf{w}^1, \mathbf{t}^1 \rangle, \dots, \langle \mathbf{w}^L, \mathbf{t}^L \rangle\}$ (i.e., the training data contains L different lexical forms of unknown words). We define the following objective function $\mathcal{L}_\Lambda$, and find Λ which maximizes $\mathcal{L}_\Lambda$ (the subscript Λ denotes being parameterized by Λ):

$$\begin{aligned} \mathcal{L}_\Lambda &= \log \prod_{l=1}^L P_\Lambda(\mathbf{t}^l | \mathbf{w}^l) + \log P(\Lambda), \\ &= \log \prod_{l=1}^L \frac{1}{Z_\Lambda(\mathbf{w}^l)} p_0(\mathbf{t}^l | \mathbf{w}^l) \exp \left\{ \sum_{i=1}^N \sum_{j=1}^N \lambda_{i,j} f_{i,j}(\mathbf{t}^l) \right\} \\ &\quad + \log P(\Lambda), \\ &= \sum_{l=1}^L \left[-\log Z_\Lambda(\mathbf{w}^l) + \log p_0(\mathbf{t}^l | \mathbf{w}^l) + \sum_{i=1}^N \sum_{j=1}^N \lambda_{i,j} f_{i,j}(\mathbf{t}^l) \right] \\ &\quad + \log P(\Lambda). \end{aligned} \quad (16)$$

The partial derivatives of the objective function are:

$$\begin{aligned} \frac{\partial \mathcal{L}_\Lambda}{\partial \lambda_{i,j}} &= \sum_{l=1}^L \left[f_{i,j}(\mathbf{t}^l) - \frac{\partial}{\partial \lambda_{i,j}} \log Z_\Lambda(\mathbf{w}^l) \right] + \frac{\partial}{\partial \lambda_{i,j}} \log P(\Lambda), \\ &= \sum_{l=1}^L \left[f_{i,j}(\mathbf{t}^l) - \sum_{\mathbf{t} \in \mathcal{T}(\mathbf{w}^l)} f_{i,j}(\mathbf{t}) P_\Lambda(\mathbf{t} | \mathbf{w}^l) \right] + \frac{\partial}{\partial \lambda_{i,j}} \log P(\Lambda). \end{aligned} \quad (17)$$

We use Gaussian priors (Chen and Rosenfeld, 1999) for $P(\Lambda)$:

$$\log P(\Lambda) = - \sum_{i=1}^N \sum_{j=1}^N \frac{\lambda_{i,j}^2}{2\sigma^2} + C, \quad \frac{\partial}{\partial \lambda_{i,j}} \log P(\Lambda) = - \frac{\lambda_{i,j}}{\sigma^2}.$$

where C is a constant and σ is set to 1 in later experiments. The optimal Λ can be obtained by quasi-Newton methods using the above $\mathcal{L}_\Lambda$ and $\frac{\partial \mathcal{L}_\Lambda}{\partial \lambda_{i,j}}$, and we use L-BFGS (Liu and Nocedal, 1989) for this purpose². However, the calculation is intractable because $Z_\Lambda(\mathbf{w}^l)$ (see Equation (9)) in Equation (16) and a term in Equation (17) contain summations over all the possible POS tags. To cope with the problem, we use the sampling technique again for the calculation, as suggested by Rosenfeld et al. (2001). $Z_\Lambda(\mathbf{w}^l)$ can be approximated using M samples $\{\mathbf{t}^{(1)}, \dots, \mathbf{t}^{(M)}\}$ generated from $p_0(\mathbf{t} | \mathbf{w}^l)$:

$$Z_\Lambda(\mathbf{w}^l) = \sum_{\mathbf{t} \in \mathcal{T}(\mathbf{w}^l)} p_0(\mathbf{t} | \mathbf{w}^l) \exp \left\{ \sum_{i=1}^N \sum_{j=1}^N \lambda_{i,j} f_{i,j}(\mathbf{t}) \right\},$$

²In later experiments, L-BFGS often did not converge completely because we used approximation with Gibbs sampling, and we stopped iteration of L-BFGS in such cases.

$$\simeq \frac{1}{M} \sum_{m=1}^M \exp \left\{ \sum_{i=1}^N \sum_{j=1}^N \lambda_{i,j} f_{i,j}(\mathbf{t}^{(m)}) \right\}. \quad (18)$$

The term in Equation (17) can also be approximated using M samples $\{\mathbf{t}^{(1)}, \dots, \mathbf{t}^{(M)}\}$ generated from $P_\Lambda(\mathbf{t} | \mathbf{w}^l)$ with Gibbs sampling:

$$\sum_{\mathbf{t} \in \mathcal{T}(\mathbf{w}^l)} f_{i,j}(\mathbf{t}) P_\Lambda(\mathbf{t} | \mathbf{w}^l) \simeq \frac{1}{M} \sum_{m=1}^M f_{i,j}(\mathbf{t}^{(m)}). \quad (19)$$

In later experiments, the initial state $\mathbf{t}^{(1)}$ in Gibbs sampling is set to the gold standard tags in the training data.

2.4 Use of Unlabeled Data

In our model, unlabeled data can be easily used by simply concatenating the test data and the unlabeled data, and decoding them in the testing phase. Intuitively, if we increase the amount of the test data, test examples with informative local features may increase. The POS tags of such examples can be easily predicted, and they are used as global features in prediction of other examples. Thus, this method uses unlabeled data in only the testing phase, and the training phase is the same as the case with no unlabeled data.

3 Experiments

3.1 Data and Procedure

We use eight corpora for our experiments; the Penn Chinese Treebank corpus 2.0 (CTB), a part of the PFR corpus (PFR), the EDR corpus (EDR), the Kyoto University corpus version 2 (KUC), the RWCP corpus (RWC), the GENIA corpus 3.02p (GEN), the SUSANNE corpus (SUS) and the Penn Treebank WSJ corpus (WSJ), (cf. Table 1). All the corpora are POS tagged corpora in Chinese(C), English(E) or Japanese(J), and they are split into three portions; training data, test data and unlabeled data. The unlabeled data is used in experiments of semi-supervised learning, and POS tags of unknown words in the unlabeled data are eliminated. Table 1 summarizes detailed information about the corpora we used: the language, the number of POS tags, the number of open class tags (POS tags that unknown words can have, described later), the sizes of training, test and unlabeled data, and the splitting method of them. For the test data and the unlabeled data, unknown words are defined as words that do not appear in the training data. The number of unknown words in the test data of each corpus is shown in Table 1, parentheses. Accuracy of POS guessing of unknown words is calculated based on how many words among them are correctly POS-guessed.

Figure 2 shows the procedure of the experiments. We split the training data into two parts; the first half as sub-training data 1 and the latter half as sub-training data 2 (Figure 2, *1). Then, we check the words that appear in the sub-training

Corpus (Lang.)	# of POS (Open Class)	# of Tokens (# of Unknown Words) [partition in the corpus]		
		Training	Test	Unlabeled
CTB (C)	34 (28)	84,937 [sec. 1–270]	7,980 (749) [sec. 271–300]	6,801 [sec. 301–325]
PFR (C)	42 (39)	304,125 [Jan. 1–Jan. 9]	370,627 (27,774) [Jan. 10–Jan. 19]	445,969 [Jan. 20–Jan. 31]
EDR (J)	15 (15)	2,550,532 [$id = 4n + 0, id = 4n + 1$]	1,280,057 (24,178) [$id = 4n + 2$]	1,274,458 [$id = 4n + 3$]
KUC (J)	40 (36)	198,514 [Jan. 1–Jan. 8]	31,302 (2,477) [Jan. 9]	41,227 [Jan. 10]
RWC (J)	66 (55)	487,333 [1–10,000th sentences]	190,571 (11,177) [10,001–14,000th sentences]	210,096 [14,001–18,672th sentences]
GEN (E)	47 (36)	243,180 [1–10,000th sentences]	123,386 (7,775) [10,001–15,000th sentences]	134,380 [15,001–20,546th sentences]
SUS (E)	125 (90)	74,902 [sec. A01–08, G01–08, J01–08, N01–08]	37,931 (5,760) [sec. A09–12, G09–12, J09–17, N09–12]	37,593 [sec. A13–20, G13–22, J21–24, N13–18]
WSJ (E)	45 (33)	912,344 [sec. 0–18]	129,654 (4,253) [sec. 22–24]	131,768 [sec. 19–21]

Table 1: Statistical Information of Corpora

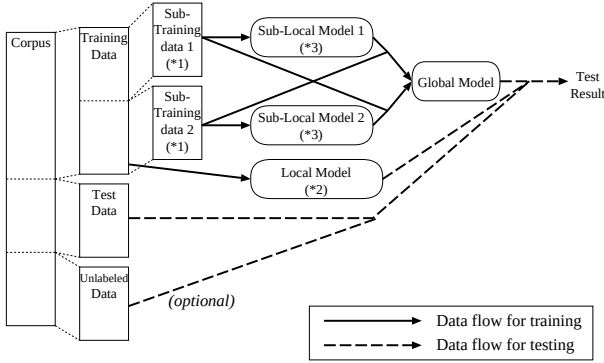


Figure 2: Experimental Procedure

data 1 but not in the sub-training data 2, or vice versa. We handle these words as (*pseudo*) unknown words in the training data. Such (two-fold) cross-validation is necessary to make training examples that contain unknown words³. POS tags that these pseudo unknown words have are defined as open class tags, and only the open class tags are considered as candidate POS tags for unknown words in the test data (i.e., N is equal to the number of the open class tags). In the training phase, we need to estimate two types of parameters; local model (parameters), which is necessary to calculate $p_0(t|w)$, and global model (parameters), i.e., $\lambda_{i,j}$. The local model parameters are estimated using all the training data (Figure 2, *2). Local

³A major method for generating such pseudo unknown words is to collect the words that appear only once in a corpus (Nagata, 1999). These words are called *hapax legomena* and known to have similar characteristics to real unknown words (Baayen and Sproat, 1996). These words are interpreted as being collected by the leave-one-out technique (which is a special case of cross-validation) as follows: One word is picked from the corpus and the rest of the corpus is considered as training data. The picked word is regarded as an unknown word if it does not exist in the training data. This procedure is iterated for all the words in the corpus. However, this approach is not applicable to our experiments because those words that appear only once in the corpus do not have global information and are useless for learning the global model, so we use the two-fold cross validation method.

model parameters and training data are necessary to estimate the global model parameters, but the global model parameters cannot be estimated from the same training data from which the local model parameters are estimated. In order to estimate the global model parameters, we firstly train sub-local models 1 and 2 from the sub-training data 1 and 2 respectively (Figure 2, *3). The sub-local models 1 and 2 are used for calculating $p_0(t|w)$ of unknown words in the sub-training data 2 and 1 respectively, when the global model parameters are estimated from the entire training data. In the testing phase, $p_0(t|w)$ of unknown words in the test data are calculated using the local model parameters which are estimated from the entire training data, and test results are obtained using the global model with the local model.

Global information cannot be used for unknown words whose lexical forms appear only once in the training or test data, so we process only non-unique unknown words (unknown words whose lexical forms appear more than once) using the proposed model. In the testing phase, POS tags of unique unknown words are determined using only the local information, by choosing POS tags which maximize $p_0(t|w)$.

Unlabeled data can be optionally used for semi-supervised learning. In that case, the test data and the unlabeled data are concatenated, and the best POS tags which maximize the probability of the mixed data are searched.

3.2 Initial Distribution

In our method, the initial distribution $p_0(t|w)$ is used for calculating the probability of t given local context w (Equation (8)). We use maximum entropy (ME) models for the initial distribution. $p_0(t|w)$ is calculated by ME models as follows (Berger et al., 1996):

$$p_0(t|w) = \frac{1}{Y(w)} \exp \left\{ \sum_{h=1}^H \alpha_h g_h(w, t) \right\}, \quad (20)$$

Language	Features
English	Prefixes of ω_0 up to four characters, suffixes of ω_0 up to four characters, ω_0 contains Arabic numerals, ω_0 contains uppercase characters, ω_0 contains hyphens.
Chinese Japanese	Prefixes of ω_0 up to two characters, suffixes of ω_0 up to two characters, $\psi_1, \psi_{ \omega_0 }, \psi_1$ & $\psi_{ \omega_0 }$, $\bigcup_{i=1}^{ \omega_0 } \{\psi_i\}$ (set of character types).
(common)	$ \omega_0 $ (length of ω_0), $\tau_{-1}, \tau_{+1}, \tau_{-2} \& \tau_{-1}, \tau_{+1} \& \tau_{+2}$, $\tau_{-1} \& \tau_{+1}, \omega_{-1} \& \tau_{-1}, \omega_{+1} \& \tau_{+1}$, $\omega_{-2} \& \tau_{-2} \& \omega_{-1} \& \tau_{-1}$, $\omega_{+1} \& \tau_{+1} \& \omega_{+2} \& \tau_{+2}$, $\omega_{-1} \& \tau_{-1} \& \omega_{+1} \& \tau_{+1}$.

Table 2: Features Used for Initial Distribution

$$Y(w) = \sum_{t=1}^N \exp \left\{ \sum_{h=1}^H \alpha_h g_h(w, t) \right\}, \quad (21)$$

where $g_h(w, t)$ is a binary feature function. We assume that each local context w contains the following information about the unknown word:

- The POS tags of the two words on each side of the unknown word: $\tau_{-2}, \tau_{-1}, \tau_{+1}, \tau_{+2}$.⁴
- The lexical forms of the unknown word itself and the two words on each side of the unknown word: $\omega_{-2}, \omega_{-1}, \omega_0, \omega_{+1}, \omega_{+2}$.
- The character types of all the characters composing the unknown word: $\psi_1, \dots, \psi_{|\omega_0|}$. We use six character types: alphabet, numeral (Arabic and Chinese numerals), symbol, Kanji (Chinese character), Hiragana (Japanese script) and Katakana (Japanese script).

A feature function $g_h(w, t)$ returns 1 if w and t satisfy certain conditions, and otherwise 0; for example:

$$g_{123}(w, t) = \begin{cases} 1 & (\omega_{-1} = \text{"President"} \text{ and } \tau_{-1} = \text{"NNP"} \text{ and } t = 5), \\ 0 & (\text{otherwise}). \end{cases}$$

The features we use are shown in Table 2, which are based on the features used by Ratnaparkhi (1996) and Uchimoto et al. (2001).

The parameters α_h in Equation (20) are estimated using all the words in the training data whose POS tags are the open class tags.

3.3 Experimental Results

The results are shown in Table 3. In the table, *local*, *local+global* and *local+global w/ unlabeled* indicate that the results were obtained using only local information, local and global information, and local and global information with the extra unlabeled data, respectively. The results using only local information were obtained by choosing POS

⁴In both the training and the testing phases, POS tags of known words are given from the corpora. When these surrounding words contain unknown words, their POS tags are represented by a special tag *Unk*.

PFR (Chinese)	
+162	vn (verbal noun)
+150	ns (place name)
+86	nz (other proper noun)
+85	j (abbreviation)
+61	nr (personal name)
...	...
-26	m (numeral)
-100	v (verb)
RWC (Japanese)	
+33	noun-proper noun-person name-family name
+32	noun-proper noun-place name
+28	noun-proper noun-organization name
+17	noun-proper noun-person name-first name
+6	noun-proper noun
+4	noun-sahen noun
...	...
-2	noun-proper noun-place name-country name
-29	noun
SUS (English)	
+13	NP (proper noun)
+6	JJ (adjective)
+2	VVD (past tense form of lexical verb)
+2	NNL (locative noun)
+2	NNJ (organization noun)
...	...
-3	NN (common noun)
-6	NNU (unit-of-measurement noun)

Table 4: Ordered List of Increased/Decreased Number of Correctly Tagged Words

tags $\hat{\mathbf{t}} = \{\hat{t}_1, \dots, \hat{t}_K\}$ which maximize the probabilities of the local model:

$$\hat{t}_k = \underset{t}{\operatorname{argmax}} p_0(t|w_k). \quad (22)$$

The table shows the accuracies, the numbers of errors, the p-values of McNemar’s test against the results using only local information, and the numbers of non-unique unknown words in the test data. On an Opteron 250 processor with 8GB of RAM, model parameter estimation and decoding without unlabeled data for the eight corpora took 117 minutes and 39 seconds in total, respectively.

In the CTB, PFR, KUC, RWC and WSJ corpora, the accuracies were improved using global information (statistically significant at $p < 0.05$), compared to the accuracies obtained using only local information. The increases of the accuracies on the English corpora (the GEN and SUS corpora) were small. Table 4 shows the increased/decreased number of correctly tagged words using global information in the PFR, RWC and SUS corpora. In the PFR (Chinese) and RWC (Japanese) corpora, many proper nouns were correctly tagged using global information. In Chinese and Japanese, proper nouns are not capitalized, therefore proper nouns are difficult to distinguish from common nouns with only local information. One reason that only the small increases were obtained with global information in the English corpora seems to be the low ambiguities of proper nouns. Many verbal nouns in PFR and a few sahen-nouns (Japanese verbal nouns) in RWC, which suffer from the problem of possibility-based POS tags, were also correctly tagged using global information. When the unlabeled data was used, the number of non-unique words in the test data increased. Compared with the case without the unlabeled data, the accu-

Corpus (Lang.)	Accuracy for Unknown Words (# of Errors) [p-value] # of Non-unique Unknown Words		
	local	local+global	local+global w/ unlabeled
CTB (C)	0.7423 (193)	0.7717 (171) [0.0000] <344>	0.7704 (172) [0.0001] <361>
PFR (C)	0.6499 (9723)	0.6690 (9193) [0.0000] <16019>	0.6785 (8930) [0.0000] <18861>
EDR (J)	0.9639 (874)	0.9643 (863) [0.1775] <4903>	0.9651 (844) [0.0034] <7770>
KUC (J)	0.7501 (619)	0.7634 (586) [0.0000] <788>	0.7562 (604) [0.0872] <936>
RWC (J)	0.7699 (2572)	0.7785 (2476) [0.0000] <5044>	0.7787 (2474) [0.0000] <5878>
GEN (E)	0.8836 (905)	0.8837 (904) [1.0000] <4094>	0.8863 (884) [0.0244] <4515>
SUS (E)	0.7934 (1190)	0.7957 (1177) [0.1878] <3210>	0.7979 (1164) [0.0116] <3583>
WSJ (E)	0.8345 (704)	0.8368 (694) [0.0162] <1412>	0.8352 (701) [0.7103] <1627>

Table 3: Results of POS Guessing of Unknown Words

Corpus (Lang.)	Mean±Standard Deviation	
	Marginal	S.A.
CTB (C)	0.7696±0.0021	0.7682±0.0028
PFR (C)	0.6707±0.0010	0.6712±0.0014
EDR (J)	0.9644±0.0001	0.9645±0.0001
KUC (J)	0.7595±0.0031	0.7612±0.0018
RWC (J)	0.7777±0.0017	0.7772±0.0020
GEN (E)	0.8841±0.0009	0.8840±0.0007
SUS (E)	0.7997±0.0038	0.7995±0.0034
WSJ (E)	0.8366±0.0013	0.8360±0.0021

Table 5: Results of Multiple Trials and Comparison to Simulated Annealing

racies increased in several corpora but decreased in the CTB, KUC and WSJ corpora.

Since our method uses Gibbs sampling in the training and the testing phases, the results are affected by the sequences of random numbers used in the sampling. In order to investigate the influence, we conduct 10 trials with different sequences of pseudo random numbers. We also conduct experiments using simulated annealing in decoding, as conducted by Finkel et al. (2005) for information extraction. We increase inverse temperature β in Equation (1) from $\beta = 1$ to $\beta \approx \infty$ with the linear cooling schedule. The results are shown in Table 5. The table shows the mean values and the standard deviations of the accuracies for the 10 trials, and *Marginal* and *S.A.* mean that decoding is conducted using Equation (11) and simulated annealing respectively. The variances caused by random numbers and the differences of the accuracies between *Marginal* and *S.A.* are relatively small.

4 Related Work

Several studies concerning the use of global information have been conducted, especially in named entity recognition, which is a similar task to POS guessing of unknown words. Chieu and Ng (2002) conducted named entity recognition using global features as well as local features. In their ME

model-based method, some global features were used such as “when the word appeared first in a position other than the beginning of sentences, the word was capitalized or not”. These global features are static and can be handled in the same manner as local features, therefore Viterbi decoding was used. The method is efficient but does not handle interactions between labels.

Finkel et al. (2005) proposed a method incorporating non-local structure for information extraction. They attempted to use *label consistency* of named entities, which is the property that named entities with the same lexical form tend to have the same label. They defined two probabilistic models; a local model based on conditional random fields and a global model based on log-linear models. Then the final model was constructed by multiplying these two models, which can be seen as unnormalized log-linear interpolation (Klakov, 1998) of the two models which are weighted equally. In their method, interactions between labels in the whole document were considered, and they used Gibbs sampling and simulated annealing for decoding. Our model is largely similar to their model. However, in their method, parameters of the global model were estimated using relative frequencies of labels or were selected by hand, while in our method, global model parameters are estimated from training data so as to fit to the data according to the objective function.

One approach for incorporating global information in natural language processing is to utilize consistency of labels, and such an approach have been used in other tasks. Takamura et al. (2005) proposed a method based on the spin models in physics for extracting semantic orientations of words. In the spin models, each electron has one of two states, *up* or *down*, and the models give probability distribution of the states. The states of electrons interact with each other and neighboring electrons tend to have the same spin. In their

method, semantic orientations (*positive* or *negative*) of words are regarded as states of spins, in order to model the property that the semantic orientation of a word tends to have the same orientation as words in its gloss. The mean field approximation was used for inference in their method.

Yarowsky (1995) studied a method for word sense disambiguation using unlabeled data. Although no probabilistic models were considered explicitly in the method, they used the property of label consistency named “one sense per discourse” for unsupervised learning together with local information named “one sense per collocation”.

There exist other approaches using global information which do not necessarily aim to use label consistency. Rosenfeld et al. (2001) proposed whole-sentence exponential language models. The method calculates the probability of a sentence s as follows:

$$P(s) = \frac{1}{Z} p_0(s) \exp \left\{ \sum_i \lambda_i f_i(s) \right\},$$

where $p_0(s)$ is an initial distribution of s and any language models such as trigram models can be used for this. $f_i(s)$ is a feature function and can handle sentence-wide features. Note that if we regard $f_{i,j}(\mathbf{t})$ in our model (Equation (7)) as a feature function, Equation (8) is essentially the same form as the above model. Their models can incorporate any sentence-wide features including syntactic features obtained by shallow parsers. They attempted to use Gibbs sampling and other sampling methods for inference, and model parameters were estimated from training data using the generalized iterative scaling algorithm with the sampling methods. Although they addressed modeling of whole sentences, the method can be directly applied to modeling of whole documents which allows us to incorporate unlabeled data easily as we have discussed. This approach, modeling whole wide-scope contexts with log-linear models and using sampling methods for inference, gives us an expressive framework and will be applied to other tasks.

5 Conclusion

In this paper, we presented a method for guessing parts-of-speech of unknown words using global information as well as local information. The method models a whole document by considering interactions between POS tags of unknown words with the same lexical form. Parameters of the model are estimated from training data using Gibbs sampling. Experimental results showed that the method improves accuracies of POS guessing of unknown words especially for Chinese and Japanese. We also applied the method to semi-supervised learning, but the results were not consistent and there is some room for improvement.

Acknowledgements

This work was supported by a grant from the National Institute of Information and Communications Technology of Japan.

References

- Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. 2003. An introduction to MCMC for Machine Learning. *Machine Learning*, 50:5–43.
- Masayuki Asahara. 2003. *Corpus-based Japanese morphological analysis*. Nara Institute of Science and Technology, Doctor's Thesis.
- Harald Baayen and Richard Sproat. 1996. Estimating Lexical Priors for Low-Frequency Morphologically Ambiguous Forms. *Computational Linguistics*, 22(2):155–166.
- Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–71.
- Stanley Chen and Ronald Rosenfeld. 1999. A Gaussian Prior for Smoothing Maximum Entropy Models. Technical Report CMUCS-99-108, Carnegie Mellon University.
- Chao-jan Chen, Ming-hong Bai, and Keh-Jiann Chen. 1997. Category Guessing for Chinese Unknown Words. In *Proceedings of NLPRS '97*, pages 35–40.
- Hai Leong Chieu and Hwee Tou Ng. 2002. Named Entity Recognition: A Maximum Entropy Approach Using Global Information. In *Proceedings of COLING 2002*, pages 190–196.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of ACL 2005*, pages 363–370.
- D. Klakow. 1998. Log-linear interpolation of language models. In *Proceedings of ICSLP '98*, pages 1695–1699.
- Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(3):503–528.
- David J. C. MacKay. 2003. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press.
- Jose L. Marroquin. 1985. Optimal Bayesian Estimators for Image Segmentation and Surface Reconstruction. A.I. Memo 839, MIT.
- Andrei Mikheev. 1997. Automatic Rule Induction for Unknown-Word Guessing. *Computational Linguistics*, 23(3):405–423.
- Shinsuke Mori and Makoto Nagao. 1996. Word Extraction from Corpora and Its Part-of-Speech Estimation Using Distributional Analysis. In *Proceedings of COLING '96*, pages 1119–1122.
- Masaki Nagata. 1999. A Part of Speech Estimation Method for Japanese Unknown Words using a Statistical Model of Morphology and Context. In *Proceedings of ACL '99*, pages 277–284.
- Giorgos S. Orphanos and Dimitris N. Christodoulakis. 1999. POS Disambiguation and Unknown Word Guessing with Decision Trees. In *Proceedings of EACL '99*, pages 134–141.
- Adwait Ratnaparkhi. 1996. A Maximum Entropy Model for Part-of-Speech Tagging. In *Proceedings of EMNLP '96*, pages 133–142.
- Ronald Rosenfeld, Stanley F. Chen, and Xiaojin Zhu. 2001. Whole-Sentence Exponential Language Models: A Vehicle For Linguistic-Statistical Integration. *Computers Speech and Language*, 15(1):55–73.
- Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting Semantic Orientations of Words using Spin Model. In *Proceedings of ACL 2005*, pages 133–140.
- Kiyotaka Uchimoto, Satoshi Sekine, and Hitoshi Isahara. 2001. The Unknown Word Problem: a Morphological Analysis of Japanese Using Maximum Entropy Aided by a Dictionary. In *Proceedings of EMNLP 2001*, pages 91–99.
- Shaojun Wang, Shaomin Wang, Russel Greiner, Dale Schuurmans, and Li Cheng. 2005. Exploiting Syntactic, Semantic and Lexical Regularities in Language Modeling via Directed Markov Random Fields. In *Proceedings of ICML 2005*, pages 948–955.
- David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of ACL '95*, pages 189–196.

A Clustered Global Phrase Reordering Model for Statistical Machine Translation

Masaaki Nagata

NTT Communication Science Laboratories
2-4 Hikaridai, Seika-cho, Souraku-gun
Kyoto, 619-0237 Japan
nagata.masaaki@labs.ntt.co.jp

Kuniko Saito

NTT Cyber Space Laboratories
1-1 Hikarinooka, Yokoshuka-shi
Kanagawa, 239-0847 Japan
saito.kuniko@labs.ntt.co.jp

Kazuhide Yamamoto, Kazuteru Ohashi*

Nagaoka University of Technology
1603-1, Kamitomioka, Nagaoka City
Niigata, 940-2188 Japan
ykaz@nlp.nagaokaut.ac.jp, ohashi@nlp.nagaokaut.ac.jp

Abstract

In this paper, we present a novel global reordering model that can be incorporated into standard phrase-based statistical machine translation. Unlike previous local reordering models that emphasize the reordering of adjacent phrase pairs (Tillmann and Zhang, 2005), our model explicitly models the reordering of long distances by directly estimating the parameters from the phrase alignments of bilingual training sentences. In principle, the global phrase reordering model is conditioned on the source and target phrases that are currently being translated, and the previously translated source and target phrases. To cope with sparseness, we use N-best phrase alignments and bilingual phrase clustering, and investigate a variety of combinations of conditioning factors. Through experiments, we show, that the global reordering model significantly improves the translation accuracy of a standard Japanese-English translation task.

1 Introduction

Global reordering is essential to the translation of languages with different word orders. Ideally, a model should allow the reordering of any distance, because if we are to translate from Japanese to English, the verb in the Japanese sentence must be moved from the end of the sentence to the beginning just after the subject in the English sentence.

Standard phrase-based translation systems use a word distance-based reordering model in which non-monotonic phrase alignment is penalized based on the word distance between successively translated source phrases without considering the orientation of the phrase alignment or the identities of the source and target phrases (Koehn et al., 2003; Och and Ney, 2004). (Tillmann and Zhang, 2005) introduced the notion of a *block* (a pair of source and target phrases that are translations of each other), and proposed the block orientation bigram in which the local reordering of adjacent blocks are expressed as a three-valued orientation, namely Right (monotone), Left (swapped), or Neutral. A block with neutral orientation is supposed to be less strongly linked to its predecessor block: thus in their model, the global reordering is not explicitly modeled.

In this paper, we present a global reordering model that explicitly models long distance reordering¹. It predicts four type of reordering patterns, namely MA (monotone adjacent), MG (monotone gap), RA (reverse adjacent), and RG (reverse gap). There are based on the identities of the source and target phrases currently being translated, and the previously translated source and target phrases. The parameters of the reordering model are estimated from the phrase alignments of training bilingual sentences. To cope with sparseness, we use N-best phrase alignments and bilingual phrase clustering.

In the following sections, we first describe the global phrase reordering model and its param-

*Graduated in March 2006

¹It might be misleading to call our reordering model “global” since it is at most considers two phrases. A truly global reordering model would take the entire sentence structure into account.

eter estimation method including N-best phrase alignments and bilingual phrase clustering. Next, through an experiment, we show that the global phrase reordering model significantly improves the translation accuracy of the IWSLT-2005 Japanese-English translation task (Eck and Hori, 2005).

2 Baseline Translation Model

In statistical machine translation, the translation of a source (foreign) sentence f is formulated as the search for a target (English) sentence $\hat{e}$ that maximizes the conditional probability $p(e|f)$, which can be rewritten using the Bayes rule as,

$$\hat{e} = \arg \max_e p(e|f) = \arg \max_e p(f|e)p(e)$$

where $p(f|e)$ is a translation model and $p(e)$ is a target language model.

In phrase-based statistical machine translation, the source sentence f is segmented into a sequence of I phrases $\bar{f}_1^I$, and each source phrase $\bar{f}_i$ is translated into a target phrase $\bar{e}_i$. Target phrases may be reordered. The translation model used in (Koehn et al., 2003) is the product of translation probability $\phi(\bar{f}_i|\bar{e}_i)$ and distortion probability $d(a_i - b_{i-1})$,

$$p(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^I \phi(\bar{f}_i|\bar{e}_i) d(a_i - b_{i-1}) \quad (1)$$

where a_i denotes the start position of the source phrase translated into the i -th target phrase, and b_{i-1} denotes the end position of the source phrase translated into the $(i-1)$ -th target phrase.

The translation probability is calculated from the relative frequency as,

$$\phi(\bar{f}|\bar{e}) = \frac{\text{count}(\bar{f}, \bar{e})}{\sum_{\bar{f}} \text{count}(\bar{f}, \bar{e})} \quad (2)$$

where $\text{count}(\bar{f}, \bar{e})$ is the frequency of alignments between the source phrase $\bar{f}$ and the target phrase $\bar{e}$.

(Koehn et al., 2003) used the following distortion model, which simply penalizes non-monotonic phrase alignments based on the word distance of successively translated source phrases with an appropriate value for the parameter α ,

$$d(a_i - b_{i-1}) = \alpha^{|a_i - b_{i-1} - 1|} \quad (3)$$

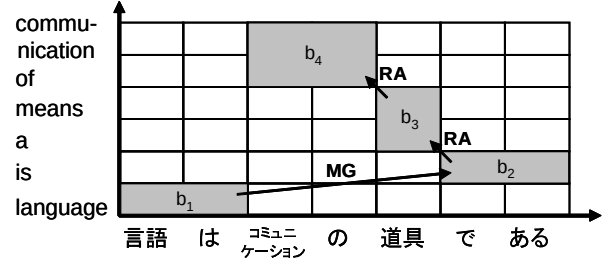


Figure 1: Phrase alignment and reordering

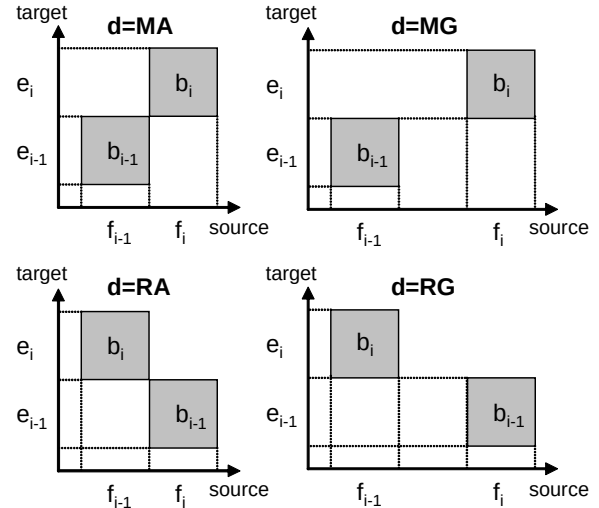


Figure 2: Four types of reordering patterns

3 The Global Phrase Reordering Model

Figure 1 shows an example of Japanese-English phrase alignment that consists of four phrase pairs. Note that the Japanese verb phrase “である” at the end of the sentence is aligned to the English verb “is” at the beginning of the sentence just after the subject. Such reordering is typical in Japanese-English translations.

Motivated by the three-valued orientation for local reordering in (Tillmann and Zhang, 2005), we define the following four types of reordering patterns, as shown in Figure 2,

- monotone adjacent (MA): The two source phrases are adjacent, and are in the same order as the two target phrases.
- monotone gap (MG): The two source phrases are not adjacent, but are in the same order as the two target phrases.
- reverse adjacent (RA): The two source phrases are adjacent, but are in the reverse order of the two target phrases.

	J-to-E	C-to-E
Monotone Adjacent	0.441	0.828
Monotone Gap	0.281	0.106
Reverse Adjacent	0.206	0.033
Reverse Gap	0.072	0.033

Table 1: Percentage of reordering patterns

- reverse gap (RG): The two source phrases are not adjacent, and are in the reverse order as the two target phrases.

For the global reordering model, we only consider the cases in which the two target phrases are adjacent because, in decoding, the target sentence is generated from left to right and phrase by phrase. If we are to generate the i -th target phrase $\bar{e}_i$ from the source phrase $\bar{f}_i$, we call $\bar{e}_i$ and $\bar{f}_i$ the current block b_i , and $\bar{e}_{i-1}$ and $\bar{f}_{i-1}$ the previous block b_{i-1} .

Table 1 shows the percentage of each reordering pattern that appeared in the N-best phrase alignments of the training bilingual sentences for the IWSLT 2005 Japanese-English and Chinese-English translation tasks (Eck and Hori, 2005). Since non-local reorderings such as monotone gap and reverse gap are more frequent in Japanese to English translations, they are worth modeling explicitly in this reordering model.

Since the probability of reordering pattern d (intended to stand for ‘distortion’) is conditioned on the current and previous blocks, the global phrase reordering model is formalized as follows:

$$p(d|\bar{e}_{i-1}, \bar{e}_i, \bar{f}_{i-1}, \bar{f}_i) \quad (4)$$

We can replace the conventional word distance-based distortion probability $d(a_i - b_{i-1})$ in Equation (1) with the global phrase reordering model in Equation (4) with minimal modification of the underlying phrase-based decoding algorithm.

4 Parameter Estimation Method

In principle, the parameters of the global phrase reordering model in Equation (4) can be estimated from the relative frequencies of respective events in the Viterbi phrase alignment of the training bilingual sentences. This straightforward estimation method, however, often suffers from sparse data problem. To cope with this sparseness, we used N-best phrase alignment and bilingual phrase

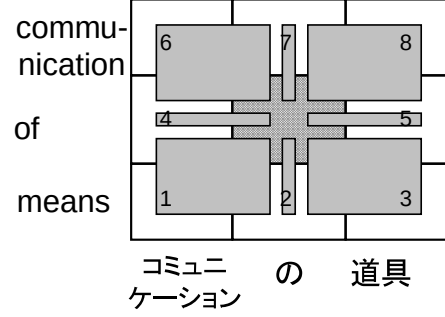


Figure 3: Expansion of a phrase pair

clustering. We also investigated various approximations of Equation (4) by reducing the conditional factors.

4.1 N-best Phrase Alignment

In order to obtain the Viterbi phrase alignment of a bilingual sentence pair, we search for the phrase segmentation and phrase alignment that maximizes the product of the phrase translation probabilities $p(\bar{f}_i|\bar{e}_i)$,

$$(\hat{f}_1^I, \hat{e}_1^I) = \arg \max_{\bar{f}_1^I, \bar{e}_1^I} \prod_{i=1}^I p(\bar{f}_i|\bar{e}_i) \quad (5)$$

Phrase translation probabilities are approximated using word translation probabilities $p(f_j|e_i)$ and $p(e_i|f_j)$ as follows,

$$p(\bar{f}|\bar{e}) = \prod_j \sum_i p(f_j|e_i) + p(e_i|f_j) \quad (6)$$

where f_j and e_i are words in the target and source phrases.

The phrase alignment based on Equation (5) can be thought of as an extension of word alignment based on the IBM Model 1 to phrase alignment. Note that bilingual phrase segmentation (phrase extraction) is also done using the same criteria. The approximation in Equation (6) is motivated by (Vogel et al., 2003). Here, we added the second term $p(e_i|f_j)$ to cope with the asymmetry between $p(f_j|e_i)$ and $p(e_i|f_j)$. The word translation probabilities are estimated using the GIZA++ (Och and Ney, 2003).

The above search is implemented in the following way:

1. All source word and target word pairs are considered to be initial phrase pairs.

2. If the phrase translation probability of the phrase pair is less than the threshold, it is deleted.
3. Each phrase pair is expanded toward the eight neighboring directions as shown in Figure 3.
4. If the phrase translation probability of the expanded phrase pair is less than the threshold, it is deleted.
5. The process of expansion and deletion is repeated until no further expansion is possible.
6. The consistent N-best phrase alignment are searched from all combinations of the above phrase pairs.

The search for consistent Viterbi phrase alignments can be implemented as a phrase-based decoder using a beam search whose outputs are constrained only to the target sentence. The consistent N-best phrase alignment can be obtained by using A* search as described in (Ueffing et al., 2002). We did not use any reordering constraints, such as IBM constraint and ITG constraint in the search for the N-best phrase alignment (Zens et al., 2004).

The thresholds used in the search are the following: the minimum phrase translation probability is 0.0001. The maximum number of translation candidates for each phrase is 20. The beam width is $1e-10$, the stack size (for each target candidate word length) is 1000. We found that, compared with the decoding of sentence translation, we have to search significantly larger space for the N-best phrase alignment.

Figure 3 shows an example of phrase pair expansion toward eight neighbors. If the current phrase pair is (の, of), the expanded phrase pairs are (コミュニケーションの, means_of), (の, means_of), (の道具, means_of), (コミュニケーションの, of), (の道具, of), (コミュニケーションの, of_communication), (の, of_communication), and (の道具, of_communication).

Figure 4 shows an example of the best three phrase alignments for a Japanese-English bilingual sentence. For the estimation of the global phrase reordering model, preliminary tests have shown that the appropriate N-best number is 20. In counting the events for the relative frequency estimation, we treat all N-best phrase alignments equally.

For comparison, we also implemented a different N-best phrase alignment method, where

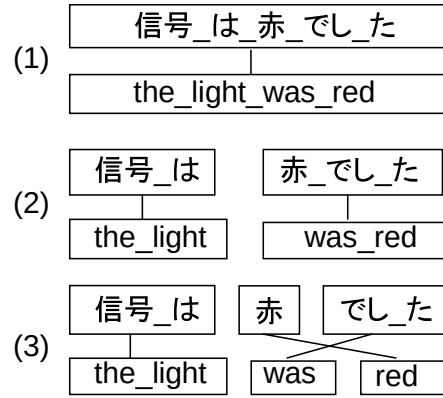


Figure 4: N-best phrase alignments

phrase pairs are extracted using the standard phrase extraction method described in (Koehn et al., 2003). We call this conventional phrase extraction method “grow-diag-final”, and the proposed phrase extraction method “ppicker” (this is intended to stand for phrase picker).

4.2 Bilingual Phrase Clustering

The second approach to cope with the sparseness in Equation (4) is to group the phrases into equivalence classes. We used a bilingual word clustering tool, mkcls (Och et al., 1999) for this purpose. It forms partitions of the vocabulary of the two languages to maximize the joint probability of the training bilingual corpus.

In order to perform bilingual phrase clustering, all words in a phrase are concatenated by an underscore ‘_’ to form a pseudo word. We then use the modified bilingual sentences as the input to mkcls. We treat all N-best phrase alignments equally. Thus, the phrase alignments in Figure 4 are converted to the following three bilingual sentence pairs.

信号は赤でした
the_light_was_red
信号は 赤 でした
the_light was_red
信号は 赤 でした
the_light was red

Preliminary tests have shown that the appropriate number of classes for the estimation of the global phrase reordering model is 20.

As a comparison, we also tried two phrase classification methods based on the part of speech of the head word (Ohashi et al., 2005). We defined (arguably) the first word of each English phrase and the last word of each Japanese phrase as the

shorthand	reordering model
baseline	$\alpha^{ a_i-b_{i-1}-1 }$
ϕ	$p(d)$
e[0]	$p(d \bar{e}_i)$
f[0]	$p(d \bar{f}_i)$
e[0]f[0]	$p(d \bar{e}_i, \bar{f}_i)$
e[-1]f[0]	$p(d \bar{e}_{i-1}, \bar{f}_i)$
e[0]f[-1,0]	$p(d \bar{e}_i, \bar{f}_{i-1}, \bar{f}_i)$
e[-1]f[-1,0]	$p(d \bar{e}_{i-1}, \bar{f}_{i-1}, \bar{f}_i)$
e[-1,0]f[0]	$p(d \bar{e}_{i-1}, \bar{e}_i, \bar{f}_i)$
e[-1,0]f[-1,0]	$p(d \bar{e}_{i-1}, \bar{e}_i, \bar{f}_{i-1}, \bar{f}_i)$

Table 2: All reordering models tried in the experiments

head word. We then used the part of speech of the head word as the phrase class. We call this method “1pos”. Since we are not sure whether it is appropriate to introduce asymmetry in head word selection, we also tried a “2pos” method, where the parts of speech of both the first and the last words are used for phrase classification.

4.3 Conditioning Factor of Reordering

The third approach to cope with sparseness in Equation (4) is to approximate the equation by reducing the conditioning factors.

Other than the baseline word distance-based reordering model and the Equation (4) itself, we tried eight different approximations of Equation (4) as shown in Table 2, where, the symbol in the left column is the shorthand for the reordering model in the right column.

The approximations are designed based on two intuitions. The current block ($\bar{e}_i$ and $\bar{f}_i$) would probably be more important than the previous block ($\bar{e}_{i-1}$ and $\bar{f}_{i-1}$). The previous target phrase ($\bar{e}_{i-1}$) might be more important than the current target phrase ($\bar{e}_i$) because the distortion model of IBM 4 is conditioned on $\bar{e}_{i-1}$, $\bar{f}_{i-1}$ and $\bar{f}_i$. The appropriate form of the global phrase reordering model is decided through experimentation.

5 Experiments

5.1 Corpus and Tools

We used the IWSLT-2005 Japanese-English translation task (Eck and Hori, 2005) for evaluating the proposed global phrase reordering model. We report results using the well-known automatic evaluation metrics Bleu (Papineni et al., 2002).

IWSLT (International Workshop on Spoken

	Sentences	Words	Vocabulary
Japanese	20,000	198,453	9,277
English	20,000	183,452	6,956

Table 3: IWSLT 2005 Japanese-English training data

Language Translation) 2005 is an evaluation campaign for spoken language translation. Its task domain encompasses basic travel conversations. 20,000 bilingual sentences are provided for training. Table 3 shows the number of words and the size of vocabulary of the training data. The average sentence length of Japanese is 9.9 words, while that of English is 9.2 words.

Two development sets, each containing 500 source sentences, are also provided and each development sentence comes with 16 reference translations. We used the second development set (devset2) for the experiments described in this paper. This 20,000 sentence corpus allows for fast experimentation and enables us to study different aspects of the proposed global phrase reordering model.

Japanese word segmentation was done using ChaSen² and English tokenization was done using a tool provided by LDC³. For the phrase classification based on the parts of speech of the head word, we used the first two layers of the Chasen’s part of speech tag for Japanese. For English part of speech tagging, we used MXPOST⁴.

Word translation probabilities are obtained by using GIZA++ (Och and Ney, 2003). For training, all English words are made in lower case. We used a back-off word trigram model as the language model. It is trained from the lowercased English side of the training corpus using a statistical language modeling toolkit, Palmkit⁵.

We implemented our own decoder based on the algorithm described in (Ueffing et al., 2002). For decoding, we used phrase translation probability, lexical translation probability, word penalty, and distortion (phrase reordering) probability. Minimum error rate training was not used for weight optimization.

The thresholds used in the decoding are the following: the minimum phrase translation probability is 0.01. The maximum number of translation

²<http://chasen.aist-nara.ac.jp/>

³<http://www.cis.upenn.edu/~treebank/tokenizer.sed>

⁴<http://www.cis.upenn.edu/~adwait/statnlp.html>

⁵<http://palmkit.sourceforge.net/>

	ppicker		grow-diag-final	
	class	lex	class	lex
baseline	0.400	0.400	0.343	0.343
ϕ	0.407	0.407	0.350	0.350
f[0]	0.417	0.410	0.362	0.356
e[0]	0.422	0.416	0.356	0.360
e[0]f[0]	0.422	0.404	0.355	0.353
e[0]f[-1,0]	0.407	0.381	0.346	0.327
e[-1,0]f[0]	0.410	0.392	0.348	0.341
e[-1,0]f[-1,0]	0.394	0.387	0.339	0.340

Table 4: BLEU score of reordering models with different phrase extraction methods

candidates for each phrase is 10. The beam width is $1e-5$, the stack size (for each target candidate word length) is 100.

5.2 Clustered and Lexicalized Model

Figure 5 shows the BLEU score of clustered and lexical reordering model with different conditioning factors. Here, “class” shows the accuracy when the identity of each phrase is represented by its class, which is obtained by the bilingual phrase clustering, while “lex” shows the accuracy when the identity of each phrases is represented by its lexical form.

The clustered reordering model “class” is generally better than the lexicalized reordering model “lex”. The accuracy of “lex” drops rapidly as the number of conditioning factors increases. The reordering models using the part of speech of the head word for phrase classification such as “1pos” and “2pos” are somewhere in between.

The best score is achieved by the clustered model when the phrase reordering pattern is conditioned on either the current target phrase $\bar{e}_i$ or the current block, namely phrase pair $\bar{e}_i$ and $\bar{f}_i$. They are significantly better than the baseline of the word distance-based reordering model.

5.3 Interaction between Phrase Extraction and Phrase Alignment

Table 4 shows the BLEU score of reordering models with different phrase extraction methods. Here, “ppicker” shows the accuracy when phrases are extracted by using the N-best phrase alignment method described in Section 4.1, while “grow-diag-final” shows the accuracy when phrases are extracted using the standard phrase extraction algorithm described in (Koehn et al., 2003).

It is obvious that, for building the global phrase reordering model, our phrase extraction method is significantly better than the conventional phrase extraction method. We assume this is because the proposed N-best phrase alignment method optimizes the combination of phrase extraction (segmentation) and phrase alignment in a sentence.

5.4 Global and Local Reordering Model

In order to show the advantages of explicitly modeling global phrase reordering, we implemented a different reordering model where the reordering pattern is classified into three values: monotone adjacent, reverse adjacent and neutral. By collapsing monotone gap and reverse gap into neutral, it can be thought of as a local reordering model similar to the block orientation bigram (Tillmann and Zhang, 2005).

Figure 6 shows the BLEU score of the local and global reordering models. Here, “class3” and “lex3” represent the three-valued local reordering model, while “class4” and “lex4” represent the four-valued global reordering model. “Class” and “lex” represent clustered and lexical models, respectively. We used “grow-diag-final” for phrase extraction in this experiment.

It is obvious that the four-valued global reordering model consistently outperformed the three-valued local reordering model under various conditioning factors.

6 Discussion

As shown in Figure 5, the reordering model of Equation (4) (indicated as e[-1,0]f[-1,0] in shorthand) suffers from a sparse data problem even if phrase clustering is used. The empirically justifiable global reordering model seems to be the following, conditioned on the classes of source and target phrases:

$$p(d|class(\bar{e}_i), class(\bar{f}_i)) \quad (7)$$

which is similar to the block orientation bigram (Tillmann and Zhang, 2005). We should note, however, that the block orientation bigram is a joint probability model for the sequence of blocks (source and target phrases) as well as their orientations (reordering pattern) whose purpose is very different from our global phrase reordering model. The advantage of the reordering model is that it can better model global phrase reordering using a four-valued reordering pattern, and it can be easily

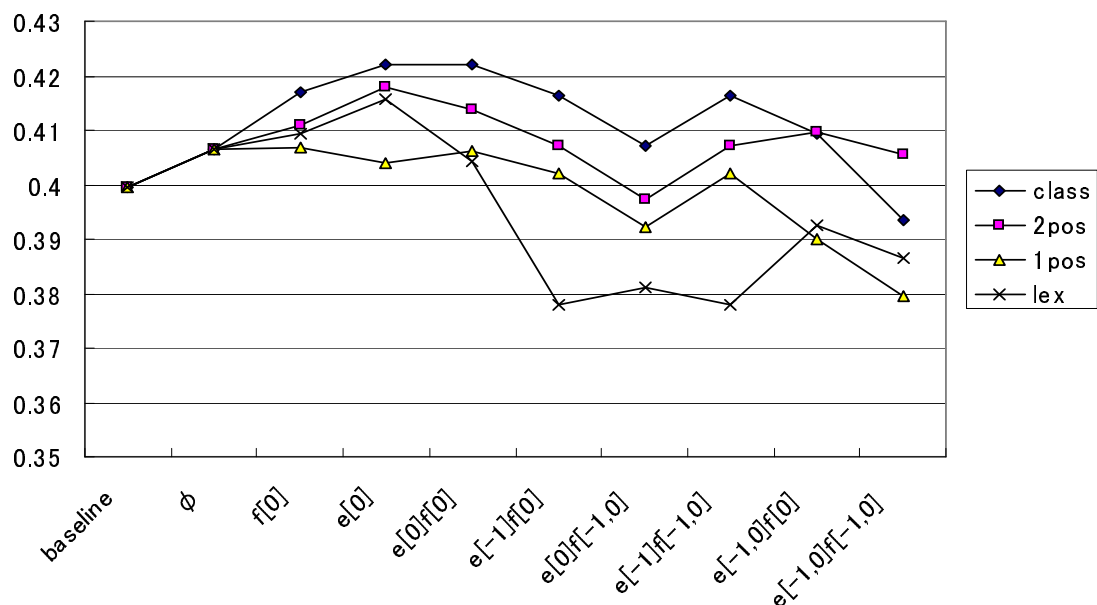


Figure 5: BLEU score for the clustered and lexical reordering model with different conditioning factors

incorporated into a standard phrase-based translation decoder.

The problem of the global phrase reordering model is the cost of parameter estimation. In particular, the N-best phrase alignment described in Section 4.1 is computationally expensive. We must devise a more efficient phrase alignment algorithm that can globally optimize both phrase segmentation (phrase extraction) and phrase alignment.

7 Conclusion

In this paper, we presented a novel global phrase reordering model, that is estimated from the N-best phrase alignment of training bilingual sentences. Through experiments, we were able to show that our reordering model offers improved translation accuracy over the baseline method.

References

- Matthias Eck and Chiori Hori. 2005. Overview of the IWSLT 2005 evaluation campaign. In *Proceedings of International Workshop on Spoken Language Translation (IWSLT 2005)*, pages 11–32.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL-03)*, pages 127–133.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.
- Franz Josef Och, Christoph Tillman, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/WVLC-99)*, pages 20–28.
- Kazuteru Ohashi, Kazuhide Yamamoto, Kuniko Saito, and Masaaki Nagata. 2005. NUT-NTT statistical machine translation system for IWSLT 2005. In *Proceedings of International Workshop on Spoken Language Translation*, pages 128–133.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, pages 311–318.
- Christoph Tillmann and Tong Zhang. 2005. A localized prediction model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 557–564.
- Nicola Ueffing, Franz Josef Och, and Hermann Ney. 2002. Generation of word graphs in statistical machine translation. In *Proceedings of the Conference*

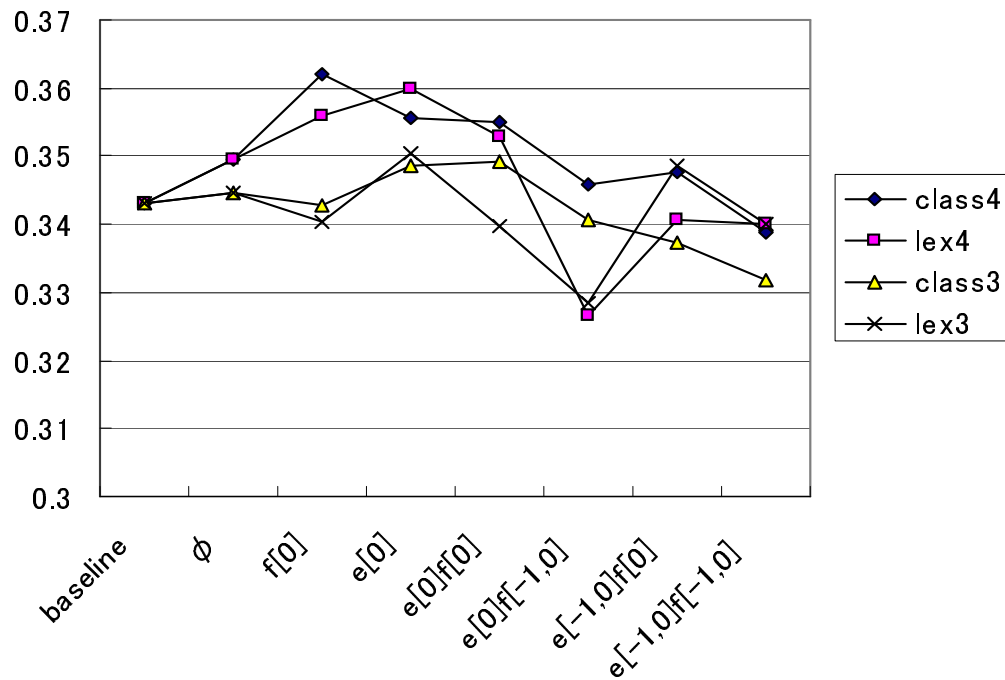


Figure 6: BLEU score of local and global reordering model

on *Empirical Methods in Natural Language Processing (EMNLP-02)*, pages 156–163.

Stephan Vogel, Ying Zhang, Fei Huang, Alicia Tribble, Ashish Venugopal, Bing Zhao, and Alex Waibel. 2003. The CMU statistical machine translation system. In *Proceedings of MT Summit IX*.

Richard Zens, Hermann Ney, Taro Watanabe, and Ei-ichiro Sumita. 2004. Reordering constraints for phrase-based statistical machine translation. In *Proceedings of 20th International Conference on Computational Linguistics (COLING-04)*, pages 205–211.

A Discriminative Global Training Algorithm for Statistical MT

Christoph Tillmann

IBM T.J. Watson Research Center
Yorktown Heights, N.Y. 10598
ctill@us.ibm.com

Tong Zhang

Yahoo! Research
New York City, N.Y. 10011
tzhang@yahoo-inc.com

Abstract

This paper presents a novel training algorithm for a linearly-scored block sequence translation model. The key component is a new procedure to directly optimize the global scoring function used by a SMT decoder. No translation, language, or distortion model probabilities are used as in earlier work on SMT. Therefore our method, which employs less domain specific knowledge, is both simpler and more extensible than previous approaches. Moreover, the training procedure treats the decoder as a black-box, and thus can be used to optimize any decoding scheme. The training algorithm is evaluated on a standard Arabic-English translation task.

1 Introduction

This paper presents a view of phrase-based SMT as a sequential process that generates block orientation sequences. A block is a pair of phrases which are translations of each other. For example, Figure 1 shows an Arabic-English translation example that uses four blocks. During decoding, we view translation as a block segmentation process, where the input sentence is segmented from left to right and the target sentence is generated from bottom to top, one block at a time. A monotone block sequence is generated except for the possibility to handle some local phrase re-ordering. In this local re-ordering model (Tillmann and Zhang, 2005; Kumar and Byrne, 2005) a block b with orientation o is generated relative to its predecessor block b' . During decoding, we maximize the score $s_w(b_1^n, o_1^n)$ of a block orientation sequence

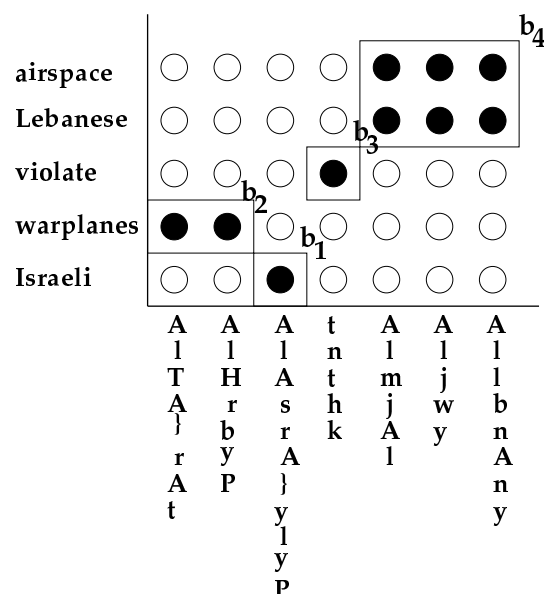


Figure 1: An Arabic-English block translation example, where the Arabic words are romanized. The following orientation sequence is generated: $o_1 = N, o_2 = L, o_3 = N, o_4 = R$.

(b_1^n, o_1^n) :

$$s_w(b_1^n, o_1^n) = \sum_{i=1}^n w^T \cdot f(b_i, o_i, b_{i-1}), \quad (1)$$

where b_i is a block, b_{i-1} is its predecessor block, and $o_i \in \{L(\text{eft}), R(\text{ight}), N(\text{eutral})\}$ is a three-valued orientation component linked to the block b_i : a block is generated to the left or the right of its predecessor block b_{i-1} , where the orientation o_{i-1} of the predecessor block is ignored. Here, n is the number of blocks in the translation. We are interested in learning the weight vector w from the training data. $f(b_i, o_i, b_{i-1})$ is a high-dimensional binary feature representation of the block orientation pair (b_i, o_i, b_{i-1}) . The block orientation se-

quence is generated under the restriction that the concatenated source phrases of the blocks b_i yield the input sentence. In modeling a block sequence, we emphasize adjacent block neighbors that have right or left orientation, since in the current experiments only local block swapping is handled (neutral orientation is used for 'detached' blocks as described in (Tillmann and Zhang, 2005)).

This paper focuses on the discriminative training of the weight vector w used in Eq. 1. The decoding process is decomposed into local decision steps based on Eq. 1, but the model is trained in a global setting as shown below. The advantage of this approach is that it can easily handle tens of millions of features, e.g. up to 35 million features for the experiments in this paper. Moreover, under this view, SMT becomes quite similar to sequential natural language annotation problems such as part-of-speech tagging and shallow parsing, and the novel training algorithm presented in this paper is actually most similar to work on training algorithms presented for these task, e.g. the on-line training algorithm presented in (McDonald et al., 2005) and the perceptron training algorithm presented in (Collins, 2002). The current approach does not use specialized probability features as in (Och, 2003) in any stage during decoder parameter training. Such probability features include language model, translation or distortion probabilities, which are commonly used in current SMT approaches¹. We are able to achieve comparable performance to (Tillmann and Zhang, 2005). The novel algorithm differs computationally from earlier work in discriminative training algorithms for SMT (Och, 2003) as follows:

- No computationally expensive N -best lists are generated during training: for each input sentence a single block sequence is generated on each iteration over the training data.
- No additional development data set is necessary as the weight vector w is trained on bilingual training data only.

The paper is structured as follows: Section 2 presents the baseline block sequence model and the feature representation. Section 3 presents the discriminative training algorithm that learns

¹A translation and distortion model is used in generating the block set used in the experiments, but these translation probabilities are not used during decoding.

a good global ranking function used during decoding. Section 4 presents results on a standard Arabic-English translation task. Finally, some discussion and future work is presented in Section 5.

2 Block Sequence Model

This paper views phrase-based SMT as a block sequence generation process. Blocks are phrase pairs consisting of target and source phrases and local phrase re-ordering is handled by including so-called block orientation. Starting point for the block-based translation model is a block set, e.g. about 9.5 million Arabic-English phrase pairs for the experiments in this paper. This block set is used to decode training sentence to obtain block orientation sequences that are used in the discriminative parameter training. Nothing but the block set and the parallel training data is used to carry out the training. We use the block set described in (Al-Onaizan et al., 2004), the use of a different block set may effect translation results.

Rather than predicting local block neighbors as in (Tillmann and Zhang, 2005), here the model parameters are trained in a global setting. Starting with a simple model, the training data is decoded multiple times: the weight vector w is trained to discriminate block sequences with a high translation score against block sequences with a high BLEU score². The high BLEU scoring block sequences are obtained as follows: the regular phrase-based decoder is modified in a way that it uses the BLEU score as optimization criterion (independent of any translation model). Here, searching for the highest BLEU scoring block sequence is restricted to local re-ordering as is the model-based decoding (as shown in Fig. 1). The BLEU score is computed with respect to the single reference translation provided by the parallel training data. A block sequence with an average BLEU score of about 0.54 is obtained for each training sentence³. The 'true' maximum BLEU block sequence as well as the high scoring

²High scoring block sequences may contain translation errors that are quantified by a lower BLEU score.

³The training BLEU score is computed for each training sentence pair separately (treating each sentence pair as a single-sentence corpus with a single reference) and then averaged over all training sentences. Although block sequences are found with a high BLEU score on average there is no guarantee to find the maximum BLEU block sequence for a given sentence pair. The target word sequence corresponding to a block sequence does not have to match the reference translation, i.e. maximum BLEU scores are quite low for some training sentences.

block sequences are represented by high dimensional feature vectors using the binary features defined below and the translation process is handled as a multi-class classification problem in which each block sequence represents a possible class. The effect of this training procedure can be seen in Figure 2: each decoding step on the training data adds a high-scoring block sequence to the discriminative training and decoding performance on the training data is improved after each iteration (along with the test data decoding performance). A theoretical justification for the novel training procedure is given in Section 3.

We now define the feature components for the block bigram feature vector $f(b_i, o_i, b_{i-1})$ in Eq. 1. Although the training algorithm can handle real-valued features as used in (Och, 2003; Tillmann and Zhang, 2005) the current paper intentionally excludes them. The current feature functions are similar to those used in common phrase-based translation systems: for them it has been shown that good translation performance can be achieved⁴. A systematic analysis of the novel training algorithm will allow us to include much more sophisticated features in future experiments, i.e. POS-based features, syntactic or hierarchical features (Chiang, 2005). The dimensionality of the feature vector $f(b_i, o_i, b_{i-1})$ depends on the number of binary features. For illustration purposes, the binary features are chosen such that they yield 1 on the example block sequence in Fig. 1. There are **phrase-based** and **word-based** features:

$$\begin{aligned} f_{1000}(b_i, o_i, b_{i-1}) &= \\ &= \begin{cases} 1 & \text{block } b_i \text{ consists of target phrase} \\ & \text{'violate' and source phrase 'tnthk'} \\ 0 & \text{otherwise} \end{cases} \\ f_{1001}(b_i, o_i, b_{i-1}) &= \\ &= \begin{cases} 1 & \text{'Lebanese' is a word in the target} \\ & \text{phrase of block } b_i \text{ and 'AllbnAny'} \\ & \text{is a word in the source phrase} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

The feature f_{1000} is a 'unigram' phrase-based feature capturing the identity of a block. Additional phrase-based features include block orientation, target and source phrase bigram features. Word-based features are used as well, e.g. feature f_{1001} captures word-to-word translation de-

pendencies similar to the use of Model 1 probabilities in (Koehn et al., 2003). Additionally, we use distortion features involving relative source word position and m -gram features for adjacent target words. These features correspond to the use of a language model, but the weights for these features are trained on the parallel training data only. For the most complex model, the number of features is about 35 million (ignoring all features that occur only once).

3 Approximate Relevant Set Method

Throughout the section, we let $\mathbf{z} = (b_1^n, o_1^n)$. Each block sequence $\mathbf{z} = (b_1^n, o_1^n)$ corresponds to a candidate translation. In the training data where target translations are given, a BLEU score $\text{Bl}(\mathbf{z})$ can be calculated for each $\mathbf{z} = (b_1^n, o_1^n)$ against the target translations. In this set up, our goal is to find a weight vector w such that the higher $s_w(\mathbf{z})$ is, the higher the corresponding BLEU score $\text{Bl}(\mathbf{z})$ should be. If we can find such a weight vector, then block decoding by searching for the highest $s_w(\mathbf{z})$ will lead to good translation with high BLEU score.

Formally, we denote a source sentence by $\mathbf{S}$, and let $V(\mathbf{S})$ be the set of possible candidate oriented block sequences $\mathbf{z} = (b_1^n, o_1^n)$ that the decoder can generate from $\mathbf{S}$. For example, in a monotone decoder, the set $V(\mathbf{S})$ contains block sequences $\{b_1^n\}$ that cover the source sentence $\mathbf{S}$ in the same order. For a decoder with local re-ordering, the candidate set $V(\mathbf{S})$ also includes additional block sequences with re-ordered block configurations that the decoder can efficiently search. Therefore depending on the specific implementation of the decoder, the set $V(\mathbf{S})$ can be different. In general, $V(\mathbf{S})$ is a subset of all possible oriented block sequences $\{(b_1^n, o_1^n)\}$ that are consistent with input sentence $\mathbf{S}$.

Given a scoring function $s_w(\cdot)$ and an input sentence $\mathbf{S}$, we can assume that the decoder implements the following decoding rule:

$$\hat{\mathbf{z}}(\mathbf{S}) = \arg \max_{\mathbf{z} \in V(\mathbf{S})} s_w(\mathbf{z}). \quad (2)$$

Let $\mathbf{S}_1, \dots, \mathbf{S}_N$ be a set of N training sentences. Each sentence $\mathbf{S}_i$ is associated with a set $V(\mathbf{S}_i)$ of possible translation block sequences that are searchable by the decoder. Each translation block sequence $\mathbf{z} \in V(\mathbf{S}_i)$ induces a translation, which is then assigned a BLEU score $\text{Bl}(\mathbf{z})$ (obtained by comparing against the target translations). The

⁴On our test set, (Tillmann and Zhang, 2005) reports a BLEU score of 37.8 and (Ittycheriah and Roukos, 2005) reports a BLEU score of 48.0.

goal of the training is to find a weight vector w such that for each training sentence $\mathbf{S}_i$, the corresponding decoder outputs $\hat{\mathbf{z}} \in V(\mathbf{S}_i)$ which has the maximum BLEU score among all $\mathbf{z} \in V(\mathbf{S}_i)$ based on Eq. 2. In other words, if $\hat{\mathbf{z}}$ maximizes the scoring function $s_w(\mathbf{z})$, then $\hat{\mathbf{z}}$ also maximizes the BLEU metric.

Based on the description, a simple idea is to learn the BLEU score $\text{Bl}(\mathbf{z})$ for each candidate block sequence $\mathbf{z}$. That is, we would like to estimate w such that $s_w(\mathbf{z}) \approx \text{Bl}(\mathbf{z})$. This can be achieved through least squares regression. It is easy to see that if we can find a weight vector w that approximates $\text{Bl}(\mathbf{z})$, then the decoding-rule in Eq. 2 automatically maximizes the BLEU score. However, it is usually difficult to estimate $\text{Bl}(\mathbf{z})$ reliably based only on a linear combination of the feature vector as in Eq. 1. We note that a good decoder does not necessarily employ a scoring function that approximates the BLEU score. Instead, we only need to make sure that the top-ranked block sequence obtained by the decoder scoring function has a high BLEU score. To formulate this idea, we attempt to find a decoding parameter such that for each sentence $\mathbf{S}$ in the training data, sequences in $V(\mathbf{S})$ with the highest BLEU scores should get $s_w(\mathbf{z})$ scores higher than those with low BLEU scores.

Denote by $V_K(\mathbf{S})$ a set of K block sequences in $V(\mathbf{S})$ with the highest BLEU scores. Our decoded result should lie in this set. We call them the “truth”. The set of the remaining sequences is $V(\mathbf{S}) - V_K(\mathbf{S})$, which we shall refer to as the “alternatives”. We look for a weight vector w that minimize the following training criterion:

$$\begin{aligned} \hat{w} = \arg \min_w & \left[\frac{1}{N} \sum_{i=1}^N \Phi(w, V_K(\mathbf{S}_i), V(\mathbf{S}_i)) \right. \\ & \left. + \lambda w^2 \right] \quad (3) \\ \Phi(w, V_K, V) &= \frac{1}{K} \sum_{\mathbf{z} \in V_K} \max_{\mathbf{z}' \in V - V_K} \psi(w, \mathbf{z}, \mathbf{z}') \\ \psi(w, \mathbf{z}, \mathbf{z}') &= \phi(s_w(\mathbf{z}), \text{Bl}(\mathbf{z}); s_w(\mathbf{z}'), \text{Bl}(\mathbf{z}')), \end{aligned}$$

where ϕ is a non-negative real-valued loss function (whose specific choice is not critical for the purposes of this paper), and $\lambda \geq 0$ is a regularization parameter. In our experiments, results are obtained using the following convex loss

$$\phi(s, b; s', b') = (b - b')(1 - (s - s'))_+^2, \quad (4)$$

where b, b' are BLEU scores, s, s' are translation scores, and $(z)_+ = \max(0, z)$. We refer to this formulation as ‘costMargin’ (cost-sensitive margin) method: for each training sentence $\mathbf{S}$ the ‘costMargin’ $\Phi(w, V_K(\mathbf{S}), V(\mathbf{S}))$ between the ‘true’ block sequence set $V_K(\mathbf{S})$ and the ‘alternative’ block sequence set $V(\mathbf{S})$ is maximized. Note that due to the truth and alternative set up, we always have $b > b'$. This loss function gives an upper bound of the error we will suffer if the order of s and s' is wrongly predicted (that is, if we predict $s \leq s'$ instead of $s > s'$). It also has the property that if for the BLEU scores $b \approx b'$ holds, then the loss value is small (proportional to $b - b'$).

A major contribution of this work is a procedure to solve Eq. 3 approximately. The main difficulty is that the search space $V(\mathbf{S})$ covered by the decoder can be extremely large. It cannot be enumerated for practical purposes. Our idea is to replace this large space by a small subspace $V^{(r)}(\mathbf{S}) \subset V(\mathbf{S})$ which we call *relevant set*. The possibility of this reduction is based on the following theoretical result.

Lemma 1 *Let $\psi(w, \mathbf{z}, \mathbf{z}')$ be a non-negative continuous piece-wise differentiable function of w , and let $\hat{w}$ be a local solution of Eq. 3. Let $\xi_i(w, \mathbf{z}) = \max_{\mathbf{z}' \in V(\mathbf{S}_i) - V_K(\mathbf{S}_i)} \psi(w, \mathbf{z}, \mathbf{z}')$, and define*

$$\begin{aligned} V^{(r)}(\mathbf{S}_i) &= \{\mathbf{z}' \in V(\mathbf{S}_i) : \exists \mathbf{z} \in V_K(\mathbf{S}_i) \text{ s.t.} \\ & \xi_i(\hat{w}, \mathbf{z}) \neq 0 \text{ \& } \psi(\hat{w}, \mathbf{z}, \mathbf{z}') = \xi_i(\hat{w}, \mathbf{z})\}. \end{aligned}$$

Then $\hat{w}$ is a local solution of

$$\min_w \left[\frac{1}{N} \sum_{i=1}^N \Phi(w, V_K(\mathbf{S}_i), V^{(r)}(\mathbf{S}_i)) + \lambda w^2 \right]. \quad (5)$$

If ϕ is a convex function of w (as in our choice), then we know that the global optimal solution remains the same if the whole decoding space V is replaced by the relevant set $V^{(r)}$.

Each subspace $V^{(r)}(\mathbf{S}_i)$ will be significantly smaller than $V(\mathbf{S}_i)$. This is because it only includes those alternatives $\mathbf{z}'$ with score $s_{\hat{w}}(\mathbf{z}')$ close to one of the selected truth. These are the most important alternatives that are easily confused with the truth. Essentially the lemma says that if the decoder works well on these difficult alternatives (relevant points), then it works well on the whole space. The idea is closely related to active learning in standard classification problems, where we

Table 1: Generic Approximate Relevant Set Method

for each data point $\mathbf{S}$ initialize truth $V_K(\mathbf{S})$ and alternative $V^{(r)}(\mathbf{S})$ for each decoding iteration $l: \ell = 1, \dots, L$ for each data point $\mathbf{S}$ select relevant points $\{\tilde{\mathbf{z}}_k\} \in V(\mathbf{S})$ (*) update $V^{(r)}(\mathbf{S}) \leftarrow V^{(r)}(\mathbf{S}) \cup \{\tilde{\mathbf{z}}_k\}$ update w by solving Eq. 5 approximately (**)

selectively pick the most important samples (often based on estimation uncertainty) for labeling in order to maximize classification performance (Lewis and Catlett, 1994). In the active learning setting, as long as we do well on the actively selected samples, we do well on the whole sample space. In our case, as long as we do well on the relevant set, the decoder will perform well.

Since the relevant set depends on the decoder parameter w , and the decoder parameter is optimized on the relevant set, it is necessary to estimate them jointly using an iterative algorithm. The basic idea is to start with a decoding parameter w , and estimate the corresponding relevant set; we then update w based on the relevant set, and iterate this process. The procedure is outlined in Table 1. We intentionally leave the implementation details of the (*) step and (**) step open. Moreover, in this general algorithm, we do not have to assume that $s_w(\mathbf{z})$ has the form of Eq. 1.

A natural question concerning the procedure is its convergence behavior. It can be shown that under mild assumptions, if we pick in (*) an alternative $\tilde{\mathbf{z}}_k \in V(\mathbf{S}) - V_K(\mathbf{S})$ for each $\mathbf{z}_k \in V_K(\mathbf{S})$ ($k = 1, \dots, K$) such that

$$\psi(w, \mathbf{z}_k, \tilde{\mathbf{z}}_k) = \max_{\mathbf{z}' \in V(\mathbf{S}) - V_K(\mathbf{S})} \psi(w, \mathbf{z}_k, \mathbf{z}'), \quad (6)$$

then the procedure converges to the solution of Eq. 3. Moreover, the rate of convergence depends only on the property of the loss function, and not on the size of $V(\mathbf{S})$. This property is critical as it shows that as long as Eq. 6 can be computed efficiently, then the Approximate Relevant Set algorithm is efficient. Moreover, it gives a bound on the size of an approximate relevant set with a certain accuracy.⁵

⁵Due to the space limitation, we will not include a for-

The approximate solution of Eq. 5 in (**) can be implemented using stochastic gradient descent (SGD), where we may simply update w as:

$$w \rightarrow w - \eta \nabla_w \psi(w, \mathbf{z}_k, \tilde{\mathbf{z}}_k).$$

The parameter $\eta > 0$ is a fixed constant often referred to as learning rate. Again, convergence results can be proved for this procedure. Due to the space limitation, we skip the formal statement as well as the corresponding analysis.

Up to this point, we have not assumed any specific form of the decoder scoring function in our algorithm. Now consider Eq. 1 used in our model. We may express it as:

$$s_w(\mathbf{z}) = w^T \cdot F(\mathbf{z}),$$

where $F(\mathbf{z}) = \sum_{i=1}^n f(b_i, o_i, b_{i-1})$. Using this feature representation and the loss function in Eq. 4, we obtain the following costMargin SGD update rule for each training data point and k :

$$w \rightarrow w + \eta \Delta \text{Bl}_k x_k (1 - w^T \cdot x_k)_+, \quad (7)$$

$$\Delta \text{Bl}_k = \text{Bl}(\mathbf{z}_k) - \text{Bl}(\tilde{\mathbf{z}}_k), \quad x_k = F(\mathbf{z}_k) - F(\tilde{\mathbf{z}}_k).$$

4 Experimental Results

We applied the novel discriminative training approach to a standard Arabic-to-English translation task. The training data comes from UN news sources. Some punctuation tokenization and some number classing are carried out on the English and the Arabic training data. We show translation results in terms of the automatic BLEU evaluation metric (Papineni et al., 2002) on the MT03 Arabic-English DARPA evaluation test set consisting of 663 sentences with 16 278 Arabic words with 4 reference translations. In order to speed up the parameter training the original training data is filtered according to the test set: all the Arabic substrings that occur in the test set are computed and the parallel training data is filtered to include only those training sentence pairs that contain at least one out of these phrases: the resulting pre-filtered training data contains about 230 thousand sentence pairs (5.52 million Arabic words and 6.76 million English words). The block set is generated using a phrase-pair selection algorithm similar to (Koehn et al., 2003; Al-Onaizan et al., 2004), which includes some heuristic filtering to

mal statement here. A detailed theoretical investigation of the method will be given in a journal paper.

increase phrase translation accuracy. Blocks that occur only once in the training data are included as well.

4.1 Practical Implementation Details

The training algorithm in Table 2 is adapted from Table 1. The training is carried out by running $L = 30$ times over the parallel training data, each time decoding all the $N = 230\,000$ training sentences and generating a single block translation sequence for each training sentence. The top five block sequences $V_5(\mathbf{S}_i)$ with the highest BLEU score are computed up-front for all training sentence pairs $\mathbf{S}_i$ and are stored separately as described in Section 2. The score-based decoding of the 230 000 training sentence pairs is carried out in parallel on 25 64-Bit Opteron machines. Here, the monotone decoding is much faster than the decoding with block swapping: the monotone decoding takes less than 0.5 hours and the decoding with swapping takes about an hour. Since the training starts with only the parallel training data and a block set, some initial block sequences have to be generated in order to initialize the global model training: for each input sentence a simple bag of blocks translation is generated. For each input interval that is matched by some block b , a single block is added to the bag-of-blocks translation $\mathbf{z}_0(\mathbf{S})$. The order in which the blocks are generated is ignored. For this block set only block and word identity features are generated, i.e. features of type f_{1000} and f_{1001} in Section 2. This step does not require the use of a decoder. The initial block sequence training data contains only a single alternative. The training procedure proceeds by iteratively decoding the training data. After each decoding step, the resulting translation block sequences are stored on disc in binary format. A block sequence generated at decoding step ℓ_1 is used in all subsequent training steps ℓ_2 , where $\ell_2 > \ell_1$. The block sequence training data after the l -th decoding step is given as $\left[(V_5(\mathbf{S}_i), V^{(r)}(\mathbf{S}_i)) \right]_{i=1}^N$, where the size $|V^{(r)}(\mathbf{S}_i)|$ of the relevant alternative set is $l + 1$. Although in order to achieve fast convergence with a theoretical guarantee, we should use Eq. 6 to update the relevant set, in reality, this idea is difficult to implement because it requires a more costly decoding step. Therefore in Table 2, we adopt an approximation, where the relevant set is updated by adding the decoder output at each stage. In this way, we are able to treat the decoding

Table 2: Relevant set method: L = number of decoding iterations, N = number of training sentences.

```

for each input sentence  $\mathbf{S}_i, i = 1, \dots, N$ 
    initialize truth  $V_5(\mathbf{S}_i)$  and alter-
    native  $V^{(r)} = \{\mathbf{z}_0(\mathbf{S}_i)\}$ 
for each decoding iteration  $l: \ell = 1, \dots, L$ 
    train  $w$  using SGD on training
    data  $\left[ (V_5(\mathbf{S}_i), V^{(r)}(\mathbf{S}_i)) \right]_{i=1}^N$ 
    for each input sentence  $\mathbf{S}_i, i = 1, \dots, N$ 
        select top-scoring sequence  $\tilde{\mathbf{z}}(\mathbf{S}_i)$  and
        update  $V^{(r)}(\mathbf{S}_i) \leftarrow V^{(r)}(\mathbf{S}_i) \cup \{\tilde{\mathbf{z}}(\mathbf{S}_i)\}$ 

```

scheme as a black box. One way to approximate Eq. 6 is to generate multiple decoding outputs and pick the most relevant points based on Eq. 6. Since the n -best list generation is computationally costly, only a single block sequence is generated for each training sentence pair, reducing the memory requirements for the training algorithm as well. Although we are not able to rigorously prove fast convergence rate for this approximation, it works well in practice, as Figure 2 shows. Theoretically this is because points achieving large values in Eq. 6 tend to have higher chances to become the top-ranked decoder output as well. The SGD-based on-line training algorithm described in Section 3, is carried out after each decoding step to generate the weight vector w for the subsequent decoding step. Since this training step is carried out on a single machine, it dominates the overall computation time. Since each iteration adds a single relevant alternative to the set $V^{(r)}(\mathbf{S}_i)$, computation time increases with the number of training iterations: the initial model is trained in a few minutes, while training the model after the 30-th iteration takes up to 5 hours for the most complex models.

Table 3 presents experimental results in terms of uncased BLEU ⁶. Two re-ordering restrictions are tested, i.e. monotone decoding ('MON'), and local block re-ordering where neighbor blocks can be swapped ('SWAP'). The 'SWAP' re-ordering uses the same features as the monotone models plus additional orientation-based and distortion-

⁶Translation performance in terms of cased BLEU is typically reduced by about 2 %.

Table 3: Translation results in terms of uncased BLEU on the training data (230 000 sentences) and the MT03 test data (670 sentences).

	Re-ordering	Features	train	test
1	'MON'	bleu	0.542	-
2		phrase	0.378	0.256
3		word	0.427	0.341
4		both	0.477	0.359
5	'SWAP'	bleu	0.594	-
6		phrase	0.441	0.295
7		word	0.455	0.359
8		both	0.479	0.363

based features. Different feature sets include word-based features, phrase-based features, and the combination of both. For the results with word-based features, the decoder still generates phrase-to-phrase translations, but all the scoring is done on the word level. Line 8 shows a BLEU score of 36.3 for the best performing system which uses all word-based and phrase-based features⁷. Line 1 and line 5 of Table 3 show the training data averaged BLEU score obtained by searching for the highest BLEU scoring block sequence for each training sentence pair as described in Section 2. Allowing local block swapping in this search procedure yields a much improved BLEU score of 0.59. The experimental results show that word-based models significantly outperform phrase-based models, the combination of word-based and phrase-based features performs better than those features types taken separately. Additionally, swap-based re-ordering slightly improves performance over monotone decoding. For all experiments, the training BLEU score remains significantly lower than the maximum obtainable BLEU score shown in line 1 and line 5. In this respect, there is significant room for improvements in terms of feature functions and alternative set generation. The word-based models perform surprisingly well, i.e. the model in line 7 uses only three feature types: model 1 features like f_{1001} in Section 2, distortion features, and target language m-gram features up to $m = 3$. Training speed varies depending on the feature types used: for the simplest model shown in line 2 of Table 3, the training takes about 12 hours, for the models us-

⁷With a margin of ± 0.014 , the differences between the results in line 4, line 7, and line 8 are not statistically significant, but the other result differences are.

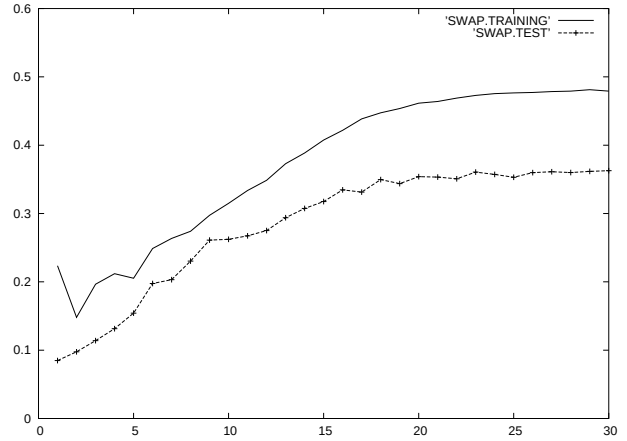


Figure 2: BLEU performance on the training set (upper graph; averaged BLEU with single reference) and the test set (lower graph; BLEU with four references) as a function of the training iteration I for the model corresponding to line 8 in Table 3.

ing word-based features shown in line 3 and line 7 training takes less than 2 days. Finally, the training for the most complex model in line 8 takes about 4 days.

Figure 2 shows the BLEU performance for the model corresponding to line 8 in Table 3 as a function of the number of training iterations. By adding top scoring alternatives in the training algorithm in Table 2, the BLEU performance on the training data improves from about 0.22 for the initial model to about 0.48 for the best model after 30 iterations. After each training iteration the test data is decoded as well. Here, the BLEU performance improves from 0.08 for the initial model to about 0.36 for the final model (we do not include the test data block sequences in the training). Table 3 shows a typical learning curve for the experiments in Table 3: the training BLEU score is much higher than the test set BLEU score despite the fact that the test set uses 4 reference translations.

5 Discussion and Future Work

The work in this paper substantially differs from previous work in SMT based on the noisy channel approach presented in (Brown et al., 1993). While error-driven training techniques are commonly used to improve the performance of phrase-based translation systems (Chiang, 2005; Och, 2003), this paper presents a novel block sequence translation approach to SMT that is similar to sequential natural language annotation problems

such as part-of-speech tagging or shallow parsing, both in modeling and parameter training. Unlike earlier approaches to SMT training, which either rely heavily on domain knowledge, or can only handle a small number of features, this approach treats the decoding process as a black box, and can optimize tens millions of parameters automatically, which makes it applicable to other problems as well. The choice of our formulation is convex, which ensures that we are able to find the global optimum even for large scale problems. The loss function in Eq. 4 may not be optimal, and using different choices may lead to future improvements. Another important direction for performance improvement is to design methods that better approximate Eq. 6. Although at this stage the system performance is not yet better than previous approaches, good translation results are achieved on a standard translation task. While being similar to (Tillmann and Zhang, 2005), the current procedure is more automated with comparable performance. The latter approach requires a decomposition of the decoding scheme into local decision steps with the inherent difficulty acknowledged in (Tillmann and Zhang, 2005). Since such limitation is not present in the current model, improved results may be obtained in the future. A perceptron-like algorithm that handles global features in the context of re-ranking is also presented in (Shen et al., 2004).

The computational requirements for the training algorithm in Table 2 can be significantly reduced. While the global training approach presented in this paper is simple, after 15 iterations or so, the alternatives that are being added to the relevant set differ very little from each other, slowing down the training considerably such that the set of possible block translations $V(S)$ might not be fully explored. As mentioned in Section 2, the current approach is still able to handle real-valued features, e.g. the language model probability. This is important since the language model can be trained on a much larger monolingual corpus.

6 Acknowledgment

This work was partially supported by the GALE project under the DARPA contract No. HR0011-06-2-0001. The authors would like to thank the anonymous reviewers for their detailed criticism on this paper.

References

- Yaser Al-Onaizan, Niyu Ge, Young-Suk Lee, Kishore Papineni, Fei Xia, and Christoph Tillmann. 2004. IBM Site Report. In *NIST 2004 MT Workshop*, Alexandria, VA, June. IBM.
- Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *CL*, 19(2):263–311.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL 2005*, pages 263–270, Ann Arbor, Michigan, June.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. EMNLP’02*, Philadelphia, PA.
- A. Ittycheriah and S. Roukos. 2005. A Maximum Entropy Word Aligner for Arabic-English MT. In *Proc. of HLT-EMNLP 06*, pages 89–96, Vancouver, British Columbia, Canada, October.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL 2003: Main Proceedings*, pages 127–133, Edmonton, Alberta, Canada, May 27 - June 1.
- Shankar Kumar and William Byrne. 2005. Local phrase reordering models for statistical machine translation. In *Proc. of HLT-EMNLP 05*, pages 161–168, Vancouver, British Columbia, Canada, October.
- D. Lewis and J. Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL’05*, pages 91–98, Ann Arbor, Michigan, June.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL’03*, pages 160–167, Sapporo, Japan.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of machine translation. In *In Proc. of ACL’02*, pages 311–318, Philadelphia, PA, July.
- Libin Shen, Anoop Sarkar, and Franz-Josef Och. 2004. Discriminative Reranking of Machine Translation. In *Proceedings of the Joint HLT and NAACL Conference (HLT 04)*, pages 177–184, Boston, MA, May.
- Christoph Tillmann and Tong Zhang. 2005. A localized prediction model for statistical machine translation. In *Proceedings of ACL’05*, pages 557–564, Ann Arbor, Michigan, June.

Phoneme-to-Text Transcription System with an Infinite Vocabulary

Shinsuke Mori Daisuke Takuma Gakuto Kurata

IBM Research, Tokyo Research Laboratory, IBM Japan, Ltd.
1623-14 Shimotsuruma Yamato-shi, 242-8502, Japan
mori@fw.ipsj.or.jp

Abstract

The noisy channel model approach is successfully applied to various natural language processing tasks. Currently the main research focus of this approach is adaptation methods, how to capture characteristics of words and expressions in a target domain given example sentences in that domain. As a solution we describe a method enlarging the vocabulary of a language model to an almost infinite size and capturing their context information. Especially the new method is suitable for languages in which words are not delimited by whitespace. We applied our method to a phoneme-to-text transcription task in Japanese and reduced about 10% of the errors in the results of an existing method.

1 Introduction

The noisy channel model approach is being successfully applied to various natural language processing (NLP) tasks, such as speech recognition (Jelinek, 1985), spelling correction (Kernighan et al., 1990), machine translation (Brown et al., 1990), etc. In this approach an NLP system is composed of two modules: one is a task-dependent part (an acoustic model for speech recognition) which describes a relationship between an input signal sequence and a word, the other is a language model (LM) which measures the likelihood of a sequence of words as a sentence in the language. Since the LM is a common part, its improvement augments the accuracies of all NLP systems based on a noisy channel model.

Recently the main research focus of LM is shifting to the adaptation method, how to capture the characteristics of words and expressions in a target domain. The standard adaptation method is to prepare a corpus in the application domain, count

the frequencies of words and word sequences, and manually annotate new words with their input signal sequences to be added to the vocabulary. It is now easy to gather machine-readable sentences in various domains because of the ease of publication and access via the Web (Kilgariff and Grefenstette, 2003). In addition, traditional machine-readable forms of medical reports or business reports are also available. When we need to develop an NLP system in various domains, there is a huge but unannotated corpus.

For languages, such as Japanese and Chinese, in which the words are not delimited by whitespace, one encounters a word identification problem before counting the frequencies of words and word sequences. To solve this problem one must have a good word segmenter in the domain of the corpus. The only robust and reliable word segmenter in the domain is, however, a word segmenter based on the statistics of the lexicons in the domain! Thus we are obliged to pay a high cost for the manual annotation of a corpus for each new subject domain.

In this paper, we propose a novel framework for building an NLP system based on a noisy channel model with an almost infinite vocabulary. In our method, first we estimate the probability of a word boundary existing between two characters at each point of a raw corpus in the target domain. Using these probabilities we regard the corpus as a stochastically segmented corpus (SSC). We then estimate word n -gram probabilities from the SSC. Then we build an NLP system, the phoneme-to-text transcription system in this paper. To describe the stochastic relationship between a character sequence and its phoneme sequence, we also propose a character-based unknown word model. With this unknown word model and a word n -gram model estimated from the SSC, the vocabulary of our LM, a set of known words with their context information, is expanded from words in a

small annotated corpus to an almost infinite size, including all substrings appearing in the large corpus in the target domain. In experiments, we estimated LMs from a relatively small annotated corpus in the general domain and a large raw corpus in the target domain. A phoneme-to-text transcription system based on our LM and unknown word model eliminated about 10% of the errors in the results of an existing method.

2 Task Complexity

In this section we explain the phoneme-to-text transcription task which our new framework is applied to.

2.1 Phoneme-to-text Transcription

To input a sentence in a language using a device with fewer keys than the alphabet we need some kind of transcription system. In French stenotypy, for example, a special keyboard with 21 keys is used to input French letters with accents (Derouault and Merialdo, 1986). A similar problem arises when we write an e-mail in any language with a mobile phone or a PDA. For languages with a much larger character set, such as Chinese, Japanese, and Korean, a transcription system called an input method is indispensable for writing on a computer (Lunde, 1998).

The task we chose for the evaluation of our method is phoneme-to-text transcription in Japanese, which can also be regarded as a pseudo-speech recognition in which the acoustic model is perfect. In order to input Japanese to a computer, the user types phoneme sequences and the computer offers possible transcription candidates in the descending order of their estimated similarities to the characters the user wants to input.¹ Then the user chooses the proper one.

2.2 Ambiguities

A phoneme sequence in Japanese (written in sans-serif font in this paper) is highly ambiguous for a computer. There are many possible word sequences with similar pronunciations. These ambiguities are mainly due to three factors:

- **Homonyms:** There are many words sharing the same phoneme sequences. In the spoken language, they are less ambiguous since they are

¹ Generally one of Japanese phonogram sets is used as phoneme. A phonogram is input by a combination of unambiguous ASCII characters.

pronounced with different intonations. Intonational signals are, however, omitted in the input of phoneme-to-text transcription.

- **Lack of word boundaries:** A word of a long sequence of phonemes can be split into several shorter words, such as frequent content words, particles, etc. (ex. a-ri-ga-to-u/thanks vs. a-ri/ant ga/is to-u/ten).
- **Variations in writing:** Some words have more than one acceptable spellings. For example, 振り込み/fu-ri-ko-mi/bank-transfer is often written as 振込/fu-ri-ko-mi omitting two verbal endings, especially in business writing.

Most of these ambiguities are not difficult to resolve for a native speaker who is familiar with the domain. So the transcription system should offer the candidate word sequences for each context and domain.

2.3 Available Resources

Generally speaking, three resources are available for a phoneme-to-text transcription based on the noisy channel model:

- **annotated corpus:** a small corpus in the general domain annotated with word boundary information and phoneme sequences for each word
- **single character dictionary:** a dictionary containing all possible phoneme sequences for each single character
- **raw corpus in the target domain:** a collection of text samples in the target domain extracted from the Web or documents in machine-readable form

3 Language Model and its Application

A stochastic LM M is a function from a sequence of characters $x \in \mathcal{X}^*$ to the probability. The summation over all possible sequences of characters must be equal to or less than 1. This probability is used as the likelihood in the NLP system.

3.1 Word N -gram Model

The most famous LM is an n -gram model based on words. In this model, a sentence is regarded as a word sequence $w_1^h (= w_1 w_2 \cdots w_h)$ and words are predicted from beginning to end:

$$M_{w,n}(w) = \prod_{i=1}^{h+1} P(w_i | w_{i-n+1}^{i-1}),$$

where w_i ($i \leq 0$) and w_{h+1} is a special symbol called a BT (boundary token). Since it is impossible to define the complete vocabulary, we prepare a special token UW for unknown words and an unknown word spelling $x_1^{h'}$ is predicted by the following character-based n -gram model after UW is predicted by $M_{w,n}$:

$$M_{x,n}(x_1^{h'}) = \prod_{i=1}^{h'+1} P(x_i | x_{i-n+1}^{i-1}), \quad (1)$$

where x_i ($i \leq 0$) and $x_{h'+1}$ is a special symbol BT. Thus, when w_i is outside of the vocabulary $\mathcal{W}$,

$$P(w_i | w_{i-n+1}^{i-1}) = M_{x,n}(w_i)P(\text{UW} | w_{i-n+1}^{i-1}).$$

3.2 Automatic Word Segmentation

Nagata (1994) proposed a stochastic word segmenter based on a word n -gram model to solve the word segmentation problem. According to this method, the word segmenter divides a sentence x into a word sequence with the highest probability

$$\hat{w} = \underset{w=x}{\operatorname{argmax}} M_{w,n}(w).$$

Nagata (1994) reported an accuracy of about 97% on a test corpus in the same domain using a learning corpus of 10,945 sentences in Japanese.

3.3 Phoneme-to-text Transcription

A phoneme-to-text transcription system based on an LM T (Mori et al., 1999) receives a phoneme sequence y and returns a list of candidate sentences $(x_1, x_2, \dots)$ in descending order of the probability $P(x|y)$:

$$T(y) = (x_1, x_2, \dots), \\ \text{where } i \leq j \Leftrightarrow P(x_i|y) \geq P(x_j|y).$$

Similar to speech recognition, the probability is decomposed into two independent parts: a pronunciation model (PM) and an LM.

$$\begin{aligned} P(x_i|y) &\geq P(x_j|y) \\ \Leftrightarrow \frac{P(y|x_i)P(x_i)}{P(y)} &\geq \frac{P(y|x_j)P(x_j)}{P(y)} \\ \Leftrightarrow P(y|x_i)P(x_i) &\geq P(y|x_j)P(x_j) \quad (2) \\ (\because P(y) &\text{ is independent of } x_i \text{ and } x_j.) \end{aligned}$$

In this formula $P(x)$ is an LM representing the likelihood of a sentence x . For the LM, we can use a word n -gram model we explained above.

The other part in the above formula $P(y|x)$ is a PM representing the probability that a given sentence x is pronounced as y . Since it is impossible to collect the phoneme sequences y for all possible sentences x , the model is decomposed into a word-based model M_y in which the words are pronounced independently

$$M_{y,w}(y|w) = \prod_{i=1}^h P(y_i|w_i), \quad (3)$$

where y_i is a phoneme sequence corresponding to the word w_i and the condition $y = y_1^h$ is met.

The probabilities $P(y_i|w_i)$ are estimated from a corpus in which each word is annotated with a phoneme sequence as follows:

$$P(y_i|w_i) = \frac{f(y_i, w_i)}{f(w_i)}, \quad (4)$$

where $f(e)$ stands for the frequency of an event e in the corpus. For unknown words no transcription model has been proposed and the phoneme-to-text transcription system (Mori et al., 1999) simply returns the phoneme sequence itself.² This is done by replacing the unknown word model based on the Japanese character set $M_{x,n}(x)$ by a model based on the phonemic alphabet $M_{y,n}(y)$.

Thus the candidate evaluation metric of a phoneme-to-text transcription (Mori et al., 1999) composed of the word n -gram model and the word-based pronunciation model is as follows:

$$\begin{aligned} P(y|x)P(x) &= \prod_{i=1}^h P(y_i|w_i)P(w_i) \\ P(y_i|w_i)P(w_i) &= \begin{cases} P(w_i|w_{i-n+1}^{i-1})P(y_i|w_i) & \text{if } w_i \in \mathcal{W}, \\ P(\text{UW}|w_{i-n+1}^{i-1})M_{y,n}(y_i) & \text{if } w_i \notin \mathcal{W}. \end{cases} \end{aligned} \quad (5)$$

4 LM Estimation from a Stochastically Segmented Corpus (SSC)

To cope with segmentation errors, the concept of stochastic segmentation is proposed (Mori and Takuma, 2004). In this section, we briefly explain a method of calculating word n -gram probabilities on a stochastically segmented corpus in the target domain. For a detailed explanation and proofs of the mathematical soundness, please refer to the paper (Mori and Takuma, 2004).

² One of the Japanese syllabaries *Katakana* is used to spell out imported words by imitating their Japanese-constrained pronunciation and the phoneme sequence itself is the correct transcription result for them. Mori et. al. (1999) reported that approximately 33.0% of the unknown words in a test corpus were imported words.

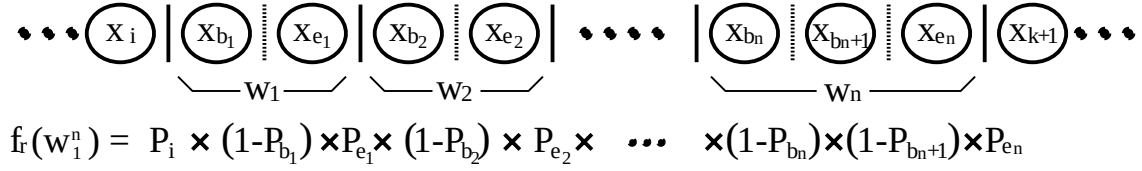


Figure 1: Word n -gram frequency in a stochastically segmented corpus (SSC).

4.1 Stochastically Segmented Corpus (SSC)

A stochastically segmented corpus (SSC) is defined as a combination of a raw corpus C_r (hereafter referred to as the character sequence $\mathbf{x}_1^{n_r}$) and word boundary probabilities P_i that a word boundary exists between two characters x_i and x_{i+1} . Since there are word boundaries before the first character and after the last character of the corpus, $P_0 = P_{n_r} = 1$.

In (Mori and Takuma, 2004), the word boundary probabilities are defined as follows. First the word boundary estimation accuracy α of an automatic word segmenter is calculated on a test corpus with word boundary information. Then the raw corpus is segmented by the word segmenter. Finally P_i is set to be α for each i where the word segmenter put a word boundary and P_i is set to be $1 - \alpha$ for each i where it did not put a word boundary. We adopted the same method in the experiments.

4.2 Word n -gram Frequency

Word n -gram frequencies on an SSC is calculated as follows:

Word 0-gram frequency: This is defined as an expected number of words in the SSC:

$$f(\cdot) = 1 + \sum_{i=1}^{n_r-1} P_i.$$

Word n -gram frequency ($n \geq 1$): Let us think of a situation (see Figure 1) in which a word sequence $\mathbf{w}_1^n$ occurs in the SSC as a subsequence beginning at the $(i+1)$ -th character and ending at the k -th character and each word w_m in the word sequence is equal to the character sequence beginning at the b_m -th character and ending at the e_m -th character ($\mathbf{x}_{b_m}^{e_m} = w_m$, $1 \leq \forall m \leq n$; $e_m + 1 = b_{m+1}$, $1 \leq \forall m \leq n-1$; $b_1 = i+1$; $e_n = k$). The word n -gram frequency of a word sequence $f_r(\mathbf{w}_1^n)$ in the SSC is defined by the summation of the stochastic frequency at each occurrence of the character sequence of the word sequence $\mathbf{w}_1^n$ over all of the

occurrences in the SSC:

$$f_r(\mathbf{w}_1^n) = \sum_{(i, \mathbf{e}_1^n) \in O_n} P_i \left[\prod_{m=1}^n \left\{ \prod_{j=b_m}^{e_m-1} (1 - P_j) \right\} P_{e_m} \right],$$

where $\mathbf{e}_1^n = (e_1, e_2, \dots, e_n)$ and $O_n = \{(i, \mathbf{e}_1^n) | \mathbf{x}_{b_m}^{e_m} = w_m, 1 \leq m \leq n\}$.

4.3 Word n -gram probability

Similar to the word n -gram probability estimation from a decisively segmented corpus, word n -gram probabilities in an SSC are estimated by the maximum likelihood estimation method as relative values of word n -gram frequencies:

$$P_r(w) = \frac{f_r(w)}{f_r(\cdot)},$$

$$P_r(w_n | \mathbf{w}_1^{n-1}) = \frac{P_r(\mathbf{w}_1^n)}{P_r(\mathbf{w}_1^{n-1})} \quad (n \geq 2).$$

5 Phoneme-to-Text Transcription with an Infinite Vocabulary

The vocabulary of an LM estimated from an SSC consists of all subsequences occurring in it. Adding a module describing a stochastic relationship between these subsequences and input signal sequences, we can build a phoneme-to-text transcription system equipped with an almost infinite vocabulary.

5.1 Word Candidate Enumeration

Given a phoneme sequence as an input, the dictionary of a phoneme-to-text transcription system described in Subsection 3.3 returns pairs of a word and a probability per Equation (4). Similarly, the dictionary of a phoneme-to-text system with an infinite vocabulary must be able to take a phoneme sequence $\mathbf{y}$ and return all possible pairs of a character sequence w and the probability $P(\mathbf{y}|w)$ as word candidates. This is done as follows:

1. First we prepare a single character dictionary containing all characters x in the language annotated with their all possible phoneme sequences $\mathcal{Y}_x = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_k\}$. For

example, the Japanese single character dictionary contains a character $x = \text{“日”}$ annotated with its all possible phoneme sequences $\mathcal{Y}_{\text{日}} = \{\text{bi, hi, jitsu, ka, ni, nichu, nit}\}$.

2. Then we build a phoneme-to-text transcription system for single characters equipped with the vocabulary consisting of the union set of phoneme sequences for all characters. Given a phoneme sequence $\mathbf{y}$, this module returns all possible character sequences w with its generation probability $P(\mathbf{y}|w)$. For example, given a subsequence of the input phoneme sequence $\mathbf{y} = \text{nittere}$, this module returns $\mathcal{W} = \{\text{日テレ, 日手レ, 日照レ, ニッテレ, ニッ手レ, ニッ照レ, \dots}\}$ as a word candidate set along with their generation probabilities.
3. There are various methods to calculate the probability $P(\mathbf{y}|w)$. The only condition is that given $w = x_1x_2 \dots x_m$, $P(\mathbf{y}|w)$ must be a stochastic language model (cf. Section 3) on the alphabet $\mathcal{Y}$. In the experiments, we assumed the uniform distribution of phoneme sequences for each character as follows:³

$$P(\mathbf{y}|w) = P(\mathbf{y}|x_1x_2 \dots x_m) = \prod_{i=1}^m \frac{1}{|\mathcal{Y}_{x_i}|}. \quad (6)$$

The module we described above receives a phoneme sequence and enumerates its decompositions to subsequences contained in the single character dictionary. This module is implemented using a dynamic programming method. In the experiments we limited the maximum length of the input to 16 phonemes.

5.2 Modeling Contexts of Word Candidates

Word n -gram probability estimated from an SSC may not be as accurate as an LM estimated from a corpus segmented appropriately by hand. Thus we use the following interpolation technique:

$$P(w_i|H_i) = \lambda_s P_s(w_i|H_i) + \lambda_r P_r(w_i|H_i),$$

where H_i is history before w_i , P_s is the probability estimated from a segmented corpus C_s , and P_r is the probability estimated by our method from a raw corpus C_r . The λ_s and λ_r are interpolation coefficients which are estimated by the deleted interpolation method (Jelinek et al., 1991).

³ More precisely, it may happen that the same phoneme sequence is generated from a character sequence in multiple ways. In this case the generation probability is calculated as the summation over all possible generations.

In the experiments, the word bi-gram model in our phoneme-to-text transcription system is combined with word bi-gram probabilities estimated from an SSC. Thus the phoneme-to-text transcription system of our new framework refers to the following LM to measure the likelihood of word sequences:

$$P(w_i) = \begin{cases} \lambda_s P_s(w_i|w_{i-1}) + \lambda_r P_r(w_i|w_{i-1}) & \text{if } w_i \in \mathcal{W}, \\ \lambda_s P_s(\mathbf{UW}|w_{i-1})M_{x,n}(w_i) + \lambda_r P_r(w_i|w_{i-1}) & \text{if } w_i \notin \mathcal{W} \wedge w_i \in \mathcal{S}_r, \\ \lambda_s P_s(\mathbf{UW}|w_{i-1})M_{x,n}(w_i), & \because P_r(w_i) = 0 \\ \text{if } w_i \notin \mathcal{W} \wedge w_i \notin \mathcal{S}_r, \end{cases} \quad (7)$$

where $\mathcal{S}_r$ is the set of all subsequences appearing in the SSC.

Our LM based on Equation (7) and an existing LM (cf. Equation (5)) behave differently when they predict an out-of-vocabulary word appearing in the SSC, that is $w_i \notin \mathcal{W} \wedge w_i \in \mathcal{S}_r$. In this case our LM has reliable context information on the OOV word to help the system choose the proper word. Our system also clearly functions better than the LM interpolated with a word n -gram model estimated from the automatic segmentation result of the corpus when the result is a wrong segmentation. For example, when the automatic segmentation result of the sequence “日テレ” (the abbreviation of Japan TV broadcasting corporation) has a word boundary between “日” and “テ,” the uni-gram probability $P(\text{日テレ})$ is equal to 0 and an OOV word “日テレ” is never enumerated as a candidate.⁴ To the contrary, using our method $P(\text{日テレ}) > 0$ when the sequence “日テレ” appears in the SSC at least once. Thus the sequence is enumerated as a candidate word. In addition, when the sequence appears frequently in the SSC, $P(\text{日テレ}) \gg 0$ and the word may appear at a high position in the candidate list even if the automatic segmenter always wrongly segments the sequence into “日” and “テレ.”

5.3 Default Character for Phoneme

In very rare cases, it happens that the input phoneme sequence cannot be decomposed into phoneme sequences in the vocabulary and those

⁴ Two word fragments “日” and “テレ” may be enumerated as word candidates. The notion of word may be necessary for the user’s facility. However, we do not discuss the necessity of the notion of word in the phoneme-to-text transcription system.

corresponding to subsequences of the SSC and, as a result, the transcription system does not output any candidate sentence. To avoid this situation, we prepare a default character for every phoneme and the transcription system also enumerates the default character for each phoneme. In Japanese from the viewpoint of transcription accuracy, it is better to set the default characters to *katakana*, which are used mainly for transliteration of imported words. Since a *katakana* is pronounced uniquely ($|\mathcal{Y}_{x_i}| = 1$),

$$P(\mathbf{y}|\mathbf{w}) = P(\mathbf{y}|x_1x_2 \cdots x_m) = 1. \quad (8)$$

From Equations (4), (6), and (8), the PM of our transcription system is as follows:

$$P(\mathbf{y}_i|w_i) = \begin{cases} \frac{f(\mathbf{y}_i, w_i)}{f(w_i)}, & \text{if } w_i \in \mathcal{W}, \\ \prod_{j=1}^m \frac{1}{|\mathcal{Y}_{x_j}|}, & \text{if } w_i \notin \mathcal{W} \wedge w_i \in \mathcal{S}_r, \\ 1, & \text{if } w_i \notin \mathcal{W} \wedge w_i \notin \mathcal{S}_r, \end{cases} \quad (9)$$

where $w_i = x_1x_2 \cdots x_m$.

5.4 Phoneme-to-Text Transcription with an Infinite Vocabulary

Finally, the transcription system with an infinite vocabulary enumerates candidate sentence $\mathbf{x} = w_1w_2 \cdots w_h$ in the descending order of the following evaluation function value composed of an LM $P(w_i)$ defined by Equation (7) and a PM $P(\mathbf{y}_i|w_i)$ defined by Equation (9):

$$P(\mathbf{y}|\mathbf{x})P(\mathbf{x}) = \prod_{i=1}^h P(\mathbf{y}_i|w_i)P(w_i)$$

Note that there are only three cases since the case decompositions in Equation (7) and Equation (9) are identical.

6 Evaluation

As an evaluation of our phoneme-to-text transcription system, we measured transcription accuracies of several systems on test corpora in two domains: one is a general domain in which we have a small annotated corpus with word boundary information and phoneme sequence for each word, and the other is a target domain in which only a large raw corpus is available. As the transcription result, we took the word sequence of the highest probability. In this section we show the results and evaluate our new framework.

Table 1: Annotated corpus in general domain

	#sentences	#words	#chars
learning	20,808	406,021	598,264
test	2,311	45,180	66,874

Table 2: Raw corpus in the target domain

	#sentences	#words	#chars
learning	797,345	—	17,645,920
test	1,000	—	20,935

6.1 Conditions on the Experiments

The segmented corpus used in our experiments is composed of articles extracted from newspapers and example sentences in a dictionary of daily conversation. Each sentence in the corpus is segmented into words and each word is annotated with a phoneme sequence. The corpus was divided into ten parts. The parameters of the model were estimated from nine of them (learning) and the model was tested on the remaining one (test). Table 1 shows the corpus size. Another corpus we used in the experiments is composed of daily business reports. This corpus is not annotated with word boundary information nor phoneme sequence for each word. For evaluation, we selected 1,000 sentences randomly and annotated them with the phoneme sequences to be used as a test set. The rest was used for LM estimation (see Table 2).

6.2 Evaluation Criterion

The criterion we used for transcription systems is precision and recall based on the number of characters in the longest common subsequence (LCS) (Aho, 1990). Let N_{COR} be the number of characters in the correct sentence, N_{SYS} be that in the output of a system, and N_{LCS} be that of the LCS of the correct sentence and the output of the system, so the recall is defined as N_{LCS}/N_{COR} and the precision as N_{LCS}/N_{SYS} .

6.3 Models for Comparison

In order to clarify the difference in the usages of the target domain corpus, we built four transcription systems and compared their accuracies. Below we explain the models in detail.

Model B: Baseline

A word bi-gram model built from the segmented general domain corpus.

Table 3: Phoneme-to-text transcription accuracy.

	word bi-gram from the annotated corpus	raw corpus usage	unknown word model	General domain		Target domain	
				Precision	Recall	Precision	Recall
$\mathcal{B}$	Yes	No	No	89.80%	92.30%	68.62%	78.40%
$\mathcal{D}$	Yes	Auto. Seg.	No	92.67%	93.42%	80.59%	86.19%
$\mathcal{D}'$	Yes	Auto. Seg.	Yes	92.52%	93.17%	90.35%	93.48%
$\mathcal{S}$	Yes	Stoch. Seg.	Yes	92.78%	93.40%	91.10%	94.09%

The vocabulary contains 10,728 words appearing in more than one corpora of the nine learning corpora. The automatic word segmenter used to build the other three models is based on the method explained in Section 3 with this LM.

Model $\mathcal{D}$: Decisive segmentation

A word bi-gram model estimated from the automatic segmentation result of the target corpus interpolated with model $\mathcal{B}$.

Model $\mathcal{D}'$: Decisive segmentation

Model $\mathcal{D}$ extended with our PM for unknown words

Model $\mathcal{S}$: Stochastic segmentation

A word bi-gram model estimated from the SSC in the target domain interpolated with model $\mathcal{B}$ and equipped with our PM for unknown words

6.4 Evaluation

Table 3 shows the transcription accuracy of the models. A comparison of the accuracies in the target domain of the Model $\mathcal{B}$ and Model $\mathcal{D}$ confirms the well known fact that even an automatic segmentation result containing errors helps an LM improve its performance. The accuracy of Model $\mathcal{D}$ in the general domain is also higher than that of Model $\mathcal{B}$. From this result we can say that over-adaptation has not occurred.

Model $\mathcal{D}'$, equipped with our PM for unknown words, is a natural extension of Model $\mathcal{D}$, a model based on an existing method. The accuracy of Model $\mathcal{D}'$ is higher than that of Model $\mathcal{D}$ in the target domain, but worse in the general domain. This is because the vocabulary of Model $\mathcal{D}'$ is enlarged with the words and the word fragments contained in the automatic segmentation result. Though no study has been reported on the method of Model $\mathcal{D}'$, below we take Model $\mathcal{D}'$ as an existing method for a more severe evaluation.

Comparing the accuracies of Model $\mathcal{D}'$ and Model $\mathcal{S}$ in both domain, it can be said that using our method we can build a more accurate model than the existing methods. The main reason is that

Table 4: Relationship between the raw corpus size and the accuracies.

Raw corpus size	Precision	Recall
1.765×10^5 chars (1/100)	89.18%	92.32%
1.765×10^6 chars (1/10)	90.33%	93.40%
1.765×10^7 chars (1/1)	91.10%	94.09%

our phoneme model PM is able to enumerate transcription candidates for out-of-vocabulary words and word n -gram probabilities estimated from the SSC helps the model choose the appropriate ones.

A detailed study of Table 3 tells us that the reduction rate of character error rate ($100\% - \text{recall}$) of Model $\mathcal{S}$ in the target domain (9.36%) is much larger than that in the general domain (3.37%). The reason for this is that the automatic word segmenter tends to make mistakes around characteristic words and expressions in the target domain and our method is much less influenced by those segmentation errors than the existing method is.

In order to clarify the relationship between the size of the SSC and the transcription accuracy, we calculated the accuracies while changing the size of the SSC (1/1, 1/10, 1/100). The result, shown in Table 4, shows that we can still achieve a further improvement just by gathering more example sentences in the target domain.

The main difference between the models is the LM part. Thus the accuracy increase is yielded by the LM improvements. This fact indicates that we can expect a similar improvement in other generative NLP systems using the noisy channel model by expanding the LM vocabulary with context information to an infinite size.

7 Related Work

The well-known methods for the unknown word problem are classified into two groups: one is to use an unknown word model and the other is to extract word candidates from a corpus before the application. Below we describe the relationship

between these methods and the proposed method.

In the method using an unknown word model, first the generation probability of an unknown word is modeled by a character n -gram, and then an NLP system, such as a morphological analyzer, searches for the best solution considering the possibility that all subsequences might be unknown words (Nagata, 1994; Bazzi and Glass, 2000). In the same way, we can build a phoneme-to-text transcription system which can enumerate unknown word candidates, but the LM is not able to refer to lexical context information to choose the appropriate word, since the unknown words are modeled to be generated from a single state. We solved this problem by allowing the LM to refer to information from an SSC.

When a machine-readable corpus in the target domain is available, we can extract word candidates from the corpus with a certain criterion and use them in application. An advantage of this method is that all of the occurrences of each candidate in the corpus are considered. Nagata (1996) proposed a method calculating word candidates with their uni-gram frequencies using a forward-backward algorithm. and reported that the accuracy of a morphological analyzer can be improved by adding the extracted words to its vocabulary. Comparing our method with this research, it can be said that our method executes the word candidate enumeration and their context calculation dynamically at the time of the solution search for an NLP task, phoneme-to-text transcription here. One of the advantages of our framework is that the system considers all substrings in the corpus as word candidates (that is the recall of the word extraction is 100%) and a higher accuracy is expected using a consistent criterion, namely the generation probability, for the word candidate enumeration process and solution search process.

The framework we propose in this paper, enlarging the vocabulary to an almost infinite size, is general and applicable to many other NLP systems based on the noisy channel model, such as speech recognition, statistical machine translation, etc. Our framework is potentially capable of improving the accuracies in these tasks as well.

8 Conclusion

In this paper we proposed a generative NLP system with an almost infinite vocabulary for languages without obvious word boundary informa-

tion in written texts. In the experiments we compared four phoneme-to-text transcription systems in Japanese. The transcription system equipped with an infinite vocabulary showed a higher accuracy than the baseline model and the model based on the existing method. These results show the efficacy of our method and tell us that our approach is promising for the phoneme-to-text transcription task or other NLP systems based on the noisy channel model.

References

- Alfred V. Aho. 1990. Algorithms for finding patterns in strings. In *Handbook of Theoretical Computer Science*, volume A: Algorithms and Complexity, pages 273–278. Elsevier Science Publishers.
- Issam Bazzi and James R. Glass. 2000. Modeling out-of-vocabulary words for robust speech recognition. In *Proc. of the ICSLP2000*.
- Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.
- Anne-Marie Derouault and Bernard Merialdo. 1986. Natural language modeling for phoneme-to-text transcription. *IEEE PAMI*, 8(6):742–749.
- Frederick Jelinek, Robert L. Mercer, and Salim Roukos. 1991. Principles of lexical language modeling for speech recognition. In *Advances in Speech Signal Processing*, chapter 21, pages 651–699. Dekker.
- Frederick Jelinek. 1985. Self-organized language modeling for speech recognition. Technical report, IBM T. J. Watson Research Center.
- Mark D. Kernighan, Kenneth W. Church, and William A. Gale. 1990. A spelling correction program based on a noisy channel model. In *Proc. of the COLING90*, pages 205–210.
- Adam Kilgariff and Gregory Grefenstette. 2003. Introduction to the special issue on the web as corpus. *Computational Linguistics*, 29(3):333–347.
- Ken Lunde. 1998. *CJKV Information Processing*. O'Reilly & Associates.
- Shinsuke Mori and Daisuke Takuma. 2004. Word n -gram probability estimation from a Japanese raw corpus. In *Proc. of the ICSLP2004*.
- Shinsuke Mori, Tsuchiya Masatoshi, Osamu Yamaji, and Makoto Nagao. 1999. Kana-kanji conversion by a stochastic model. *Transactions of IPSJ*, 40(7):2946–2953. (in Japanese).
- Masaaki Nagata. 1994. A stochastic Japanese morphological analyzer using a forward-DP backward-A* n -best search algorithm. In *Proc. of the COLING94*, pages 201–207.
- Masaaki Nagata. 1996. Automatic extraction of new words from Japanese texts using generalized forward-backward search. In *EMNLP*.

Automatic Generation of Domain Models for Call Centers from Noisy Transcriptions

Shourya Roy and L Venkata Subramaniam

IBM Research
India Research Lab
IIT Delhi, Block-1
New Delhi 110016
India
rshourya,lvsubram@in.ibm.com

Abstract

Call centers handle customer queries from various domains such as computer sales and support, mobile phones, car rental, etc. Each such domain generally has a *domain model* which is essential to handle customer complaints. These models contain common problem categories, typical customer issues and their solutions, greeting styles. Currently these models are manually created over time. Towards this, we propose an unsupervised technique to generate domain models automatically from call transcriptions. We use a state of the art Automatic Speech Recognition system to transcribe the calls between agents and customers, which still results in high word error rates (40%) and show that even from these noisy transcriptions of calls we can automatically build a domain model. The domain model is comprised of primarily a *topic taxonomy* where every node is characterized by *topic(s)*, *typical Questions-Answers (Q&As)*, *typical actions* and *call statistics*. We show how such a domain model can be used for topic identification of unseen calls. We also propose applications for aiding agents while handling calls and for agent monitoring based on the domain model.

1 Introduction

Call center is a general term for help desks, information lines and customer service centers. Many companies today operate call centers to handle customer issues. It includes dialog-based (both voice and online chat) and email support a user receives from a professional agent. Call centers have become a central focus of most companies as they allow them to be in direct contact with their

customers to solve product-related and services-related issues and also for grievance redress. A typical call center agent handles over a hundred calls in a day. Gigabytes of data is produced every day in the form of speech audio, speech transcripts, email, etc. This data is valuable for doing analysis at many levels, e.g., to obtain statistics about the type of problems and issues associated with different products and services. This data can also be used to evaluate agents and train them to improve their performance.

Today's call centers handle a wide variety of domains such as computer sales and support, mobile phones and apparels. To analyze the calls in any domain, analysts need to identify the key issues in the domain. Further, there may be variations within a domain, say mobile phones, based on the service providers. The analysts generate a *domain model* through inspection of the call records (audio, transcripts and emails). Such a model can include a listing of the call categories, types of problems solved in each category, listing of the customer issues, typical questions-answers, appropriate call opening and closing styles, etc. In essence, these models provide a structured view of the domain. Manually building such models for various domains may become prohibitively resource intensive. Another important point to note is that these models are *dynamic* in nature and change over time. As a new version of a mobile phone is introduced, software is launched in a country, a sudden attack of a virus, the model may need to be refined. Hence, an automated way of creating and maintaining such a model is important.

In this paper, we have tried to formalize the essential aspects of a domain model. It comprises of primarily a *topic taxonomy* where every node is characterized by *topic(s)*, *typical Questions-*

Answers (Q&As), typical actions and call statistics. To build the model, we first automatically transcribe the calls. Current automatic speech recognition technology for telephone calls have moderate to high word error rates (Padmanabhan et al., 2002). We applied various feature engineering techniques to combat the noise introduced by the speech recognition system and applied text clustering techniques to group topically similar calls together. Using clustering at different granularity and identifying the relationship between groups at different granularity we generate a taxonomy of call types. This taxonomy is augmented with various meta information related to each node as mentioned above. Such a model can be used for identification of topics of unseen calls. Towards this, we envision an *aiding tool* for agents to increase agent effectiveness and an administrative tool for agent appraisal and training.

Organization of the paper: We start by describing related work in relevant areas. Section 3 talks about the call center dataset and the speech recognition system used. The following section contains the definition and describes an unsupervised mechanism for building a topical model from automatically transcribed calls. Section 5 demonstrates the usability of such a topical model and proposes possible applications. Section 6 concludes the paper.

2 Background and Related Work

In this work, we are trying to bridge the gap between a few seemingly unrelated research areas viz. (1) Automatic Speech Recognition(ASR), (2) Text Clustering and Automatic Taxonomy Generation (ATG) and (3) Call Center Analytics. We present some relevant work done in each of these areas.

Automatic Speech Recognition(ASR): Automatic transcription of telephonic conversations is proven to be more difficult than the transcription of read speech. According to (Padmanabhan et al., 2002), word-error rates are in the range of 7-8% for read speech whereas for telephonic speech it is more than 30%. This degradation is due to the spontaneity of speech as well as the telephone channel. Most speech recognition systems perform well when trained for a particular accent (Lawson et al., 2003). However, with call centers now being located in different parts of the world, the requirement of handling different ac-

cents by the same speech recognition system further increases word error rates.

Automatic Taxonomy Generation (ATG): In recent years there has been some work relating to mining domain specific documents to build an ontology. Mostly these systems rely on parsing (both shallow and deep) to extract relationships between key concepts within the domain. The ontology is constructed from this by linking the extracted concepts and relations (Jiang and Tan, 2005). However, the documents contain well formed sentences which allow for parsers to be used.

Call Center Analytics: A lot of work on automatic call type classification for the purpose of categorizing calls (Tang et al., 2003), call routing (Kuo and Lee, 2003; Haffner et al., 2003), obtaining call log summaries (Douglas et al., 2005), agent assisting and monitoring (Mishne et al., 2005) has appeared in the past. In some cases, they have modeled these as *text classification* problems where topic labels are manually obtained (Tang et al., 2003) and used to put the calls into different buckets. Extraction of key phrases, which can be used as features, from the noisy transcribed calls is an important issue. For manually transcribed calls, which do not have any noise, in (Mishne et al., 2005) a phrase level significance estimate is obtained by combining word level estimates that were computed by comparing the frequency of a word in a domain-specific corpus to its frequency in an open-domain corpus. In (Wright et al., 1997) phrase level significance was obtained for noisy transcribed data where the phrases are clustered and combined into finite state machines. Other approaches use n-gram features with stop word removal and minimum support (Kuo and Lee, 2003; Douglas et al., 2005). In (Bechet et al., 2004) call center dialogs have been clustered to learn about dialog traces that are similar.

Our Contribution: In the call center scenario, the authors are not aware of any work that deals with automatically generating a taxonomy from transcribed calls. In this paper, we have tried to formalize the essential aspects of a domain model. We show an unsupervised method for building a domain model from noisy unlabeled data, which is available in abundance. This hierarchical domain model contains summarized topic specific details for topics of different granularity. We show how such a model can be used for topic identification of unseen calls. We propose two applications for

aiding agents while handling calls and for agent monitoring based on the domain model.

3 Issues with Call Center Data

We obtained telephonic conversation data collected from the internal IT help desk of a company. The calls correspond to users making specific queries regarding problems with computer software such as *Lotus Notes*, *Net Client*, *MS Office*, *MS Windows*, etc. Under these broad categories users faced specific problems e.g. in *Lotus Notes* users had problems with their *passwords*, *mail archiving*, *replication*, *installation*, etc. It is possible that many of the sub problem categories are similar, e.g. *password* issues can occur with *Lotus Notes*, *Net Client* and *MS Windows*.

We obtained automatic transcriptions of the dialogs using an Automatic Speech Recognition (ASR) system. The transcription server, used for transcribing the call center data, is an IBM research prototype. The speech recognition system was trained on 300 hours of data comprising of help desk calls sampled at 6KHz. The transcription output comprises information about the recognized words along with their durations, i.e., beginning and ending times of the words. Further, speaker turns are marked, so the agent and customer portions of speech are demarcated without exactly naming which part is the agent and which the customer. It should be noted that the call center agents and the customers were of different nationalities having varied accents and this further made the job of the speech recognizer hard. The resultant transcriptions have a word error rate of about 40%. This high error rate implies that many wrong deletions of actual words and wrong insertion of dictionary words have taken place. Also often speaker turns are not correctly identified and voice portions of both speakers are assigned to a single speaker. Apart from speech recognition errors there are other issues related to spontaneous speech recognition in the transcriptions. There are no punctuation marks, silence periods are marked but it is not possible to find sentence boundaries based on these. There are repeats, false starts, a lot of pause filling words such as *um* and *uh*, etc. Portion of a transcribed call is shown in figure 1. Generally, at these noise levels such data is hard to interpret by a human. We used over 2000 calls that have been automatically transcribed for our analysis. The average duration of a call is about 9

```
SPEAKER 1: windows thanks for calling and you can
learn yes i don't mind it so then i went to
SPEAKER 2: well and ok bring the machine front
end loaded with a standard um and that's um it's
a desktop machine and i did that everything was
working wonderfully um I went ahead connected
into my my network um so i i changed my network
settings to um to my home network so i i can you
know it's showing me for my workroom um and then
it is said it had to reboot in order for changes
to take effect so i rebooted and now it's asking
me for a password which i never i never said
anything up
SPEAKER 1: ok just press the escape key i can
doesn't do anything can you pull up so that i mean
```

Figure 1: Partial transcript of a help desk dialog

minutes. For 125 of these calls, *call topics* were manually assigned.

4 Generation of Domain Model

Fig 2 shows the steps for generating a domain model in the call center scenario. This section explains different modules shown in the figure.

4.1 Description of Model

We propose the *Domain Model* to be comprised of primarily a *topic taxonomy* where every node is characterized by *topic(s)*, *typical Questions-Answers (Q&As)*, *typical actions* and *call statistics*. Generating such a taxonomy manually from scratch requires significant effort. Further, the changing nature of customer problems requires frequent changes to the taxonomy. In the next subsection, we show that meaningful taxonomies can be built without any manual supervision from a collection of noisy call transcriptions.

4.2 Taxonomy Generation

As mentioned in section 3, automatically transcribed data is noisy and requires a good amount of *feature engineering* before applying any text analytics technique. Each transcription is passed through a *Feature Engineering Component* to perform noise removal. We performed a sequence of cleansing operations to remove *stopwords* such as *the*, *of*, *seven*, *dot*, *january*, *hello*. We also remove *pause filling words* such as *um*, *uh*, *huh*. The remaining words in every transcription are passed through a *stemmer* (using Porter's stemming algo-

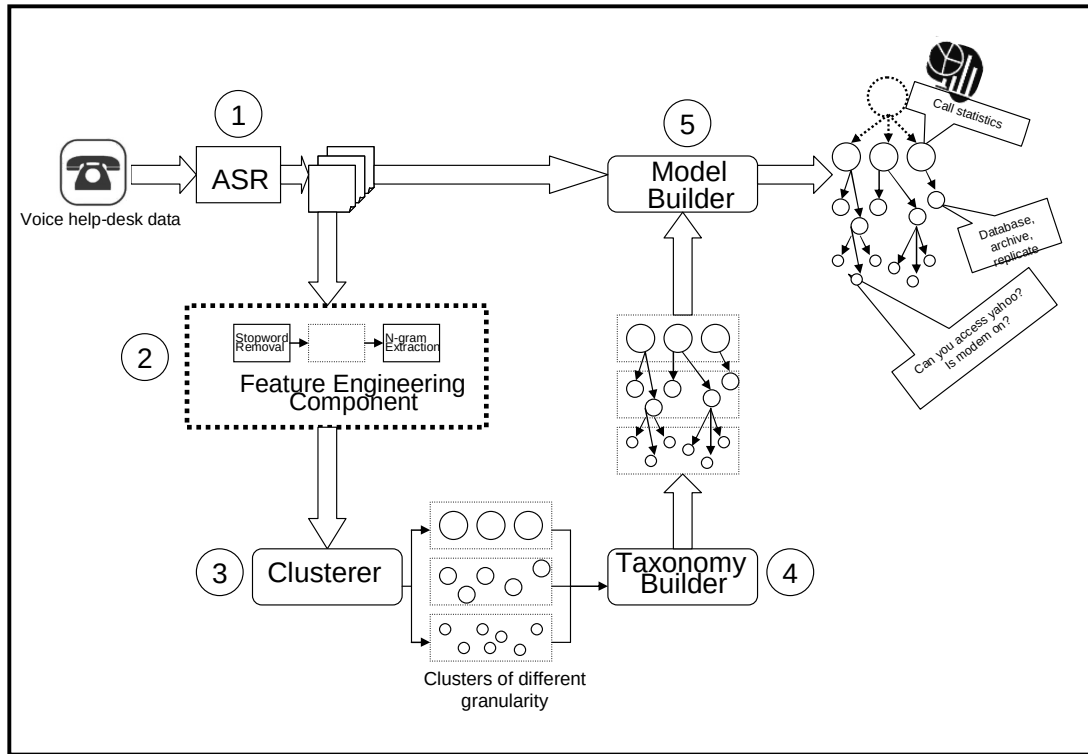


Figure 2: 5 Steps to automatically build domain model from a collection of telephonic conversation recordings

rithm¹) to extract the root form of every word e.g. *call* from *called*. We extract all n -grams which occur more frequently than a threshold and do not contain any stopword. We observed that using all n -grams without thresholding deteriorates the quality of the generated taxonomy. *a t & t*, *lotus notes*, and *expense reimbursement* are some examples of extracted n -grams.

The *Clusterer* generates individual levels of the taxonomy by using text clustering. We used CLUTO package² for doing text clustering. We experimented with all the available clustering functions in CLUTO but no one clustering algorithm consistently outperformed others. Also, there was not much difference between various algorithms based on the available goodness metrics. Hence, we used the default *repeated bisection* technique with *cosine* function as the similarity metric. We ran this algorithm on a collection of 2000 transcriptions multiple times. First we generate 5 clusters from the 2000 transcriptions. Next we generate 10 clusters from the same set of transcriptions and so on. At the finest level we split them into 100 clusters. To generate the topic

taxonomy, these sets containing 5 to 100 clusters are passed through the *Taxonomy Builder* component. This component (1) removes clusters containing less than n documents (2) introduces directed edges from cluster v_1 to v_2 if v_1 and v_2 share at least one document between them, and where v_2 is one level finer than v_1 . Now v_1 and v_2 become nodes in adjacent layers in the taxonomy. Here we found the taxonomy to be a tree but in general it can be a DAG. Now onwards, each node in the taxonomy will be referred to as a *topic*.

This kind of top-down approach was preferred over a bottom-up approach because it not only gives the linkage between clusters of various granularity but also gives the most *descriptive* and *discriminative* set of features associated with each node. CLUTO defines descriptive (and discriminative) features as the set of features which contribute the most to the average similarity (dissimilarity) between documents belonging to the same cluster (different clusters). In general, there is a large overlap between descriptive and discriminative features. These features, *topic features*, are later used for generating topic specific information. Figure 3 shows a part of the taxonomy obtained from the IT help desk dataset. The labels

¹<http://www.tartarus.org/~martin/PorterStemmer>

²<http://glaros.dtc.umn.edu/gkhome/views/cluto>

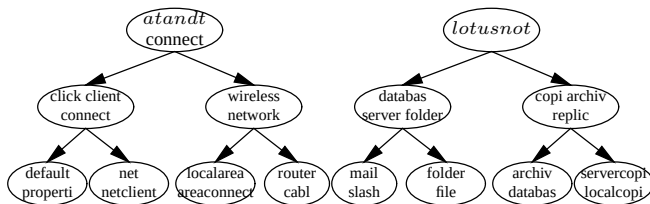


Figure 3: A part of the automatically generated ontology along with descriptive features.

shown in Figure 3 are the most descriptive and discriminative features of a node given the labels of its ancestors.

4.3 Topic Specific Information

The *Model Builder* component in Figure 2 creates an augmented taxonomy with topic specific information extracted from noisy transcriptions. Topic specific information includes phrases that describe *typical actions*, *typical Q&As* and *call statistics* (for each topic in the taxonomy).

Typical Actions: Actions correspond to typical issues raised by the customer, problems and strategies for solving them. We observed that action related phrases are mostly found around topic features. Hence, we start by searching and collecting all the phrases containing topic words from the documents belonging to the topic. We define a 10-word window around the topic features and harvest all phrases from the documents. The set of collected phrases are then searched for n -grams with support above a preset threshold. For example, both the 10-grams *note in click button to set up for all stops* and *to action settings and click the button to set up* increase the support count of the 5-gram *click button to set up*.

The search for the n -grams proceeds based on a threshold on a distance function that counts the insertions necessary to match the two phrases. For example *can you* is closer to *can < ... > you* than to *can < ... > < ... > you*. Longer n -grams are allowed a higher distance threshold than shorter n -grams. After this stage we extracted all the phrases that frequently occur within the cluster.

In the second step, *phrase tiling and ordering*, we prune and merge the extracted phrases and order them. Tiling constructs longer n -grams from sequences of overlapping shorter n -grams. We noted that the phrases have more meaning if they are ordered by their appearance. For example, if *go to the program menu* typically appears before *select options from program menu* then it is more

```
thank you for calling this is
problem with our serial number software
Q: may i have your serial number
Q: how may i help you today
A: i'm having trouble with my at&t network
.....
.....
click on advance log in properties
i want you to right click
create a connection across an existing internet
connection
in d. n. s. use default network
.....
.....
Q: would you like to have your ticket
A: ticket number is two
thank you for calling and have a great day
thank you for calling bye bye
anything else i can help you with
have a great day you too
```

Figure 4: Topic specific information

useful to present them in the order of their appearance. We establish this order based on the average turn number where a phrase occurs.

Typical Questions-Answers: To understand a customer's issue the agent needs to ask the right set of questions. Asking the right questions is the key to effective call handling. We search for all the questions within a topic by defining question templates. The question templates basically look for all phrases beginning with *how*, *what*, *can I*, *can you*, *were there*, etc. This set comprised of 127 such templates for questions. All 10-word phrases conforming to the question templates are collected and phrase harvesting, tiling and ordering is done on them as described above. For the answers we search for phrases in the vicinity immediately following the question.

Figure 4 shows a part of the topic specific information that has been generated for the *default properti* node in Fig 3. There are 123 documents in this node. We have selected phrases that occur at least 5 times in these 123 documents. We have captured the general opening and closing styles used by the agents in addition to typical actions and Q&As for the topic. In this node the documents pertain to queries on setting up a new A T & T network connection. Most of the topic specific issues that have been captured relate to the agent

leading the customer through the steps for setting up the connection. In the absence of tagged dataset we could not quantify our observation. However, when we compared the automatically generated topic specific information to the extracted information from the hand labeled calls, we noted that almost all the issues have been captured. In fact there are some issues in the automatically generated set that are missing from the hand labeled set. The following observations can be made from the topic specific information that has been generated:

- The phrases that have been captured turn out to be quite well formed. Even though the ASR system introduces a lot of noise, the resulting phrases when collected over the clusters are clean.
- Some phrases appear in multiple forms *thank you for calling how can i help you, how may i help you today, thanks for calling can i be of help today*. While tiling is able to merge matching phrases, semantically similar phrases are not merged.
- The list of topic specific phrases, as already noted, matched and at times was more exhaustive than similar hand generated sets.

Call Statistics: We compute various aggregate statistics for each node in the topic taxonomy as part of the model viz. (1) *average call duration*(in seconds), (2) *average transcription length*(number of words) (3) *average number of speaker turns* and (4) *number of calls*. We observed that call durations and number of speaker turns varies significantly from one topic to another. Figure 5 shows average call duration and corresponding average transcription lengths for a few interesting topics. It can be seen that in topic *cluster-1*, which is about *expense reimbursement* and related stuff, most of the queries can be answered quickly in standard ways. However, some *connection* related issues (topic *cluster-5*) require more information from customers and are generally longer in duration. Interestingly, topic *cluster-2* and topic *cluster-4* have similar average call durations but quite different average transcription lengths. On investigation we found that *cluster-4* is primarily about *printer* related queries where the customer many a times is not ready with details like printer name, ip address of the printer, resulting in long hold time whereas for *cluster-2*, which is about *online courses*, users

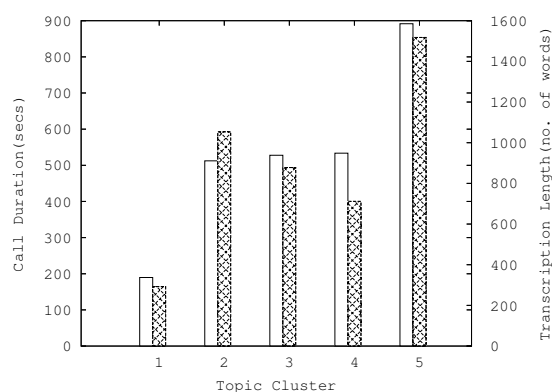


Figure 5: Call duration and transcription length for some topic clusters

generally have details like course name, etc. ready with them and are interactive in nature.

We build a hierarchical index of type $\{topic \rightarrow information\}$ based on this automatically generated model for each topic in the *topic taxonomy*. An entry of this index contains topic specific information viz. (1) *typical Q&As*, (2) *typical actions*, and (3) *call statistics*. As we go down this hierarchical index the information associated with each topic becomes more and more specific. In (Mishne et al., 2005) a manually developed collection of issues and their solutions is indexed so that they can be matched to the call topic. In our work the indexed collection is automatically obtained from the call transcriptions. Also, our index is more useful because of its hierarchical nature where information can be obtained for topics of various granularity unlike (Mishne et al., 2005) where there is no concept of topics at all.

5 Application of Domain Model

Information retrieval from spoken dialog data is an important requirement for call centers. Call centers constantly endeavor to improve the call handling efficiency and identify key problem areas. The described model provides a comprehensive and structured view of the domain that can be used to do both. It encodes three levels of information about the domain:

- **General:** The taxonomy along with the labels gives a general view of the domain. The general information can be used to monitor trends on how the number of calls in different categories change over time e.g. daily, weekly, monthly.

- **Topic level:** This includes a listing of the specific issues related to the topic, typical customer questions and problems, usual strategies for solving the problems, average call durations, etc. It can be used to identify primary issues, problems and solutions pertaining to any category.
- **Dialog level:** This includes information on how agents typically open and close calls, ask questions and guide customers, average number of speaker turns, etc. The dialog level information can be used to monitor whether agents are using courteous language in their calls, whether they ask pertinent questions, etc.

The $\{topic \rightarrow information\}$ index requires identification of the topic for each call to make use of information available in the model. Below we show examples of the use of the model for topic identification.

5.1 Topic Identification

Many of the customer complaints can be categorized into coarse as well as fine topic categories by listening to only the initial part of the call. Exploiting this observation we do *fast topic identification* using a simple technique based on distribution of topic specific *descriptive* and *discriminative* features (Sec 4.2) within the initial portion of the call. Figure 6 shows variation in prediction accuracy using this technique as a function of the fraction of a call observed for 5, 10 and 25 clusters verified over the 125 hand-labeled transcriptions. It can be seen, at coarse level, nearly 70% prediction accuracy can be achieved by listening to the initial 30% of the call and more than 80% of the calls can be correctly categorized by listening only to the first half of the call. Also calls related to some categories can be quickly detected compared to some other clusters as shown in Figure 7.

5.2 Aiding and Administrative Tool

Using the techniques presented in this paper so far it is possible to put together many applications for a call center. In this section we give some example applications and describe ways in which they can be implemented. Based on the hierarchical model described in Section 4 and topic identification mentioned in the last sub-section we describe

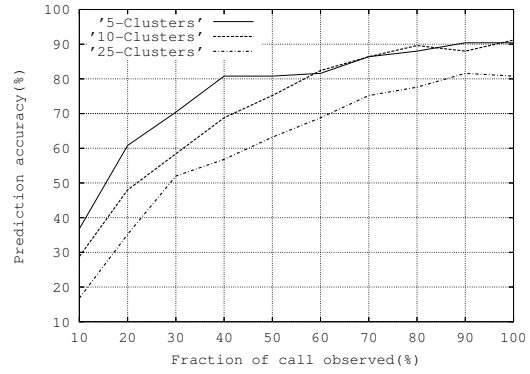


Figure 6: Variation in prediction accuracy with fraction of call observed for 5, 10 and 25 clusters

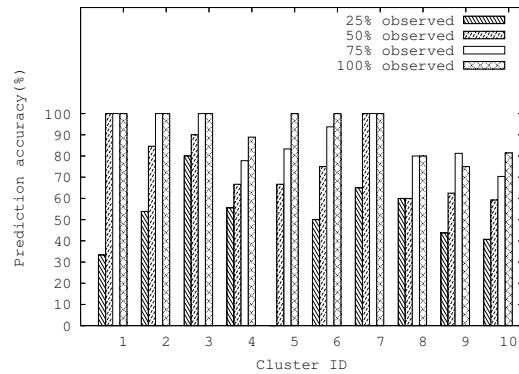


Figure 7: Cluster wise variation in prediction accuracy for 10 clusters

(1) a tool capable of aiding agents for efficient handling of calls to improve customer satisfaction as well as to reduce call handling time, (2) an administrative tool for agent appraisal and training.

Agent aiding is done based on the automatically generated domain model. The hierarchical nature of the model helps to provide generic to specific information to the agent as the call progresses. During call handling the agent can be provided the automatically generated taxonomy and the agent can get relevant information associated with different nodes by say clicking on the nodes. For example, once the agent identifies a call to be about $\{lotusnot\}$ in Fig 3 then he can see the generic *Lotus Notes* related Q&As and actions. By interacting further with the customer the agent identifies it to be of $\{copi archiv replic\}$ topic and typical Q&As and actions change accordingly. Finally, the agent narrows down to the topic as $\{servercopi localcopi\}$ and suggest solution for replication problem in *Lotus Notes*.

The concept of **administrative tool** is primarily driven by Dialog and Topic level information. We envision this post-processing tool to be used

for comparing completed individual calls with corresponding topics based on the distribution of Q&As, actions and call statistics. Based on the topic level information we can check whether the agent identified the issues and offered the known solutions on a given topic. We can use the dialog level information to check whether the agent used courteous opening and closing sentences. Calls that deviate from the topic specific distributions, can be identified in this way and agents handling these calls can be offered further training on the *subject matter*, *courtesy*, etc. This kind of post-processing tool can also help us to catch *abnormally long calls*, *agents with high average call handle time*, etc.

6 Discussion and Future Work

We have shown that it is possible to retrieve useful information from noisy transcriptions of call center voice conversations. We have shown that the extracted information can be put in the form of a model that succinctly captures the domain and provides a comprehensive view of it. We briefly showed through experiments that this model is an accurate description of the domain. We have also suggested useful scenarios where the model can be used to aid and improve call center performance.

A call center handles several hundred-thousand calls per year in various domains. It is very difficult to monitor the performance based on manual processing of the calls. The framework presented in this paper, allows a large part of this work to be automated. A domain specific model that is automatically learnt and updated based on the voice conversations allows the call center to identify problem areas quickly and allocate resources more effectively.

In future we would like to semantically cluster the topic specific information so that redundant topics are eliminated from the list. We can use Automatic Taxonomy Generation(ATG) algorithms for document summarization (Kumnamuru et al., 2004) to build topic taxonomies. We would also like to link our model to technical manuals, catalogs, etc. already available on the different topics in the given domain.

Acknowledgements: We thank our colleagues Raghuram Krishnapuram and Sreeram Balakrishnan for helpful discussions. We also thank Olivier Siohan from the IBM T. J. Watson Research Center for providing us with call transcriptions.

References

- F. Bechet, G. Riccardi and D. Hakkani-Tur 2004. Mining Spoken Dialogue Corpora for System Evaluation and Modeling. *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. July, Barcelona, Spain.
- S. Douglas, D. Agarwal, T. Alonso, R. M. Bell, M. Gilbert, D. F. Swayne and C. Volinsky. 2005. Mining Customer Care Dialogs for “Daily News”. *IEEE Trans. on Speech and Audio Processing*, 13(5):652–660.
- P. Haffner, G. Tur and J. H. Wright 2003. Optimizing SVMs for Complex Call Classification. *IEEE International Conference on Acoustics, Speech, and Signal Processing*. April 6-10, Hong Kong.
- X. Jiang and A.-H. Tan. 2005. Mining Ontological Knowledge from Domain-Specific Text Documents. *IEEE International Conference on Data Mining*, November 26-30, New Orleans, Louisiana, USA.
- K. Kumnamuru, R. Lotlikar, S. Roy, K. Singal and R. Krishnapuram. 2004. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. *International Conference on World Wide Web*. New York, NY, USA.
- H.-K J. Kuo and C.-H. Lee. 2003. Discriminative Training of Natural Language Call Routers. *IEEE Trans. on Speech and Audio Processing*, 11(1):24–35.
- A. D. Lawson, D. M. Harris, J. J. Grieco. 2003. Effect of Foreign Accent on Speech Recognition in the NATO N-4 Corpus. *Eurospeech*. September 1-4, Geneva, Switzerland.
- G. Mishne, D. Carmel, R. Hoory, A. Roytman and A. Soffer. 2005. Automatic Analysis of Call-center Conversations. *Conference on Information and Knowledge Management*. October 31-November 5, Bremen, Germany.
- M. Padmanabhan, G. Saon, J. Huang, B. Kingsbury and L. Mangu.. 2002. Automatic Speech Recognition Performance on a Voicemail Transcription Task. *IEEE Trans. on Speech and Audio Processing*, 10(7):433–442.
- M. Tang, B. Pellom and K. Hacioglu. 2003. Call-type Classification and Unsupervised Training for the Call Center Domain. *Automatic Speech Recognition and Understanding Workshop*. November 30-December 4, St. Thomas, U S Virgin Islands.
- J. Wright, A. Gorin and G. Riccardi. 1997. Automatic Acquisition of Salient Grammar Fragments for Call-type Classification. *Eurospeech*. September, Rhodes, Greece.

Proximity in Context: an empirically grounded computational model of proximity for processing topological spatial expressions*

John D. Kelleher

Dublin Institute of Technology

Dublin, Ireland

john.kelleher@comp.dit.ie

Geert-Jan M. Kruijff

DFKI GmbH

Saarbrücken, Germany

gj@dfki.de

Fintan J. Costello

University College Dublin

Dublin, Ireland

fintan.costello@ucd.ie

Abstract

The paper presents a new model for context-dependent interpretation of linguistic expressions about spatial proximity between objects in a natural scene. The paper discusses novel psycholinguistic experimental data that tests and verifies the model. The model has been implemented, and enables a conversational robot to identify objects in a scene through topological spatial relations (e.g. “X near Y”). The model can help motivate the choice between topological and projective prepositions.

1 Introduction

Our long-term goal is to develop conversational robots with which we can have natural, fluent situated dialog. An inherent aspect of such situated dialog is reference to aspects of the physical environment in which the agents are situated. In this paper, we present a computational model which provides a context-dependent analysis of the environment in terms of *spatial proximity*. We show how we can use this model to ground spatial language that uses topological prepositions (“the ball near the box”) to identify objects in a scene.

Proximity is ubiquitous in situated dialog, but there are deeper “cognitive” reasons for why we need a context-dependent model of proximity to facilitate fluent dialog with a conversational robot. This has to do with the cognitive load that processing proximity expressions imposes. Consider the examples in (1). Psycholinguistic data indicates that a spatial proximity expression (1b) presents a heavier cognitive load than a referring expression identifying an object purely on physical features (1a) yet is easier to process than a projective expression (1c) (van der Sluis and Krahmer, 2004).

The research reported here was supported by the CoSy project, EU FP6 IST “Cognitive Systems” FP6-004250-IP.

- (1) a. the blue ball
- b. the ball near the box
- c. the ball to the right of the box

One explanation for this preference is that feature-based descriptions are easier to resolve perceptually, with a further distinction among features as given in Figure 1, cf. (Dale and Reiter, 1995). On the other hand, the interpretation and realization of spatial expressions requires effort and attention (Logan, 1994; Logan, 1995).

Similarly we can distinguish between the cognitive loads of processing different forms of spatial relations.

Focusing on static prepositions, topological prepositions

have a lower cognitive load than projective prepositions. Topological prepositions (e.g. “at”, “near”) describe proximity to an object. Projective prepositions (e.g. “above”) describe a region in a particular direction from the object. Projective prepositions impose a higher cognitive load because we need to consider different spatial frames of reference (Krahmer and Theune, 1999; Moratz and Tenbrink, 2006). Now, if we want a robot to interact with other agents in a way that obeys the Principle of Minimal Cooperative Effort (Clark and Wilkes-Gibbs, 1986), it should adopt the simplest means to (spatially) refer to an object. However, research on spatial language in human-robot interaction has primarily focused on the use of projective prepositions.

We currently lack a comprehensive model for topological prepositions. Without such a model,

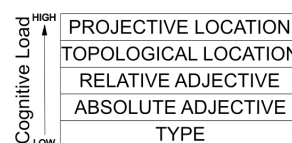


Figure 1: Cognitive load

a robot cannot interpret spatial proximity expressions nor motivate their contextually and pragmatically appropriate use. In this paper, we present a model that addresses this problem. The model uses energy functions, modulated by visual and discourse salience, to model how spatial templates associated with other landmarks may interfere to establish what are contextually appropriate ways to locate a target relative to these landmarks. The model enables grounding of spatial expressions using spatial proximity to refer to objects in the environment. We focus on expressions using topological prepositions such as “near” or “at”.

Terminology. We use the term **target** (T) to refer to the object that is being located by a spatial expression, and **landmark** (L) to refer to the object relative to which the target’s location is described: “[The man]_T near [the table]_L.” A **dis-tractor** is any object in the visual context that is neither landmark nor target.

Overview §2 presents contextual effects we can observe in grounding spatial expressions, including the effect of interference on whether two objects may be considered proximal. §3 discusses a model that accounts for all these effects, and §4 describes an experiment to test the model. §5 shows how we use the model in linguistic interpretation.

2 Data

Below we discuss previous psycholinguistic experiments, focusing on how contextual factors such as distance, size, and salience may affect proximity. We also present novel examples, showing that the location of other objects in a scene may interfere with the acceptability of a proximal description to locate a target relative to a landmark. These examples motivate the model in §3.

1.74	1.90	2.84	3.16	2.34	1.81	2.13
2.61	3.84	4.66	4.97	4.90	3.56	3.26
4.06	5.56	7.55	7.97	7.29	4.80	3.91
4.47	5.91	8.52	O	7.90	6.13	4.46
3.47	4.81	6.94	7.56	7.31	5.59	3.63
3.25	4.03	4.50	4.78	4.41	3.47	3.10
1.84	2.23	2.03	3.06	2.53	2.13	2.00

Figure 2: 7-by-7 cell grid with mean goodness ratings for the relation *the X is near O* as a function of the position occupied by X.

Spatial reasoning is a complex activity that in-

volves at least two levels of processing: a *geometric level* where metric, topological, and projective properties are handled, (Herskovits, 1986); and a *functional level* where the normal function of an entity affects the spatial relationships attributed to it in a context, cf. (Coventry and Garrod, 2004). We focus on geometric factors.

Although a lot of experimental work has been done on spatial reasoning and language (cf. (Coventry and Garrod, 2004)), only Logan and Sadler (1996) examined topological prepositions in a context where functional factors were excluded. They introduced the notion of a **spatial template**. The template is centred on the landmark and identifies for each point in its space the acceptability of the spatial relationship between the landmark and the target appearing at that point being described by the preposition. Logan & Sadler examined various spatial prepositions this way. In their experiments, a human subject was shown sentences of the form “the X is [relation] the O”, each with a picture of a spatial configuration of an **O** in the center of an invisible 7-by-7 cell grid, and an **X** in one of the 48 surrounding positions. The subject then had to rate how well the sentence described the picture, on a scale from 1(bad) to 9(good). Figure 2 gives the mean goodness rating for the relation “near to” as a function of the position occupied by X (Logan and Sadler, 1996). It is clear from Figure 2 that ratings diminish as the distance between X and O increases, but also that even at the extremes of the grid the ratings were still above 1 (min. rating).

Besides distance there are also other factors that determine the applicability of a proximal relation. For example, given prototypical size, the region denoted by “near the building” is larger than that of “near the apple” (Gapp, 1994). Moreover, an object’s salience influences the determination of the proximal region associated with it (Regier and Carlson, 2001; Roy, 2002).

Finally, the two scenes in Figure 3 show interference as a contextual factor. For the scene on the left we can use “the blue box is near the black box” to describe object (c). This seems inappropriate in the scene on the right. Placing an object (d) beside (b) appears to interfere with the appropriateness of using a proximal relation to locate (c) relative to (b), even though the absolute distance between (c) and (b) has not changed.

Thus, there is empirical evidence for several



Figure 3: Proximity and distance

contextual factors determining the applicability of a proximal description. We argued that the location of other distractor objects in context may also interfere with this applicability. The model in §3 captures all these factors, and is evaluated in §4.

3 Computational Model

Below we describe a model of relative proximity that uses (1) the distance between objects, (2) the size and salience of the landmark object, and (3) the location of other objects in the scene. Our model is based on first computing absolute proximity between each point and each landmark in a scene, and then combining or overlaying the resulting absolute proximity fields to compute the relative proximity of each point to each landmark.

3.1 Computing absolute proximity fields

We first compute for each landmark an *absolute proximity field* giving each point's proximity to that landmark, independent of proximity to any other landmark. We compute fields on the projection of the scene onto the 2D-plane, a 2D-array *ARRAY* of points. At each point P in *ARRAY*, the absolute proximity for landmark L is

$$prox_{abs} = (1 - dist_{normalised}(L, P, ARRAY)) * salience(L). \quad (1)$$

In this equation the absolute proximity for a point P and a landmark L is a function of both the distance between the point and the location of the landmark, and the salience of the landmark.

To represent distance we use a normalised distance function $dist_{normalised}(L, P, ARRAY)$, which returns a value between 0 and 1.¹ The smaller the distance between L and P , the higher the absolute proximity value returned, i.e. the more acceptable it is to say that P is close to L . In this way, this component of the absolute proximity field captures the gradual gradation in applicability evident in Logan and Sadler (1996).

¹We normalise by computing the distance between the two points, and then dividing this distance it by the maximum distance between point L and any point in the scene.

We model the influence of visual and discourse salience on absolute proximity as a function $salience(L)$, returning a value between 0 and 1 that represents the relative salience of the landmark L in the scene (2). The relative salience of an object is the average of its visual salience (S_{vis}) and discourse salience (S_{disc}),

$$salience(L) = (S_{vis}(L) + S_{disc}(L))/2 \quad (2)$$

Visual salience S_{vis} is computed using the algorithm of Kelleher and van Genabith (2004). Computing a relative salience for each object in a scene is based on its perceivable size and its centrality relative to the viewer's focus of attention. The algorithm returns scores in the range of 0 to 1. As the algorithm captures object size we can model the effect of landmark size on proximity through the salience component of absolute proximity. The discourse salience (S_{disc}) of an object is computed based on recency of mention (Hajicová, 1993) except we represent the maximum overall salience in the scene as 1, and use 0 to indicate that the landmark is not salient in the current context.

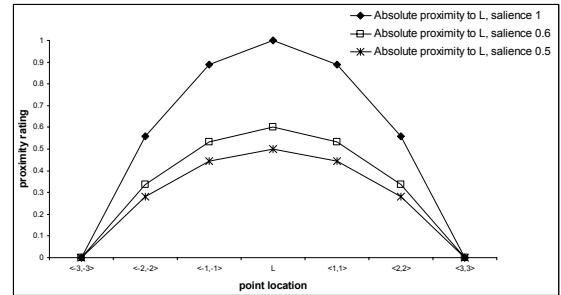


Figure 4: Absolute proximity ratings for landmark L centered in a 2D plane, points ranging from plane's upper-left corner ($\langle -3, -3 \rangle$) to lower right corner ($\langle 3, 3 \rangle$).

Figure 4 shows computed absolute proximity with salience values of 1, 0.6, and 0.5, for points from the upper-left to the lower-right of a 2D plane, with the landmark at the center of that plane. The graph shows how salience influences absolute proximity in our model: for a landmark with high salience, points far from the landmark can still have high absolute proximity to it.

3.2 Computing relative proximity fields

Once we have constructed absolute proximity fields for the landmarks in a scene, our next step is to overlay these fields to produce a measure of

relative proximity to each landmark at each point. For this we first select a landmark, and then iterate over each point in the scene comparing the absolute proximity of the selected landmark at that point with the absolute proximity of all other landmarks at that point. The relative proximity of a selected landmark at a point is equal to the absolute proximity field for that landmark at that point, minus the highest absolute proximity field for any other landmark at that point (see Equation 3).

$$prox_{rel}(P, L) = prox_{abs}(P, L) - \underset{\forall L_X \neq L}{MAX} prox_{abs}(P, L_X) \quad (3)$$

The idea here is that the other landmark with the highest absolute proximity is acting in competition with the selected landmark. If that other landmark's absolute proximity is higher than the absolute proximity of the selected landmark, the selected landmark's *relative* proximity for the point will be negative. If the competing landmark's absolute proximity is slightly lower than the absolute proximity of the selected landmark, the selected landmark's *relative* proximity for the point will be positive, but low. Only when the competing landmark's absolute proximity is significantly lower than the absolute proximity of the selected landmark will the selected landmark have a high relative proximity for the point in question.

In (3) the proximity of a given point to a selected landmark rises as that point's distance from the landmark decreases (the closer the point is to the landmark, the higher its proximity score for the landmark will be), but *falls* as that point's distance from some other landmark decreases (the closer the point is to some other landmark, the lower its proximity score *for the selected landmark* will be). Figure 5 shows the relative proximity fields of two landmarks, L1 and L2, computed using (3), in a 1-dimensional (linear) space. The two landmarks have different degrees of salience: a salience of 0.5 for L1 and of 0.6 for L2 (represented by the different sizes of the landmarks). In this figure, any point where the relative proximity for one particular landmark is above the zero line represents a point which is proximal to that landmark, rather than to the other landmark. The extent to which that point is above zero represents its degree of proximity to that landmark. The overall proximal area for a given landmark is the overall area for which its relative proximity field is above zero. The left and right borders of the figure represent

the boundaries (walls) of the area.

Figure 5 illustrates three main points. First, the overall size of a landmark's proximal area is a function of the landmark's position relative to the other landmark and to the boundaries. For example, landmark L2 has a large open space between it and the right boundary: Most of this space falls into the proximal area for that landmark. Landmark L1 falls into quite a narrow space between the left boundary and L2. L1 thus has a much smaller proximal area in the figure than L2. Second, the relative proximity field for some landmark is a function of that landmark's salience. This can be seen in Figure 5 by considering the space between the two landmarks. In that space the width of the proximal area for L2 is greater than that of L1, because L2 is more salient.

The third point concerns areas of ambiguous proximity in Figure 5: areas in which neither of the landmarks have a significantly higher relative proximity than the other. There are two such areas in the Figure. The first is between the two landmarks, in the region where one relative proximity field line crosses the other. These points are ambiguous in terms of relative proximity because these points are equidistant from those two landmarks. The second ambiguous area is at the extreme right of the space shown in Figure 5. This area is ambiguous because this area is distant from both landmarks: points in this area would not be judged proximal to either landmark. The question of ambiguity in relative proximity judgments is considered in more detail in §5.

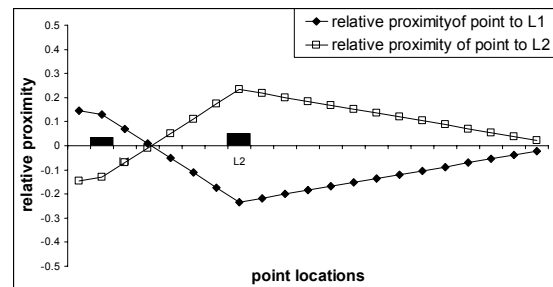


Figure 5: Graph of relative proximity fields for two landmarks L1 and L2. Relative proximity fields were computed with salience scores of 0.5 for L1 and 0.6 for L2.

4 Experiment

Below we describe an experiment which tests our approach (§3) to relative proximity by examining

the changes in people's judgements of the appropriateness of the expression *near* being used to describe the relationship between a target and landmark object in an image where a second, distractor landmark is present. All objects in these images were coloured shapes, a circle, triangle or square.

4.1 Material and Procedure

All images used in this experiment contained a central landmark object and a target object, usually with a third distractor object. The landmark was always placed in the middle of a 7-by-7 grid. Images were divided into 8 groups of 6 images each. Each image in a group contained the target object placed in one of 6 different cells on the grid, numbered from 1 to 6. Figure 6 shows how we number these target positions according to their nearness to the landmark.

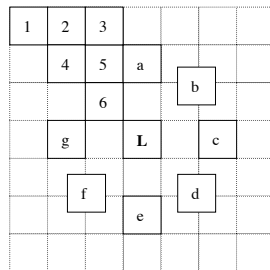


Figure 6: Relative locations of landmark (L) target positions (1..6) and distractor landmark positions (a..g) in images used in the experiment.

Groups are organised according to the presence and position of a distractor object. In group *a* the distractor is directly above the landmark, in group *b* the distractor is rotated 45 degrees clockwise from the vertical, in group *c* it is directly to the right of the landmark, in *d* it is rotated 135 degrees clockwise from the vertical, and so on. The distractor object is always the same distance from the central landmark. In addition to the distractor groups *a,b,c,d,e,f* and *g*, there is an eighth group, group *x*, in which no distractor object occurs.

In the experiment, each image was displayed with a sentence of the form *The ___ is near the ___*, with a description of the target and landmark respectively. The sentence was presented under the image. 12 participants took part in this experiment. Participants were asked to rate the acceptability of the sentence as a description of the image using a 10-point scale, with zero denoting not acceptable at all; four or five denoting moderately

acceptable; and nine perfectly acceptable.

4.2 Results and Discussion

We assess participants' responses by comparing their average proximity judgments with those predicted by the absolute proximity equation (Equation 1), and by the relative proximity equation (Equation 3). For both equations we assume that all objects have a salience score of 1. With salience equal to 1, the absolute proximity equation relates proximity between target and landmark objects to the distance between those two objects, so that the closer the target is to the landmark the higher its proximity will be. With salience equal to 1, the relative proximity equation relates proximity to both distance between target and landmark and distance between target and distractor, so that the proximity of a given target object to a landmark rises as that target's distance from the landmark decreases but *falls* as the target's distance from some other distractor object decreases.

Figure 7 shows graphs comparing participants' proximity ratings with the proximity scores computed by Equation 1 (the absolute proximity equation), and by Equation 3 (the relative proximity equation), for the images in group *x* and in the other 7 groups. In the first graph there is no difference between the proximity scores computed by the two equations, since, when there is no distractor object present the relative proximity equation reduces to the absolute proximity equation. The correlation between both computed proximity scores and participants' average proximity scores for this group is quite high ($r = 0.95$). For the remaining 7 groups the proximity value computed from Equation 1 gives a fair match to people's proximity judgements for target objects (the average correlation across these seven groups in Figure 7 is around $r = 0.93$). However, relative proximity score as computed in Equation 3 significantly improves the correlation in each graph, giving an average correlation across the seven groups of around $r = 0.99$ (all correlations in Figure 7 are significant $p < 0.01$).

Given that the correlations for both Equation 1 and Equation 3 are high we examined whether the results returned by Equation 3 were reliably closer to human judgements than those from Equation 1. For the 42 images where a distractor object was present we recorded which equation gave a result that was closer to participants' normalised aver-

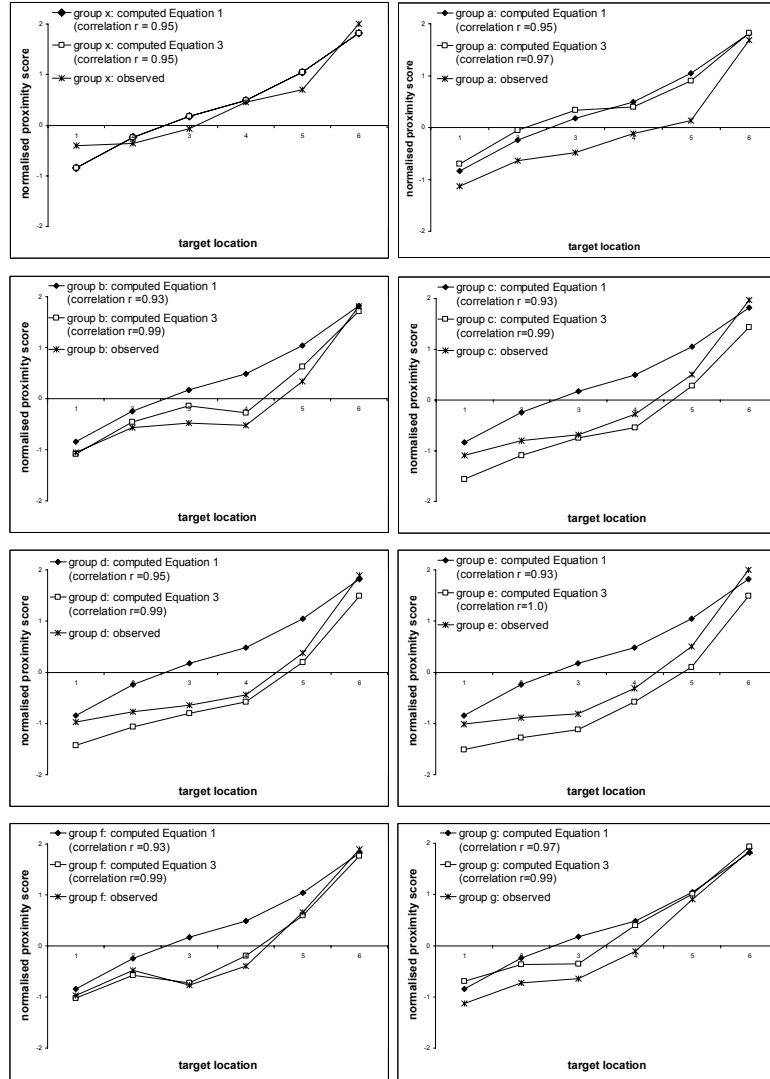


Figure 7: comparison between normalised proximity scores observed and computed for each group.

age for that image. In 28 cases Equation 3 was closer, while in 14 Equation 1 was closer (a 2:1 advantage for Equation 3, significant in a sign test: $n+ = 28, n- = 14, Z = 2.2, p < 0.05$). We conclude that proximity judgements for objects in our experiment are best represented by relative proximity as computed in Equation 3. These results support our ‘relative’ model of proximity.²

It is interesting to note that Equation 3 overestimates proximity in the cases (*a*, *b* and *g*)

²Note that, in order to display the relationship between proximity values given by participants, computed in Equation 1, and computed in Equation 3, the values displayed in Figure 7 are normalised so that proximity values have a mean of 0 and a standard deviation of 1. This normalisation simply means that all values fall in the same region of the scale, and can be easily compared visually.

where the distractor object is closest to the targets and slightly underestimates proximity in all other cases. We will investigate this in future work.

5 Expressing spatial proximity

We use the model of §3 to interpret spatial references to objects. A fundamental requirement for processing situated dialogue is that linguistic meaning provides enough information to establish the *visual grounding* of spatial expressions: How can the robot relate the meaning of a spatial expression to a scene it visually perceives, so it can locate the objects which the expression applies to?

Approaches agree here on the need for ontologically rich representations, but differ in how these are to be visually grounded. Oates et al. (2000)

and Roy (2002) use machine learning to obtain a statistical mapping between visual and linguistic features. Gorniak and Roy (2004) use manually constructed mappings between linguistic constructions, and probabilistic functions which evaluate whether an object can act as referent, whereas DeVault and Stone (2004) use symbolic constraint resolution. Our approach to visual grounding of language is similar to the latter two approaches.

We use a Combinatory Categorical Grammar (CCG) (Baldrige and Kruijff, 2003) to describe the relation between the syntactic structure of an utterance and its meaning. We model meaning as an ontologically richly sorted, relational structure, using a description logic-like framework (Baldrige and Kruijff, 2002). We use OpenCCG for parsing and realization.³

(2) the box near the ball

```
@_{b:phys-obj} (box
  & <Delimitation>unique
  & <Number>singular
  & <Quantification>specific_singular)
& @_{b:phys-obj} <Location>(r : region & near
  & <Proximity>proximal
  & <Positioning>static)
& @_{r:region} <FromWhere>(b1 : phys - obj
  & ball
  & <Delimitation>unique
  & <Number>singular
  & <Quantification>specific_singular)
```

Example (2) shows the meaning representation for “the box near the ball”. It consists of several, related *elementary predicates* (EPs). One type of EP represents a discourse referent as a proposition with a handle: $@_{\{b:phys-obj\}}$ (**box**) means that the referent b is a physical object, namely a **box**. Another type of EP states dependencies between referents as modal relations, e.g. $@_{\{b:phys-obj\}} \langle Location \rangle (r : region \ \& \ \text{near})$ means that discourse referent b (the box) is located in a region r that is near to a landmark. We represent regions explicitly to enable later reference to the region using deictic reference (e.g. “there”). Within each EP we can have semantic features, e.g. the region r characterizes a static location of b and expresses *proximity* to a landmark. Example (2) gives a ball in the context as the landmark.

We use the sorting information in the utterance’s meaning (e.g. *phys-obj*, *region*) for further

interpretation using ontology-based spatial reasoning. This yields several inferences that need to hold for the scene, like DeVault and Stone (2004). Where we differ is in how we check whether these inferences hold. Like Gorniak and Roy (2004), we map these conditions onto the energy landscape computed by the proximity field functions. This enables us to take into account inhibition effects arising in the actual situated context, unlike Gorniak & Roy or DeVault & Stone.

We convert relative proximity fields into proximal regions anchored to landmarks to contextually interpret linguistic meaning. We must decide whether a landmark’s relative proximity score at a given point indicates that it is “near” or “close to” or “at” or “beside” the landmark. For this we iterate over each point in the scene, and compare the relative proximity scores of the different landmarks at each point. If the primary landmark’s (i.e., the landmark with the highest relative proximity at the point) relative proximity exceeds the next highest relative proximity score by more than a predefined confidence interval the point is in the vague region anchored around the primary landmark. Otherwise, we take it as ambiguous and not in the proximal region that is being interpreted. The motivation for the confidence interval is to capture situations where the difference in relative proximity scores between the primary landmark and one or more landmarks at a given point is relatively small. Figure 8 illustrates the parsing of a scene into the regions “near” two landmarks. The relative proximity fields of the two landmarks are identical to those in Figure 5, using a confidence interval of 0.1. Ambiguous points are where the proximity ambiguity series is plotted at 0.5. The regions “near” each landmark are those areas of the graph where each landmark’s relative proximity series is the highest plot on the graph.

Figure 8 illustrates an important aspect of our model: the comparison of relative proximity fields naturally defines the extent of vague proximal regions. For example, see the region right of L2 in Figure 8. The extent of L2’s proximal region in this direction is bounded by the interference effect of L1’s relative proximity field. Because the landmarks’ relative proximity scores converge, the area on the far right of the image is ambiguous with respect to which landmark it is proximal to. In effect, the model captures the fact that the area is relatively distant from both landmarks. Follow-

³<http://www.sf.net/openccg/>

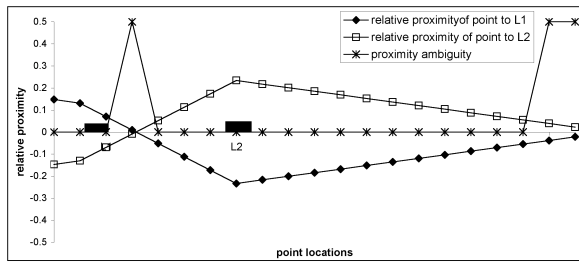


Figure 8: Graph of ambiguous regions overlaid on relative proximity fields for landmarks L1 and L2, with confidence interval=0.1 and different salience scores for L1 (0.5) and L2 (0.6). Locations of landmarks are marked on the X-axis.

ing the cognitive load model (§1), objects located in this region should be described with a projective relation such as “to the right of L2” rather than a proximal relation like “near L2”, see Kelleher and Kruijff (2006).

6 Conclusions

We addressed the issue of how we can provide a context-dependent interpretation of spatial expressions that identify objects based on proximity in a visual scene. We discussed available psycholinguistic data to substantiate the usefulness of having such a model for interpreting and generating fluent situated dialogue between a human and a robot, and that we need a context-dependent representation of what is (situationally) appropriate to consider proximal to a landmark. Context-dependence thereby involves salience of landmarks as well as inhibition effects between landmarks. We presented a model in which we can address these issues, and we exemplified how logical forms representing the meaning of spatial proximity expressions can be grounded in this model. We tested and verified the model using a psycholinguistic experiment. Future work will examine whether the model can be used to describe the semantics of nouns (such as *corner*) that express vague spatial extent, and how the model relates to the functional aspects of spatial reasoning.

References

- J. Baldridge and G.J.M. Kruijff. 2002. Coupling CCG and hybrid logic dependency semantics. In *Proceedings of ACL 2002*, Philadelphia, Pennsylvania.
- J. Baldridge and G.J.M. Kruijff. 2003. Multi-modal combinatory categorial grammar. In *Proceedings of EACL 2003*, Budapest, Hungary.
- H. Clark and D. Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22:1–39.
- K.R. Coventry and S. Garrod. 2004. *Saying, Seeing and Acting. The Psychological Semantics of Spatial Prepositions*. Essays in Cognitive Psychology Series. Lawrence Erlbaum Associates.
- R. Dale and E. Reiter. 1995. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive Science*, 18:233–263.
- D. DeVault and M. Stone. 2004. Interpreting vague utterances in context. In *Proceedings of COLING 2004*, volume 2, pages 1247–1253, Geneva, Switzerland.
- K.P. Gapp. 1994. Basic meanings of spatial relations: Computation and evaluation in 3d space. In *Proceedings of AAAI-94*, pages 1393–1398.
- P. Gorniak and D. Roy. 2004. Grounded semantic composition for visual scenes. *Journal of Artificial Intelligence Research*, 21:429–470.
- E. Hajicová. 1993. *Issues of Sentence Structure and Discourse Patterns*, volume 2 of *Theoretical and Computational Linguistics*. Charles University Press.
- A. Herskovits. 1986. *Language and spatial cognition: An interdisciplinary study of prepositions in English*. Studies in Natural Language Processing. Cambridge University Press.
- J.D. Kelleher and G.J. Kruijff. 2006. Incremental generation of spatial referring expressions in situated dialog. In *Proceedings ACL/COLING '06*, Sydney, Australia.
- J. Kelleher and J. van Genabith. 2004. Visual salience and reference resolution in simulated 3d environments. *AI Review*, 21(3-4):253–267.
- E. Krahmer and M. Theune. 1999. Efficient generation of descriptions in context. In R. Kibble and K. van Deemter, editors, *Workshop on the Generation of Nominals, ESS-LLI'99*, Utrecht, The Netherlands.
- G.D. Logan and D.D. Sadler. 1996. A computational analysis of the apprehension of spatial relations. In M. Bloom, P. and Peterson, L. Nadell, and M. Garrett, editors, *Language and Space*, pages 493–529. MIT Press.
- G.D. Logan. 1994. Spatial attention and the apprehension of spatial relations. *Journal of Experimental Psychology: Human Perception and Performance*, 20:1015–1036.
- G.D. Logan. 1995. Linguistic and conceptual control of visual spatial attention. *Cognitive Psychology*, 12:523–533.
- R. Moratz and T. Tenbrink. 2006. Spatial reference in linguistic human-robot interaction: Iterative, empirically supported development of a model of projective relations. *Spatial Cognition and Computation*.
- T. Oates, Z. Eyller-Walker, and P.R. Cohen. 2000. Toward natural language interfaces for robotic agents: Grounding linguistic meaning in sensors. In *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 227–228.
- T. Regier and L. Carlson. 2001. Grounding spatial language in perception: An empirical and computational investigation. *Journal of Experimental Psychology: General*, 130(2):273–298.
- D.K. Roy. 2002. Learning words and syntax for a scene description task. *Computer Speech and Language*, 16(3).
- I.F. van der Sluis and E.J. Krahmer. 2004. The influence of target size and distance on the production of speech and gesture in multimodal referring expressions. In R. Kibble and K. van Deemter, editors, *ICSLP04*.

Machine Learning of Temporal Relations

Inderjeet Mani^{¥§}, Marc Verhagen[¶], Ben Wellner^{¥¶}
Chong Min Lee[§] and James Pustejovsky[¶]

[¥]The MITRE Corporation

202 Burlington Road, Bedford, MA 01730, USA

[§]Department of Linguistics, Georgetown University

37th and O Streets, Washington, DC 20036, USA

[¶]Department of Computer Science, Brandeis University

415 South St., Waltham, MA 02254, USA

{imani, wellner}@mitre.org, {marc, jamesp}@cs.brandeis.edu, cml54@georgetown.edu

Abstract

This paper investigates a machine learning approach for temporally ordering and anchoring events in natural language texts. To address data sparseness, we used temporal reasoning as an over-sampling method to dramatically expand the amount of training data, resulting in predictive accuracy on link labeling as high as 93% using a Maximum Entropy classifier on human annotated data. This method compared favorably against a series of increasingly sophisticated baselines involving expansion of rules derived from human intuitions.

1 Introduction

The growing interest in practical NLP applications such as question-answering and text summarization places increasing demands on the processing of temporal information. In multi-document summarization of news articles, it can be useful to know the relative order of events so as to merge and present information from multiple news sources correctly. In question-answering, one would like to be able to ask when an event occurs, or what events occurred prior to a particular event.

A wealth of prior research by (Passoneau 1988), (Webber 1988), (Hwang and Schubert 1992), (Kamp and Reyle 1993), (Lascarides and Asher 1993), (Hitzeman et al. 1995), (Kehler 2000) and others, has explored the different knowledge sources used in inferring the temporal ordering of events, including temporal adverbials, tense, aspect, rhetorical relations, pragmatic conventions, and background knowledge. For example, the narrative convention of events being described in the order in which they occur

is followed in (1), but overridden by means of a discourse relation, Explanation in (2).

(1) Max *stood up*. John *greeted* him.

(2) Max *fell*. John *pushed* him.

In addition to discourse relations, which often require inferences based on world knowledge, the ordering decisions humans carry out appear to involve a variety of knowledge sources, including tense and grammatical aspect (3a), lexical aspect (3b), and temporal adverbials (3c):

(3a) Max *entered* the room. He *had drunk* a lot of wine.

(3b) Max *entered* the room. Mary *was seated* behind the desk.

(3c) The company *announced* Tuesday that third-quarter sales *had fallen*.

Clearly, substantial linguistic processing may be required for a system to make these inferences, and world knowledge is hard to make available to a domain-independent program. An important strategy in this area is of course the development of annotated corpora than can facilitate the machine learning of such ordering inferences.

This paper¹ investigates a machine learning approach for temporally ordering events in natural language texts. In Section 2, we describe the annotation scheme and annotated corpora, and the challenges posed by them. A basic learning approach is described in Section 3. To address data sparseness, we used temporal reasoning as an over-sampling method to dramatically expand the amount of training data.

As we will discuss in Section 5, there are no standard algorithms for making these inferences that we can compare against. We believe strongly that in such situations, it's worthwhile for computational linguists to devote consider-

¹Research at Georgetown and Brandeis on this problem was funded in part by a grant from the ARDA AQUAINT Program, Phase II.

able effort to developing insightful baselines. Our work is, accordingly, evaluated in comparison against four baselines: (i) the usual majority class statistical baseline, shown along with each result, (ii) a more sophisticated baseline that uses hand-coded rules (Section 4.1), (iii) a hybrid baseline based on hand-coded rules expanded with Google-induced rules (Section 4.2), and (iv) a machine learning version that learns from imperfect annotation produced by (ii) (Section 4.3).

2 Annotation Scheme and Corpora

2.1 TimeML

TimeML (Pustejovsky et al. 2005) (www.timeml.org) is an annotation scheme for markup of events, times, and their temporal relations in news articles. The TimeML scheme flags tensed verbs, adjectives, and nominals with EVENT tags with various attributes, including the class of event, tense, grammatical aspect, polarity (negative or positive), any modal operators which govern the event being tagged, and cardinality of the event if it's mentioned more than once. Likewise, time expressions are flagged and their values normalized, based on TIMEX3, an extension of the ACE (2004) (tern.mitre.org) TIMEX2 annotation scheme.

For temporal relations, TimeML defines a TLINK tag that links tagged events to other events and/or times. For example, given (3a), a TLINK tag **orders** an instance of the event of entering to an instance of the drinking with the relation type AFTER. Likewise, given the sentence (3c), a TLINK tag will **anchor** the event instance of announcing to the time expression *Tuesday* (whose normalized value will be inferred from context), with the relation IS_INCLUDED. These inferences are shown (in slightly abbreviated form) in the annotations in (4) and (5).

(4) Max <EVENT eventID="e1" class="occurrence" tense="past" aspect="none">entered</EVENT> the room. He <EVENT eventID="e2" class="occurrence" tense="past" aspect="perfect">had drunk</EVENT> a lot of wine.

<TLINK eventID="e1" relatedToEventID="e2" relType="AFTER"/>

(5) The company <EVENT eventID="e1" class="reporting" tense="past" aspect="none">announced</EVENT> <TIMEX3 tid="t2" type="DATE" temporalFunction="false" value="1998-01-08">Tuesday </TIMEX3> that third-

quarter sales <EVENT eventID="e2" class="occurrence" tense="past" aspect="perfect">had fallen</EVENT>. <TLINK eventID="e1" relatedToEventID="e2" relType="AFTER"/> <TLINK eventID="e1" relatedToTimeID="t2" relType="IS_INCLUDED"/>

The anchor relation is an Event-Time TLINK, and the order relation is an Event-Event TLINK. TimeML uses 14 temporal relations in the TLINK *RelTypes*, which reduce to a disjunctive classification of 6 temporal relations *RelTypes* = {SIMULTANEOUS, IBEFORE, BEFORE, BEGINS, ENDS, INCLUDES}. An event or time is SIMULTANEOUS with another event or time if they occupy the same time interval. An event or time INCLUDES another event or time if the latter occupies a proper subinterval of the former. These 6 relations and their inverses map one-to-one to 12 of Allen's 13 basic relations (Allen 1984)². There has been a considerable amount of activity related to this scheme; we focus here on some of the challenges posed by the TLINK annotation, the part that is directly relevant to the temporal ordering and anchoring problems.

2.2 Challenges

The annotation of TimeML information is on a par with other challenging semantic annotation schemes, like PropBank, RST annotation, etc., where high inter-annotator reliability is crucial but not always achievable without massive pre-processing to reduce the user's workload. In TimeML, inter-annotator agreement for time expressions and events is 0.83 and 0.78 (average of Precision and Recall) respectively, but on TLINKs it is 0.55 (P&R average), due to the large number of event pairs that can be selected for comparison. The time complexity of the human TLINK annotation task is quadratic in the number of events and times in the document.

Two corpora have been released based on TimeML: the TimeBank (Pustejovsky et al. 2003) (we use version 1.2.a) with 186 documents and

²Of the 14 TLINK relations, the 6 inverse relations are redundant. In order to have a disjunctive classification, SIMULTANEOUS and IDENTITY are collapsed, since IDENTITY is a subtype of SIMULTANEOUS. (Specifically, X and Y are identical if they are simultaneous *and* coreferential.) DURING and IS_INCLUDED are collapsed since DURING is a subtype of IS_INCLUDED that anchors events to times that are durations. IBEFORE (immediately before) corresponds to Allen's MEETS. Allen's OVERLAPS relation is not represented in TimeML. More details can be found at timeml.org.

64,077 words of text, and the Opinion Corpus (www.timeml.org), with 73 documents and 38,709 words. The TimeBank was developed in the early stages of TimeML development, and was partitioned across five annotators with different levels of expertise. The Opinion Corpus was developed very recently, and was partitioned across just two highly trained annotators, and could therefore be expected to be less noisy. In our experiments, we merged the two datasets to produce a single corpus, called OTC.

Table 1 shows the distribution of EVENTS and TIMES, and TLINK *RelTypes*³ in the OTC. The majority class percentages are shown in parentheses. It can be seen that BEFORE and SIMULTANEOUS together form a majority of event-ordering (Event-Event) links, whereas most of the event anchoring (Event-Time) links are INCLUDES.

12750 Events, 2114 Times		
Relation	Event-Event	Event-Time
IBEFOR	131	15
BEGINS	160	112
ENDS	208	159
SIMULTANEOUS	1528	77
INCLUDES	950	3001 (65.3%)
BEFORE	3170 (51.6%)	1229
TOTAL	6147	4593

Table 1. TLINK Class Distributions in OTC Corpus

The lack of TLINK coverage in human annotation could be helped by preprocessing, provided it meets some threshold of accuracy. Given the availability of a corpus like OTC, it is natural to try a machine learning approach to see if it can be used to provide that preprocessing. However, the noise in the corpus and the sparseness of links present challenges to a learning approach.

3 Machine Learning Approach

3.1 Initial Learner

There are several sub-problems related to inferring event anchoring and event ordering. Once a tagger has tagged the events and times, the first task (A) is to link events and/or times, and the second task (B) is to label the links. Task A is hard to evaluate since, in the absence of massive preprocessing, many links are ignored by the human in creating the annotated corpora. In addition,

a program, as a baseline, can trivially link all tagged events and times, getting 100% recall on Task A. We focus here on Task B, the labeling task. In the case of humans, in fact, when a TLINK is posited by both annotators between the same pairs of events or times, the inter-annotator agreement on the labels is a .77 average of P&R. To ensure replicability of results, we assume perfect (i.e., OTC-supplied) events, times, and links.

Thus, we can consider TLINK inference as the following classification problem: *given an ordered pair of elements X and Y, where X and Y are events or times which the human has related temporally via a TLINK, the classifier has to assign a label in RelTypes*. Using *RelTypes* instead of $RelTypes \cup \{NONE\}$ also avoids the problem of heavily skewing the data towards the NONE class.

To construct feature vectors for machine learning, we took each TLINK in the corpus and used the given TimeML features, with the TLINK class being the vector’s class feature. For replicability by other users of these corpora, and to be able to isolate the effect of components, we used ‘perfect’ features; no feature engineering was attempted. The features were, for each event in an event-ordering pair, the *event-class*, *aspect*, *modality*, *tense* and *negation* (all nominal features); *event string*, and *signal* (a preposition/adverb, e.g., *reported on Tuesday*), which are string features, and contextual features indicating whether the *same tense* and *same aspect* are true of both elements in the event pair. For event-time links, we used the above event and signal features along with TIMEX3 time features.

For learning, we used an off-the-shelf Maximum Entropy (ME) classifier (from Carafe, available at sourceforge.net/projects/carafe). As shown in the **UNCLOSED (ME)** column in Table 2⁴, accuracy of the unclosed ME classifier does not go above 77%, though it’s always better than the majority class (in parentheses). We also tried a variety of other classifiers, including the SMO support-vector machine and the naïve Bayes tools in WEKA (www.weka.net.nz). SMO performance (but not naïve Bayes) was comparable with ME, with SMO trailing it in a few cases (to save space, we report just ME performance). It’s possible that feature engineering could improve performance, but since this is “perfect” data, the result is not encouraging.

³The number of TLINKs shown is based on the number of TLINK vectors extracted from the OTC.

⁴All machine learning results, except for ME-C in Table 4, use 10-fold cross-validation. ‘Accuracy’ in tables is Predictive Accuracy.

	UNCLOSED (ME)						CLOSED (ME-C)					
	Event-Event			Event-Time			Event-Event			Event-Time		
Accuracy:	62.5 (51.6)			76.13 (65.3)			93.1 (75.2)			88.25 (62.3)		
Relation	Prec	Rec	F	Prec	Rec	F	Prec	Rec	F	Prec	Rec	F
<i>IBEFOR</i>	50.00	27.27	35.39	0	0	0	77.78	60.86	68.29	0	0	0
<i>BEGINS</i>	50.00	41.18	45.16	60.00	50.00	54.54	85.25	82.54	83.87	76.47	74.28	75.36
<i>ENDS</i>	94.74	66.67	78.26	41.67	27.78	33.33	87.83	94.20	90.90	79.31	77.97	78.62
<i>SIMULTANEOUS</i>	50.35	50.00	50.17	33.33	20.00	25.00	62.50	38.60	47.72	73.68	56.00	63.63
<i>INCLUDES</i>	47.88	34.34	40.00	80.92	62.72	84.29	90.41	88.23	89.30	86.07	80.78	83.34
<i>BEFORE</i>	68.85	79.24	73.68	70.47	62.72	66.37	94.95	97.26	96.09	90.16	93.56	91.83

Table 2. Machine learning results using unclosed and closed data

3.2 Expanding Training Data using Temporal Reasoning

To expand our training set, we use a temporal closure component SputLink (Verhagen 2004), that takes known temporal relations in a text and derives new implied relations from them, in effect making explicit what was implicit. SputLink was inspired by (Setzer and Gaizauskas 2000) and is based on Allen’s interval algebra, taking into account the limitations on that algebra that were pointed out by (Vilain et al. 1990). It is basically a constraint propagation algorithm that uses a transitivity table to model the compositional behavior of all pairs of relations in a document. SputLink’s transitivity table is represented by 745 axioms. An example axiom:

If relation(A, B) = BEFORE &&
relation(B, C) = INCLUDES
then infer relation(A, C) = BEFORE

Once the TLINKs in each document in the corpus are closed using SputLink, the same vector generation procedure and feature representation described in Section 3.1 are used. The effect of closing the TLINKs on the corpus has a dramatic impact on learning. Table 2, in the **CLOSED (ME-C)** column shows that accuracies for this method (called ME-C, for Maximum Entropy learning with closure) are now in the high 80’s and low 90’s, and still outperform the closed majority class (shown in parentheses).

What is the reason for the improvement?⁵ One reason is the dramatic increase in the amount of training data. The more connected the initial un-

closed graph for a document is in TLINKs, the greater the impact in terms of closure. When the OTC is closed, the number of TLINKs goes up by more than 11 times, from 6147 Event-Event and 4593 Event-Time TLINKs to 91,157 Event-Event and 29,963 Event-Time TLINKs. The number of BEFORE links goes up from 3170 (51.6%) Event-Event and 1229 Event-Time TLINKs (26.75%) to 68585 (75.2%) Event-Event and 18665 (62.3%) Event-Time TLINKs, making BEFORE the majority class in the closed data for both Event-Event and Event-Time TLINKs. There are only an average of 0.84 TLINKs per event before closure, but after closure it shoots up to 9.49 TLINKs per event. (Note that as a result, the majority class percentages for the closed data have changed from the unclosed data.)

Being able to bootstrap more training data is of course very useful. However, we need to dig deeper to investigate how the increase in data affected the machine learning. The improvement provided by temporal closure can be explained by three factors: (1) closure effectively creates a new classification problem with many more instances, providing more data to train on; (2) the class distribution is further skewed which results in a higher majority class baseline (3) closure produces additional data in such a way as to increase the frequencies and statistical power of existing features in the unclosed data, as opposed to adding new features. For example, with unclosed data, given A BEFORE B and B BEFORE C, closure generates A BEFORE C which provides more significance for the features related to A and C appearing as first and second arguments, respectively, in a BEFORE relation.

In order to help determine the effects of the above factors, we carried out two experiments in which we sampled 6145 vectors from the *closed*

⁵Interestingly, performance does not improve for SIMULTANEOUS. The reason for this might be due to the relatively modest increase in SIMULTANEOUS relations from applying closure (roughly factor of 2).

data – i.e. approximately the number of Event-Event vectors in the unclosed data. This effectively removed the contribution of factor (1) above. The first experiment (Closed Class Distribution) simply sampled 6145 instances uniformly from the closed instances, while the second experiment (Unclosed Class Distribution) sampled instances according to the same distribution as the unclosed data. Table 3 shows these results. The greater class distribution skew in the closed data clearly contributes to improved accuracy. However, when using the same class distribution as the unclosed data (removing factor (2) from above), the accuracy, 76%, is *higher than using the full unclosed data*. This indicates that closure does indeed help according to factor (3).

4 Comparison against Baselines

4.1 Hand-Coded Rules

Humans have strong intuitions about rules for temporal ordering, as we indicated in discussing sentences (1) to (3). Such intuitions led to the development of pattern matching rules incorporated in a TLINK tagger called GTag. GTag takes a document with TimeML tags, along with syntactic information from part-of-speech tagging and chunking from Carafe, and then uses 187 syntactic and lexical rules to infer and label TLINKs between tagged events and other tagged events or times. The tagger takes pairs of TLINKable items (event and/or time) and searches for the single most-confident rule to apply to it, if any, to produce a labeled TLINK between those items. Each (if-then) rule has a left-hand side which consists of a conjunction of tests based on TimeML-related feature combinations (TimeML features along with part-of-speech and chunk-related features), and a right-hand side which is an assignment to one of the TimeML TLINK classes.

The rule patterns are grouped into several different classes: (i) the event is anchored with or without a signal to a time expression within the same clause, e.g., (3c), (ii) the event is anchored without a signal to the document date (as is often the case for reporting verbs in news), (iii) an event is linked to another event in the same sentence, e.g., (3c), and (iv) the event in a main clause of one sentence is anchored with a signal or tense/aspect cue to an event in the main clause of the previous sentence, e.g., (1-2), (3a-b).

The performance of this baseline is shown in Table 4 (line GTag). The top most accurate rule (87% accuracy) was GTag Rule 6.6, which links

a past-tense event verb joined by a conjunction to another past-tense event verb as being BEFORE the latter (e.g., *they traveled and slept the night ..*):

```
If sameSentence=YES &&
    sentenceType=ANY &&
    conjBetweenEvents=YES &&
    arg1.class=EVENT &&
    arg2.class=EVENT &&
    arg1.tense=PAST &&
    arg2.tense=PAST &&
    arg1.aspect=NONE &&
    arg2.aspect=NONE &&
    arg1.pos=VB &&
    arg2.pos=VB &&
    arg1.firstVbEvent=ANY &&
    arg2.firstVbEvent=ANY
then infer relation=BEFORE
```

The vast majority of the intuition-bred rules have very low accuracy compared to ME-C, with intuitions failing for various feature combinations and relations (for relations, for example, GTag lacks rules for IBEFORE, STARTS, and ENDS). The bottom-line here is that even when heuristic preferences are intuited, those preferences need to be guided by empirical data, whereas hand-coded rules are relatively ignorant of the distributions that are found in data.

4.2 Adding Google-Induced Lexical Rules

One might argue that the above baseline is too weak, since it doesn't allow for a rich set of lexical relations. For example, pushing can result in falling, killing always results in death, and so forth. These kinds of defeasible rules have been investigated in the semantics literature, including the work of Lascares and Asher cited in Section 1.

However, rather than hand-creating lexical rules and running into the same limitations as with GTag's rules, we used an empirically-derived resource called VerbOcean (Chklovski and Pantel 2004), available at <http://semantics.isi.edu/ocean>. This resource consists of lexical relations mined from Google searches. The mining uses a set of lexical and syntactic patterns to test for pairs of verb strongly associated on the Web in an asymmetric 'happens-before' relation. For example, the system discovers that *marriage* happens-before *divorce*, and that *tie* happens-before *untie*.

We automatically extracted all the 'happens-before' relations from the VerbOcean resource at the above web site, and then automatically converted those relations to GTag format, producing 4,199 rules. Here is one such converted rule:

```

If arg1.class=EVENT &&
  arg2.class=EVENT &&
  arg1.word=learn &&
  arg2.word=forget &&
then infer relation=BEFORE

```

Adding these lexical rules to GTag (with morphological normalization being added for rule matching on word features) amounts to a considerable augmentation of the rule-set, by a factor of 22. GTag with this augmented rule-set might be a useful baseline to consider, since one would expect the gigantic size of the Google ‘corpus’ to yield fairly robust, broad-coverage rules.

What if both a core GTag rule and a VerbOcean-derived rule could both apply? We assume the one with the higher confidence is chosen. However, we don’t have enough data to reliably estimate rule confidences for the original GTag rules; so, for the purposes of VerbOcean rule integration, we assigned either the original VerbOcean rules as having greater confidence than

the original GTag rules in case of a conflict (i.e., a preference for the more specific rule), or vice-versa.

The results are shown in Table 4 (lines GTag+VerbOcean). The combined rule set, under both voting schemes, had no statistically significant difference in accuracy from the original GTag rule set. So, ME-C beat this baseline as well.

The reason VerbOcean didn’t help is again one of data sparseness, due to most verbs occurring rarely in the OTC. There were only 19 occasions when a happens-before pair from VerbOcean correctly matched a human BEFORE TLINK, of which 6 involved the same rule being right twice (including *learn* happens-before *forget*, a rule which students are especially familiar with!), with the rest being right just once. There were only 5 occasions when a VerbOcean rule incorrectly matched a human BEFORE TLINK, involving just three rules.

	Closed Class Distribution				UnClosed Class Distribution			
Relation	Prec	Rec	F	Accuracy	Prec	Rec	F	Accuracy
<i>IBEFORE</i>	100.0	100.0	100.0	87.20 (72.03)	83.33	58.82	68.96	76.0 (40.95)
<i>BEGINS</i>	0	0	0		72.72	50.0	59.25	
<i>ENDS</i>	66.66	57.14	61.53		62.50	50.0	55.55	
<i>SIMULTANEOUS</i>	14.28	6.66	9.09		60.54	66.41	63.34	
<i>INCLUDES</i>	73.91	77.98	75.89		75.75	77.31	76.53	
<i>BEFORE</i>	90.68	92.60	91.63		84.09	84.61	84.35	

Table 3. Machine Learning from subsamples of the closed data

Baseline	Accuracy	
	Event-Event	Event-Time
GTag	63.43	72.46
GTag+VerbOcean - GTag overriding VerbOcean	64.80	74.02
GTag+VerbOcean - VerbOcean overriding GTag	64.22	73.37
GTag+closure+ME-C	53.84 (57.00)	67.37 (67.59)

Table 4. Accuracy of ‘Intuition’ Derived Baselines

4.3 Learning from Hand-Coded Rules Baseline

The previous baseline was a hybrid confidence-based combination of corpus-induced lexical relations with hand-created rules for temporal ordering. One could consider another obvious hybrid, namely learning from annotations created by GTag-annotated corpora. Since the intuitive baseline fares badly, this may not be that attractive. However, the dramatic impact of closure could help offset the limited coverage provided by human intuitions.

Table 4 (line GTag+closure+ME-C) shows the results of closing the TLINKs produced by GTag’s annotation and then training ME from the resulting data. The results here are evaluated against a held-out test set. We can see that even after closure, the baseline of learning from unclosed human annotations is much poorer than ME-C, and is in fact substantially worse than the majority class on event ordering.

This means that for preprocessing new data sets to produce noisily annotated data for this classification task, it is far better to use machine-learning from closed human annotations rather

than machine-learning from closed annotations produced by an intuitive baseline.

5 Related Work

Our approach of classifying pairs independently during learning does not take into account dependencies between pairs. For example, a classifier may label $\langle X, Y \rangle$ as BEFORE. Given the pair $\langle X, Z \rangle$, such a classifier has no idea if $\langle Y, Z \rangle$ has been classified as BEFORE, in which case, through closure, $\langle X, Z \rangle$ should be classified as BEFORE. This can result in the classifier producing an inconsistently annotated text. The machine learning approach of (Cohen et al. 1999) addresses this, but their approach is limited to total orderings involving BEFORE, whereas TLINKs introduce partial orderings involving BEFORE and five other relations. Future research will investigate methods for tighter integration of temporal reasoning and statistical classification.

The only closely comparable machine-learning approach to the problem of TLINK extraction was that of (Boguraev and Ando 2005), who trained a classifier on Timebank 1.1 for event anchoring for events and times within the same sentence, obtaining an F-measure (for tasks A and B together) of 53.1. Other work in machine-learning and hand-coded approaches, while interesting, is harder to compare in terms of accuracy since they do not use common task definitions, annotation standards, and evaluation measures. (Li et al. 2004) obtained 78-88% accuracy on ordering within-sentence temporal relations in Chinese texts. (Mani et al. 2003) obtained 80.2 F-measure training a decision tree on 2069 clauses in anchoring events to reference times that were inferred for each clause. (Berglund et al. 2006) use a document-level evaluation approach pioneered by (Setzer and Gaizauskas 2000), which uses a distinct evaluation metric. Finally, (Lapata and Lascarides 2004) use found data to successfully learn which (possibly ambiguous) temporal markers connect a main and subordinate clause, without inferring underlying temporal relations.

In terms of hand-coded approaches, (Mani and Wilson 2000) used a baseline method of blindly propagating TempEx time values to events based on proximity, obtaining 59.4% on a small sample of 8,505 words of text. (Filatova and Hovy 2001) obtained 82% accuracy on ‘timestamping’ clauses for a single type of event/topic on a data set of 172 clauses. (Schilder and Habel 2001)

report 84% accuracy inferring temporal relations in German data, and (Li et al. 2001) report 93% accuracy on extracting temporal relations in Chinese. Because these accuracies are on different data sets and metrics, they cannot be compared directly with our methods.

Recently, researchers have developed other tools for automatically tagging aspects of TimeML, including EVENT (Sauri et al. 2005) at 0.80 F-measure and TIMEX3⁶ tags at 0.82-0.85 F-measure. In addition, the TERN competition (tern.mitre.org) has shown very high (close to .95 F-measures) for TIMEX2 tagging, which is fairly similar to TIMEX3. These results suggest the time is ripe for exploiting ‘imperfect’ features in our machine learning approach.

6 Conclusion

Our research has uncovered one new finding: semantic reasoning (in this case, logical axioms for temporal closure), can be extremely valuable in addressing data sparseness. Without it, performance on this task of learning temporal relations is poor; with it, it is excellent. We showed that temporal reasoning can be used as an over-sampling method to dramatically expand the amount of training data for TLINK labeling, resulting in labeling predictive accuracy as high as 93% using an off-the-shelf Maximum Entropy classifier. Future research will investigate this effect further, as well as examine factors that enhance or mitigate this effect in different corpora.

The paper showed that ME-C performed significantly better than a series of increasingly sophisticated baselines involving expansion of rules derived from human intuitions. Our results in these comparisons confirm the lessons learned from the corpus-based revolution, namely that rules based on intuition alone are prone to incompleteness and are hard to tune without access to the distributions found in empirical data. Clearly, lexical rules have a role to play in semantic and pragmatic reasoning from language, as in the discussion of example (2) in Section 1. Such rules, when mined by robust, large corpus-based methods, as in the Google-derived VerbOcean, are clearly relevant, but too specific to apply more than a few times in the OTC corpus.

It may be possible to acquire confidence weights for at least some of the intuitive rules in GTag from Google searches, so that we have a

⁶http://complingone.georgetown.edu/~linguist/GU_TIME_DOWNLOAD.HTML

level field for integrating confidence weights from the fairly general GTag rules and the fairly specific VerbOcean-like lexical rules. Further, the GTag and VerbOcean rules could be incorporated as features for machine learning, along with features from automatic preprocessing.

We have taken pains to use freely downloadable resources like Carafe, VerbOcean, and WEKA to help others easily replicate and quickly ramp up a system. To further facilitate further research, our tools as well as labeled vectors (unclosed as well as closed) are available for others to experiment with.

References

- James Allen. 1984. Towards a General Theory of Action and Time. *Artificial Intelligence*, 23, 2, 123-154.
- Anders Berglund, Richard Johansson and Pierre Nugues. 2006. A Machine Learning Approach to Extract Temporal Information from Texts in Swedish and Generate Animated 3D Scenes. *Proceedings of EACL-2006*.
- Branimir Boguraev and Rie Kubota Ando. 2005. TimeML-Compliant Text Analysis for Temporal Reasoning. *Proceedings of IJCAI-05*, 997-1003.
- Timothy Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations. *Proceedings of EMNLP-04*. <http://semantics.isi.edu/ocean>
- W. Cohen, R. Schapire, and Y. Singer. 1999. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243-270, 1999.
- Janet Hitzeman, Marc Moens and Clare Grover. 1995. Algorithms for Analyzing the Temporal Structure of Discourse. *Proceedings of EACL'95*, Dublin, Ireland, 253-260.
- C.H. Hwang and L. K. Schubert. 1992. Tense Trees as the fine structure of discourse. *Proceedings of ACL'1992*, 232-240.
- Hans Kamp and Uwe Ryle. 1993. *From Discourse to Logic (Part 2)*. Dordrecht: Kluwer.
- Andrew Kehler. 2000. Resolving Temporal Relations using Tense Meaning and Discourse Interpretation, in M. Faller, S. Kaufmann, and M. Pauly, (eds.), *Formalizing the Dynamics of Information*, CSLI Publications, Stanford.
- Mirella Lapata and Alex Lascarides. 2004. Inferring Sentence-internal Temporal Relations. In *Proceedings of the North American Chapter of the Association of Computational Linguistics*, 153-160.
- Alex Lascarides and Nicholas Asher. 1993. Temporal Relations, Discourse Structure, and Commonsense Entailment. *Linguistics and Philosophy* 16, 437-494.
- Wenjie Li, Kam-Fai Wong, Guihong Cao and Chunfa Yuan. 2004. Applying Machine Learning to Chinese Temporal Relation Resolution. *Proceedings of ACL'2004*, 582-588.
- Inderjeet Mani, Barry Schiffman, and Jianping Zhang. 2003. Inferring Temporal Ordering of Events in News. Short Paper. *Proceedings of HLT-NAACL'03*, 55-57.
- Inderjeet Mani and George Wilson. 2000. Robust Temporal Processing of News. *Proceedings of ACL'2000*.
- Rebecca J. Passonneau. A Computational Model of the Semantics of Tense and Aspect. *Computational Linguistics*, 14, 2, 1988, 44-60.
- James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, David Day, Lisa Ferro, Robert Gaizauskas, Marcia Lazo, Andrea Setzer, and Beth Sundheim. 2003. The TimeBank Corpus. *Corpus Linguistics*, 647-656.
- James Pustejovsky, Bob Ingria, Roser Sauri, Jose Castano, Jessica Littman, Rob Gaizauskas, Andrea Setzer, G. Katz, and I. Mani. 2005. The Specification Language TimeML. In I. Mani, J. Pustejovsky, and R. Gaizauskas, (eds.), *The Language of Time: A Reader*. Oxford University Press.
- Roser Sauri, Robert Knippen, Marc Verhagen and James Pustejovsky. 2005. Evita: A Robust Event Recognizer for QA Systems. Short Paper. *Proceedings of HLT/EMNLP 2005*: 700-707.
- Frank Schilder and Christof Habel. 2005. From temporal expressions to temporal information: semantic tagging of news messages. In I. Mani, J. Pustejovsky, and R. Gaizauskas, (eds.), *The Language of Time: A Reader*. Oxford University Press.
- Andrea Setzer and Robert Gaizauskas. 2000. Annotating Events and Temporal Information in Newswire Texts. *Proceedings of LREC-2000*, 1287-1294.
- Marc Verhagen. 2004. *Times Between The Lines*. Ph.D. Dissertation, Department of Computer Science, Brandeis University.
- Marc Vilain, Henry Kautz, and Peter Van Beek. 1989. Constraint propagation algorithms for temporal reasoning: A revised report. In D. S. Weld and J. de Kleer (eds.), *Readings in Qualitative Reasoning about Physical Systems*, Morgan-Kaufman, 373-381.
- Bonnie Webber. 1988. Tense as Discourse Anaphor. *Computational Linguistics*, 14, 2, 1988, 61-73.

An End-to-End Discriminative Approach to Machine Translation

Percy Liang

Alexandre Bouchard-Côté

Dan Klein

Ben Taskar

Computer Science Division, EECS Department

University of California at Berkeley

Berkeley, CA 94720

{pliang, bouchard, klein, taskar}@cs.berkeley.edu

Abstract

We present a perceptron-style discriminative approach to machine translation in which large feature sets can be exploited. Unlike discriminative reranking approaches, our system can take advantage of learned features in all stages of decoding. We first discuss several challenges to error-driven discriminative approaches. In particular, we explore different ways of updating parameters given a training example. We find that making frequent but smaller updates is preferable to making fewer but larger updates. Then, we discuss an array of features and show both how they quantitatively increase BLEU score and how they qualitatively interact on specific examples. One particular feature we investigate is a novel way to introduce learning into the initial phrase extraction process, which has previously been entirely heuristic.

1 Introduction

The generative, noisy-channel paradigm has historically served as the foundation for most of the work in statistical machine translation (Brown et al., 1994). At the same time, discriminative methods have provided substantial improvements over generative models on a wide range of NLP tasks. They allow one to easily encode domain knowledge in the form of features. Moreover, parameters are tuned to directly minimize error rather than to maximize joint likelihood, which may not correspond well to the task objective.

In this paper, we present an end-to-end discriminative approach to machine translation. The proposed system is phrase-based, as in Koehn et al. (2003), but uses an online perceptron training scheme to learn model parameters. Unlike minimum error rate training (Och, 2003), our system is able to exploit large numbers of specific features in the same manner as static reranking systems (Shen et al., 2004; Och et al., 2004). However, unlike static rerankers, our system does not rely on a baseline translation system. Instead, it updates based on its own n -best lists. As parameter

estimates improve, the system produces better n -best lists, which can in turn enable better updates in future training iterations. In this paper, we focus on two aspects of the problem of discriminative translation: the inherent difficulty of learning from reference translations, and the challenge of engineering effective features for this task.

Discriminative learning from reference translations is inherently problematic because standard discriminative methods need to know which outputs are correct and which are not. However, a proposed translation that differs from a reference translation need not be incorrect. It may differ in word choice, literalness, or style, yet be fully acceptable. Pushing our system to avoid such alternate translations is undesirable. On the other hand, even if a system produces a reference translation, it may do so by abusing the hidden structure (sentence segmentation and alignment). We can therefore never be entirely sure whether or not a proposed output is safe to update towards. We discuss this issue in detail in Section 5, where we show that conservative updates (which push the system towards a local variant of the current prediction) are more effective than more aggressive updates (which try to directly update towards the reference).

The second major contribution of this work is an investigation of an array of features for our model. We show how our features quantitatively increase BLEU score, as well as how they qualitatively interact on specific examples. We first consider learning weights for individual phrases and part-of-speech patterns, showing gains from each. We then present a novel way to parameterize and introduce learning into the initial phrase extraction process. In particular, we introduce *alignment constellation* features, which allow us to weight phrases based on the word alignment pattern that led to their extraction. This kind of

feature provides a potential way to initially extract phrases more aggressively and then later down-weight undesirable patterns, essentially learning a weighted extraction heuristic. Finally, we use POS features to parameterize a distortion model in a limited distortion decoder (Zens and Ney, 2004; Tillmann and Zhang, 2005). We show that overall, BLEU score increases from 28.4 to 29.6 on French-English.

2 Approach

2.1 Translation as structured classification

Machine translation can be seen as a structured classification task, in which the goal is to learn a mapping from an input (French) sentence $\mathbf{x}$ to an output (English) sentence $\mathbf{y}$. Given this setup, discriminative methods allow us to define a broad class of features Φ that operate on $(\mathbf{x}, \mathbf{y})$. For example, some features would measure the fluency of $\mathbf{y}$ and others would measure the faithfulness of $\mathbf{y}$ as a translation of $\mathbf{x}$.

However, the translation task in this framework differs from traditional applications of discriminative structured classification such as POS tagging and parsing in a fundamental way. Whereas in POS tagging, there is a one-to-one correspondence between the words $\mathbf{x}$ and the tags $\mathbf{y}$, the correspondence between $\mathbf{x}$ and $\mathbf{y}$ in machine translation is not only much more complex, but is in fact unknown. Therefore, we introduce a hidden correspondence structure $\mathbf{h}$ and work with the feature vector $\Phi(\mathbf{x}, \mathbf{y}, \mathbf{h})$.

The phrase-based model of Koehn et al. (2003) is an instance of this framework. In their model, the correspondence $\mathbf{h}$ consists of (1) the segmentation of the input sentence into phrases, (2) the segmentation of the output sentence into the same number of phrases, and (3) a bijection between the input and output phrases. The feature vector $\Phi(\mathbf{x}, \mathbf{y}, \mathbf{h})$ contains four components: the log probability of the output sentence $\mathbf{y}$ under a language model, the score of translating $\mathbf{x}$ into $\mathbf{y}$ based on a phrase table, a distortion score, and a length penalty.¹ In Section 6, we vastly increase the number of features to take advantage of the full power of discriminative training.

Another example of this framework is the hierarchical model of Chiang (2005). In this model the correspondence $\mathbf{h}$ is a synchronous parse tree

¹More components can be added to the feature vector if additional language models or phrase tables are available.

over input and output sentences, and features include the scores of various productions used in the tree.

Given features Φ and a corresponding set of parameters $\mathbf{w}$, a standard classification rule f is to return the highest scoring output sentence $\mathbf{y}$, maximizing over correspondences $\mathbf{h}$:

$$f(\mathbf{x}; \mathbf{w}) = \operatorname{argmax}_{\mathbf{y}, \mathbf{h}} \mathbf{w} \cdot \Phi(\mathbf{x}, \mathbf{y}, \mathbf{h}). \quad (1)$$

In the phrase-based model, computing the argmax exactly is intractable, so we approximate f with beam decoding.

2.2 Perceptron-based training

To tune the parameters $\mathbf{w}$ of the model, we use the averaged perceptron algorithm (Collins, 2002) because of its efficiency and past success on various NLP tasks (Collins and Roark, 2004; Roark et al., 2004). In principle, $\mathbf{w}$ could have been tuned by maximizing conditional probability or maximizing margin. However, these two options require either marginalization or numerical optimization, neither of which is tractable over the space of output sentences $\mathbf{y}$ and correspondences $\mathbf{h}$. In contrast, the perceptron algorithm requires only a decoder that computes $f(\mathbf{x}; \mathbf{w})$.

Recall the traditional perceptron update rule on an example $(\mathbf{x}_i, \mathbf{y}_i)$ is

$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(\mathbf{x}_i, \mathbf{y}_t) - \Phi(\mathbf{x}_i, \mathbf{y}_p), \quad (2)$$

where $\mathbf{y}_t = \mathbf{y}_i$ is the *target output* and $\mathbf{y}_p = f(\mathbf{x}_i; \mathbf{w}) = \operatorname{argmax}_{\mathbf{y}} \mathbf{w} \cdot \Phi(\mathbf{x}_i, \mathbf{y})$ is the prediction using the current parameters $\mathbf{w}$.

We adapt this update rule to work with hidden variables as follows:

$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(\mathbf{x}_i, \mathbf{y}_t, \mathbf{h}_t) - \Phi(\mathbf{x}_i, \mathbf{y}_p, \mathbf{h}_p), \quad (3)$$

where $(\mathbf{y}_p, \mathbf{h}_p)$ is the argmax computation in Equation 1, and $(\mathbf{y}_t, \mathbf{h}_t)$ is the *target* that we update towards. If $(\mathbf{y}_t, \mathbf{h}_t)$ is the same argmax computation with the additional constraint that $\mathbf{y}_t = \mathbf{y}_i$, then Equation 3 can be interpreted as a Viterbi approximation to the stochastic gradient

$$\mathbb{E}_{P(\mathbf{h}|\mathbf{x}_i, \mathbf{y}_i; \mathbf{w})} \Phi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}) - \mathbb{E}_{P(\mathbf{y}, \mathbf{h}|\mathbf{x}_i; \mathbf{w})} \Phi(\mathbf{x}_i, \mathbf{y}, \mathbf{h})$$

for the following conditional likelihood objective:

$$P(\mathbf{y}_i | \mathbf{x}_i) \propto \sum_{\mathbf{h}} \exp(\mathbf{w} \cdot \Phi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h})).$$

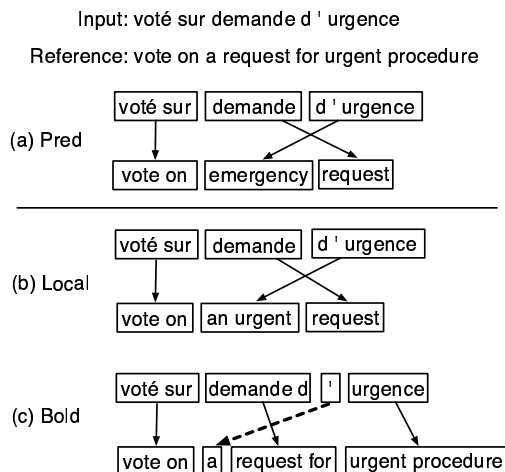


Figure 1: Given the current prediction (a), there are two possible updates, local (b) and bold (c). Although the bold update (c) reaches the reference translation, a bad correspondence is used. The local update (b) does not reach the reference, but is more reasonable than (c).

Discriminative training with hidden variables has been handled in this probabilistic framework (Quattoni et al., 2004; Koo and Collins, 2005), but we choose Equation 3 for efficiency.

It turns out that using the Viterbi approximation (which we call *bold* updating) is not always the best strategy. To appreciate the difficulty, consider the example in Figure 1. Suppose we make the prediction (a) with the current set of parameters. There are often several acceptable output translations $\mathbf{y}$, for example, (b) and (c). Since (c)’s output matches the reference translation, should we update towards (c)? In this case, the answer is negative. The problem with (c) is that the correspondence $\mathbf{h}$ contains an incorrect alignment ($\cdot, a$). However, since $\mathbf{h}$ is unobserved, the training procedure has no way of knowing this. While the output in (b) is farther from the reference, its correspondence $\mathbf{h}$ is much more reasonable. In short, it does not suffice for $\mathbf{y}_t$ to be good; both $\mathbf{y}_t$ and $\mathbf{h}_t$ need to be good. A major challenge in using the perceptron algorithm for machine translation is determining the target $(\mathbf{y}_t, \mathbf{h}_t)$ in Equation 3. Section 5 discusses possible targets to update towards.

3 Dataset

Our experiments were done on the French-English portion of the Europarl corpus (Koehn, 2002),

Dataset	TRAIN	DEV	TEST
Years	'99–'01	'02	'03
# sentences	67K	first 1K	first 1K
# words (unk.)	715K	10.4K (35)	10.8K (48)

Table 1: The Europarl dataset split we used and various statistics on length 5–15 sentences. The number of French word tokens is given, along with the number that were not seen among the 414K total sentences in TRAIN (which includes all lengths).

which consists of European parliamentary proceedings from 1996 to 2003.

We split the data into three sets according to Table 1. TRAIN served two purposes: it was used to construct the features, and the 5–15 length sentences were used for tuning the parameters of those features. DEV, which consisted of the first 1K length 5–15 sentences in 2002, was used to evaluate the performance of the system as we developed it. Note that the DEV set was not used to tune any parameters; tuning was done exclusively on TRAIN. At the end we ran our models once on TEST to get final numbers.²

4 Models

Our experiments used phrase-based models (Koehn et al., 2003), which require a translation table and language model for decoding and feature computation. To facilitate comparison with previous work, we created the translation tables using the same techniques as Koehn et al. (2003).³ The language model was a Kneser-Ney interpolated trigram model generated using the SRILM toolkit (Stolcke, 2002). We built our own phrase-based beam decoder that can handle arbitrary features.⁴ The contributions of features are incrementally added into the score as decoding

²We also experimented with several combinations of jack-knifing to prevent overfitting, in which we selected features on TRAIN-OLD (1996–1998 Europarl corpus) and tuned the parameters on TRAIN, or vice-versa. However, it turned out that using TRAIN-OLD was suboptimal since that data is less relevant to DEV. Another alternative is to combine TRAIN-OLD and TRAIN into one dual-purpose dataset. The differences between this and our current approach were inconclusive.

³In other words, we used GIZA++ to construct a word alignment in each direction and a growth heuristic to combine them. We extracted all the substrings that are closed under this high-quality word alignment and computed surface statistics from cooccurrences counts.

⁴In our experiments, we used a beam size of 10, which we found to be only slightly worse than using a beam of 100.

proceeds.

We experimented with two levels of distortion: *monotonic*, where the phrasal alignment is monotonic (but word reordering is still possible within a phrase) and *limited distortion*, where only adjacent phrases are allowed to exchange positions (Zens and Ney, 2004). In the future, we plan to explore our discriminative framework on a full distortion model (Koehn et al., 2003) or even a hierarchical model (Chiang, 2005).

Throughout the following experiments, we trained the perceptron algorithm for 10 iterations. The weights were initialized to 1 on the translation table, 1 on the language model (the blanket features in Section 6), and 0 elsewhere. The next two sections give experiments on the two key components of a discriminative machine translation system: choosing the proper update strategy (Section 5) and including powerful features (Section 6).

5 Update strategies

This section describes the importance of choosing a good update strategy—the difference in BLEU score can be as large as 1.2 between different strategies. An update strategy specifies the *target* ($\mathbf{y}_t, \mathbf{h}_t$) that we update towards (Equation 3) given the current set of parameters and a provided reference translation ($\mathbf{x}_i, \mathbf{y}_i$). As mentioned in Section 2.2, faithful output (i.e. $\mathbf{y}_t = \mathbf{y}_i$) does not imply that updating towards ($\mathbf{y}_t, \mathbf{h}_t$) is desirable. In fact, such a constrained target might not even be *reachable* by the decoder, for example, if the reference is very non-literal.

We explored the following three ways to choose the target ($\mathbf{y}_t, \mathbf{h}_t$):

- **Bold updating:** Update towards the highest scoring option ($\mathbf{y}, \mathbf{h}$), where $\mathbf{y}$ is constrained to be the reference $\mathbf{y}_i$ but $\mathbf{h}$ is unconstrained. Examples not reachable by the decoder are skipped.
- **Local updating:** Generate an n -best list using the current parameters. Update towards the option with the highest BLEU score.⁵

⁵Since BLEU score (k -BLEU with $k = 4$) involves computing a geometric mean over i -grams, $i = 1, \dots, k$, it is zero if the translation does not have at least one k -gram in common with the reference translation. Since a BLEU score of zero is both unhelpful for choosing from the n -best and common when computed on just a single example, we instead used a smoothed version for choosing the target: $\sum_{i=1}^4 \frac{i \cdot \text{BLEU}(x, y)}{2^{4-i+1}}$. We still report NIST’s usual 4-gram BLEU.

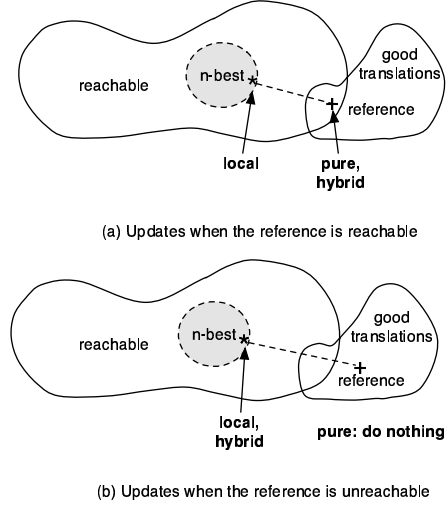


Figure 2: The three update strategies under two scenarios.

- **Hybrid updating:** Do a bold update if the reference is reachable. Otherwise, do a local update.

Figure 2 shows the space of translations schematically. On each training example, our decoder produces an n -best list. The reference translation may or may not be reachable.

Bold updating most resembles the traditional perceptron update rule (Equation 2). We are ensured that the target output $\mathbf{y}$ will be correct, although the correspondence $\mathbf{h}$ might be bad. Another weakness of bold updating is that we might not make full use of the training data.

Local updating uses every example, but its steps are more cautious. It can be viewed as “dynamic reranking,” where parameters are updated using the best option on the n -best list, similar to standard static reranking. The key difference is that, unlike static reranking, the parameter updates propagate back to the baseline classifier, so that the n -best list improves over time. In this regard, dynamic reranking remedies one of the main weaknesses of static reranking, which is that the performance of the system is directly limited by the quality of the baseline classifier.

Hybrid updating combines the two strategies: it makes full use of the training data as in local updating, but still tries to make swift progress towards the reference translation as in bold updating.

We conducted experiments to see which of the updating strategies worked best. We trained on

Decoder	Bold	Local	Hybrid
Monotonic	34.3	34.6	34.5
Limited distortion	33.5	34.7	33.6

Table 2: Comparison of BLEU scores between different updating strategies for the monotonic and limited distortion decoders on DEV.

5000 of the 67K available examples, using the BLANKET+LEX+POS feature set (Section 6). Table 2 shows that local updating is the most effective, especially when using the limited distortion decoder.

In bold updating, only a small fraction of the 5000 examples (1296 for the monotonic decoder and 1601 for the limited distortion decoder) had reachable reference translations, and, therefore, contributed to parameter updates. One might therefore hypothesize that local updating performs better simply because it is able to leverage more data. This is not the full story, however, since the hybrid approach (which makes the same number of updates) performs significantly worse than local updating when using the limited distortion decoder.

To see the problem with bold updating, recall the example in Figure 1. Bold updating tries to reach the reference at all costs, even if it means abusing the hidden correspondence in the process. In the example, the alignment $(', a)$ is unreasonable, but the algorithm has no way to recognize this. Local updating is much more stable since it only updates towards sentences in the n -best list.

When using the limited distortion decoder, bold updating is even more problematic because the added flexibility of phrase swaps allows more preposterous alignments to be produced. Limited distortion decoding actually performs *worse* than monotonic decoding with bold updating, but better with local updating.

Another difference between bold updating and local updating is that the BLEU score on the training data is dramatically higher for bold updating than for local (or hybrid) updating: 80 for the former versus 40 for the latter. This is not surprising given that bold updating aggressively tries to obtain the references. However, what is surprising is that although bold updating appears to be overfitting severely, its BLEU score on the DEV does not suffer much in the monotonic case.

Model	DEV BLEU	TEST BLEU
Monotonic		
BLANKET (untuned)	33.0	28.3
BLANKET	33.4	28.4
BLANKET+LEX	35.0	29.2
BLANKET+LEX+POS	35.3	29.6
Pharaoh (MERT)	34.5	28.8
Full-distortion		
Pharaoh (MERT)	34.9	29.5

Table 3: Main results on our system with different feature sets compared to minimum error-rate trained Pharaoh.

6 Features

This section shows that by adding an array of expressive features and discriminatively learning their weights, we can obtain a 2.3 increase in BLEU score on DEV. We add these features incrementally, first tuning blanket features (Section 6.1), then adding lexical features (Section 6.2), and finally adding part-of-speech (POS) features (Section 6.3). Table 3 summarizes the performance gains.

For the experiments in this section, we used the local updating strategy and the monotonic decoder for efficiency. We train on all 67K of the length 5–15 sentences in TRAIN.⁶

6.1 Blanket features

The blanket features (BLANKET) consist of the translation log-probability and the language model log-probability, which are two of the components of the Pharaoh model (Section 2.1). After discriminative training, the relative weight of these two features is roughly 2:1, resulting in a BLEU score increase from 33.0 (setting both weights to 1) to 33.4.

The following simple example gives a flavor of the discriminative approach. The untuned system translated the French phrase *trente-cinq langues* into *five languages* in a DEV example. Although the probability $P(\text{five} \mid \text{trente-cinq}) = 0.065$ is rightly much smaller than $P(\text{thirty-five} \mid \text{trente-cinq}) = 0.279$, the language model favors *five languages* over *thirty-five languages*. The trained system downweights the language model and recovers the correct translation.

⁶We used sentences of length 5–15 to facilitate comparisons with Koehn et al. (2003) and to enable rapid experimentation with various feature sets. Experiments on sentences of length 5–50 showed similar gains in performance.

6.2 Lexical features

The blanket features provide a rough guide for translation, but they are far too coarse to fix specific mistakes. We therefore add *lexical features* (LEX) to allow for more fine-grained control. These features come in two varieties. Lexical phrase features indicate the presence of a specific translation phrase, such as (*y a-t-il, are there*), and lexical language model features indicate the presence of a specific output n-gram, such as *of the*. Lexical language model features have been exploited successfully in discriminative language modeling to improve speech recognition performance (Roark et al., 2004). We confirm the utility of the two kinds of lexical features: BLANKET+LEX achieves a BLEU score of 35.0, an improvement of 1.6 over BLANKET.

To understand the effect of adding lexical features, consider the ten with highest and lowest weights after training:

64	any comments ?	-55	(des, of)
63	(y a-t-il, are there)	-52	(y a-t-il, are there any)
62	there any comments	-42	there any of
57	any comments	-39	of comments
46	(des, any)	-38	of comments ?

These features can in fact be traced back to the following example:

Input	y a-t-il	des	observations ?
B	are there any	of	comments ?
B+L	are there	any	comments ?

The second and third rows are the outputs of BLANKET (wrong) and BLANKET+LEX (correct), respectively. The correction can be accredited to two changes in feature weights. First, the lexical feature (*y a-t-il, are there any*) has been assigned a negative weight and (*y a-t-il, are there*) a positive weight to counter the fact that the former phrase incorrectly had a higher score in the original translation table. Second, (*des, of*) is preferred over (*des, any*), even though the former is a better translation in isolation. This apparent degradation causes no problems, because when *des* should actually be translated to *of*, these words are usually embedded in larger phrases, in which case the isolated translation probability plays no role.

Another example of a related phenomenon is the following:

Input	...	pour cela que	j ' ai	voté favorablement	.
B	...	for that	i have	voted in favour	.
B+L	...	for this reason	i	voted in favour	.

Counterintuitively, the phrase pair (*j ' ai, I have*) ends up with a very negative weight. The reason behind this is that in French,

j ' ai is often used in a paraphrastic construction which should be translated into the simple past in English. For that to happen, *j ' ai* needs to be aligned with *I*. Since (*j ' ai, I*) has a small score compare to (*j ' ai, I have*) in the original translation table, downweighting the latter pair allows this sentence to be translated correctly.

A general trend is that literal phrase translations are downweighted. Lessening the pressure to literally translate certain phrases allows the language model to fill in the gaps appropriately with suitable non-literal translations. This point highlights the strength of discriminative training: weights are jointly tuned to account for the intricate interactions between overlapping phrases, which is something not achievable by estimating the weights directly from surface statistics.

6.3 Part-of-speech features

While lexical features are useful for eliminating specific errors, they have limited ability to generalize to related phrases. This suggests the use of similar features which are abstracted to the POS level.⁷ In our experiments, we used the TreeTagger POS tagger (Schmid, 1994), which ships pre-trained on several languages, to map each word to its majority POS tag. We could also relatively easily base our features on context-dependent POS tags: the entire input sentence is available before decoding begins, and the output sentence is decoded left-to-right and could be tagged incrementally.

Where we had lexical phrase features, such as (*la réalisation du droit, the right*), we now also have their POS abstractions, for instance (DT NN IN NN, DT NN). This phrase pair is undesirable, not because of particular lexical facts about *la réalisation*, but because dropping a nominal head is generally to be avoided. The lexical language model features have similar POS counterparts. With these two kinds of POS features, we obtained an 0.3 increase in BLEU score from BLANKET+LEX to BLANKET+LEX+POS.

Finally, when we use the limited distortion decoder, it is important to learn when to swap adjacent phrases. Unlike Pharaoh, which simply has a uniform penalty for swaps, we would like to use context—in particular, POS information. For example, we would like to know that if a (JJ, JJ)

⁷We also tried using word clusters (Brown et al., 1992) instead of POS but found that POS was more helpful.

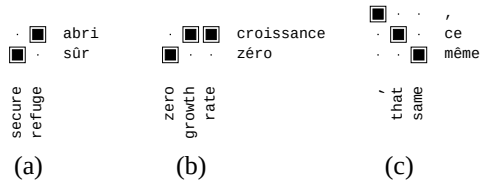


Figure 3: Three constellation features with example phrase pairs. Constellations (a) and (b) have large positive weights and (c) has a large negative weight.

phrase is constructed after a (NN, NN) phrase, they are reasonable candidates for swapping because of regular word-order differences between French and English. While the bulk of our results are presented for the monotonic case, the limited distortion results of Table 2 use these lexical swap features; without parameterized swap features, accuracy was below the untuned monotonic baseline.

An interesting statistic is the number of nonzero feature weights that were learned using each feature set. BLANKET has only 4 features, while BLANKET+LEX has 1.55 million features.⁸ Remarkably, BLANKET+LEX+POS has fewer features—only 1.24 million. This is an effect of generalization ability—POS information somewhat reduces the need for specific lexical features.

6.4 Alignment constellation features

Koehn et al. (2003) demonstrated that choosing the appropriate heuristic for extracting phrases is very important. They showed that the difference in BLEU score between various heuristics was as large as 2.0.

The process of phrase extraction is difficult to optimize in a non-discriminative setting: many heuristics have been proposed (Koehn et al., 2003), but it is not obvious which one should be chosen for a given language pair. We propose a natural way to handle this part of the translation pipeline. The idea is to push the learning process all the way down to the phrase extraction by parameterizing the phrase extraction heuristic itself.

The heuristics in Koehn et al. (2003) decide whether to extract a given phrase pair based on the underlying word alignments (see Figure 3 for three examples), which we call *constellations*. Since we do not know which constellations correspond to

⁸Both the language model and translation table components have two features, one for known words and one for unknown words.

Features	-CONST	+CONST
BLANKET	31.8	32.2
BLANKET+LEX	32.2	32.5
BLANKET+LEX+POS	32.3	32.5

Table 4: DEV BLEU score increase resulting from adding constellation features.

good phrase pairs, we introduce an alignment constellation feature to indicate the presence of a particular alignment constellation.⁹

Table 4 details the effect of adding constellation features on top of our previous feature sets.¹⁰ We get a minor increase in BLEU score from each feature set, although there is no gain by adding POS features in addition to constellation features, probably because POS and constellation features provide redundant information for French-English translations.

It is interesting to look at the constellations with highest and lowest weights, which are perhaps surprising at first glance. At the top of the list are word inversions (Figure 3 (a) and (b)), while long monotonic constellations fall at the bottom of the list (c). Although monotonic translations are much more frequent than word inversions in our dataset, when translations are monotonic, shorter segmentations are preferred. This phenomenon is another manifestation of the complex interaction of phrase segmentations.

7 Final results

The last column of Table 3 shows the performance of our methods on the final TEST set. Our best test BLEU score is 29.6 using BLANKET+LEX+POS, an increase of 1.3 BLEU over our untuned feature set BLANKET. The discrepancy between DEV performance and TEST performance is due to temporal distance from TRAIN and high variance in BLEU score.¹¹

We also compared our model with Pharaoh (Koehn et al., 2003). We tuned Pharaoh’s four parameters using minimum error rate training (Och, 2003) on DEV.¹² We obtained an increase of 0.8

⁹As in the POS features, we map each phrase pair to its majority constellation.

¹⁰Due to time constraints, we ran these experiments on 5000 training examples using bold updating.

¹¹For example, the DEV BLEU score for BLANKET+LEX ranges from 28.6 to 33.2, depending on which block of 1000 sentences we chose.

¹²We used the training scripts from the 2006 MT Shared Task. We still tuned our model parameters on TRAIN and

BLEU over the Pharaoh, run with the monotone flag.¹³ Even though we are using a monotonic decoder, our best results are still slightly better than the version of Pharaoh that permits arbitrary distortion.

8 Related work

In machine translation, most discriminative approaches currently fall into two general categories. The first approach is to reuse the components of a generative model, but tune their relative weights in a discriminative fashion (Och and Ney, 2002; Och, 2003; Chiang, 2005). This approach only works in practice with a small handful of parameters.

The second approach is to use reranking, in which a baseline classifier generates an n -best list of candidate translations, and a separate discriminative classifier chooses amongst them (Shen et al., 2004; Och et al., 2004). The major limitation of a reranking system is its dependence on the underlying baseline system, which bounds the potential improvement from discriminative training. In machine translation, this limitation is a real concern; it is common for all translations on moderately-sized n -best lists to be of poor quality. For instance, Och et al. (2004) reported that a 1000-best list was required to achieve performance gains from reranking. In contrast, the decoder in our system can use the feature weights learned in the previous iteration.

Tillmann and Zhang (2005) present a discriminative approach based on local models. Their formulation explicitly decomposed the score of a translation into a sequence of local decisions, while our formulation allows global estimation.

9 Conclusion

We have presented a novel end-to-end discriminative system for machine translation. We studied update strategies, an important issue in on-line discriminative training for MT, and conclude that making many smaller (conservative) updates is better than making few large (aggressive) updates. We also investigated the effect of adding many expressive features, which yielded a 0.8 increase in BLEU score over monotonic Pharaoh.

Acknowledgments We would like to thank our reviewers for their comments. This work was sup-

only used DEV to optimize the number of training iterations.

¹³This result is significant with p -value 0.0585 based on approximate randomization (Riezler and Maxwell, 2005).

ported by a FQRNT fellowship to second author and a Microsoft Research New Faculty Fellowship to the third author.

References

- Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-Based n -gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479.
- Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1994. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19:263–311.
- David Chiang. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *ACL 2005*.
- Michael Collins and Brian Roark. 2004. Incremental Parsing with the Perceptron Algorithm. In *ACL 2004*.
- Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *EMNLP 2002*.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *HLT-NAACL 2003*.
- Philipp Koehn. 2002. Europarl: A Multilingual Corpus for Evaluation of Machine Translation.
- Terry Koo and Michael Collins. 2005. Hidden-Variable Models for Discriminative Reranking. In *EMNLP 2005*.
- Franz Josef Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *ACL 2002*.
- Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, and Anoop Sarkar. 2004. A Smorgasbord of Features for Statistical Machine Translation. In *HLT-NAACL 2004*.
- Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *ACL 2003*.
- Ariadna Quattoni, Michael Collins, and Trevor Darrell. 2004. Conditional Random Fields for Object Recognition. In *NIPS 2004*.
- Stefan Riezler and John T. Maxwell. 2005. On Some Pitfalls in Automatic Evaluation and Significance Testing for MT. In *Workshop on Intrinsic and Extrinsic Evaluation Methods for MT and Summarization (MTSE)*.
- Brian Roark, Murat Saraclar, Michael Collins, and Mark Johnson. 2004. Discriminative Language Modeling with Conditional Random Fields and the Perceptron Algorithm. In *ACL 2004*.
- Helmut Schmid. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*.
- Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative Reranking for Machine Translation. In *HLT-NAACL 2004*.
- Andreas Stolcke. 2002. SRILM: An Extensible Language Modeling Toolkit. In *ICSLP 2002*.
- Christoph Tillmann and Tong Zhang. 2005. A Localized Prediction Model for Statistical Machine Translation. In *ACL 2005*.
- Richard Zens and Hermann Ney. 2004. Improvements in Statistical Phrase-Based Translation. In *HLT-NAACL 2004*.

Semi-Supervised Training for Statistical Word Alignment

Alexander Fraser

ISI / University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
fraser@isi.edu

Daniel Marcu

ISI / University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
marcu@isi.edu

Abstract

We introduce a semi-supervised approach to training for statistical machine translation that alternates the traditional Expectation Maximization step that is applied on a large training corpus with a discriminative step aimed at increasing word-alignment quality on a small, manually word-aligned sub-corpus. We show that our algorithm leads not only to improved alignments but also to machine translation outputs of higher quality.

1 Introduction

The most widely applied training procedure for statistical machine translation — IBM model 4 (Brown et al., 1993) unsupervised training followed by post-processing with symmetrization heuristics (Och and Ney, 2003) — yields low quality word alignments. When compared with gold standard parallel data which was manually aligned using a high-recall/precision methodology (Melamed, 1998), the word-level alignments produced automatically have an F-measure accuracy of 64.6 and 76.4% (see Section 2 for details).

In this paper, we improve word alignment and, subsequently, MT accuracy by developing a range of increasingly sophisticated methods:

1. We first recast the problem of estimating the IBM models (Brown et al., 1993) in a discriminative framework, which leads to an initial increase in word-alignment accuracy.
2. We extend the IBM models with new (sub)models, which leads to additional increases in word-alignment accuracy. In the process, we also show that these improvements are explained not only by the power

of the new models, but also by a novel search procedure for the alignment of highest probability.

3. Finally, we propose a training procedure that interleaves discriminative training with maximum likelihood training.

These steps lead to word alignments of higher accuracy which, in our case, correlate with higher MT accuracy.

The rest of the paper is organized as follows. In Section 2, we review the data sets we use to validate experimentally our algorithms and the associated baselines. In Section 3, we present iteratively our contributions that eventually lead to absolute increases in alignment quality of 4.8% for French/English and 4.8% for Arabic/English, as measured using F-measure for large word alignment tasks. These contributions pertain to the casting of the training procedure in the discriminative framework (Section 3.1); the IBM model extensions and modified search procedure for the Viterbi alignments (Section 3.2); and the interleaved, minimum error/maximum likelihood, training algorithm (Section 4). In Section 5, we assess the impact that our improved alignments have on MT quality. We conclude with a comparison of our work with previous research on discriminative training for word alignment and a short discussion of semi-supervised learning.

2 Data Sets and Baseline

We conduct experiments on alignment and translation tasks using Arabic/English and French/English data sets (see Table 1 for details). Both sets have training data and two gold standard word alignments for small samples of the training data, which we use as the alignment

		ARABIC/ENGLISH		FRENCH/ENGLISH	
		A	E	F	E
TRAINING	SENTS	3,713,753		2,842,184	
	WORDS	102,473,086	119,994,972	75,794,254	67,366,819
	VOCAB	489,534	231,255	149,568	114,907
	SINGLETONS	199,749	104,155	60,651	47,765
ALIGN DISCR.	SENTS	100		110	
	WORDS	1,712	2,010	1,888	1,726
	LINKS	2,129		2,292	
ALIGN TEST	SENTS	55		110	
	WORDS	1,004	1,210	1,899	1,716
	LINKS	1,368		2,176	
MAX BLEU	SENTS	728 (4 REFERENCES)		833 (1 REFERENCE)	
	WORDS	17664	22.0K TO 24.5K	20,562	17,454
TRANS. TEST	SENTS	663 (4 REFERENCES)		2,380 (1 REFERENCE)	
	WORDS	16,075	19.0K TO 21.6K	58,990	49,182

Table 1: Datasets

SYSTEM	F-MEASURE F TO E	F-MEASURE E TO F	F-MEASURE BEST SYMM.
A/E MODEL 4: ITERATION 4	65.6 / 60.5	53.6 / 50.2	69.1 / 64.6 (UNION)
F/E MODEL 4: ITERATION 4	73.8 / 75.1	74.2 / 73.5	76.5 / 76.4 (REFINED)

Table 2: Baseline Results. F-measures are presented on both the alignment discriminative training set and the alignment test set sub-corpora, separated by /.

discriminative training set and alignment test set. Translation quality is evaluated by translating a held-out translation test set. An additional translation set called the Maximum BLEU set is employed by the SMT system to train the weights associated with the components of its log-linear model (Och, 2003).

The training corpora are publicly available: both the Arabic/English data and the French/English Hansards were released by LDC. We created the manual word alignments ourselves, following the Blinker guidelines (Melamed, 1998).

To train our baseline systems we follow a standard procedure. The models were trained two times, first using French or Arabic as the source language and then using English as the source language. For each training direction, we run GIZA++ (Och and Ney, 2003), specifying 5 iterations of Model 1, 4 iterations of the HMM model (Vogel et al., 1996), and 4 iterations of Model 4. We quantify the quality of the resulting hypothesized alignments with F-measure using the manually aligned sets.

We present the results for three different conditions in Table 2. For the “F to E” direction the models assign non-zero probability to alignments consisting of links from one Foreign word to zero or more English words, while for “E to F” the models assign non-zero probability to alignments

consisting of links from one English word to zero or more Foreign words. It is standard practice to improve the final alignments by combining the “F to E” and “E to F” directions using symmetrization heuristics. We use the “union”, “refined” and “intersection” heuristics defined in (Och and Ney, 2003) which are used in conjunction with IBM Model 4 as the baseline in virtually all recent work on word alignment. In Table 2, we report the best symmetrized results.

The low F-measure scores of the baselines motivate our work.

3 Improving Word Alignments

3.1 Discriminative Reranking of the IBM Models

We reinterpret the five groups of parameters of Model 4 listed in the first five lines of Table 3 as sub-models of a log-linear model (see Equation 1). Each sub-model h_m has an associated weight λ_m . Given a vector of these weights λ , the alignment search problem, i.e. the search to return the best alignment $\hat{a}$ of the sentences e and f according to the model, is specified by Equation 2.

$$p_{\lambda}(f, a|e) = \frac{\exp(\sum_i \lambda_i h_i(a, e, f))}{\sum_{a', f'} \exp(\sum_i \lambda_i h_i(a', e, f'))} \quad (1)$$

$$\hat{a} = \operatorname{argmax}_a \sum_i \lambda_i h_i(f, a, e) \quad (2)$$

m	Model 4	Description	m	Description
1	$t(f e)$	translation probs, f and e are words	9	translation table using approx. stems
2	$n(\phi e)$	fertility probs, ϕ is number of words generated by e	10	backoff fertility (fertility estimated over all e)
3	<i>null</i>	parameters used in generating Foreign words which are unaligned	11	backoff fertility for words with count ≤ 5
4	$d_1(\Delta j)$	movement probs of leftmost Foreign word translated from a particular e	12	translation table from HMM iteration 4
5	$d_{>1}(\Delta j)$	movement probs of other Foreign words translated from a particular e	13	zero fertility English word penalty
6		translation table from refined combination of both alignments	14	non-zero fertility English word penalty
7		translation table from union of both alignments	15	NULL Foreign word penalty
8		translation table from intersection of both alignments	16	non-NULL Foreign word penalty

Table 3: Sub-Models. Note that sub-models 1 to 5 are IBM Model 4, sub-models 6 to 16 are new.

Log-linear models are often trained to maximize entropy, but we will train our model directly on the final performance criterion. We use 1-F-measure as our error function, comparing hypothesized word alignments for the discriminative training set with the gold standard.

Och (2003) has described an efficient exact one-dimensional error minimization technique for a similar search problem in machine translation. The technique involves calculating a piecewise constant function $f_m(x)$ which evaluates the error of the hypotheses which would be picked by equation 2 from a set of hypotheses if we hold all weights constant, except for the weight λ_m (which is set to x).

The discriminative reranking algorithm is initialized with the parameters of the sub-models θ , an initial choice of the λ vector, gold standard word alignments (labels) for the alignment discriminative training set, the constant N specifying the N -best list size used¹, and an empty master set of hypothesized alignments. The algorithm is a three step loop:

1. Enrich the master set of hypothesized alignments by producing an N -best list using λ . If all of the hypotheses in the N -best list are already in the master set, the algorithm has converged, so terminate the loop.
2. Consider the current λ vector and 999 additional randomly generated vectors, setting λ to the vector with lowest error on the master set.
3. Repeatedly run Och’s one-dimensional error minimization step until there is no further error reduction (this results in a new vector λ).

¹ $N = 128$ for our experiments

3.2 Improvements to the Model and Search

3.2.1 New Sources of Knowledge

We define new sub-models to model factors not captured by Model 4. These are lines 6 to 16 of Table 3, where we use the “E to F” alignment direction as an example. We use word-level translation tables informed by both the “E to F” and the “F to E” translation directions derived using the three symmetrization heuristics, the “E to F” translation table from the final iteration of the HMM model and an “E to F” translation table derived using approximative stemming. The approximative stemming sub-model (sub-model 9) uses the first 4 letters of each vocabulary item as the stem for English and French while for Arabic we use the full word as the stem. We also use sub-models for backed off fertility, and direct penalization of unaligned English words (“zero fertility”) and aligned English words, and unaligned Foreign words (“NULL-generated” words) and aligned Foreign words. This is a small sampling of the kinds of knowledge sources we can use in this framework; many others have been proposed in the literature.

Table 4 shows an evaluation of discriminative reranking. We observe:

1. The first line is the starting point, which is the Viterbi alignment of the 4th iteration of HMM training.
2. The 1-to-many alignments generated by discriminatively reranking Model 4 are better than the 1-to-many alignments of four iterations of Model 4.
3. The 1-to-many alignments of the discriminatively reranked extended model are much better than four iterations of Model 4.

SYSTEM	F-MEASURE F TO E	F-MEASURE E TO F	F-MEASURE BEST SYMM.
A/E LAST ITERATION HMM	58.6 / 54.4	47.7 / 39.9	62.1 / 57.0 (UNION)
A/E MODEL 4 RERANKING	65.3 / 59.5	55.7 / 51.4	69.7 / 64.6 (UNION)
A/E EXTENDED MODEL RERANKING	68.4 / 62.2	61.6 / 57.7	72.0 / 66.4 (UNION)
A/E MODEL 4: ITERATION 4	65.6 / 60.5	53.6 / 50.2	69.1 / 64.6 (UNION)
F/E LAST ITERATION HMM	72.4 / 73.9	71.5 / 71.8	76.4 / 77.3 (REFINED)
F/E MODEL 4 RERANKING	77.9 / 77.9	78.4 / 77.7	79.2 / 79.4 (REFINED)
F/E EXTENDED MODEL RERANKING	78.7 / 80.2	79.3 / 79.6	79.6 / 80.4 (REFINED)
F/E MODEL 4: ITERATION 4	73.8 / 75.1	74.2 / 73.5	76.5 / 76.4 (REFINED)

Table 4: Discriminative Reranking with Improved Search. F-measures are presented on both the alignment discriminative training set and the alignment test set sub-corpora, separated by /.

- The discriminatively reranked extended model outperforms four iterations of Model 4 in both cases with the best heuristic symmetrization, but some of the gain is lost as we are optimizing the F-measure of the 1-to-many alignments rather than the F-measure of the many-to-many alignments directly.

Overall, the results show our approach is better than or competitive with running four iterations of unsupervised Model 4 training.

3.2.2 New Alignment Search Algorithm

Brown et al. (1993) introduced operations defining a hillclimbing search appropriate for Model 4. Their search starts with a complete hypothesis and exhaustively applies two operations to it, selecting the best improved hypothesis it can find (or terminating if no improved hypothesis is found). This search makes many search errors². We developed a new alignment algorithm to reduce search errors:

- We perform an initial hillclimbing search (as in the baseline algorithm) but construct a priority queue of possible other candidate alignments to consider.
- Alignments which are expanded are marked so that they will not be returned to at a future point in the search.
- The alignment search operates by considering complete hypotheses so it is an “anytime” algorithm (meaning that it always has a current best guess). Timers can therefore be used to terminate the processing of the priority queue of candidate alignments.

The first two improvements are related to the well-known Tabu local search algorithm (Glover,

²A search error in a word aligner is a failure to find the best alignment according to the model, i.e. in our case a failure to maximize Equation 2.

1986). The third improvement is important for restricting total time used when producing alignments for large training corpora.

We performed two experiments. The first evaluates the number of search errors. For each corpus we sampled 1000 sentence pairs randomly, with no sentence length restriction. Model 4 parameters are estimated from the final HMM Viterbi alignment of these sentence pairs. We then search to try to find the Model 4 Viterbi alignment with both the new and old algorithms, allowing them both to process for the same amount of time. The percentage of known search errors is the percentage of sentences from our sample in which we were able to find a more probable candidate by applying our new algorithm using 24 hours of computation for just the 1000 sample sentences. Table 5 presents the results, showing that our new algorithm reduced search errors in all cases, but further reduction could be obtained. The second experiment shows the impact of the new search on discriminative reranking of Model 4 (see Table 6). Reduced search errors lead to a better fit of the discriminative training corpus.

4 Semi-Supervised Training for Word Alignments

Intuitively, in approximate EM training for Model 4 (Brown et al., 1993), the E-step corresponds to calculating the probability of all alignments according to the current model estimate, while the M-step is the creation of a new model estimate given a probability distribution over alignments (calculated in the E-step).

In the E-step ideally all possible alignments should be enumerated and labeled with $p(a|e, f)$, but this is intractable. For the M-step, we would like to count over all possible alignments for each sentence pair, weighted by their probability according to the model estimated at the previous

SYSTEM	F TO E ERRORS %	E TO F ERRORS %
A/E OLD	19.4	22.3
A/E NEW	8.5	15.3
F/E OLD	32.5	25.9
F/E NEW	13.7	10.4

Table 5: Comparison of New Search Algorithm with Old Search Algorithm

SYSTEM	F-MEASURE F TO E	F-MEASURE E TO F	F-MEASURE BEST SYMM.
A/E MODEL 4 RERANKING OLD	64.1 / 58.1	54.0 / 48.8	67.9 / 63.0 (UNION)
A/E MODEL 4 RERANKING NEW	65.3 / 59.5	55.7 / 51.4	69.7 / 64.6 (UNION)
F/E MODEL 4 RERANKING OLD	77.3 / 77.8	78.3 / 77.2	79.2 / 79.1 (REFINED)
F/E MODEL 4 RERANKING NEW	77.9 / 77.9	78.4 / 77.7	79.2 / 79.4 (REFINED)

Table 6: Impact of Improved Search on Discriminative Reranking of Model 4

step. Because this is not tractable, we make the assumption that the single assumed Viterbi alignment can be used to update our estimate in the M-step. This approximation is called Viterbi training. Neal and Hinton (1998) analyze approximate EM training and motivate this type of variant.

We extend approximate EM training to perform a new type of training which we call Minimum Error / Maximum Likelihood Training. The intuition behind this approach to semi-supervised training is that we wish to obtain the advantages of both discriminative training (error minimization) and approximate EM (which allows us to estimate a large numbers of parameters even though we have very few gold standard word alignments). We introduce the EMD algorithm, in which discriminative training is used to control the contributions of sub-models (thereby minimizing error), while a procedure similar to one step of approximate EM is used to estimate the large number of sub-model parameters.

A brief sketch of the EMD algorithm applied to our extended model is presented in Figure 1. Parameters have a superscript t representing their value at iteration t . We initialize the algorithm with the gold standard word alignments (labels) of the word alignment discriminative training set, an initial λ , N , and the starting alignments (the iteration 4 HMM Viterbi alignment). In line 2, we make iteration 0 estimates of the 5 sub-models of Model 4 and the 6 heuristic sub-models which are iteration dependent. In line 3, we run discriminative training using the algorithm from Section 3.1. In line 4, we measure the error of the resulting λ vector. In the main loop in line 7 we align the full training set (similar to the E-step of EM), while in line 8 we estimate the iteration-dependent sub-models (similar to the M-step of EM). Then

```

1: Algorithm EMD(labels,  $\lambda'$ ,  $N$ , starting alignments)
2: estimate  $\theta_m^0$  for  $m = 1$  to 11
3:  $\lambda^0 = \text{Discrim}(\theta^0, \lambda', \text{labels}, N)$ 
4:  $e^0 = E(\lambda^0, \text{labels})$ 
5:  $t = 1$ 
6: loop
7:   align full training set using  $\lambda^{t-1}$  and  $\theta_m^{t-1}$ 
8:   estimate  $\theta_m^t$  for  $m = 1$  to 11
9:    $\lambda^t = \text{Discrim}(\theta^t, \lambda'', \text{labels}, N)$ 
10:   $e^t = E(\lambda^t, \text{labels})$ 
11:  if  $e^t \geq e^{t-1}$  then
12:    terminate loop
13:  end if
14:   $t = t + 1$ 
15: end loop
16: return hypothesized alignments of full training set

```

Figure 1: Sketch of the EMD algorithm

we perform discriminative reranking in line 9 and check for convergence in lines 10 and 11 (convergence means that error was not decreased from the previous iteration). The output of the algorithm is new hypothesized alignments of the training corpus.

Table 7 evaluates the EMD semi-supervised training algorithm. We observe:

1. In both cases there is improved F-measure on the second iteration of semi-supervised training, indicating that the EMD algorithm performs better than one step discriminative reranking.
2. The French/English data set has converged³ after the second iteration.
3. The Arabic/English data set converged after improvement for the first, second and third iterations.

We also performed an additional experiment for French/English aimed at understanding the potential contribution of the word aligned data without

³Convergence is achieved because error on the word alignment discriminative training set does not improve.

SYSTEM	F-MEASURE F TO E	F-MEASURE E TO F	BEST SYMM.
A/E STARTING POINT	58.6 / 54.4	47.7 / 39.9	62.1 / 57.0 (UNION)
A/E EMD: ITERATION 1	68.4 / 62.2	61.6 / 57.7	72.0 / 66.4 (UNION)
A/E EMD: ITERATION 2	69.8 / 63.1	64.1 / 59.5	74.1 / 68.1 (UNION)
A/E EMD: ITERATION 3	70.6 / 65.4	64.3 / 59.2	74.7 / 69.4 (UNION)
F/E STARTING POINT	72.4 / 73.9	71.5 / 71.8	76.4 / 77.3 (REFINED)
F/E EMD: ITERATION 1	78.7 / 80.2	79.3 / 79.6	79.6 / 80.4 (REFINED)
F/E EMD: ITERATION 2	79.4 / 80.5	79.8 / 80.5	79.9 / 81.2 (REFINED)

Table 7: Semi-Supervised Training Task F-measure

the new algorithm⁴. Like Ittycheriah and Roukos (2005), we converted the alignment discriminative training corpus links into a special corpus consisting of parallel sentences where each sentence consists only of a single word involved in the link. We found that the information in the links was “washed out” by the rest of the data and resulted in no change in the alignment test set’s F-Measure. Callison-Burch et al. (2004) showed in their work on combining alignments of lower and higher quality that the alignments of higher quality should be given a much higher weight than the lower quality alignments. Using this insight, we found that adding 10,000 copies of the special corpus to our training data resulted in the highest alignment test set gain, which was a small gain of 0.6 F-Measure. This result suggests that while the link information is useful for improving F-Measure, our improved methods for training are producing much larger improvements.

5 Improvement of MT Quality

The symmetrized alignments from the last iteration of EMD were used to build phrasal SMT systems, as were the symmetrized Model 4 alignments (the baseline). Aside from the final alignment, all other resources were held constant between the baseline and contrastive SMT systems, including those based on lower level alignments models such as IBM Model 1. For all of our experiments, we use two language models, one built using the English portion of the training data and the other built using additional English news data. We run Maximum BLEU (Och, 2003) for 25 iterations individually for each system.

Table 8 shows our results. We report BLEU (Papineni et al., 2001) multiplied by 100. We also show the F-measure after heuristic symmetrization of the alignment test sets. The table shows that

⁴We would like to thank an anonymous reviewer for suggesting that this experiment would be useful even when using a small discriminative training corpus.

our algorithm produces heuristically symmetrized final alignments of improved F-measure. Using these alignments in our phrasal SMT system, we produced a statistically significant BLEU improvement (at a 95% confidence interval a gain of 0.78 is necessary) on the French/English task and a statistically significant BLEU improvement on the Arabic/English task (at a 95% confidence interval a gain of 1.2 is necessary).

5.1 Error Criterion

The error criterion we used for all experiments is $1 - \text{F-measure}$. The formula for F-measure is shown in Equation 3. (Fraser and Marcu, 2006) established that tuning the trade-off between Precision and Recall in the F-Measure formula will lead to the best BLEU results. We tuned α by building a collection of alignments using our baseline system, measuring Precision and Recall against the alignment discriminative training set, building SMT systems and measuring resulting BLEU scores, and then searching for an appropriate α setting. We searched $\alpha = 0.1, 0.2, \dots, 0.9$ and set α so that the resulting F-measure tracks BLEU to the best extent possible. The best settings were $\alpha = 0.2$ for Arabic/English and $\alpha = 0.7$ for French/English, and these settings of α were used for every result reported in this paper. See (Fraser and Marcu, 2006) for further details.

$$F(A, S, \alpha) = \frac{1}{\frac{\alpha}{\text{Precision}(A, S)} + \frac{(1-\alpha)}{\text{Recall}(A, S)}} \quad (3)$$

6 Previous Research

Previous work on discriminative training for word-alignment differed most strongly from our approach in that it generally views word-alignment as a supervised task. Examples of this perspective include (Liu et al., 2005; Ittycheriah and Roukos, 2005; Moore, 2005; Taskar et al., 2005). All of these also used knowledge from one of the IBM Models in order to obtain competitive results

SYSTEM	BLEU	F-MEASURE
A/E UNSUP. MODEL 4 UNION	49.16	64.6
A/E EMD 3 UNION	50.84	69.4
F/E UNSUP. MODEL 4 REFINED	30.63	76.4
F/E EMD 2 REFINED	31.56	81.2

Table 8: Evaluation of Translation Quality

with the baseline (with the exception of (Moore, 2005)). We interleave discriminative training with EM and are therefore performing semi-supervised training. We show that semi-supervised training leads to better word alignments than running unsupervised training followed by discriminative training.

Another important difference with previous work is that we are concerned with generating many-to-many word alignments. Cherry and Lin (2003) and Taskar et al. (2005) compared their results with Model 4 using “intersection” by looking at AER (with the “Sure” versus “Possible” link distinction), and restricted themselves to considering 1-to-1 alignments. However, “union” and “refined” alignments, which are many-to-many, are what are used to build competitive phrasal SMT systems, because “intersection” performs poorly, despite having been shown to have the best AER scores for the French/English corpus we are using (Och and Ney, 2003). (Fraser and Marcu, 2006) recently found serious problems with AER both empirically and analytically, which explains why optimizing AER frequently results in poor machine translation performance.

Finally, we show better MT results by using F-measure with a tuned α value. The only previous discriminative approach which has been shown to produce translations of similar or better quality to those produced by the symmetrized baseline was (Ittycheriah and Roukos, 2005). They had access to 5000 gold standard word alignments, considerably more than the 100 or 110 gold standard word alignments used here. They also invested significant effort in sub-model engineering (producing both sub-models specific to Arabic/English alignment and sub-models which would be useful for other language pairs), while we use sub-models which are simple extensions of Model 4 and language independent.

The problem of semi-supervised learning is often defined as “using unlabeled data to help supervised learning” (Seeger, 2000). Most work on semi-supervised learning uses underlying distribu-

tions with a relatively small number of parameters. An initial model is estimated in a supervised fashion using the labeled data, and this supervised model is used to attach labels (or a probability distribution over labels) to the unlabeled data, then a new supervised model is estimated, and this is iterated. If these techniques are applied when there are a small number of labels in relation to the number of parameters used, they will suffer from the “overconfident pseudo-labeling problem” (Seeger, 2000), where the initial labels of poor quality assigned to the unlabeled data will dominate the model estimated in the M-step. However, there are tasks with large numbers of parameters where there are sufficient labels. Nigam et al. (2000) addressed a text classification task. They estimate a Naive Bayes classifier over the labeled data and use it to provide initial MAP estimates for unlabeled documents, followed by EM to further refine the model. Callison-Burch et al. (2004) examined the issue of semi-supervised training for word alignment, but under a scenario where they simulated sufficient gold standard word alignments to follow an approach similar to Nigam et al. (2000). We do not have enough labels for this approach.

We are aware of two approaches to semi-supervised learning which are more similar in spirit to ours. Ivanov et al. (2001) used discriminative training in a reinforcement learning context in a similar way to our adding of a discriminative training step to an unsupervised context. A large body of work uses semi-supervised learning for clustering by imposing constraints on clusters. For instance, in (Basu et al., 2004), the clustering system was supplied with pairs of instances labeled as belonging to the same or different clusters.

7 Conclusion

We presented a semi-supervised algorithm based on IBM Model 4, with modeling and search extensions, which produces alignments of improved F-measure over unsupervised Model 4 training.

We used these alignments to produce translations of higher quality.

The semi-supervised learning literature generally addresses augmenting supervised learning tasks with unlabeled data (Seeger, 2000). In contrast, we augmented an unsupervised learning task with labeled data. We hope that Minimum Error / Maximum Likelihood training using the EMD algorithm can be used for a wide diversity of tasks where there is not enough labeled data to allow supervised estimation of an initial model of reasonable quality.

8 Acknowledgments

This work was partially supported under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022. We would like to thank the USC Center for High Performance Computing and Communications.

References

- Sugato Basu, Mikhail Bilenko, and Raymond J. Mooney. 2004. A probabilistic framework for semi-supervised clustering. In *KDD '04: Proc. of the ACM SIGKDD international conference on knowledge discovery and data mining*, pages 59–68, New York. ACM Press.
- Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.
- Chris Callison-Burch, David Talbot, and Miles Osborne. 2004. Statistical machine translation with word- and sentence-aligned parallel corpora. In *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July.
- Colin Cherry and Dekang Lin. 2003. A probability model to improve word alignment. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan, July.
- Alexander Fraser and Daniel Marcu. 2006. Measuring word alignment quality for statistical machine translation. In *Technical Report ISI-TR-616*. Available at <http://www.isi.edu/fraser/research.html>, ISI/University of Southern California, May.
- Fred Glover. 1986. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13(5):533–549.
- Abraham Ittycheriah and Salim Roukos. 2005. A maximum entropy word aligner for Arabic-English machine translation. In *Proc. of Human Language Technology Conf. and Conf. on Empirical Methods in Natural Language Processing*, Vancouver, BC.
- Yuri A. Ivanov, Bruce Blumberg, and Alex Pentland. 2001. Expectation maximization for weakly labeled data. In *ICML '01: Proc. of the Eighteenth International Conf. on Machine Learning*, pages 218–225.
- Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 459–466, Ann Arbor, MI.
- I. Dan Melamed. 1998. Manual annotation of translational equivalence: The blinker project. Technical Report 98-07, Institute for Research in Cognitive Science, Philadelphia, PA.
- Robert C. Moore. 2005. A discriminative framework for bilingual word alignment. In *Proc. of Human Language Technology Conf. and Conf. on Empirical Methods in Natural Language Processing*, Vancouver, BC, October.
- Radford M. Neal and Geoffrey E. Hinton. 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*. Kluwer.
- Kamal Nigam, Andrew K. McCallum, Sebastian Thrun, and Tom M. Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.
- Kishore A. Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, September.
- Matthias Seeger. 2000. Learning with labeled and unlabeled data. In *Technical report, 2000*. Available at <http://www.dai.ed.ac.uk/seeger/papers.html>.
- Ben Taskar, Simon Lacoste-Julien, and Dan Klein. 2005. A discriminative matching approach to word alignment. In *Proc. of Human Language Technology Conf. and Conf. on Empirical Methods in Natural Language Processing*, Vancouver, BC, October.
- Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING '96: The 16th Int. Conf. on Computational Linguistics*, pages 836–841, Copenhagen, Denmark, August.

Left-to-Right Target Generation for Hierarchical Phrase-based Translation

Taro Watanabe Hajime Tsukada Hideki Isozaki

2-4, Hikaridai, Seika-cho, Soraku-gun,

Kyoto, JAPAN 619-0237

{taro, tsukada, isozaki}@cslab.kecl.ntt.co.jp

Abstract

We present a hierarchical phrase-based statistical machine translation in which a target sentence is efficiently generated in left-to-right order. The model is a class of synchronous-CFG with a Greibach Normal Form-like structure for the projected production rule: The paired target-side of a production rule takes a phrase prefixed form. The decoder for the target-normalized form is based on an Early-style top down parser on the source side. The target-normalized form coupled with our top down parser implies a left-to-right generation of translations which enables us a straightforward integration with ngram language models. Our model was experimented on a Japanese-to-English newswire translation task, and showed statistically significant performance improvements against a phrase-based translation system.

1 Introduction

In a classical statistical machine translation, a foreign language sentence $f_1^J = f_1, f_2, \dots, f_J$ is translated into another language, i.e. English, $e_1^I = e_1, e_2, \dots, e_I$ by seeking a maximum likely solution of:

$$\hat{e}_1^I = \operatorname{argmax}_{e_1^I} \Pr(e_1^I | f_1^J) \quad (1)$$

$$= \operatorname{argmax}_{e_1^I} \Pr(f_1^J | e_1^I) \Pr(e_1^I) \quad (2)$$

The source channel approach in Equation 2 independently decomposes translation knowledge into

a translation model and a language model, respectively (Brown et al., 1993). The former represents the correspondence between two languages and the latter contributes to the fluency of English.

In the state of the art statistical machine translation, the posterior probability $\Pr(e_1^I | f_1^J)$ is directly maximized using a log-linear combination of feature functions (Och and Ney, 2002):

$$\hat{e}_1^I = \operatorname{argmax}_{e_1^I} \frac{\exp\left(\sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J)\right)}{\sum_{e_1^{I'}} \exp\left(\sum_{m=1}^M \lambda_m h_m(e_1^{I'}, f_1^J)\right)} \quad (3)$$

where $h_m(e_1^I, f_1^J)$ is a feature function, such as a ngram language model or a translation model. When decoding, the denominator is dropped since it depends only on f_1^J . Feature function scaling factors λ_m are optimized based on a maximum likely approach (Och and Ney, 2002) or on a direct error minimization approach (Och, 2003). This modeling allows the integration of various feature functions depending on the scenario of how a translation is constituted.

A phrase-based translation model is one of the modern approaches which exploits a phrase, a contiguous sequence of words, as a unit of translation (Koehn et al., 2003; Zens and Ney, 2003; Tillman, 2004). The idea is based on a word-based source channel modeling of Brown et al. (1993): It assumes that e_1^I is segmented into a sequence of K phrases $\bar{e}_1^K$. Each phrase $\bar{e}_k$ is transformed into $\bar{f}_k$. The translated phrases are reordered to form f_1^J . One of the benefits of the modeling is that the phrase translation unit preserves localized word reordering. However, it cannot hypothesize a long-distance reordering required for linguistically divergent language pairs. For instance, when translating Japanese to English, a Japanese SOV structure has to be reordered to match with an En-

glish SVO structure. Such a sentence-wise movement cannot be realized within the phrase-based modeling.

Chiang (2005) introduced a hierarchical phrase-based translation model that combined the strength of the phrase-based approach and a synchronous-CFG formalism (Aho and Ullman, 1969): A rewrite system initiated from a start symbol which synchronously rewrites paired non-terminals. Their translation model is a binarized synchronous-CFG, or a rank-2 of synchronous-CFG, in which the right-hand side of a production rule contains at most two non-terminals. The form can be regarded as a phrase translation pair with at most two holes instantiated with other phrases. The hierarchically combined phrases provide a sort of reordering constraints that is not directly modeled by a phrase-based model.

Rules are induced from a bilingual corpus without linguistic clues first by extracting phrase translation pairs, and then by generalizing extracted phrases with holes (Chiang, 2005). Even in a phrase-based model, the number of phrases extracted from a bilingual corpus is quadratic to the length of bilingual sentences. The grammar size for the hierarchical phrase-based model will be further exploded, since there exists numerous combination of inserting holes to each rule. The spuriously increasing grammar size will be problematic for decoding without certain heuristics, such as a length based thresholding.

The integration with a ngram language model further increases the cost of decoding especially when incorporating a higher order ngram, such as 5-gram. In the hierarchical phrase-based model (Chiang, 2005), and an inversion transduction grammar (ITG) (Wu, 1997), the problem is resolved by restricting to a binarized form where at most two non-terminals are allowed in the right-hand side. However, Huang et al. (2005) reported that the computational complexity for decoding amounted to $O(J^{3+3(n-1)})$ with n -gram even using a hook technique. The complexity lies in memorizing the ngram's context for each constituent. The order of ngram would be a dominant factor for higher order ngrams.

As an alternative to a binarized form, we present a target-normalized hierarchical phrase-based translation model. The model is a class of a hierarchical phrase-based model, but constrained so that the English part of the right-hand side

is restricted to a Greibach Normal Form (GNF)-like structure: A contiguous sequence of terminals, or a phrase, is followed by a string of non-terminals. The target-normalized form reduces the number of rules extracted from a bilingual corpus, but still preserves the strength of the phrase-based approach. An integration with ngram language model is straightforward, since the model generates a translation in left-to-right order. Our decoder is based on an Earley-style top down parsing on the foreign language side. The projected English-side is generated in left-to-right order synchronized with the derivation of the foreign language side. The decoder's implementation is taken after a decoder for an existing phrase-based model with a simple modification to account for production rules. Experimental results on a Japanese-to-English newswire translation task showed significant improvement against a phrase-based modeling.

2 Translation Model

A weighted synchronous-CFG is a rewrite system consisting of production rules whose right-hand side is paired (Aho and Ullman, 1969):

$$X \leftarrow \langle \gamma, \alpha, \sim \rangle \quad (4)$$

where X is a non-terminal, γ and α are strings of terminals and non-terminals. For notational simplicity, we assume that γ and α correspond to the foreign language side and the English side, respectively. $\sim$ is a one-to-one correspondence for the non-terminals appeared in γ and α . Starting from an initial non-terminal, each rule rewrites non-terminals in γ and α that are associated with $\sim$.

Chiang (2005) proposed a hierarchical phrase-based translation model, a binary synchronous-CFG, which restricted the form of production rules as follows:

- Only two types of non-terminals allowed: S and X .
- Both of the strings γ and α must contain at least one terminal item.
- Rules may have at most two non-terminals but non-terminals cannot be adjacent for the foreign language side γ .

The production rules are induced from a bilingual corpus with the help of word alignments. To alleviate a data sparseness problem, glue rules are

added that prefer combining hierarchical phrases in a serial manner:

$$S \rightarrow \langle S_{\boxed{1}} X_{\boxed{2}}, S_{\boxed{1}} X_{\boxed{2}} \rangle \quad (5)$$

$$S \rightarrow \langle X_{\boxed{1}}, X_{\boxed{1}} \rangle \quad (6)$$

where boxed indices indicate non-terminal's linkages represented in $\sim$.

Our model is based on Chiang (2005)'s framework, but further restricts the form of production rules so that the aligned right-hand side α follows a GNF-like structure:

$$X \leftarrow \langle \gamma, \bar{b}\beta, \sim \rangle \quad (7)$$

where $\bar{b}$ is a string of terminals, or a phrase, and β is a (possibly empty) string of non-terminals. The foreign language at right-hand side γ still takes an arbitrary string of terminals and non-terminals. The use of a phrase $\bar{b}$ as a prefix keeps the strength of the phrase-base framework. A contiguous English side coupled with a (possibly) discontinuous foreign language side preserves a phrase-bounded local word reordering. At the same time, the target-normalized framework still combines phrases hierarchically in a restricted manner.

The target-normalized form can be regarded as a type of rule in which certain non-terminals are always instantiated with phrase translation pairs. Thus, we will be able to reduce the number of rules induced from a bilingual corpus, which, in turn, help reducing the decoding complexity.

The contiguous phrase-prefixed form generates English in left-to-right order. Therefore, a decoder can easily hypothesize a derivation tree integrated with a ngram language model even with higher order.

Note that we do not imply arbitrary synchronous-CFGs are transformed into the target normalized form. The form simply restricts the grammar extracted from a bilingual corpus explained in the next section.

2.1 Rule Extraction

We present an algorithm to extract production rules from a bilingual corpus. The procedure is based on those for the hierarchical phrase-based translation model (Chiang, 2005).

First, a bilingual corpus is annotated with word alignments using the method of Koehn et al. (2003). Many-to-many word alignments are induced by running a one-to-many word alignment

model, such as GIZA++ (Och and Ney, 2003), in both directions and by combining the results based on a heuristic (Koehn et al., 2003).

Second, phrase translation pairs are extracted from the word alignment corpus (Koehn et al., 2003). The method exhaustively extracts phrase pairs (f_j^{j+m}, e_i^{i+n}) from a sentence pair (f_1^J, e_1^I) that do not violate the word alignment constraints a :

$$\exists(i', j') \in a : j' \in [j, j+m], i' \in [i, i+n]$$

$$\nexists(i', j') \in a : j' \in [j, j+m], i' \notin [i, i+n]$$

$$\nexists(i', j') \in a : j' \notin [j, j+m], i' \in [i, i+n]$$

Third, based on the extracted phrases, production rules are accumulated by computing the "holes" for contiguous phrases (Chiang, 2005):

1. A phrase pair $(\bar{f}, \bar{e})$ constitutes a rule

$$X \rightarrow \langle \bar{f}, \bar{e} \rangle$$

2. A rule $X \rightarrow \langle \gamma, \alpha \rangle$ and a phrase pair $(\bar{f}, \bar{e})$ s.t. $\gamma = \gamma' f \gamma''$ and $\alpha = \bar{e}' \bar{e} \beta$ constitutes a rule

$$X \rightarrow \langle \gamma' X_{\boxed{k}} \gamma'', \bar{e}' X_{\boxed{k}} \beta \rangle$$

Following Chiang (2005), we applied constraints when inducing rules with non-terminals:

- At least one foreign word must be aligned to an English word.
- Adjacent non-terminals are not allowed for the foreign language side.

2.2 Phrase-based Rules

The rule extraction procedure described in Section 2.1 is a corpus-based, therefore will be easily suffered from a data sparseness problem. The hierarchical phrase-based model avoided this problem by introducing the glue rules 5 and 6 that combined hierarchical phrases sequentially (Chiang, 2005).

We use a different method of generalizing production rules. When production rules without non-terminals are extracted in step 1 of Section 2.1,

$$X \rightarrow \langle \bar{f}, \bar{e} \rangle \quad (8)$$

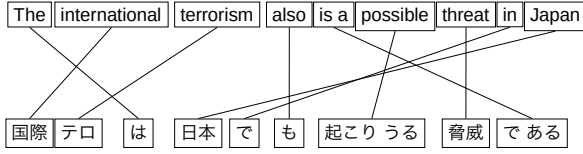
then, we also add production rules as follows:

$$X \rightarrow \langle \bar{f} X_{\boxed{1}}, \bar{e} X_{\boxed{1}} \rangle \quad (9)$$

$$X \rightarrow \langle X_{\boxed{1}} \bar{f}, X_{\boxed{1}} \bar{e} \rangle \quad (10)$$

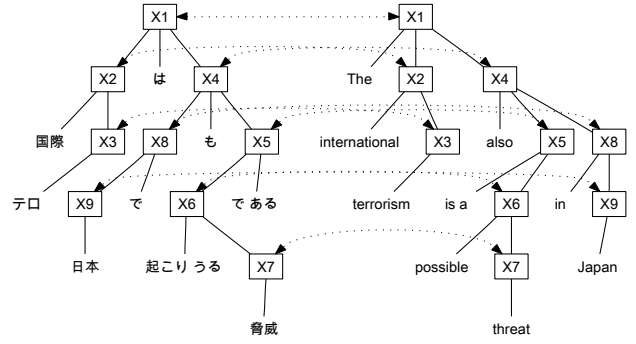
$$X \rightarrow \langle X_{\boxed{1}} \bar{f} X_{\boxed{2}}, X_{\boxed{1}} \bar{e} X_{\boxed{2}} \rangle \quad (11)$$

$$X \rightarrow \langle X_{\boxed{2}} \bar{f} X_{\boxed{1}}, X_{\boxed{2}} \bar{e} X_{\boxed{1}} \rangle \quad (12)$$



Reference translation: “International terrorism is a threat even to Japan”

(a) Translation by a phrase-based model.



(b) A derivation tree representation for Figure 1(a). Indices in non-terminal X represent the order to perform rewriting.

Figure 1: An example of Japanese-to-English translation by a phrase-based model.

We call them phrase-based rules, since four types of rules are generalized directly from phrase translation pairs.

The class of rules roughly corresponds to the reordering constraints used in a phrase-based model during decoding. Rules 8 and 9 are sufficient to realize a monotone decoding in which phrase translation pairs are simply combined sequentially. With rules 10 and 11, the non-terminal $X_{\square}$ behaves as a place holder where certain number of foreign words are skipped. Therefore, those rules realize a window size constraint used in many phrase-based models (Koehn et al., 2003). The rule 12 further gives an extra freedom for the phrase pair reordering. The rules 8 through 12 can be interpreted as ITG-constraints where phrase translation pairs are hierarchically combined either in a monotonic way or in an inverted manner (Zens and Ney, 2003; Wu, 1997). Thus, by controlling what types of phrase-based rules employed in a grammar, we will be able to simulate a phrase-based translation model with various constraints. This reduction is rather natural in that a finite state transducer, or a phrase-based model, is a subclass of a synchronous-CFG.

Figure 1(a) shows an example Japanese-to-English translation by a phrase-based model described in Section 5. Using the phrase-based rules, the translation results is represented as a derivation tree in Figure 1(b).

3 Decoding

Our decoder is an Earley-style top down parser on the foreign language side with a beam search strategy. Given an input sentence f_1^J , the decoder seeks for the best English according to Equation 3 using the feature functions described in Section 4. The English output sentence is generated in left-

to-right order in accordance with the derivation of the foreign language side synchronized with the cardinality of already translated foreign word positions.

The decoding process is very similar to those described in (Koehn et al., 2003): It starts from an initial empty hypothesis. From an existing hypothesis, new hypothesis is generated by consuming a production rule that covers untranslated foreign word positions. The score for the newly generated hypothesis is updated by combining the scores of feature functions described in Section 4. The English side of the rule is simply concatenated to form a new prefix of English sentence. Hypotheses that consumed m foreign words are stored in a priority queue Q_m .

Hypotheses in Q_m undergo two types of pruning: A histogram pruning preserves at most M hypotheses in Q_m . A threshold pruning discards a hypotheses whose score is below the maximum score of Q_m multiplied with a threshold value τ . Rules are constrained by their foreign word span of a non-terminal. For a rule consisting of more than two non-terminals, we constrained so that at least one non-terminal should span at most κ words.

The decoder is characterized as a weighted synchronous-CFG implemented with a push-down automaton rather a weighted finite state transducer (Aho and Ullman, 1969). Each hypothesis maintains following knowledge:

- A prefix of English sentence. For space efficiency, the prefix is represented as a word graph.
- Partial contexts for each feature function. For instance, to compute a 5-gram language model feature, we keep the consecutive last four words of an English prefix.

- A stack that keeps track of the uncovered foreign word spans. The stack for an initial hypothesis is initialized with span $[1, J]$.

When extending a hypothesis, the associated stack structure is popped. The popped foreign word span $[j^l, j^r]$ is used to locate the rules for uncovered foreign word positions. We assume that the decoder accumulates all the applicable rules from a large database and stores the extracted rules in a chart structure. The decoder identifies what rules to consume when extending a hypothesis using the chart structure. A new hypothesis is created with an updated stack by pushing foreign non-terminal spans: For each rule spanning $[j^l, j^r]$ at foreign-side with non-terminal spans of $[k_1^l, k_1^r], [k_2^l, k_2^r], \dots$, the non-terminal spans are pushed in the reverse order of the projected English side. For example, A rule with foreign word non-terminal spans:

$$X \rightarrow \langle X_{\square} : [k_2^l, k_2^r] \bar{f} X_{\square} : [k_1^l, k_1^r], \bar{e} X_{\square} X_{\square} \rangle$$

will update a stack by pushing the foreign word spans $[k_2^l, k_2^r]$ and $[k_1^l, k_1^r]$ in order. This ordering assures that, when popped, the English-side will be generated in left-to-right order. A hypothesis with an empty stack implies that the hypothesis has covered all the foreign words.

Figure 2 illustrates the decoding process for the derivation tree in Figure 1(b). Starting from the initial hypothesis of $[1, 11]$, the stack is updated in accordance with non-terminal's spans. The span is popped and the rule with the foreign word span $[1, 11]$ is looked up from the chart structure. The stack structure for the newly created hypothesis is updated by pushing non-terminal spans $[4, 11]$ and $[1, 2]$.

Our decoder is based on an in-house developed phrase-based decoder which uses a bit vector to represent uncovered foreign word positions for each hypothesis. We basically replaced the bit vector structure to the stack structure: Almost no modification was required for the word graph structure and the beam search strategy implemented for a phrase-based modeling. The use of a stack structure directly models a synchronous-CFG formalism realized as a push-down automaton, while the bit vector implementation is conceptualized as a finite state transducer. The cost of decoding with the proposed model is cubic to foreign language sentence length.

Rules	Stack
	$[1, 11]$
$X : [1, 11] \rightarrow \langle X_{\square} : [1, 2] \text{ は } X_{\square} : [4, 11], \text{ The } X_{\square} X_{\square} \rangle$	$[1, 2]$ $[4, 11]$
$X : [1, 2] \rightarrow \langle \text{国際 } X_{\square} : [2, 2], \text{ international } X_{\square} \rangle$	$[2, 2]$ $[4, 11]$
$X : [2, 2] \rightarrow \langle \text{テロ, terrorism} \rangle$	$[4, 11]$
$X : [4, 11] \rightarrow \langle X_{\square} : [4, 5] \text{ も } X_{\square} : [7, 11], \text{ also } X_{\square} X_{\square} \rangle$	$[7, 11]$ $[4, 5]$
$X : [7, 11] \rightarrow \langle X_{\square} : [7, 9] \text{ である, is a } X_{\square} \rangle$	$[7, 9]$ $[4, 5]$
$X : [7, 9] \rightarrow \langle \text{起こりうる } X_{\square} : [9, 9], \text{ possible } X_{\square} \rangle$	$[9, 9]$ $[4, 5]$
$X : [9, 9] \rightarrow \langle \text{脅威, threat} \rangle$	$[4, 5]$
$X : [4, 5] \rightarrow \langle X_{\square} : [4, 4] \text{ で, in } X_{\square} \rangle$	$[4, 4]$
$X : [4, 4] \rightarrow \langle \text{日本, Japan} \rangle$	

Figure 2: An example decoding process of Figure 1(b) with a stack to keep track of foreign word spans.

4 Feature Functions

The decoder for our translation model uses a log-linear combination of feature functions, or sub-models, to seek for the maximum likely translation according to Equation 3. This section describes the models experimented in Section 5, mainly consisting of count-based models, lexicon-based models, a language model, reordering models and length-based models.

4.1 Count-based Models

Main feature functions $h_{\phi}(f_1^J | e_1^I, \mathcal{D})$ and $h_{\phi}(e_1^I | f_1^J, \mathcal{D})$ estimate the likelihood of two sentences f_1^J and e_1^I over a derivation tree $\mathcal{D}$. We assume that the production rules in $\mathcal{D}$ are independent of each other:

$$h_{\phi}(f_1^J | e_1^I, \mathcal{D}) = \log \prod_{\langle \gamma, \alpha \rangle \in \mathcal{D}} \phi(\gamma | \alpha) \quad (13)$$

$\phi(\gamma | \alpha)$ is estimated through the relative frequency on a given bilingual corpus.

$$\phi(\gamma | \alpha) = \frac{\text{count}(\gamma, \alpha)}{\sum_{\gamma} \text{count}(\gamma, \alpha)} \quad (14)$$

where $\text{count}(\cdot)$ represents the cooccurrence frequency of rules γ and α .

The relative count-based probabilities for the phrase-based rules are simply adopted from the original probabilities of phrase translation pairs.

4.2 Lexicon-based Models

We define lexically weighted feature functions $h_w(f_1^J | e_1^I, \mathcal{D})$ and $h_w(e_1^I | f_1^J, \mathcal{D})$ applying the independence assumption of production rules as in

Equation 13.

$$h_w(f_1^J | e_1^I, \mathcal{D}) = \log \prod_{\langle \gamma, \alpha \rangle \in \mathcal{D}} p_w(\gamma | \alpha) \quad (15)$$

The lexical weight $p_w(\gamma | \alpha)$ is computed from word alignments a inside γ and α (Koehn et al., 2003):

$$p_w(\gamma | \alpha, a) = \prod_{i=1}^{|\alpha|} \frac{1}{|\{j | (i, j) \in a\}|} \sum_{\forall (i, j) \in a} t(\gamma_j | \alpha_i) \quad (16)$$

where $t(\cdot)$ is a lexicon model trained from the word alignment annotated bilingual corpus discussed in Section 2.1. The alignment a also includes non-terminal correspondence with $t(X_{\overline{a}} | X_{\overline{b}}) = 1$. If we observed multiple alignment instances for γ and α , then, we take the maximum of the weights.

$$p_w(\gamma | \alpha) = \max_a p_w(\gamma | \alpha, a) \quad (17)$$

4.3 Language Model

We used mixed-cased n-gram language model. In case of 5-gram language model, the feature function is expressed as follows:

$$h_{lm}(e_1^I) = \log \prod_i p_n(e_i | e_{i-4} e_{i-3} e_{i-2} e_{i-1}) \quad (18)$$

4.4 Reordering Models

In order to limit the reorderings, two feature functions are employed based on the backtracking of rules during the top-down parsing on foreign language side.

$$h_h(e_1^I, f_1^J, \mathcal{D}) = \sum_{\mathcal{D}_i \in \text{back}(\mathcal{D})} \text{height}(\mathcal{D}_i) \quad (19)$$

$$h_w(e_1^I, f_1^J, \mathcal{D}) = \sum_{\mathcal{D}_i \in \text{back}(\mathcal{D})} \text{width}(\mathcal{D}_i) \quad (20)$$

where $\text{back}(\mathcal{D})$ is a set of subtrees backtracked during the derivation of $\mathcal{D}$, and $\text{height}(\mathcal{D}_i)$ and $\text{width}(\mathcal{D}_i)$ refer the height and width of subtree $\mathcal{D}_i$, respectively. In Figure 1(b), for instance, a rule of $X_{\overline{1}}$ with non-terminals $X_{\overline{2}}$ and $X_{\overline{4}}$, two rules $X_{\overline{2}}$ and $X_{\overline{3}}$ spanning two terminal symbols should be backtracked to proceed to $X_{\overline{4}}$. The rationale is that positive scaling factors prefer a deeper structure whereby negative scaling factors prefer a mono-tonized structure.

4.5 Length-based Models

Three trivial length-based feature functions were used in our experiment.

$$h_l(e_1^I) = I \quad (21)$$

$$h_r(\mathcal{D}) = \text{rule}(\mathcal{D}) \quad (22)$$

$$h_p(\mathcal{D}) = \text{phrase}(\mathcal{D}) \quad (23)$$

Table 1: Japanese/English news corpus

		Japanese	English
train	sentence	175,384	
	dictionary	+ 1,329,519	
	words	8,373,478	7,222,726
	vocabulary	297,646	397,592
dev.	sentence	1,500	
	words	47,081	39,117
	OOV	45	149
test	sentence	1,500	
	words	47,033	38,707
	OOV	51	127

Table 2: Phrases/rules extracted from the Japanese/English bilingual corpus. Figures do not include phrase-based rules.

	# rules/phrases
Phrase	5,433,091
Normalized-2	6,225,630
Normalized-3	6,233,294
Hierarchical	12,824,387

where $\text{rule}(\mathcal{D})$ and $\text{phrase}(\mathcal{D})$ are the number of production rules extracted in Section 2.1 and phrase-based rules generalized in Section 2.2, respectively. The English length feature function controls the length of output sentence. Two feature functions based on rule’s counts are hypothesized to control whether to incorporate a production rule or a phrase-based rule into $\mathcal{D}$.

5 Experiments

The bilingual corpus used for our experiments was obtained from an automatically sentence aligned Japanese/English Yomiuri newspaper corpus consisting of 180K sentence pairs (refer to Table 1) (Utiyama and Isahara, 2003). From one-to-one aligned sentences, 1,500 sentence pairs were sampled for a development set and a test set¹. Since the bilingual corpus is rather small, especially for the newspaper translation domain, Japanese/English dictionaries consisting of 1.3M entries were added into a training set to alleviate an OOV problem².

Word alignments were annotated by a HMM translation model (Och and Ney, 2003). After

¹Japanese sentences were segmented by MeCab available from <http://mecab.sourceforge.jp>.

²The dictionary entries were compiled from JEDICT/JNAMEDICT and an in-house developed dictionary.

the annotation via Viterbi alignments with refinements, phrases translation pairs and production rules were extracted (refer to Table 2). We performed the rule extraction using the hierarchical phrase-based constraint (Hierarchical) and our proposed target-normalized form with 2 and 3 non-terminals (Normalized-2 and Normalized-3). Phrase translation pairs were also extracted for comparison (Phrase). We did not threshold the extracted phrases or rules by their length. Table 2 shows that Normalized-2 extracted slightly larger number of rules than those for phrase-based model. Including three non-terminals did not increase the grammar size. The hierarchical phrase-based translation model extracts twice as large as our target-normalized formalism. The target-normalized form is restrictive in that non-terminals should be consecutive for the English-side. This property prohibits spuriously extracted production rules.

Mixed-casing 3-gram/5-gram language models were estimated from LDC English GigaWord 2 together with the 100K English articles of Yomiuri newspaper that were used neither for development nor test sets ³.

We run the decoder for the target-normalized hierarchical phrase-based model consisting of at most two non-terminals, since adding rules with three non-terminals did not increase the grammar size. ITG-constraint simulated phrase-based rules were also included into our grammar. The foreign word span size was thresholded so that at least one non-terminal should span at most 7 words.

Our phrase-based model employed all feature functions for the hierarchical phrase-based system with additional feature functions:

- A distortion model that penalizes the reordering of phrases by the number of words skipped $|j - (j' + m') - 1|$, where j is the foreign word position for a phrase f_j^{j+m} translated immediately after a phrase for $f_{j'}^{j'+m'}$ (Koehn et al., 2003).
- Lexicalized reordering models constrain the reordering of phrases whether to favor monotone, swap or discontinuous positions (Tillman, 2004).

The phrase-based decoder’s reordering was constrained by ITG-constraints with a window size of

³We used SRI ngram language modeling toolkit with limited vocabulary size.

Table 3: Results for the Japanese-to-English newswire translation task.

		BLEU [%]	NIST
Phrase	3-gram	7.14	3.21
	5-gram	7.33	3.19
Normalized-2	3-gram	10.00	4.11
	5-gram	10.26	4.20

7.

The translation results are summarized in Table 3. Two systems were contrasted by 3-gram and 5-gram language models. Results were evaluated by ngram precision based metrics, BLEU and NIST, on the casing preserved single reference test set. Feature function scaling factors for each system were optimized on BLEU score under the development set using a downhill simplex method. The differences of translation qualities are statistically significant at the 95% confidence level (Koehn, 2004). Although the figures presented in Table 3 are rather low, we found that Normalized-2 resulted in statistically significant improvement over Phrase. Figure 3 shows some translation results from the test set.

6 Conclusion

The target-normalized hierarchical phrase-based model is based on a more general hierarchical phrase-based model (Chiang, 2005). The hierarchically combined phrases can be regarded as an instance of phrase-based model with a place holder to constraint reordering. Such reordering was realized either by an additional constraint for decoding, such as window constraints, IBM constraints or ITG-constraints (Zens and Ney, 2003), or by lexicalized reordering feature functions (Tillman, 2004). In the hierarchical phrase-based model, such reordering is explicitly represented in each rule.

As experimented in Section 5, the use of the target-normalized form reduced the grammar size, but still outperformed a phrase-based system. Furthermore, the target-normalized form coupled with our top down parsing on the foreign language side allows an easier integration with ngram language model. A decoder can be implemented based on a phrase-based model by employing a stack structure to keep track of untranslated foreign word spans.

The target-normalized form can be interpreted

Reference:	Japan needs to learn a lesson from history to ensure that it not repeat its mistakes .
Phrase:	At the same time , it never mistakes that it is necessary to learn lessons from the history of criminal .
Normalized-2:	It is necessary to learn lessons from history so as not to repeat similar mistakes in the future .
Reference:	The ministries will dispatch design and construction experts to China to train local engineers and to research technology that is appropriate to China's economic situation .
Phrase:	Japan sent specialists to train local technicians to the project , in addition to the situation in China and its design methods by exception of study .
Normalized-2:	Japan will send experts to study the situation in China , and train Chinese engineers , construction design and construction methods of the recipient from .
Reference:	The Health and Welfare Ministry has decided to invoke the Disaster Relief Law in extending relief measures to the village and the city of Niigata .
Phrase:	The Health and Welfare Ministry in that the Japanese people in the village are made law .
Normalized-2:	The Health and Welfare Ministry decided to apply the Disaster Relief Law to the village in Niigata .

Figure 3: Sample translations from two systems: Phrase and Normalized-2

as a set of rules that reorders the foreign language to match with English language sequentially. Collins et al. (2005) presented a method with hand-coded rules. Our method directly learns such serialization rules from a bilingual corpus without linguistic clues.

The translation quality presented in Section 5 are rather low due to the limited size of the bilingual corpus, and also because of the linguistic difference of two languages. As our future work, we are in the process of experimenting our model for other languages with rich resources, such as Chinese and Arabic, as well as similar language pairs, such as French and English. Additional feature functions will be also investigated that were proved successful for phrase-based models together with feature functions useful for a tree-based modeling.

Acknowledgement

We would like to thank to our colleagues, especially to Hideto Kazawa and Jun Suzuki, for useful discussions on the hierarchical phrase-based translation.

References

- Alfred V. Aho and Jeffrey D. Ullman. 1969. Syntax directed translations and the pushdown assembler. *J. Comput. Syst. Sci.*, 3(1):37–56.
- 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.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL 2005*, pages 263–270, Ann Arbor, Michigan, June.
- Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proc. of ACL 2005*, pages 531–540, Ann Arbor, Michigan, June.
- Liang Huang, Hao Zhang, and Daniel Gildea. 2005. Machine translation as lexicalized parsing with hooks. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 65–73, Vancouver, British Columbia, October.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of NAACL 2003*, pages 48–54, Edmonton, Canada.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP 2004*, pages 388–395, Barcelona, Spain, July.
- Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of ACL 2002*, pages 295–302.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL 2003*, pages 160–167.
- Christoph Tillman. 2004. A unigram orientation model for statistical machine translation. In *HLT-NAACL 2004: Short Papers*, pages 101–104, Boston, Massachusetts, USA, May 2 - May 7.
- Masao Utiyama and Hitoshi Isahara. 2003. Reliable measures for aligning Japanese-English news articles and sentences. In *Proc. of ACL 2003*, pages 72–79.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Comput. Linguist.*, 23(3):377–403.
- Richard Zens and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proc. of ACL 2003*, pages 144–151.

You Can't Beat Frequency (Unless You Use Linguistic Knowledge) – A Qualitative Evaluation of Association Measures for Collocation and Term Extraction

Joachim Wermter Udo Hahn

Jena University Language & Information Engineering (JULIE) Lab
D-07743 Jena, Germany
{wermter|hahn}@coling-uni-jena.de

Abstract

In the past years, a number of lexical association measures have been studied to help extract new scientific terminology or general-language collocations. The implicit assumption of this research was that newly designed term measures involving more sophisticated statistical criteria would outperform simple counts of co-occurrence frequencies. We here explicitly test this assumption. By way of four qualitative criteria, we show that purely statistics-based measures reveal virtually no difference compared with frequency of occurrence counts, while linguistically more informed metrics do reveal such a marked difference.

1 Introduction

Research on domain-specific automatic term recognition (ATR) and on general-language collocation extraction (CE) has gone mostly separate ways in the last decade although their underlying procedures and goals turn out to be rather similar. In both cases, linguistic filters (POS taggers, phrase chunkers, (shallow) parsers) initially collect candidates from large text corpora and then frequency- or statistics-based evidence or association measures yield scores indicating to what degree a candidate qualifies as a term or a collocation. While term mining and collocation mining, as a whole, involve almost the same analytical processing steps, such as orthographic and morphological normalization, normalization of term or collocation variation etc., it is exactly the measure which grades termhood or collocativity of a candidate on which alternative approaches diverge.

Still, the output of such mining algorithms look similar. It is typically constituted by a ranked list on which, ideally, the true terms or collocations are placed in the top portion of the list, while the non-terms / non-collocations occur in its bottom portion.

While there have been lots of approaches to come up with a fully adequate ATR/CE metric (cf. Section 2), we have made observations in our experiments that seem to indicate that simplicity rules, i.e., frequency of occurrence is the dominating factor for the ranking in the result lists even when much smarter statistical machinery is employed. In this paper, we will discuss data which reveals that purely statistics-based measures exhibit virtually no difference compared with frequency of occurrence counts, while linguistically more informed measures do reveal such a marked difference – for the problem of term and collocation mining at least.

2 Related Work

Although there has been a fair amount of work employing linguistically sophisticated analysis of candidate items (e.g., on CE by Lin (1998) and Lin (1999) as well as on ATR by Daille (1996), Jacquemin (1999), and Jacquemin (2001)), these approaches are limited by the difficulty to port grammatical specifications to other domains (in the case of ATR) or by the error-proneness of full general-language parsers (in the case of CE). Therefore, most recent approaches in both areas have backed off to more shallow linguistic filtering techniques, such as POS tagging and phrase chunking (e.g., Frantzi et al. (2000), Krenn and Evert (2001), Nenadić et al. (2004), Wermter and Hahn (2005)).

After linguistic filtering, various measures are employed in the literature for grading the termhood / collocativity of collected candidates. Among the most widespread ones, both for ATR and CE, are statistical and information-theoretic measures, such as t-test, log-likelihood, entropy, and mutual information. Their prominence is also reflected by the fact that a whole chapter of a widely used textbook on statistical NLP (*viz.* Chapter 5 (Collocations) in Manning and Schütze (1999)) is devoted to them. In addition, the C-value (Frantzi et al., 2000) – basically a frequency-based approach – has been another widely used measure for multi-word ATR. Recently, more linguistically informed algorithms have been introduced both for CE (Wermter and Hahn, 2004) and for ATR (Wermter and Hahn, 2005), which have been shown to outperform several of the statistics-only metrics.

3 Methods and Experiments

3.1 Qualitative Criteria

Because various metrics assign a score to the candidates indicating as to what degree they qualify as a collocation or term (or not), these candidates should ideally be ranked in such a way that the following two conditions are met:

- true collocations or terms (i.e., the true positives) are ranked in the upper portion of the output list.
- non-collocations or non-terms (i.e., the true negatives) are ranked in the lower part of the output list.¹

While a trivial solution to the problem might be to simply count the number of occurrences of candidates in the data, employing more sophisticated statistics-based / information-theoretic or even linguistically-motivated algorithms for grading term and collocation candidates is guided by the assumption that this additional level of sophistication yields more adequate rankings relative to these two conditions.

Several studies (e.g., Evert and Krenn (2001), Krenn and Evert (2001), Frantzi et al. (2000), Wermter and Hahn (2004)), however, have already observed that ranking the candidates merely by their frequency of occurrence fares quite well

¹Obviously, this goal is similar to ranking documents according to their relevance for information retrieval.

compared with various more sophisticated association measures (AMs such as t-test, log-likelihood, etc.). In particular, the precision/recall value comparison between the various AMs exhibits a rather inconclusive picture in Evert and Krenn (2001) and Krenn and Evert (2001) as to whether sophisticated statistical AMs are actually more viable than frequency counting.

Commonly used statistical significance testing (e.g., the McNemar or the Wilcoxon sign rank tests; see (Sachs, 1984)) does not seem to provide an appropriate evaluation ground either. Although Evert and Krenn (2001) and Wermter and Hahn (2004) provide significance testing of some AMs with respect to mere frequency counting for collocation extraction, they do not differentiate whether this is due to differences in the ranking of true positives or true negatives or a combination thereof.² As for studies on ATR (e.g., Wermter and Hahn (2005) or Nenadić et al. (2004)), no statistical testing of the term extraction algorithms to mere frequency counting was performed.

But after all, these kinds of commonly used statistical significance tests may not provide the right machinery in the first place. By design, they are rather limited (or focused) in their scope in that they just check whether a null hypothesis can be rejected or not. In such a sense, they do not provide a way to determine, e.g., to which *degree of magnitude* some differences pertain and thus do not offer the facilities to devise qualitative criteria to test whether an AM is superior to co-occurrence frequency counting.

The purpose of this study is therefore to postulate a set of criteria for the *qualitative* testing of differences among the various CE and ATR metrics. We do this by taking up the two conditions above which state that a good CE or ATR algorithm would rank most of the true positives in a candidate set in the upper portion and most of the true negatives in the lower portion of the output. Thus, compared to co-occurrence frequency counting, a *superior* CE/ATR algorithm should achieve the following four objectives:

²In particular Evert and Krenn (2001) use the chi-square test which assumes independent samples and is thus not really suitable for testing the significance of differences of two or more measures which are typically run on the same set of candidates (i.e., a dependent sample). Wermter and Hahn (2004) use the McNemar test for dependent samples, which only examines the differences in which two metrics do not coincide.

1. keep the true positives in the upper portion
2. keep the true negatives in the lower portion
3. *demote* true negatives from the upper portion
4. *promote* true positives from the lower portion.

We take these to be four qualitative criteria by which the merit of a certain AM against mere occurrence frequency counting can be determined.

3.2 Data Sets

For collocation extraction (CE), we used the data set provided by Wermter and Hahn (2004) which consists of a 114-million-word German newspaper corpus. After shallow syntactic analysis, the authors extracted Preposition-Noun-Verb (PNV) combinations occurring at least ten times and had them classified by human judges as to whether they constituted a valid collocation or not, resulting in 8644 PNV-combinations with 13.7% true positives. As for domain-specific automatic term recognition (ATR), we used a biomedical term candidate set put forth by Wermter and Hahn (2005), who, after shallow syntactic analysis, extracted 31,017 trigram term candidates occurring at least eight times out of a 104-million-word MEDLINE corpus. Checking these term candidates against the 2004 edition UMLS Metathesaurus (UMLS, 2004)³ resulted in 11.6% true positives. This information is summarized in Table 1.

	Collocations	Terms
domain	newspaper	biomedicine
language	German	English
linguistic type	PP-Verb combinations	noun phrases (trigrams)
corpus size	114 million	104 million
cutoff	10	8
# candidates	8,644	31,017
# true positives	1,180 (13.7%)	3,590 (11.6%)
# true negatives	7,464 (86.3%)	27,427 (88.4%)

Table 1: Data sets for Collocation Extraction (CE) and Automatic Term Discovery (ATR)

³The UMLS Metathesaurus is an extensive and carefully curated terminological resource for the biomedical domain.

3.3 The Association Measures

We examined both standard statistics-based and more recent linguistically rooted association measures against mere frequency of occurrence counting (henceforth referred to as Frequency). As the standard statistical AM, we selected the t-test (see also Manning and Schütze (1999) for a description on its use in CE and ATR) because it has been shown to be the best-performing statistics-only measure for CE (cf. Evert and Krenn (2001) and Krenn and Evert (2001)) and also for ATR (see Wermter and Hahn (2005)).

Concerning more recent linguistically grounded AMs, we looked at limited syntagmatic modifiability (LSM) for CE (Wermter and Hahn, 2004) and limited paradigmatic modifiability (LPM) for ATR (Wermter and Hahn, 2005). LSM exploits the well-known linguistic property that collocations are much less modifiable with additional lexical material (supplements) than non-collocations. For each collocation candidate, LSM determines the lexical supplement with the highest probability, which results in a higher collocativity score for those candidates with a particularly characteristic lexical supplement. LPM assumes that domain-specific terms are linguistically more fixed and show less distributional variation than common noun phrases. Taking n-gram term candidates, it determines the likelihood of precluding the appearance of alternative tokens in various token slot combinations, which results in higher scores for more constrained candidates. All measures assign a score to the candidates and thus produce a ranked output list.

3.4 Experimental Setup

In order to determine any potential merit of the above measures, we use the four criteria described in Section 3.1 and qualitatively compare the different rankings given to true positives and true negatives by an AM and by Frequency. For this purpose, we chose the middle rank as a mark to divide a ranked output list into an upper portion and a lower portion. Then we looked at the true positives (TPs) and true negatives (TNs) assigned to these portions by Frequency and quantified, according to the criteria postulated in Section 3.1, to what degree the other AMs changed these rankings (or not). In order to better quantify the degrees of movement, we partitioned both the upper and the lower portions into three further subportions.

	Association Measure	upper portion (ranks 1 - 4322)			lower portion (ranks 4323 - 8644)		
		0% - 16.7%	16.7% - 33.3%	33.3% - 50%	50% - 66.7%	66.7% - 83.3%	83.3% - 100%
Criterion 1 (905 TPs)	Freq	545 (60.2%)	216 (23.9%)	144 (15.9%)	0	0	0
	t-test	540 (59.7%)	198 (21.9%)	115 (12.7%)	9 (1.0%)	12 (1.3%)	12 (1.3%)
	LSM	606 (67.0%)	237 (26.2%)	35 (3.9%)	10 (1.1%)	12 (1.3%)	5 (0.6%)
Criterion 2 (4047 TNs)	Freq	0	0	0	1361 (33.6%)	1357 (33.5%)	1329 (32.8%)
	t-test	0	34 (0.8%)	613 (15.2%)	1121 (27.7%)	1100 (27.2%)	1179 (29.1%)
	LSM	118 (2.9%)	506 (12.5%)	726 (17.9%)	808 (20.0%)	800 (19.8%)	1089 (26.9%)
Criterion 3 (3417 TNs)	Freq	896 (26.2%)	1225 (35.9%)	1296 (37.9%)	0	0	0
	t-test	901 (26.4%)	1243 (36.4%)	932 (27.3%)	95 (2.8%)	47 (1.4%)	199 (5.8%)
	LSM	835 (24.4%)	1150 (33.7%)	342 (10.0%)	218 (6.4%)	378 (11.1%)	494 (14.5%)
Criterion 4 (275 TPs)	Freq	0	0	0	113 (41.1%)	85 (30.9%)	77 (28.0%)
	t-test	0	0	31 (11.3%)	88 (32.6%)	59 (21.5%)	95 (34.5%)
	LSM	0	10 (3.6%)	144 (52.4%)	85 (30.9%)	27 (9.8%)	9 (3.3%)

Table 2: Results on the four qualitative criteria for Collocation Extraction (CE)

	Association Measure	upper portion (ranks 1 - 15508)			lower portion (ranks 15509 - 31017)		
		0% - 16.7%	16.7% - 33.3%	33.3% - 50%	50% - 66.7%	66.7% - 83.3%	83.3% - 100%
Criterion 1 (2469 TPs)	Freq	1252 (50.7%)	702 (28.4%)	515 (20.9%)	0	0	0
	t-test	1283 (52.0%)	709 (28.7%)	446 (18.1%)	13 (0.5%)	2 (0.1%)	16 (0.6%)
	LPM	1346 (54.5%)	513 (20.8%)	301 (12.2%)	163 (6.6%)	95 (3.8%)	51 (2.1%)
Criterion 2 (14387 TNs)	Freq	0	0	0	4732 (32.9%)	4822 (33.5%)	4833 (33.6%)
	t-test	0	0	580 (4.0%)	4407 (30.6%)	4743 (33.0%)	4657 (32.4%)
	LPM	1009 (7.0%)	1698 (11.8%)	2190 (15.2%)	2628 (18.3%)	3029 (21.1%)	3834 (26.6%)
Criterion 3 (13040 TNs)	Freq	3917 (30.0%)	4467 (34.3%)	4656 (35.7%)	0	0	0
	t-test	3885 (29.8%)	4460 (34.2%)	4048 (31.0%)	315 (2.4%)	76 (0.6%)	256 (2.0%)
	LPM	2545 (19.5%)	2712 (20.8%)	2492 (19.1%)	2200 (16.9%)	1908 (14.6%)	1182 (9.1%)
Criterion 4 (1121 TPs)	Freq	0	0	0	438 (39.1%)	347 (31.0%)	336 (30.0%)
	t-test	0	0	97 (8.7%)	436 (38.9%)	348 (31.0%)	240 (21.4%)
	LPM	268 (23.9%)	246 (21.9%)	188 (16.8%)	180 (16.1%)	137 (12.2%)	102 (9.1%)

Table 3: Results on the four qualitative criteria for Automatic Term Discovery (ATR)

4 Results and Discussion

The first two criteria examine how conservative an association measure is with respect to Frequency, i.e., a superior AM at least should keep the status-quo (or even improve it) by keeping the true positives in the upper portion and the true negatives in the lower one. In meeting criteria 1 for CE, Table 2 shows that t-test behaves very similar to Frequency in keeping roughly the same amount of TPs in each of the upper three subportions. LSM even promotes its TPs from the third into the first two upper subportion (i.e., by a 7- and 2-point increase in the first and in the second subportion as well as a 12-point decrease in the third subportion, compared to Frequency).

With respect to the same criterion for ATR (see Table 3), Frequency and t-test again show quite similar distributions of TPs in the top three subportions. LPM, on the other hand, demonstrates a modest increase (by 4 points) in the top upper subportion, but decreases in the second and third one so that a small fraction of TPs gets demoted to the lower three subportions (6.6%, 3.8% and 2.1%).

Regarding criterion 2 for CE (see Table 2), t-

test’s share of TNs in the lower three subportions is slightly less than that of Frequency, leading to a 15-point increase in the adjacent third upper subportion. This local ”spilling over” to the upper portion is comparatively small considering the change that occurs with respect to LSM. Here, TNs appear in the second (12.5%) and the third (17.9%) upper subportions. For ATR, t-test once more shows a very similar distribution compared to Frequency, whereas LPM again promotes some of its lower TNs into the upper subportions (7%, 11.8% and 15.2%).

Criteria 3 and 4 examine the kinds of re-rankings (i.e., demoting upper portion TNs and promoting lower portion TPs) which an AM needs to perform in order to qualify as being superior to Frequency. These criteria look at how well an AM is able to undo the unfavorable ranking of TPs and TNs by Frequency. As for criterion 3 (the demotion of TNs from the upper portion) in CE, Table 2 shows that t-test is only marginally able to undo the unfavorable rankings in its third upper subportion (11 percentage points less of TNs). This causes a small fraction of TNs getting demoted to

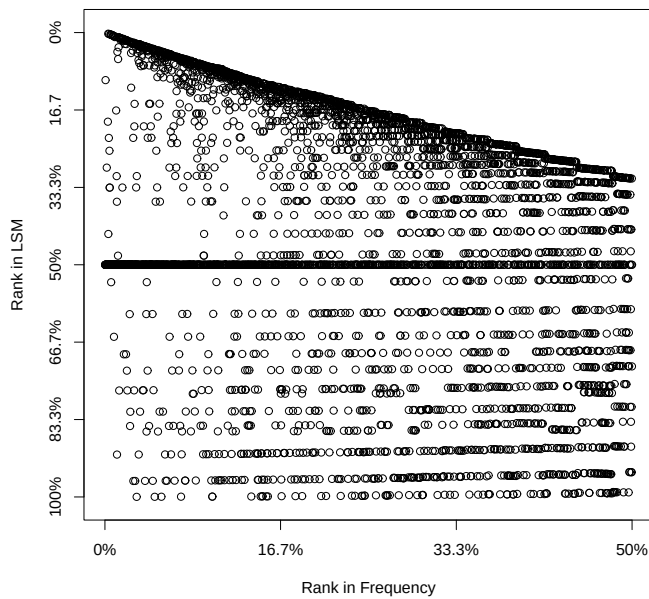


Figure 1: Collocations: True negatives moved from upper to lower portion (LSM rank compared to Frequency rank)

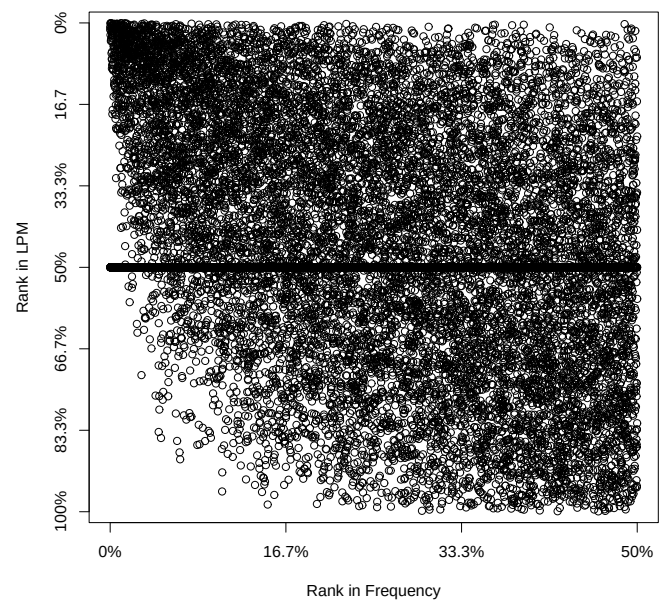


Figure 3: Terms: True negatives moved from upper to lower portion (LPM rank compared to Frequency rank)

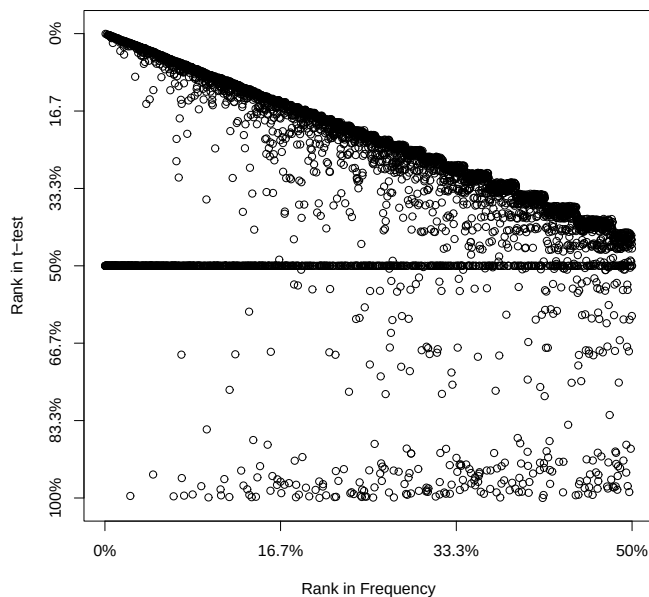


Figure 2: Collocations: True negatives moved from upper to lower portion (t-test rank compared to Frequency rank)

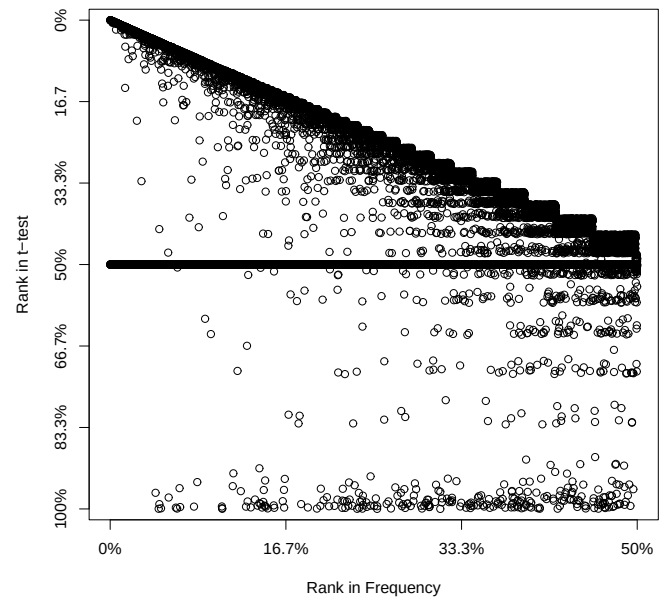


Figure 4: Terms: True negatives moved from upper to lower portion (t-test rank compared to Frequency rank)

the lower three subportions (viz. 2.8%, 1.4%, and 5.8%).

A view from another angle on this rather slight re-ranking is offered by the scatterplot in Figure 2, in which the rankings of the upper portion TNs

of Frequency are plotted against their ranking in t-test. Here it can be seen that, in terms of the rank subportions considered, the t-test TNs are concentrated along the same line as the Frequency TNs, with only a few being able to break this line and

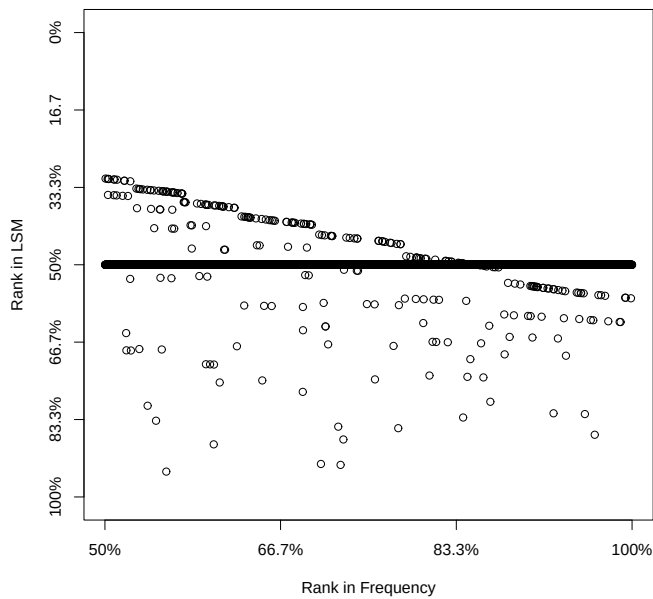


Figure 5: Collocations: True positives moved from lower to upper portion (LSM rank compared to Frequency rank)

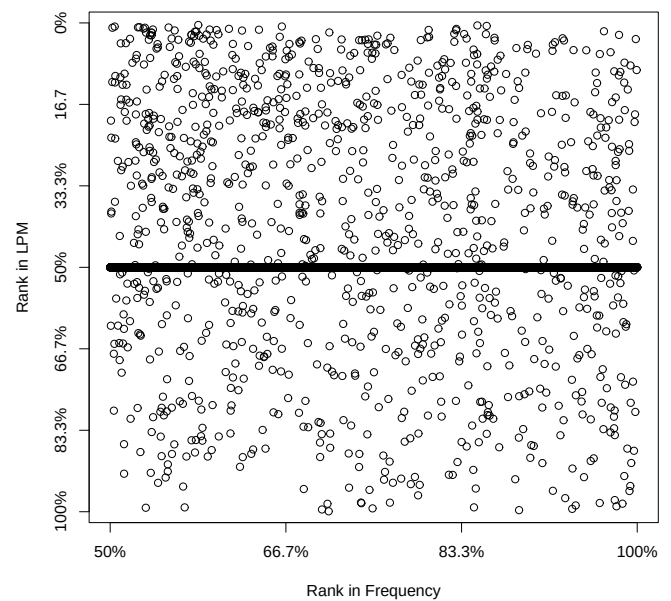


Figure 7: Terms: True positives moved from lower to upper portion (LPM rank compared to Frequency rank)

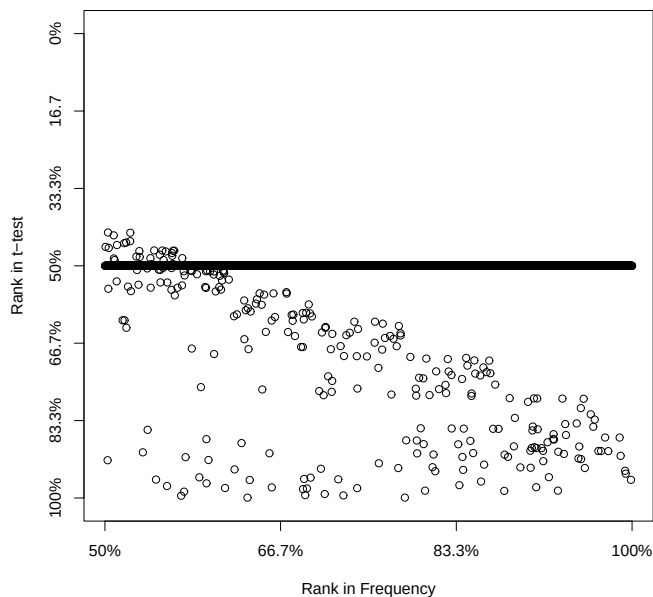


Figure 6: Collocations: True positives moved from lower to upper portion (t-test rank compared to Frequency rank)

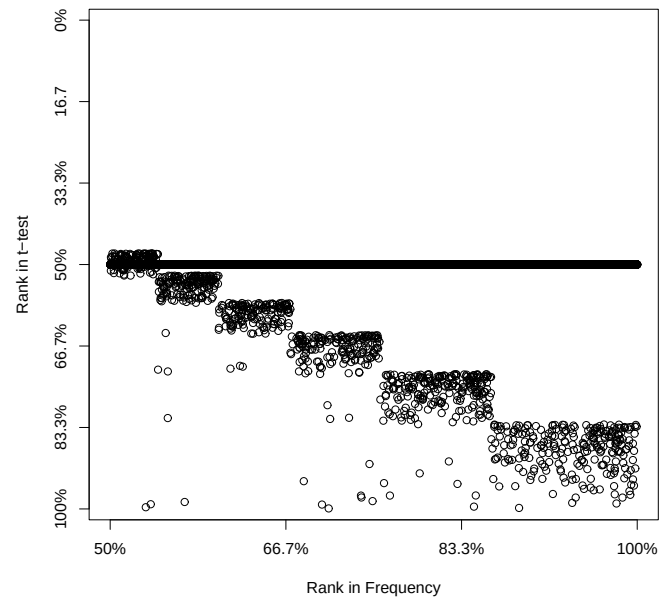


Figure 8: Terms: True positives moved from lower to upper portion (t-test rank compared to Frequency rank)

get demoted to a lower subportion.

A strikingly similar picture holds for this criterion in ATR: as can be witnessed from Figure 4, the vast majority of upper portion t-test TNs is stuck on the same line as in Frequency. The sim-

ilarity of t-test in both CE and ATR is even more remarkable given the fact in the actual number of upper portion TNs is more than four times higher in ATR (13040) than in CE (3076). A look at the actual figures in Table 3 indicates that t-test is even

less able to deviate from Frequency's TN distribution (i.e., the third upper subportion is only occupied by 4.7 points less TNs, with the other two subportions essentially remaining the same as in Frequency).

The two linguistically rooted measures, LSM for CE and LPM for ATR, offer quite a different picture regarding this criterion. With LSM, almost one third (32%) of the upper portion TNs get demoted to the three lower portions (see Table 2); with LPM, this proportion even amounts to 40.6% (see Table 3). The scatterplots in Figure 1 and Figure 3 visualize this from another perspective: in particular, LPM completely breaks the original Frequency ranking pattern and scatters the upper portion TNs in almost all possible directions, with the vast majority of them thus getting demoted to a lower rank than in Frequency. Although LSM stays more in line, still substantially more upper portion TNs get demoted than with t-test.

With regard to Criterion 4 (the promotion of TPs from the lower portion) in CE, t-test manages to promote 11.3% of its lower portion TPs to the adjacent third upper subportion, but at the same time demotes more TPs to the third lower subportion (34.5% compared to 28% in Frequency; see Table 2). Figure 6 thus shows the t-test TPs to be a bit more dispersed in the lower portion. For ATR, the t-test distribution of TPs differs even less from Frequency. Table 3 reveals that only 8.7% of the lower portion TPs get promoted to the adjacent third upper portion. The staggered groupinlpr g of lower portion t-test TPs (visualized in the respective scatterplot in Figure 8) actually indicates that there are certain plateaus beyond which the TPs cannot get promoted.

The two non-standard measures, LSM and LPM, once more present a very different picture. Regarding LSM, 56% of all lower portion TPs get promoted to the upper three subportions. The majority of these (52.4%) gets placed the third upper subportion. This can also be seen in the respective scatterplot in Figure 5 which shows a marked concentration of lower portion TPs in the third upper subportion. With respect to LPM, even 62.6% of all lower portion TPs make it to the upper portions – with the majority (23.9%) even getting promoted to the first upper subportion. The respective scatterplot in Figure 7 additionally shows that this upward movement of TPs, like the downward movement of TNs in Figure 3, is quite dispersed.

5 Conclusions

For lexical processing, the automatic identification of terms and collocations constitutes a research theme that has been dealt with by employing increasingly complex probabilistic criteria (t-test, mutual information, log-likelihood etc.). This trend is also reflected by their prominent status in standard textbooks on statistical NLP. The implicit justification in using these statistics-only metrics was that they would markedly outperform frequency of co-occurrence counting. We devised four qualitative criteria for explicitly testing this assumption. Using the best performing standard association measure (t-test) as a *pars pro toto*, our study indicates that the statistical sophistication does not pay off when compared with simple frequency of co-occurrence counting.

This pattern changes, however, when probabilistic measures incorporate additional linguistic knowledge about the distributional properties of terms and the modifiability properties of collocations. Our results show that these augmented metrics reveal a marked difference compared to frequency of occurrence counts – to a larger degree with respect to automatic term recognition, to a slightly lesser degree for collocation extraction.

References

- Béatrice Daille. 1996. Study and implementation of combined techniques for automatic extraction of terminology. In Judith L. Klavans and Philip Resnik, editors, *The Balancing Act: Combining Statistical and Symbolic Approaches to Language*, pages 49–66. Cambridge, MA: MIT Press.
- Stefan Evert and Brigitte Krenn. 2001. Methods for the qualitative evaluation of lexical association measures. In *ACL'01/EACL'01 – Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics and the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 188–195. Toulouse, France, July 9–11, 2001. San Francisco, CA: Morgan Kaufmann.
- Katerina T. Frantzi, Sophia Ananiadou, and Hideki Mima. 2000. Automatic recognition of multi-word terms: The C-value/NC-value method. *International Journal on Digital Libraries*, 3(2):115–130.
- Christian Jacquemin. 1999. Syntagmatic and paradigmatic representations of term variation. In *Proceedings of the 37rd Annual Meeting of the Association for Computational Linguistics*, pages 341–348. College Park, MD, USA, 20–26 June 1999. San Francisco, CA: Morgan Kaufmann.

- Christian Jacquemin. 2001. *Spotting and Discovering Terms through NLP*. Mass.: MIT Press.
- Brigitte Krenn and Stefan Evert. 2001. Can we do better than frequency? A case study on extracting pp-verb collocations. In *Proceedings of the ACL Workshop on Collocations*. Toulouse, France.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *COLING/ACL'98 – Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics & 17th International Conference on Computational Linguistics*, volume 2, pages 768–774. Montréal, Quebec, Canada, August 10-14, 1998. San Francisco, CA: Morgan Kaufmann.
- Dekang Lin. 1999. Automatic identification of non-compositional phrases. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 317–324. College Park, MD, USA, 20-26 June 1999. San Francisco, CA: Morgan Kaufmann.
- Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. Cambridge, MA; London, U.K.: Bradford Book & MIT Press.
- Goran Nenadić, Sophia Ananiadou, and John McNaught. 2004. Enhancing automatic term recognition through recognition of variation. In *COLING Geneva 2004 – Proceedings of the 20th International Conference on Computational Linguistics*, pages 604–610. Geneva, Switzerland, August 23-27, 2004. Association for Computational Linguistics.
- Lothar Sachs. 1984. *Applied Statistics: A Handbook of Techniques*. New York: Springer, 2nd edition.
- UMLS. 2004. *Unified Medical Language System*. Bethesda, MD: National Library of Medicine.
- Joachim Wermter and Udo Hahn. 2004. Collocation extraction based on modifiability statistics. In *COLING Geneva 2004 – Proceedings of the 20th International Conference on Computational Linguistics*, volume 2, pages 980–986. Geneva, Switzerland, August 23-27, 2004. Association for Computational Linguistics.
- Joachim Wermter and Udo Hahn. 2005. Paradigmatic modifiability statistics for the extraction of complex multi-word terms. In *HLT-EMNLP'05 – Proceedings of the 5th Human Language Technology Conference and 2005 Conference on Empirical Methods in Natural Language Processing*, pages 843–850. Vancouver, Canada, October 6-8, 2005. Association for Computational Linguistics.

Ontologizing Semantic Relations

Marco Pennacchiotti

ART Group - DISP
University of Rome “Tor Vergata”
Viale del Politecnico 1
Rome, Italy
pennacchiotti@info.uniroma2.it

Patrick Pantel

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA90292
pantel@isi.edu

Abstract

Many algorithms have been developed to harvest lexical semantic resources, however few have linked the mined knowledge into formal knowledge repositories. In this paper, we propose two algorithms for automatically ontologizing (attaching) semantic relations into WordNet. We present an empirical evaluation on the task of attaching *part-of* and *causation* relations, showing an improvement on *F*-score over a baseline model.

1 Introduction

NLP researchers have developed many algorithms for mining knowledge from text and the Web, including facts (Etzioni et al. 2005), semantic lexicons (Riloff and Shepherd 1997), concept lists (Lin and Pantel 2002), and word similarity lists (Hindle 1990). Many recent efforts have also focused on extracting binary semantic relations between entities, such as entailments (Szpektor et al. 2004), *is-a* (Ravi-chandran and Hovy 2002), *part-of* (Girju et al. 2003), and other relations.

The output of most of these systems is flat lists of lexical semantic knowledge such as “*Italy is-a country*” and “*orange similar-to blue*”. However, using this knowledge beyond simple keyword matching, for example in inferences, requires it to be linked into formal semantic repositories such as ontologies or term banks like WordNet (Fellbaum 1998).

Pantel (2005) defined the task of *ontologizing* a lexical semantic resource as linking its terms to the concepts in a WordNet-like hierarchy. For example, “*orange similar-to blue*” ontologizes in WordNet to “*orange#2 similar-to blue#1*” and “*orange#2 similar-to blue#2*”. In his framework,

Pantel proposed a method of inducing ontological co-occurrence vectors¹ which are subsequently used to ontologize unknown terms into WordNet with 74% accuracy.

In this paper, we take the next step and explore two algorithms for ontologizing binary semantic relations into WordNet and we present empirical results on the task of attaching *part-of* and *causation* relations. Formally, given an instance (x, r, y) of a binary relation r between terms x and y , the ontologizing task is to identify the WordNet senses of x and y where r holds. For example, the instance $(proton, PART-OF, element)$ ontologizes into WordNet as $(proton\#1, PART-OF, element\#2)$.

The first algorithm that we explore, called the *anchoring approach*, was suggested as a promising avenue of future work in (Pantel 2005). This bottom up algorithm is based on the intuition that x can be disambiguated by retrieving the set of terms that occur in the same relation r with y and then finding the senses of x that are most similar to this set. The assumption is that terms occurring in the same relation will tend to have similar meaning. In this paper, we propose a measure of similarity to capture this intuition.

In contrast to anchoring, our second algorithm, called the *clustering approach*, takes a top-down view. Given a relation r , suppose that we are given every *conceptual instance* of r , i.e., instances of r in the upper ontology like $(particles\#1, PART-OF, substances\#1)$. An instance (x, r, y) can then be ontologized easily by finding the senses of x and y that are subsumed by ancestors linked by a conceptual instance of r . For example, the instance $(proton, PART-OF, element)$ ontologizes to $(proton\#1, PART-OF, element\#2)$ since *proton#1* is subsumed by *particles* and *element#2* is subsumed by *substances*. The problem then is to automatically infer the set of con-

¹ The ontological co-occurrence vector of a concept consists of all lexical co-occurrences with the concept in a corpus.

ceptual instances. In this paper, we develop a clustering algorithm for generalizing a set of relation instances to conceptual instances by looking up the WordNet hypernymy hierarchy for common ancestors, as specific as possible, that subsume as many instances as possible. An instance is then attached to its senses that are subsumed by the highest scoring conceptual instances.

2 Relevant Work

Several researchers have worked on ontologizing semantic resources. Most recently, Pantel (2005) developed a method to propagate lexical co-occurrence vectors to WordNet synsets, forming ontological co-occurrence vectors. Adopting an extension of the distributional hypothesis (Harris 1985), the co-occurrence vectors are used to compute the similarity between synset/synset and between lexical term/synset. An unknown term is then attached to the WordNet synset whose co-occurrence vector is most similar to the term's co-occurrence vector. Though the author suggests a method for attaching more complex lexical structures like binary semantic relations, the paper focused only on attaching terms.

Basili (2000) proposed an unsupervised method to infer semantic classes (WordNet synsets) for terms in domain-specific verb relations. These relations, such as (x, EXPAND, y) are first automatically learnt from a corpus. The semantic classes of x and y are then inferred using *conceptual density* (Agirre and Rigau 1996), a WordNet-based measure applied to all instantiation of x and y in the corpus. Semantic classes represent possible common generalizations of the verb arguments. At the end of the process, a set of syntactic-semantic patterns are available for each verb, such as:

$(\text{social\_group\#1}, \text{expand}, \text{act\#2})$
 $(\text{instrumentality\#2}, \text{expand}, \text{act\#2})$

The method is successful on specific relations with few instances (such as domain verb relations) while its value on generic and frequent relations, such as *part-of*, was untested.

Girju et al. (2003) presented a highly supervised machine learning algorithm to infer semantic constraints on *part-of* relations, such as $(\text{object\#1}, \text{PART-OF}, \text{social\_event\#1})$. These constraints are then used as selectional restrictions in harvesting *part-of* instances from ambiguous lexical patterns, like “ X of Y ”. The approach shows high performance in terms of precision and recall, but, as the authors acknowledge, it

requires large human effort during the training phase.

Others have also made significant additions to WordNet. For example, in eXtended WordNet (Harabagiu et al. 1999), the glosses in WordNet are enriched by disambiguating the nouns, verbs, adverbs, and adjectives with synsets. Another work has enriched WordNet synsets with topically related words extracted from the Web (Agirre et al. 2001). Finally, the general task of word sense disambiguation (Gale et al. 1991) is relevant since there the task is to ontologize each term in a passage into a WordNet-like sense inventory. If we had a large collection of sense-tagged text, then our mining algorithms could directly discover WordNet attachment points at harvest time. However, since there is little high precision sense-tagged corpora, methods are required to ontologize semantic resources without fully disambiguating text.

3 Ontologizing Semantic Relations

Given an instance (x, r, y) of a binary relation r between terms x and y , the ontologizing task is to identify the senses of x and y where r holds. In this paper, we focus on WordNet 2.0 senses, though any similar term bank would apply.

Let S_x and S_y be the sets of all WordNet senses of x and y . A *sense pair*, s_{xy} , is defined as any pair of senses of x and y : $s_{xy} = \{s_x, s_y\}$ where $s_x \in S_x$ and $s_y \in S_y$. The set of all sense pairs S_{xy} consists of all permutations between senses in S_x and S_y .

In order to attach a relation instance (x, r, y) into WordNet, one must:

- *Disambiguate* x and y , that is, find the subsets $S'_x \subseteq S_x$ and $S'_y \subseteq S_y$ for which the relation r holds; and
- *Instantiate* the relation in WordNet, using the synsets corresponding to all correct permutations between the senses in S'_x and S'_y . We denote this set of *attachment points* as S'_{xy} .

If S_x or S_y is empty, no attachments are produced.

For example, the instance $(\text{study}, \text{PART-OF}, \text{report})$ is ontologized into WordNet through the senses $S'_x = \{\text{survey\#1}, \text{study\#2}\}$ and $S'_y = \{\text{report\#1}\}$. The final attachment points S'_{xy} are:

$(\text{survey\#1}, \text{PART-OF}, \text{report\#1})$
 $(\text{study\#1}, \text{PART-OF}, \text{report\#1})$

Unlike common algorithms for word sense disambiguation, here it is important to take into consideration the semantic dependency between the two terms x and y . For example, an entity that is *part-of* a study has to be some kind of informa-

tion. This knowledge about mutual selectional preference (the preferred semantic class that fills a certain relation role, as x or y) can be exploited to ontologize the instance.

In the following sections, we propose two algorithms for ontologizing binary semantic relations.

3.1 Method 1: Anchor Approach

Given an instance (x, r, y) , this approach fixes the term y , called the *anchor*, and then disambiguates x by looking at all other terms that occur in the relation r with y . Based on the principle of distributional similarity (Harris 1985), the algorithm assumes that the words that occur in the same relation r with y will be more similar to the correct sense(s) of x than the incorrect ones. After disambiguating x , the process is then inverted with x as the anchor to disambiguate y .

In the first step, y is fixed and the algorithm retrieves the set of all other terms X' that occur in an instance (x', r, y) , $x' \in X^2$. For example, given the instance (*reflections*, *PART-OF*, *book*), and a resource containing the following relations:

(*false allegations*, *PART-OF*, *book*)
(*stories*, *PART-OF*, *book*)
(*expert analysis*, *PART-OF*, *book*)
(*conclusions*, *PART-OF*, *book*)

the resulting set X' would be: {*allegations*, *stories*, *analysis*, *conclusions*}.

All possible permutations, $S_{xx'}$, between the senses of x and the senses of each term in X' , called $S_{x'}$, are computed. For each sense pair $\{s_x, s_{x'}\} \in S_{xx'}$, a similarity score $r(s_x, s_{x'})$ is calculated using WordNet:

$$r(s_x, s_{x'}) = \frac{1}{d(s_x, s_{x'}) + 1} \times f(s_{x'})$$

where the distance $d(s_x, s_{x'})$ is the length of the shortest path connecting the two synsets in the hypernymy hierarchy of WordNet, and $f(s_{x'})$ is the number of times sense $s_{x'}$ occurs in any of the instances of X' . Note that if no connection between two synsets exists, then $r(s_x, s_{x'}) = 0$.

The overall sense score for each sense s_x of x is calculated as:

$$r(s_x) = \sum_{s_{x'} \in S_{x'}} r(s_x, s_{x'})$$

Finally, the algorithm inverts the process by setting x as the anchor and computes $r(s_y)$ for

each sense of y . All possible permutations of senses are computed and scored by averaging $r(s_x)$ and $r(s_y)$. Permutations scoring higher than a threshold τ_1 are selected as the attachment points in WordNet. We experimentally set $\tau_1 = 0.02$.

3.2 Method 2: Clustering Approach

The main idea of the clustering approach is to leverage the lexical behaviors of the two terms in an instance as a whole. The assumption is that the general meaning of the relation is derived from the combination of the two terms.

The algorithm is divided in two main phases. In the first phase, *semantic clusters* are built using the WordNet senses of all instances. A semantic cluster is defined by the set of instances that have a common semantic generalization. We denote the *conceptual instance* of the semantic cluster as the pair of WordNet synsets that represents this generalization. For example the following two *part-of* instances:

(*second section*, *PART-OF*, *Los Angeles-area news*)
(*Sandag study*, *PART-OF*, *report*)

are in a common cluster represented by the following conceptual instance:

[*writing#2*, *PART-OF*, *message#2*]

since *writing#2* is a hypernym of both *section* and *study*, and *message#2* is a hypernym of *news* and *report*³.

In the second phase, the algorithm attaches an instance into WordNet by using WordNet distance metrics and frequency scores to select the best cluster for each instance. A good cluster is one that:

- achieves a good trade-off between generality and specificity; and
- disambiguates among the senses of x and y using the other instances' senses as support.

For example, given the instance (*second section*, *PART-OF*, *Los Angeles-area news*) and the following conceptual instances:

[*writing#2*, *PART-OF*, *message#2*]
[*object#1*, *PART-OF*, *message#2*]
[*writing#2*, *PART-OF*, *communication#2*]
[*social_group#1*, *PART-OF*, *broadcast#2*]
[*organization#*, *PART-OF*, *message#2*]

the first conceptual instance should be scored highest since it is both not too generic nor too specific and is supported by the instance (*Sandag study*, *PART-OF*, *report*), i.e., the conceptual instance subsumes both instances. The second and

² For semantic relations between complex terms, like (*expert analysis*, *PART-OF*, *book*), only the head noun of terms are recorded, like "*analysis*". As a future work, we plan to use the whole term if it is present in WordNet.

³ Again, here, we use the syntactic head of each term for generalization since we assume that it drives the meaning of the term itself.

the third conceptual instances should be scored lower since they are too generic, while the last two should be scored lower since the sense for *section* and *news* are not supported by other instances. The system then outputs, for each instance, the set of sense pairs that are subsumed by the highest scoring conceptual instance. In the previous example:

(*section*#1, *PART-OF*, *news*#1)
 (*section*#1, *PART-OF*, *news*#2)
 (*section*#1, *PART-OF*, *news*#3)

are selected, as they are subsumed by [*writing*#2, *PART-OF*, *message*#2]. These sense pairs are then retained as *attachment points* into WordNet.

Below, we describe each phase in more detail.

Phase 1: Cluster Building

Given an instance (x, r, y) , all sense pair permutations $s_{xy} = \{s_x, s_y\}$ are retrieved from WordNet. A set of *candidate conceptual instances*, C_{xy} , is formed for each instance from the permutation of each WordNet ancestor of s_x and s_y , following the hypernymy link, up to degree τ_2 .

Each candidate conceptual instance, $c = \{c_x, c_y\}$, is scored by its degree of generalization as follows:

$$r(c) = \frac{1}{(n_x + 1) \times (n_y + 1)}$$

where n_i is the number of hypernymy links needed to go from s_i to c_i , for $i \in \{x, y\}$. $r(c)$ ranges from $[0, 1]$ and is highest when little generalization is needed.

For example, the instance (*Sandag study*, *PART-OF*, *report*) produces 70 sense pairs since *study* has 10 senses and *report* has 7 senses. Assuming $\tau_2=1$, the instance sense (*survey*#1, *PART-OF*, *report*#1) has the following set of candidate conceptual instances:

C_{xy}	n_x	n_y	$r(c)$
(<i>survey</i> #1, <i>PART-OF</i> , <i>report</i> #1)	0	0	1
(<i>survey</i> #1, <i>PART-OF</i> , <i>document</i> #1)	0	1	0.5
(<i>examination</i> #1, <i>PART-OF</i> , <i>report</i> #1)	1	0	0.5
(<i>examination</i> #1, <i>PART-OF</i> , <i>document</i> #1)	1	1	0.25

Finally, each candidate conceptual instance c forms a cluster of all instances (x, r, y) that have some sense pair s_x and s_y as hyponyms of c . Note also that candidate conceptual instances may be subsumed by other candidate conceptual instances. Let G_c refer to the set of all candidate conceptual instances subsumed by candidate conceptual instance c .

Intuitively, better candidate conceptual instances are those that subsume both many instances and other candidate conceptual instances, but at the same time that have the least distance from subsumed instances. We capture this intuition with the following score of c :

$$score(c) = \frac{\sum_{g \in G_c} r(g)}{|G_c|} \times \log|I_c| \times \log|G_c|$$

where I_c is the set of instances subsumed by c . We experimented with different variations of this score and found that it is important to put more weight on the distance between subsumed conceptual instances than the actual number of subsumed instances. Without the $\log$ terms, the highest scoring conceptual instances are too generic (i.e., they are too high up in the ontology).

Phase 2: Attachment Points Selection

In this phase, we utilize the conceptual instances of the previous phase to attach each instance (x, r, y) into WordNet.

At the end of Phase 1, an instance can be clustered in different conceptual instances. In order to select an attachment, the algorithm selects the sense pair of x and y that is subsumed by the highest scoring candidate conceptual instance. It and all other sense pairs that are subsumed by this conceptual instance are then retained as the final attachment points.

As a side effect, a final set of conceptual instances is obtained by deleting from each candidate those instances that are subsumed by a higher scoring conceptual instance. Remaining conceptual instances are then re-scored using $score(c)$. The final set of conceptual instances thus contains unambiguous sense pairs.

4 Experimental Results

In this section we provide an empirical evaluation of our two algorithms.

4.1 Experimental Setup

Researchers have developed many algorithms for harvesting semantic relations from corpora and the Web. For the purposes of this paper, we may choose any one of them and manually validate its mined relations. We choose *Espresso*⁴, a general-purpose, broad, and accurate corpus harvesting algorithm requiring minimal supervision. Adopt-

⁴ Reference suppressed – the paper introducing *Espresso* has also been submitted to COLING/ACL 2006.

SYSTEM	PRECISION	RECALL	F-SCORE
BL	54.0%	31.3%	39.6%
AN	40.7%	47.3%	43.8%
CL	57.4%	49.6%	53.2%

Table 1. System precision, recall and *F*-score on the *part-of* relation.

ing a bootstrapping approach, *Espresso* takes as input a few seed instances of a particular relation and iteratively learns surface patterns to extract more instances.

Test Sets

We experiment with two relations: *part-of* and *causation*. The *causation* relation occurs when an entity produces an effect or is responsible for events or results, for example (*virus*, *CAUSE*, *influenza*) and (*burning fuel*, *CAUSE*, *pollution*). We manually built five seed relation instances for both relations and apply *Espresso* to a dataset consisting of a sample of articles from the Aquaint (TREC-9) newswire text collection. The sample consists of 55.7 million words extracted from the Los Angeles Times data files. *Espresso* extracted 1,468 *part-of* instances and 1,129 *causation* instances. We manually validated the output and randomly selected 200 correct relation instances of each relation for ontologizing into WordNet 2.0.

Gold Standard

We manually built a gold standard of all correct attachments of the test sets in WordNet. For each relation instance (x, r, y), two human annotators selected from all sense permutations of x and y the correct attachment points in WordNet. For example, for (*synthetic material*, *PART-OF*, *filter*), the judges selected the following attachment points: (*synthetic material*#1, *PART-OF*, *filter*#1) and (*synthetic material*#1, *PART-OF*, *filter*#2). The kappa statistic (Siegel and Castellan Jr. 1988) on the two relations together was $K = 0.73$.

Systems

The following three systems are evaluated:

- **BL**: the baseline system that attaches each relation instance to the first (most common) WordNet sense of both terms;
- **AN**: the *anchor approach* described in Section 3.1.
- **CL**: the *clustering approach* described in Section 3.2.

SYSTEM	PRECISION	RECALL	F-SCORE
BL	45.0%	25.0%	32.1%
AN	41.7%	32.4%	36.5%
CL	40.0%	32.6%	35.9%

Table 2. System precision, recall and *F*-score on the *causation* relation.

4.2 Precision, Recall and *F*-score

For both the *part-of* and *causation* relations, we apply the three systems described above and compare their attachment performance using precision, recall, and *F*-score. Using the manually built gold standard, the precision of a system on a given relation instance is measured as the percentage of correct attachments and recall is measured as the percentage of correct attachments retrieved by the system. Overall system precision and recall are then computed by averaging the precision and recall of each relation instance.

Table 1 and Table 2 report the results on the *part-of* and *causation* relations. We experimentally set the CL generalization parameter τ_2 to 5 and the τ_1 parameter for AN to 0.02.

4.3 Discussion

For both relations, CL and AN outperform the baseline in overall *F*-score. For *part-of*, Table 1 shows that CL outperforms BL by 13.6% in *F*-score and AN by 9.4%. For *causation*, Table 2 shows that AN outperforms BL by 4.4% on *F*-score and CL by 0.6%.

The good results of the CL method on the *part-of* relation suggest that instances of this relation are particularly amenable to be clustered. The generality of the *part-of* relation in fact allows the creation of fairly natural clusters, corresponding to different sub-types of *part-of*, as those proposed in (Winston 1983). The *causation* relation, however, being more difficult to define at a semantic level (Girju 2003), is less easy to cluster and thus to disambiguate.

Both CL and AN have better recall than BL, but precision results vary with CL beating BL only on the *part-of* relation. Overall, the system performances suggest that ontologizing semantic relations into WordNet is in general not easy.

The better results of CL and AN with respect to BL suggest that the use of comparative semantic analysis among corpus instances is a good way to carry out disambiguation. Yet, the BL

method shows surprisingly good results. This indicates that also a simple method based on word sense usage in language can be valuable. An interesting avenue of future work is to better combine these two different views in a single system.

The low recall results for CL are mostly attributed to the fact that in Phase 2 only the best scoring cluster is retained for each instance. This means that instances with multiple senses that do not have a common generalization are not captured. For example the *part-of* instance (*wings*, *PART-OF*, *chicken*) should cluster both in [*body_part#1*, *PART-OF*, *animal#1*] and [*body_part#1*, *PART-OF*, *food#2*], but only the best scoring one is retained.

5 Conceptual Instances: Other Uses

Our *clustering approach* from Section 3.2 is enabled by learning conceptual instances – relations between mid-level ontological concepts. Beyond the ontologizing task, conceptual instances may be useful for several other tasks. In this section, we discuss some of these opportunities and present small qualitative evaluations.

Conceptual instances represent common semantic generalizations of a particular relation. For example, below are two possible conceptual instances for the *part-of* relation:

[*person#1*, *PART-OF*, *organization#1*]
[*act#1*, *PART-OF*, *plan#1*]

The first conceptual instance in the example subsumes all the *part-of* instances in which one or more persons are part of an organization, such as:

(*president Brown*, *PART-OF*, *executive council*)
(*representatives*, *PART-OF*, *organization*)
(*students*, *PART-OF*, *orchestra*)
(*players*, *PART-OF*, *Metro League*)

Below, we present three possible ways of exploiting these conceptual instances.

Support to Relation Extraction Tools

Conceptual instances may be used to support relation extraction algorithms such as *Espresso*.

Most minimally supervised harvesting algorithms do not exploit *generic patterns*, i.e. those patterns with high recall but low precision, since they cannot separate correct and incorrect relation instances. For example, the pattern “*X of Y*” extracts many correct relation instances like “*wheel of the car*” but also many incorrect ones like “*house of representatives*”.

Girju et al. (2003) described a highly supervised algorithm for learning semantic constraints

on *generic patterns*, leading to a very significant increase in system recall without deteriorating precision. Conceptual instances can be used to automatically learn such semantic constraints by acting as a filter for generic patterns, retaining only those instances that are subsumed by high scoring conceptual instances. Effectively, conceptual instances are used as *selectional restrictions* for the relation. For example, our system discards the following incorrect instances:

(*week*, *CAUSE*, *coalition*)
(*demeanor*, *CAUSE*, *vacuum*)

as they are both part of the very low scoring conceptual instance [*abstraction#6*, *CAUSE*, *state#1*].

Ontology Learning from Text

Each conceptual instance can be viewed as a formal specification of the relation at hand. For example, Winston (1983) manually identified six sub-types of the *part-of* relation: *member-collection*, *component-integral object*, *portion-mass*, *stuff-object*, *feature-activity* and *place-area*. Such classifications are useful in applications and tasks where a semantically rich organization of knowledge is required. Conceptual instances can be viewed as an automatic derivation of such a classification based on corpus usage. Moreover, conceptual instances can be used to improve the ontology learning process itself. For example, our *clustering approach* can be seen as an *inductive* step producing conceptual instances that are then used in a *deductive* step to learn new instances. An algorithm could iterate between the induction/deduction cycle until no new relation instances and conceptual instances can be inferred.

Word Sense Disambiguation

Word Sense Disambiguation (WSD) systems can exploit the selectional restrictions identified by conceptual instances to disambiguate ambiguous terms occurring in particular contexts. For example, given the sentence:

“*the board is composed by members of different countries*”

and a harvesting algorithm that extracts the *part-of* relation (*members*, *PART-OF*, *board*), the system could infer the correct senses for *board* and *members* by looking at their closest conceptual instance. In our system, we would infer the attachment (*member#1*, *PART-OF*, *board#1*) since it is part of the highest scoring conceptual instance [*person#1*, *PART-OF*, *organization#1*].

CONCEPTUAL INSTANCE	SCORE	# INSTANCES	INSTANCES
[multitude#3, PART-OF, group#1]	2.04	10	(ordinary people, PART-OF, Democratic Revolutionary Party) (unlicensed people, PART-OF, underground economy) (young people, PART-OF, commission) (air mass, PART-OF, cold front)
[person#1, PART-OF, organization#1]	1.71	43	(foreign ministers, PART-OF, council) (students, PART-OF, orchestra) (socialists, PART-OF, Iraqi National Joint Action Committee) (players, PART-OF, Metro League)
[act#2, PART-OF, plan#1]	1.60	16	(major concessions, PART-OF, new plan) (attacks, PART-OF, coordinated terrorist plan) (visit, PART-OF, exchange program) (survey, PART-OF, project)
[communication#2, PART-OF, book#1]	1.14	10	(hints, PART-OF, booklet) (soup recipes, PART-OF, book) (information, PART-OF, instruction manual) (extensive expert analysis, PART-OF, book)
[compound#2, PART-OF, waste#1]	0.57	3	(salts, PART-OF, powdery white waste) (lime, PART-OF, powdery white waste) (resin, PART-OF, waste)

Table 3. Sample of the highest scoring conceptual instances learned for the *part-of* relation. For each conceptual instance, we report the score(c), the number of instances, and some example instances.

5.1 Qualitative Evaluation

Table 3 and Table 4 list samples of the highest ranking conceptual instances obtained by our system for the *part-of* and *causation* relations. Below we provide a small evaluation to verify:

- the *correctness* of the conceptual instances. Incorrect conceptual instances such as [attribute#2, CAUSE, state#4], discovered by our system, can impede WSD and extraction tools where precise selectional restrictions are needed; and
- the *accuracy* of the conceptual instances. Sometimes, an instance is incorrectly attached to a correct conceptual instance. For example, the instance (air mass, PART-OF, cold front) is incorrectly clustered in [group#1, PART-OF, multitude#3] since mass and front both have a sense that is descendant of group#1 and multitude#3. However, these are not the correct senses of mass and front for which the *part-of* relation holds.

For evaluating *correctness*, we manually verify how many correct conceptual instances are produced by Phase 2 of the clustering approach described in Section 3.2. The claim is that a *correct* conceptual instance is one for which the relation holds for all possible subsumed senses. For example, the conceptual instance [group#1, PART-OF, multitude#3] is correct, as the relation holds for every semantic subsumption of the two senses. An example of an incorrect conceptual instance is [state#4, CAUSE, abstraction#6] since it subsumes the incorrect instance (audience, CAUSE, new context). A manual evaluation of the highest scoring 200 conceptual instances, generated on our test sets described in Section 4.1,

showed 82% correctness for the *part-of* relation and 86% for *causation*.

For estimating the overall clustering *accuracy*, we evaluated the number of correctly clustered instances in each conceptual instance. For example, the instance (business people, PART-OF, committee) is correctly clustered in [multitude#3, PART-OF, group#1] and the instance (law, PART-OF, constitutional pitfalls) is incorrectly clustered in [group#1, PART-OF, artifact#1]. We estimated the overall accuracy by manually judging the instances attached to 10 randomly sampled conceptual instances. The accuracy for *part-of* is 84% and for *causation* it is 76.6%.

6 Conclusions

In this paper, we proposed two algorithms for automatically ontologizing binary semantic relations into WordNet: an *anchoring approach* and a *clustering approach*. Experiments on the *part-of* and *causation* relations showed promising results. Both algorithms outperformed the baseline on *F*-score. Our best results were on the *part-of* relation where the *clustering approach* achieved 13.6% higher *F*-score than the baseline.

The induction of conceptual instances has opened the way for many avenues of future work. We intend to pursue the ideas presented in Section 5 for using conceptual instances to: i) support knowledge acquisition tools by learning semantic constraints on extracting patterns; ii) support ontology learning from text; and iii) improve word sense disambiguation through selectional restrictions. Also, we will try different similarity score functions for both the clustering and the anchor approaches, as those surveyed in Corley and Mihalcea (2005).

CONCEPTUAL INSTANCE	SCORE	# INSTANCES	INSTANCES
[change#3, CAUSE, state#4]	1.49	17	(separation, CAUSE, anxiety) (demotion, CAUSE, roster vacancy) (budget cuts, CAUSE, enrollment declines) (reduced flow, CAUSE, vacuum)
[act#2, CAUSE, state#3]	0.81	20	(oil drilling, CAUSE, air pollution) (workplace exposure, CAUSE, genetic injury) (industrial emissions, CAUSE, air pollution) (long recovery, CAUSE, great stress)
[person#1, CAUSE, act#2]	0.64	12	(homeowners, CAUSE, water waste) (needlelike puncture, CAUSE, physician) (group member, CAUSE, controversy) (children, CAUSE, property damage)
[organism#1, CAUSE, disease#1]	0.03	4	(parasites, CAUSE, pneumonia) (virus, CAUSE, influenza) (chemical agents, CAUSE, pneumonia) (genetic mutation, CAUSE, Dwarfism)

Table 4. Sample of the highest scoring conceptual instances learned for the *causation* relation. For each conceptual instance, we report score(c), the number of instances, and some example instances.

The algorithms described in this paper may be applied to ontologize many lexical resources of semantic relations, no matter the harvesting algorithm used to mine them. In doing so, we have the potential to quickly enrich our ontologies, like WordNet, thus reducing the knowledge acquisition bottleneck. It is our hope that we will be able to leverage these enriched resources, albeit with some noisy additions, to improve performance on knowledge-rich problems such as question answering and textual entailment.

References

- Agirre, E. and Rigau, G. 1996. Word sense disambiguation using conceptual density. In *Proceedings of COLING-96*. pp. 16-22. Copenhagen, Denmark.
- Agirre, E.; Ansa, O.; Martinez, D.; and Hovy, E. 2001. Enriching WordNet concepts with topic signatures. In *Proceedings of NAACL Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*. Pittsburgh, PA.
- Basili, R.; Pazienza, M.T.; and Vindigni, M. 2000. Corpus-driven learning of event recognition rules. In *Proceedings of Workshop on Machine Learning and Information Extraction (ECAI-00)*.
- Corley, C. and Mihalcea, R. 2005. Measuring the Semantic Similarity of Texts. In *Proceedings of the ACL Workshop on Empirical Modelling of Semantic Equivalence and Entailment*. Ann Arbor, MI.
- Etzioni, O.; Cafarella, M.J.; Downey, D.; Popescu, A.-M.; Shaked, T.; Soderland, S.; Weld, D.S.; and Yates, A. 2005. Unsupervised named-entity extraction from the Web: An experimental study. *Artificial Intelligence*, 165(1): 91-134.
- Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.
- Gale, W.; Church, K.; and Yarowsky, D. 1992. A method for disambiguating word senses in a large corpus. *Computers and Humanities*, 26:415-439.
- Girju, R.; Badulescu, A.; and Moldovan, D. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of HLT/NAACL-03*. pp. 80-87. Edmonton, Canada.
- Girju, R. 2003. Automatic Detection of Causal Relations for Question Answering. In *Proceedings of ACL Workshop on Multilingual Summarization and Question Answering*. Sapporo, Japan.
- Harabagiu, S.; Miller, G.; and Moldovan, D. 1999. WordNet 2 - A Morphologically and Semantically Enhanced Resource. In *Proceedings of SIGLEX-99*. pp.1-8. University of Maryland.
- Harris, Z. 1985. Distributional structure. In: Katz, J. J. (ed.) *The Philosophy of Linguistics*. New York: Oxford University Press. pp. 26-47.
- Hindle, D. 1990. Noun classification from predicate-argument structures. In *Proceedings of ACL-90*. pp. 268-275. Pittsburgh, PA.
- Lin, D. and Pantel, P. 2002. Concept discovery from text. In *Proceedings of COLING-02*. pp. 577-583. Taipei, Taiwan.
- Pantel, P. 2005. Inducing Ontological Co-occurrence Vectors. In *Proceedings of ACL-05*. pp. 125-132. Ann Arbor, MI.
- Ravichandran, D. and Hovy, E.H. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL-2002*. pp. 41-47. Philadelphia, PA.
- Riloff, E. and Shepherd, J. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of EMNLP-97*.
- Siegel, S. and Castellan Jr., N. J. 1988. *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill.
- Szpektor, I.; Tanev, H.; Dagan, I.; and Coppola, B. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP-04*. Barcelona, Spain.
- Winston, M.; Chaffin, R.; and Hermann, D. 1987. A taxonomy of part-whole relations. *Cognitive Science*, 11:417-444.

Semantic Taxonomy Induction from Heterogenous Evidence

Rion Snow

Computer Science Department
Stanford University
Stanford, CA 94305
rion@cs.stanford.edu

Daniel Jurafsky

Linguistics Department
Stanford University
Stanford, CA 94305
jurafsky@stanford.edu

Andrew Y. Ng

Computer Science Department
Stanford University
Stanford, CA 94305
ang@cs.stanford.edu

Abstract

We propose a novel algorithm for inducing semantic taxonomies. Previous algorithms for taxonomy induction have typically focused on independent classifiers for discovering new single relationships based on hand-constructed or automatically discovered textual patterns. By contrast, our algorithm flexibly incorporates evidence from multiple classifiers over heterogenous relationships to optimize the entire structure of the taxonomy, using knowledge of a word's coordinate terms to help in determining its hypernyms, and vice versa. We apply our algorithm on the problem of sense-disambiguated noun hyponym acquisition, where we combine the predictions of hypernym and coordinate term classifiers with the knowledge in a preexisting semantic taxonomy (WordNet 2.1). We add 10,000 novel synsets to WordNet 2.1 at 84% precision, a relative error reduction of 70% over a non-joint algorithm using the same component classifiers. Finally, we show that a taxonomy built using our algorithm shows a 23% relative F-score improvement over WordNet 2.1 on an independent testset of hypernym pairs.

1 Introduction

The goal of capturing structured relational knowledge about lexical terms has been the motivating force underlying many projects in lexical acquisition, information extraction, and the construction of semantic taxonomies. Broad-coverage semantic taxonomies such as WordNet (Fellbaum, 1998) and CYC (Lenat, 1995) have been constructed by hand at great cost; while a crucial source of knowledge about the relations between words, these taxonomies still suffer from sparse coverage.

Many algorithms with the potential for automatically extending lexical resources have been proposed, including work in lexical acquisition (Riloff and Shepherd, 1997; Roark and Charniak, 1998) and in discovering instances, named entities, and alternate glosses (Etzioni et al., 2005; Pasça, 2005). Additionally, a wide variety of relationship-specific classifiers have been proposed, including pattern-based classifiers for hyponyms (Hearst, 1992), meronyms (Girju, 2003),

synonyms (Lin et al., 2003), a variety of verb relations (Chklovski and Pantel, 2004), and general purpose analogy relations (Turney et al., 2003). Such classifiers use hand-written or automatically-induced patterns like *Such NP_y as NP_x or NP_y like NP_x* to determine, for example that NP_y is a hyponym of NP_x (i.e., NP_y IS-A NP_x). While such classifiers have achieved some degree of success, they frequently lack the global knowledge necessary to integrate their predictions into a complex taxonomy with multiple relations.

Past work on semantic taxonomy induction includes the noun hypernym hierarchy created in (Carballo, 2001), the part-whole taxonomies in (Girju, 2003), and a great deal of recent work described in (Buitelaar et al., 2005). Such work has typically either focused on only inferring small taxonomies over a single relation, or as in (Carballo, 2001), has used evidence for multiple relations independently from one another, by for example first focusing strictly on inferring clusters of coordinate terms, and then by inferring hypernyms over those clusters.

Another major shortfall in previous techniques for taxonomy induction has been the inability to handle lexical ambiguity. Previous approaches have typically sidestepped the issue of polysemy altogether by making the assumption of only a single sense per word, and inferring taxonomies explicitly over words and not senses. Enforcing a false monosemy has the downside of making potentially erroneous inferences; for example, collapsing the polysemous term *Bush* into a single sense might lead one to infer by transitivity that a *rose bush* is a kind of *U.S. president*.

Our approach simultaneously provides a solution to the problems of jointly considering evidence about multiple relationships as well as lexical ambiguity within a single probabilistic framework. The key contribution of this work is to offer a solution to two crucial problems in taxonomy in-

duction and hyponym acquisition: the problem of combining heterogeneous sources of evidence in a flexible way, and the problem of correctly identifying the appropriate word sense of each new word added to the taxonomy.¹

2 A Probabilistic Framework for Taxonomy Induction

In section 2.1 we introduce our definitions for taxonomies, relations, and the *taxonomic constraints* that enforce dependencies between relations; in section 2.2 we give a probabilistic model for defining the conditional probability of a set of relational evidence given a taxonomy; in section 2.3 we formulate a local search algorithm to find the taxonomy maximizing this conditional probability; and in section 2.4 we extend our framework to deal with lexical ambiguity.

2.1 Taxonomies, Relations, and Taxonomic Constraints

We define a taxonomy $\mathbf{T}$ as a set of pairwise relations $\mathbf{R}$ over some domain of objects $\mathbf{D}_{\mathbf{T}}$. For example, the relations in WordNet include *hypernymy*, *holonymy*, *verb entailment*, and many others; the objects of WordNet between which these relations hold are its word senses or *synsets*. We define that each relation $R \in \mathbf{R}$ is a set of ordered or unordered pairs of objects $(i, j) \in \mathbf{D}_{\mathbf{T}}$; we define $R_{ij} \in \mathbf{T}$ if relationship R holds over objects (i, j) in $\mathbf{T}$.

Relations for Hyponym Acquisition

For the case of hyponym acquisition, the objects in our taxonomy are WordNet *synsets*. In this paper we focus on two of the many possible relationships between senses: the hypernym relation and the coordinate term relation. We treat the hypernym or ISA relation as atomic; we use the notation H_{ij}^n if a sense j is the n -th ancestor of a sense i in the hypernym hierarchy. We will simply use H_{ij} to indicate that j is an ancestor of i at some unspecified level. Two senses are typically considered to be “coordinate terms” or “taxonomic sisters” if they share an immediate parent in the hypernym hierarchy. We generalize this notion of siblinghood to state that two senses i and j are (m, n) -cousins if their closest *least common*

subsumer (LCS)² is within exactly m and n links, respectively.³ We use the notation C_{ij}^{mn} to denote that i and j are (m, n) -cousins. Thus coordinate terms are $(1, 1)$ -cousins; technically the hypernym relation may also be seen as a specific case of this representation; an immediate parent in the hypernym hierarchy is a $(1, 0)$ -cousin, and the k -th ancestor is a $(k, 0)$ -cousin.

Taxonomic Constraints

A semantic taxonomy such as WordNet enforces certain *taxonomic constraints* which disallow particular taxonomies $\mathbf{T}$. For example, the ISA transitivity constraint in WordNet requires that each synset inherits the hypernyms of its hypernym, and the part-inheritance constraint requires that each synset inherits the meronyms of its hypernyms.

For the case of hyponym acquisition we enforce the following two taxonomic constraints on the hypernym and (m, n) -cousin relations:

1. ISA Transitivity:

$$H_{ij}^m \wedge H_{jk}^n \Rightarrow H_{ik}^{m+n}.$$

2. Definition of (m, n) -cousinhood:

$$C_{ij}^{mn} \Leftrightarrow \exists k. k = LCS(i, j) \wedge H_{ik}^m \wedge H_{jk}^n.$$

Constraint (1) requires that each synset inherits the hypernyms of its direct hypernym; constraint (2) simply defines the (m, n) -cousin relation in terms of the atomic hypernym relation.

The addition of any new hypernym relation to a preexisting taxonomy will usually necessitate the addition of a set of other novel relations as implied by the taxonomic constraints. We refer to the full set of novel relations implied by a new link R_{ij} as $\mathbf{I}(R_{ij})$; we discuss the efficient computation of the set of implied links for the purpose of hyponym acquisition in Section 3.4.

2.2 A Probabilistic Formulation

We propose that the event $R_{ij} \in \mathbf{T}$ has some prior probability $P(R_{ij} \in \mathbf{T})$, and $P(R_{ij} \in$

¹The taxonomies discussed in this paper are available for download at <http://ai.stanford.edu/~rion/swn>.

²A least common subsumer $LCS(i, j)$ is defined as a synset that is an ancestor in the hypernym hierarchy of both i and j which has no child that is also an ancestor of both i and j . When there is more than one LCS (due to multiple inheritance), we refer to the *closest LCS*, i.e., the LCS that minimizes the maximum distance to i and j .

³An (m, n) -cousin for $m \geq 2$ corresponds to the English kinship relation “ $(m - 1)$ -th cousin $|m - n|$ -times removed.”

$\mathbf{T}) + P(R_{ij} \notin \mathbf{T}) = 1$. We define the probability of the taxonomy as a whole as the joint probability of its component relations; given a partition of all possible relations $\mathbf{R} = \{A, B\}$ where $A \in \mathbf{T}$ and $B \notin \mathbf{T}$, we define:

$$P(\mathbf{T}) = P(A \in \mathbf{T}, B \notin \mathbf{T}).$$

We assume that we have some set of observed evidence $\mathbf{E}$ consisting of observed features over pairs of objects in some domain $\mathbf{D}_{\mathbf{E}}$; we'll begin with the assumption that our features are over pairs of words, and that the objects in the taxonomy also correspond directly to words.⁴ Given a set of features $E_{ij}^R \in \mathbf{E}$, we assume we have some model for inferring $P(R_{ij} \in \mathbf{T} | E_{ij}^R)$, i.e., the posterior probability of the event $R_{ij} \in \mathbf{T}$ given the corresponding evidence E_{ij}^R for that relation. For example, evidence for the hypernym relation E_{ij}^H might be the set of all observed lexico-syntactic patterns containing i and j in all sentences in some corpus.

For simplicity we make the following independence assumptions: first, we assume that each item of observed evidence E_{ij}^R is independent of all other observed evidence given the taxonomy $\mathbf{T}$, i.e., $P(\mathbf{E} | \mathbf{T}) = \prod_{E_{ij}^R \in \mathbf{E}} P(E_{ij}^R | \mathbf{T})$.

Further, we assume that each item of observed evidence E_{ij}^R depends on the taxonomy $\mathbf{T}$ only by way of the corresponding relation R_{ij} , i.e.,

$$P(E_{ij}^R | \mathbf{T}) = \begin{cases} P(E_{ij}^R | R_{ij} \in \mathbf{T}) & \text{if } R_{ij} \in \mathbf{T} \\ P(E_{ij}^R | R_{ij} \notin \mathbf{T}) & \text{if } R_{ij} \notin \mathbf{T} \end{cases}$$

For example, if our evidence E_{ij}^H is a set of observed lexico-syntactic patterns indicative of hypernymy between two words i and j , we assume that whatever dependence the relations in $\mathbf{T}$ have on our observations may be explained entirely by dependence on the existence or non-existence of the single hypernym relation $H(i, j)$.

Applying these two independence assumptions we may express the conditional probability of our evidence given the taxonomy:

$$P(\mathbf{E} | \mathbf{T}) = \prod_{R_{ij} \in \mathbf{T}} P(E_{ij}^R | R_{ij} \in \mathbf{T}) \cdot \prod_{R_{ij} \notin \mathbf{T}} P(E_{ij}^R | R_{ij} \notin \mathbf{T}).$$

Rewriting the conditional probability in terms of our estimates of the posterior probabilities

⁴In section 2.4 we drop this assumption, extending our model to manage lexical ambiguity.

$P(R_{ij} | E_{ij}^R)$ using Bayes Rule, we obtain:

$$P(\mathbf{E} | \mathbf{T}) = \prod_{R_{ij} \in \mathbf{T}} \frac{P(R_{ij} \in \mathbf{T} | E_{ij}^R) P(E_{ij}^R)}{P(R_{ij} \in \mathbf{T})} \cdot \prod_{R_{ij} \notin \mathbf{T}} \frac{P(R_{ij} \notin \mathbf{T} | E_{ij}^R) P(E_{ij}^R)}{P(R_{ij} \notin \mathbf{T})}.$$

Within our model we define the goal of taxonomy induction to be to find the taxonomy $\hat{\mathbf{T}}$ that maximizes the conditional probability of our observations $\mathbf{E}$ given the relationships of $\mathbf{T}$, i.e., to find

$$\hat{\mathbf{T}} = \arg \max_{\mathbf{T}} P(\mathbf{E} | \mathbf{T}).$$

2.3 Local Search Over Taxonomies

We propose a search algorithm for finding $\hat{\mathbf{T}}$ for the case of hyponym acquisition. We assume we begin with some initial (possibly empty) taxonomy $\mathbf{T}$. We restrict our consideration of possible new taxonomies to those created by the single operation $\text{ADD-RELATION}(R_{ij}, \mathbf{T})$, which adds the single relation R_{ij} to $\mathbf{T}$.

We define the multiplicative change $\Delta_{\mathbf{T}}(R_{ij})$ to the conditional probability $P(\mathbf{E} | \mathbf{T})$ given the addition of a single relation R_{ij} :

$$\begin{aligned} \Delta_{\mathbf{T}}(R_{ij}) &= P(\mathbf{E} | \mathbf{T}') / P(\mathbf{E} | \mathbf{T}) \\ &= \frac{P(R_{ij} \in \mathbf{T} | E_{ij}^R) P(E_{ij}^R)}{P(R_{ij} \notin \mathbf{T} | E_{ij}^R) P(E_{ij}^R)} \cdot \frac{P(R_{ij} \notin \mathbf{T})}{P(R_{ij} \in \mathbf{T})} \\ &= k \left(\frac{P(R_{ij} \in \mathbf{T} | E_{ij}^R)}{1 - P(R_{ij} \in \mathbf{T} | E_{ij}^R)} \right). \end{aligned}$$

Here k is the inverse odds of the prior on the event $R_{ij} \in \mathbf{T}$; we consider this to be a constant independent of i, j , and the taxonomy $\mathbf{T}$.

To enforce the taxonomic constraints in $\mathbf{T}$, for each application of the ADD-RELATION operator we must add all new relations in the implied set $\mathbf{I}(R_{ij})$ not already in $\mathbf{T}$.⁵ Thus we define the multiplicative change of the full set of implied relations as the product over all new relations:

$$\Delta_{\mathbf{T}}(\mathbf{I}(R_{ij})) = \prod_{R \in \mathbf{I}(R_{ij})} \Delta_{\mathbf{T}}(R).$$

⁵For example, in order to add the new synset *microsoft* under the noun synset *company#n#1* in WordNet 2.1, we must necessarily add the new relations $H^2(\text{microsoft}, \text{institution}\#n\#1)$ $C^{11}(\text{microsoft}, \text{dotcom}\#n\#1)$, and so on.

This definition leads to the following best-first search algorithm for hyponym acquisition, which at each iteration defines the new taxonomy as the union of the previous taxonomy $\mathbf{T}$ and the set of novel relations implied by the relation R_{ij} that maximizes $\Delta_{\mathbf{T}}(\mathbf{I}(R_{ij}))$ and thus maximizes the conditional probability of the evidence over all possible single relations:

$$\begin{aligned} & \text{WHILE } \max_{R_{ij} \notin \mathbf{T}} \Delta_{\mathbf{T}}(\mathbf{I}(R_{ij})) > 1 \\ & \quad \mathbf{T} \leftarrow \mathbf{T} \cup \mathbf{I}(\arg \max_{R_{ij} \notin \mathbf{T}} \Delta_{\mathbf{T}}(\mathbf{I}(R_{ij}))). \end{aligned}$$

2.4 Extending the Model to Manage Lexical Ambiguity

Since word senses are not directly observable, if the objects in the taxonomy are word senses (as in WordNet), we must extend our model to allow for a many-to-many mapping (e.g., a word-to-sense mapping) between $\mathbf{D}_{\mathbf{E}}$ and $\mathbf{D}_{\mathbf{T}}$. For this setting we assume we know the function $\text{senses}(i)$, mapping from the word i to all of i 's possible corresponding senses.

We assume that each set of word-pair evidence E_{ij}^R we possess is in fact sense-pair evidence E_{kl}^R for a specific pair of senses $k_0 \in \text{senses}(i), l_0 \in \text{senses}(j)$. Further, we assume that a new relation between two words is probable only between the correct sense pair, i.e.:

$$P(R_{kl}|E_{ij}^R) = 1\{k = k_0, l = l_0\} \cdot P(R_{ij}|E_{ij}^R).$$

When computing the conditional probability of a specific new relation $R_{kl} \in \mathbf{I}(R_{ab})$, we assume that the relevant sense pair k_0, l_0 is the one which maximizes the probability of the new relation, i.e. for $k \in \text{senses}(i), l \in \text{senses}(j)$,

$$(k_0, l_0) = \arg \max_{k, l} P(R_{kl} \in \mathbf{T} | E_{ij}^R).$$

Our independence assumptions for this extension need only to be changed slightly; we now assume that the evidence E_{ij}^R depends on the taxonomy $\mathbf{T}$ via only a single relation between sense-pairs R_{kl} . Using this revised independence assumption the derivation for best-first search over taxonomies for hyponym acquisition remains unchanged. One side effect of this revised independence assumption is that the addition of the single “sense-collapsed” relation R_{kl} in the taxonomy $\mathbf{T}$ will explain the evidence E_{ij}^R for the relation over words i and j now that such evidence has been revealed to concern only the specific senses k and l .

3 Extending WordNet

We demonstrate the ability of our model to use evidence from multiple relations to extend WordNet with novel noun hyponyms. While in principle we could use any number of relations, for simplicity we consider two primary sources of evidence: the probability of two words in WordNet being in a hypernym relation, and the probability of two words in WordNet being in a coordinate relation.

In sections 3.1 and 3.2 we describe the construction of our hypernym and coordinate classifiers, respectively; in section 3.3 we outline the efficient algorithm we use to perform local search over hyponym-extended WordNets; and in section 3.4 we give an example of the implicit structure-based word sense disambiguation performed within our framework.

3.1 Hyponym Classification

Our classifier for the hypernym relation is derived from the “hyponym-only” classifier described in (Snow et al., 2005). The features used for predicting the hypernym relationship are obtained by parsing a large corpus of newswire and encyclopedia text with MINIPAR (Lin, 1998). From the resulting dependency trees the evidence E_{ij}^H for each word pair (i, j) is constructed; the evidence takes the form of a vector of counts of occurrences that each labeled syntactic dependency path was found as the shortest path connecting i and j in some dependency tree. The labeled training set is constructed by labeling the collected feature vectors as positive “known hypernym” or negative “known non-hypernym” examples using WordNet 2.0; 49,922 feature vectors were labeled as positive training examples, and 800,828 noun pairs were labeled as negative training examples. The model for predicting $P(H_{ij}|E_{ij}^H)$ is then trained using logistic regression, predicting the noun-pair hypernymy label from WordNet from the feature vector of lexico-syntactic patterns.

The hypernym classifier described above predicts the probability of the generalized hypernym-ancestor relation over words $P(H_{ij}|E_{ij}^H)$. For the purposes of taxonomy induction, we would prefer an ancestor-distance specific set of classifiers over senses, i.e., for $k \in \text{senses}(i), l \in \text{senses}(j)$, the set of classifiers estimating $\{P(H_{kl}^1|E_{ij}^H), P(H_{kl}^2|E_{ij}^H), \dots\}$.

One problem that arises from directly assigning the probability $P(H_{ij}^n|E_{ij}^H) \propto P(H_{ij}|E_{ij}^H)$ for all n is the possibility of adding a novel hyponym to an overly-specific hypernym, which might still satisfy $P(H_{ij}^n|E_{ij}^H)$ for a very large n . In order to discourage unnecessary overspecification, we penalize each probability $P(H_{ij}^k|E_{ij}^H)$ by a factor λ^{k-1} for some $\lambda < 1$, and renormalize: $P(H_{ij}^k|E_{ij}^H) \propto \lambda^{k-1}P(H_{ij}|E_{ij}^H)$. In our experiments we set $\lambda = 0.95$.

3.2 (m, n) -cousin Classification

The classifier for learning coordinate terms relies on the notion of *distributional similarity*, i.e., the idea that two words with similar meanings will be used in similar contexts (Hindle, 1990). We extend this notion to suggest that words with similar meanings should be near each other in a semantic taxonomy, and in particular will likely share a hypernym as a near parent.

Our classifier for (m, n) -cousins is derived from the algorithm and corpus given in (Ravichandran et al., 2005). In that work an efficient randomized algorithm is derived for computing clusters of similar nouns. We use a set of more than 1000 distinct clusters of English nouns collected by their algorithm over 70 million webpages⁶, with each noun i having a score representing its cosine similarity to the centroid c of the cluster to which it belongs, $\cos(\theta(i, c))$.

We use the cluster scores of noun pairs as input to our own algorithm for predicting the (m, n) -cousin relationship between the senses of two words i and j . If two words i and j appear in a cluster together, with cluster centroid c , we set our single coordinate input feature to be the minimum cluster score $\min(\cos(\theta(i, c)), \cos(\theta(j, c)))$, and zero otherwise. For each such noun pair feature, we construct a labeled training set of (m, n) -cousin relation labels from WordNet 2.1. We define a noun pair (i, j) to be a “known (m, n) -cousin” if for some senses $k \in \text{senses}(i), l \in \text{senses}(j)$, $C_{ij}^{mn} \in \text{WordNet}$; if more than one such relation exists, we assume the relation with smallest sum $m + n$, breaking ties by smallest absolute difference $|m - n|$. We consider all such labeled relationships from WordNet with $0 \leq m, n \leq 7$; pairs of words that have no corresponding pair of synsets connected in the hypernym hi-

erarchy, or with $\min(m, n) > 7$, are assigned to a single class C^∞ . Further, due to the symmetry of the similarity score, we merge each class $C^{mn} = C^{mn} \cup C^{nm}$; this implies that the resulting classifier will predict, as expected given a symmetric input, $P(C_{kl}^{mn}|E_{ij}^C) = P(C_{kl}^{nm}|E_{ij}^C)$.

We find 333,473 noun synset pairs in our training set with similarity score greater than 0.15. We next apply softmax regression to learn a classifier that predicts $P(C_{ij}^{mn}|E_{ij}^C)$, predicting the WordNet class labels from the single similarity score derived from the noun pair’s cluster similarity.

3.3 Details of our Implementation

Hyponym acquisition is among the simplest and most straightforward of the possible applications of our model; here we show how we efficiently implement our algorithm for this problem. First, we identify the set of all the word pairs (i, j) over which we have hypernym and/or coordinate evidence, and which might represent additions of a novel hyponym to the WordNet 2.1 taxonomy (i.e., that has a known noun hypernym and an unknown hyponym, or has a known noun coordinate term and an unknown coordinate term). This yields a list of 95,000 single links over threshold $P(R_{ij}) > 0.12$.

For each unknown hyponym i we may have several pieces of evidence; for example, for the unknown term *continental* we have 21 relevant pieces of hypernym evidence, with links to possible hypernyms $\{\text{carrier}, \text{airline}, \text{unit}, \dots\}$; and we have 5 pieces of coordinate evidence, with links to possible coordinate terms $\{\text{airline}, \text{american eagle}, \text{airbus}, \dots\}$.

For each proposed hypernym or coordinate link involved with the novel hyponym i , we compute the set of candidate hypernyms for i ; in practice we consider all senses of the immediate hypernym j for each potential novel hypernym, and all senses of the coordinate term k and its first two hypernym ancestors for each potential coordinate.

In the *continental* example, from the 26 individual pieces of evidence over words we construct the set of 99 unique synsets that we will consider as possible hypernyms; these include the two senses of the word *airline*, the ten senses of the word *carrier*, and so forth.

Next, we iterate through each of the possible hypernym synsets l under which we might add the new word i ; for each synset l we com-

⁶As a preprocessing step we hand-edit the clusters to remove those containing non-English words, terms related to adult content, and other webpage-specific clusters.

pute the change in taxonomy score resulting from adding the implied relations $\mathbf{I}(H_{il}^1)$ required by the taxonomic constraints of $\mathbf{T}$. Since typically our set of all evidence involving i will be much smaller than the set of possible relations in $\mathbf{I}(H_{il}^1)$, we may efficiently check whether, for each sense $s \in \text{senses}(w)$, for all words where we have some evidence E_{iw}^R , whether s participates in some relation with i in the set of implied relations $\mathbf{I}(H_{il}^1)$.⁷ If there is more than one sense $s \in \text{senses}(w)$, we add to $\mathbf{I}(H_{il}^1)$ the single relationship R_{is} that maximizes the taxonomy likelihood, i.e. $\arg \max_{s \in \text{senses}(w)} \Delta_{\mathbf{T}}(R_{is})$.

3.4 Hypernym Sense Disambiguation

A major strength of our model is its ability to correctly choose the sense of a hypernym to which to add a novel hyponym, despite collecting evidence over untagged word pairs. In our algorithm word sense disambiguation is an implicit side-effect of our algorithm; since our algorithm chooses to add the single link which, with its implied links, yields the most likely taxonomy, and since each distinct synset in WordNet has a different immediate neighborhood of relations, our algorithm simply disambiguates each node based on its surrounding structural information.

As an example of sense disambiguation in practice, consider our example of *continental*. Suppose we are iterating through each of the 99 possible synsets under which we might add *continental* as a hyponym, and we come to the synset *airline#n#2* in WordNet 2.1, i.e. “a commercial organization serving as a common carrier.” In this case we will iterate through each piece of hypernym and coordinate evidence; we find that the relation $H(\text{continental}, \text{carrier})$ is satisfied with high probability for the specific synset *carrier#n#5*, the grandparent of *airline#n#2*; thus the factor $\Delta_{\mathbf{T}}(H^3(\text{continental}, \text{carrier#n#5}))$ is included in the factor of the set of implied relations $\Delta_{\mathbf{T}}(\mathbf{I}(H^1(\text{continental}, \text{airline#n#2})))$.

Suppose we instead evaluate the *first* synset of *airline*, i.e., *airline#n#1*, with the gloss “a hose that carries air under pressure.” For this synset none of the other 20 relationships directly implied by hypernym evidence or the 5 relationships implied by the coordinate ev-

idence are implied by adding the single link $H^1(\text{continental}, \text{airline#n#1})$; thus the resulting change in the set of implied links given by the correct “carrier” sense of *airline* is much higher than that of the “hose” sense. In fact it is the largest of all the 99 considered hypernym links for *continental*; $H^1(\text{continental}, \text{airline#n#2})$ is link #18,736 added to the taxonomy by our algorithm.

4 Evaluation

In order to evaluate our framework for taxonomy induction, we have applied hyponym acquisition to construct several distinct taxonomies, starting with the base of WordNet 2.1 and only adding novel noun hyponyms. Further, we have constructed taxonomies using a baseline algorithm, which uses the identical hypernym and coordinate classifiers used in our joint algorithm, but which does not combine the evidence of the classifiers.

In section 4.1 we describe our evaluation methodology; in sections 4.2 and 4.3 we analyze the fine-grained precision and disambiguation precision of our algorithm compared to the baseline; in section 4.4 we compare the coarse-grained precision of our links (motivated by categories defined by the WordNet *supersenses*) against the baseline algorithm and against an “oracle” for named entity recognition.

Finally, in section 4.5 we evaluate the taxonomies inferred by our algorithm directly against the WordNet 2.1 taxonomy; we perform this evaluation by testing each taxonomy on a set of human judgments of hypernym and non-hypernym noun pairs sampled from newswire text.

4.1 Methodology

We evaluate the quality of our acquired hyponyms by direct judgment. In four separate annotation sessions, two judges labeled {50,100,100,100} samples uniformly generated from the first {100,1000,10000,20000} single links added by our algorithm.

For the direct measure of fine-grained precision, we simply ask for each link $H(X, Y)$ added by the system, is X a Y ? In addition to the fine-grained precision, we give a coarse-grained evaluation, inspired by the idea of supersense-tagging in (Ciarmita and Johnson, 2003). The 26 supersenses used in WordNet 2.1 are listed in Table 1; we label a hyponym link as correct in the coarse-grained evaluation if the novel hyponym is placed under the appropriate supersense. This evaluation task

⁷Checking whether or not $R_{is} \in \mathbf{I}(H_{il}^1)$ may be efficiently computed by checking whether s is in the hypernym ancestors of l or if it shares a least common subsumer with l within 7 steps.

1 Tops	8 communication	15 object	22 relation
2 act	9 event	16 person	23 shape
3 animal	10 feeling	17 phenomenon	24 state
4 artifact	11 food	18 plant	25 substance
5 attribute	12 group	19 possession	26 time
6 body	13 location	20 process	
7 cognition	14 motive	21 quantity	

Table 1: The 26 WordNet supersenses

is similar to a fine-grained Named Entity Recognition (Fleischman and Hovy, 2002) task with 26 categories; for example, if our algorithm mistakenly inserts a novel non-capital city under the hyponym *state capital*, it will inherit the correct supersense *location*. Finally, we evaluate the ability of our algorithm to correctly choose the appropriate sense of the hypernym under which a novel hyponym is being added. Our labelers categorize each candidate sense-disambiguated hypernym synset suggested by our algorithm into the following categories:

- c_1 : Correct sense-disambiguated hypernym.
- c_2 : Correct hypernym word, but incorrect sense of that word.
- c_3 : Incorrect hypernym, but correct supersense.
- c_4 : Any other relation is considered incorrect.

A single hyponym/hypernym pair is allowed to be simultaneously labeled 2 and 3.

4.2 Fine-grained evaluation

Table 2 displays the results of our evaluation of fine-grained precision for the baseline non-joint algorithm (**Base**) and our joint algorithm (**Joint**), as well as the relative error reduction (**ER**) of our algorithm over the baseline. We use the minimum of the two judges’ scores. Here we define fine-grained precision as $c_1/total$. We see that our joint algorithm strongly outperforms the baseline, and has high precision for predicting novel hyponyms up to 10,000 links.

4.3 Hypernym sense disambiguation

Also in Table 2 we compare the sense disambiguation precision of our algorithm and the baseline. Here we measure the precision of sense-disambiguation among all examples where each algorithm found a correct hyponym word; our calculation for disambiguation precision is $c_1/(c_1 + c_2)$. Again our joint algorithm outperforms the baseline algorithm at all levels of recall. Interestingly the baseline disambiguation precision improves with higher recall; this may

#Links	Fine-grained Pre.			Disambiguation Pre.		
	Base	Joint	ER	Base	Joint	ER
100	0.60	1.00	100%	0.86	1.00	100%
1000	0.52	0.93	85%	0.84	1.00	100%
10000	0.46	0.84	70%	0.90	1.00	100%
20000	0.46	0.68	41%	0.94	0.98	68%

Table 2: Fine-grained and disambiguation precision and error reduction for hyponym acquisition

# Links	NER Oracle	Base	Joint	ER vs. NER	ER vs. Base
100	1.00	0.72	1.00	0%	100%
1000	0.69	0.68	0.99	97%	85%
10000	0.45	0.69	0.96	93%	70%
20000	0.54	0.69	0.92	83%	41%

Table 3: Coarse-grained precision and error reduction vs. Non-joint baseline and NER Oracle

be attributed to the observation that the highest-confidence hypernyms predicted by individual classifiers are likely to be polysemous, whereas hypernyms of lower confidence are more frequently monosemous (and thus trivially easy to disambiguate).

4.4 Coarse-grained evaluation

We compute coarse-grained precision as $(c_1 + c_3)/total$. Inferring the correct coarse-grained supersense of a novel hyponym can be viewed as a fine-grained (26-category) Named Entity Recognition task; our algorithm for taxonomy induction can thus be viewed as performing high-accuracy fine-grained NER. Here we compare against both the baseline non-joint algorithm as well as an “oracle” algorithm for Named Entity Recognition, which perfectly classifies the supersense of all nouns that fall under the four supersenses $\{person, group, location, quantity\}$, but works only for those supersenses. Table 3 shows the results of this coarse-grained evaluation. We see that the baseline non-joint algorithm has higher precision than the NER oracle as 10,000 and 20,000 links; however, both are significantly outperformed by our joint algorithm, which maintains high coarse-grained precision (92%) even at 20,000 links.

4.5 Comparison of inferred taxonomies and WordNet

For our final evaluation we compare our learned taxonomies directly against the currently existing hypernym links in WordNet 2.1. In order to compare taxonomies we use a hand-labeled test

	WN	+10K	+20K	+30K	+40K
PRE	0.524	0.524	0.574	0.583	0.571
REC	0.165	0.165	0.203	0.211	0.211
F	0.251	0.251	0.300	0.309	0.307

Table 4: Taxonomy hypernym classification vs. WordNet 2.1 on hand-labeled testset

set of over 5,000 noun pairs, randomly-sampled from newswire corpora (described in (Snow et al., 2005)). We measured the performance of both our inferred taxonomies and WordNet against this test set.⁸ The performance and comparison of the best WordNet classifier vs. our taxonomies is given in Table 4. Our best-performing inferred taxonomy on this test set is achieved after adding 30,000 novel hyponyms, achieving an 23% relative improvement in F-score over the WN2.1 classifier.

5 Conclusions

We have presented an algorithm for inducing semantic taxonomies which attempts to globally optimize the entire structure of the taxonomy. Our probabilistic architecture also includes a new model for learning coordinate terms based on (m, n) -cousin classification. The model’s ability to integrate heterogeneous evidence from different classifiers offers a solution to the key problem of choosing the correct word sense to which to attach a new hypernym.

Acknowledgements

Thanks to Christiane Fellbaum, Rajat Raina, Bill MacCartney, and Allison Buckley for useful discussions and assistance annotating data. Rion Snow is supported by an NDSEG Fellowship sponsored by the DOD and AFOSR. This work was supported in part by the Disruptive Technology Office (DTO)’s Advanced Question Answering for Intelligence (AQUAINT) Program.

References

P. Buitelaar, P. Cimiano and B. Magnini. 2005. *Ontology Learning from Text: Methods, Evaluation and Applications*. Volume 123 *Frontiers in Artificial Intelligence and Applications*.

S. Caraballo. 2001. *Automatic Acquisition of a Hypernym-Labeled Noun Hierarchy from Text*. Brown University Ph.D. Thesis.

⁸We found that the WordNet 2.1 model achieving the highest F-score used only the first sense of each hyponym, and allowed a maximum distance of 4 edges between each hyponym and its hypernym.

S. Cederberg and D. Widdows. 2003. Using LSA and Noun Coordination Information to Improve the Precision and Recall of Automatic Hyponymy Extraction. *Proc. CoNLL-2003*, pp. 111–118.

T. Chklovski and P. Pantel. 2004. VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations. *Proc. EMNLP-2004*.

M. Ciaramita and M. Johnson. 2003. Supersense Tagging of Unknown Nouns in WordNet. *Proc. EMNLP-2003*.

O. Etzioni, M. Cafarella, D. Downey, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. 2005. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. *Artificial Intelligence*, 165(1):91–134.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

R. Girju, A. Badulescu, and D. Moldovan. 2003. Learning Semantic Constraints for the Automatic Discovery of Part-Whole Relations. *Proc. HLT-03*.

M. Fleischman and E. Hovy. 2002. Fine grained classification of named entities. *Proc. COLING-02*.

M. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. *Proc. COLING-92*.

D. Hindle. 1990. Noun classification from predicate-argument structures. *Proc. ACL-90*.

D. Lenat. 1995. CYC: A Large-Scale Investment in Knowledge Infrastructure, *Communications of the ACM*, 38:11, 33–35.

D. Lin. 1998. Dependency-based Evaluation of MINIPAR. *Workshop on the Evaluation of Parsing Systems, Granada, Spain*.

D. Lin, S. Zhao, L. Qin and M. Zhou. 2003. Identifying Synonyms among Distributionally Similar Words. *Proc. IJCAI-03*.

M. Pasca. 2005. Finding Instance Names and Alternative Glosses on the Web: WordNet Reloaded. *CI-Cling 2005*, pp. 280-292.

D. Ravichandran, P. Pantel, and E. Hovy. 2002. Randomized Algorithms and NLP: Using Locality Sensitive Hash Function for High Speed Noun Clustering. *Proc. ACL-2002*.

E. Riloff and J. Shepherd. 1997. A Corpus-Based Approach for Building Semantic Lexicons. *Proc. EMNLP-1997*.

B. Roark and E. Charniak. 1998. Noun-phrase co-occurrence statistics for semi-automatic-semantic lexicon construction. *Proc. ACL-1998*.

R. Snow, D. Jurafsky, and A. Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. *NIPS 2005*.

P. Turney, M. Littman, J. Bigham, and V. Shnayder. 2003. Combining independent modules to solve multiple-choice synonym and analogy problems. *Proc. RANLP-2003*, pp. 482–489.

Names and Similarities on the Web: Fact Extraction in the Fast Lane

Marius Paşca

Google Inc.
Mountain View, CA 94043
mars@google.com

Dekang Lin

Google Inc.
Mountain View, CA 94043
lindek@google.com

Jeffrey Bigham*

University of Washington
Seattle, WA 98195
jbigham@cs.washington.edu

Andrei Lifchits*

University of British Columbia
Vancouver, BC V6T 1Z4
alifchit@cs.ubc.ca

Alpa Jain*

Columbia University
New York, NY 10027
alpa@cs.columbia.edu

Abstract

In a new approach to large-scale extraction of facts from unstructured text, distributional similarities become an integral part of both the iterative acquisition of high-coverage contextual extraction patterns, and the validation and ranking of candidate facts. The evaluation measures the quality and coverage of facts extracted from one hundred million Web documents, starting from ten seed facts and using no additional knowledge, lexicons or complex tools.

1 Introduction

1.1 Background

The potential impact of structured fact repositories containing billions of relations among named entities on Web search is enormous. They enable the pursuit of new search paradigms, the processing of database-like queries, and alternative methods of presenting search results. The preparation of exhaustive lists of hand-written extraction rules is impractical given the need for domain-independent extraction of many types of facts from unstructured text. In contrast, the idea of bootstrapping for relation and information extraction was first proposed in (Riloff and Jones, 1999), and successfully applied to the construction of semantic lexicons (Thelen and Riloff, 2002), named entity recognition (Collins and Singer, 1999), extraction of binary relations (Agichtein and Gravano, 2000), and acquisition of structured data for tasks such as Question Answering (Lita and Carbonell, 2004; Fleischman et al., 2003). In the context of fact extraction, the resulting iterative acquisition

framework starts from a small set of seed facts, finds contextual patterns that extract the seed facts from the underlying text collection, identifies a larger set of candidate facts that are extracted by the patterns, and adds the best candidate facts to the previous seed set.

1.2 Contributions

Figure 1 describes an architecture geared towards large-scale fact extraction. The architecture is similar to other instances of bootstrapping for information extraction. The main processing stages are the acquisition of contextual extraction patterns given the seed facts, acquisition of candidate facts given the extraction patterns, scoring and ranking of the patterns, and scoring and ranking of the candidate facts, a subset of which is added to the seed set of the next round.

Within the existing iterative acquisition framework, our first contribution is a method for automatically generating generalized contextual extraction patterns, based on dynamically-computed classes of similar words. Traditionally, the acquisition of contextual extraction patterns requires hundreds or thousands of consecutive iterations over the entire text collection (Lita and Carbonell, 2004), often using relatively expensive or restrictive tools such as shallow syntactic parsers (Riloff and Jones, 1999; Thelen and Riloff, 2002) or named entity recognizers (Agichtein and Gravano, 2000). Comparatively, generalized extraction patterns achieve exponentially higher coverage in early iterations. The extraction of large sets of candidate facts opens the possibility of fast-growth iterative extraction, as opposed to the de-facto strategy of conservatively growing the seed set by as few as five items (Thelen and Riloff, 2002) after each iteration.

*Work done during internships at Google Inc.

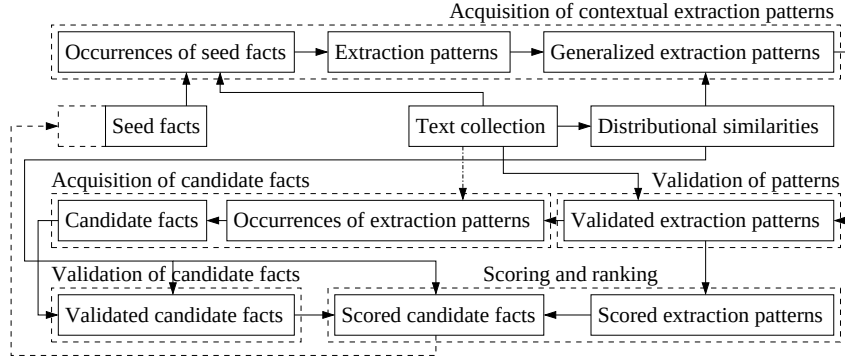


Figure 1: Large-scale fact extraction architecture

The second contribution of the paper is a method for domain-independent validation and ranking of candidate facts, based on a similarity measure of each candidate fact relative to the set of seed facts. Whereas previous studies assume clean text collections such as news corpora (Thelen and Riloff, 2002; Agichtein and Gravano, 2000; Hasegawa et al., 2004), the validation is essential for low-quality sets of candidate facts collected from noisy Web documents. Without it, the addition of spurious candidate facts to the seed set would result in a quick divergence of the iterative acquisition towards irrelevant information (Agichtein and Gravano, 2000). Furthermore, the finer-grained ranking induced by similarities is necessary in fast-growth iterative acquisition, whereas previously proposed ranking criteria (Thelen and Riloff, 2002; Lita and Carbonell, 2004) are implicitly designed for slow growth of the seed set.

2 Similarities for Pattern Acquisition

2.1 Generalization via Word Similarities

The extraction patterns are acquired by matching the pairs of phrases from the seed set into document sentences. The patterns consist of contiguous sequences of sentence terms, but otherwise differ from the types of patterns proposed in earlier work in two respects. First, the terms of a pattern are either regular words or, for higher generality, any word from a class of similar words. Second, the amount of textual context encoded in a pattern is limited to the sequence of terms between (i.e., infix) the pair of phrases from a seed fact that could be matched in a document sentence, thus excluding any context to the left (i.e., prefix) and to the right (i.e., postfix) of the seed.

The pattern shown at the top of Figure 2, which

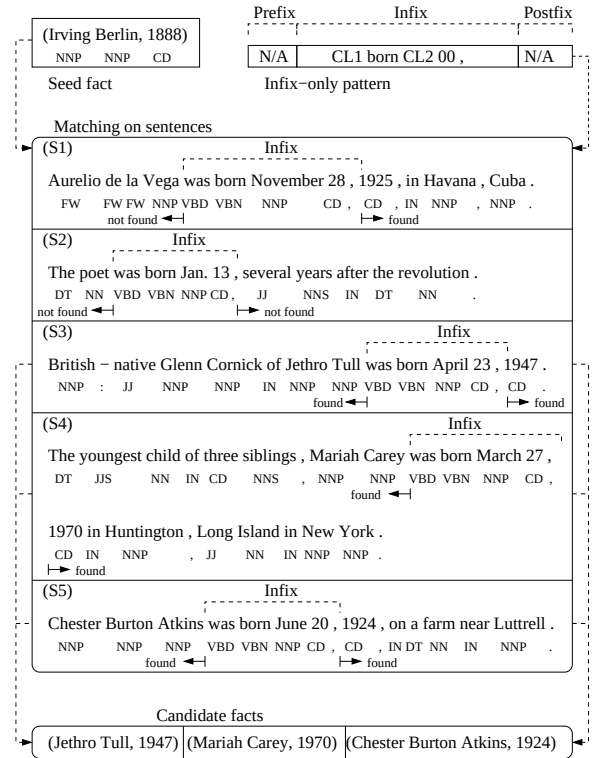


Figure 2: Extraction via infix-only patterns

contains the sequence [CL1 born CL2 00 .], illustrates the use of classes of distributionally similar words within extraction patterns. The first word class in the sequence, CL1, consists of words such as {was, is, could}, whereas the second class includes {February, April, June, Aug., November} and other similar words. The classes of words are computed on the fly over all sequences of terms in the extracted patterns, on top of a large set of pairwise similarities among words (Lin, 1998) extracted in advance from around 50 million news articles indexed by the Google search engine over three years. All digits in both patterns and sentences are replaced with a common marker, such

that any two numerical values with the same number of digits will overlap during matching.

Many methods have been proposed to compute distributional similarity between words, e.g., (Hindle, 1990), (Pereira et al., 1993), (Grefenstette, 1994) and (Lin, 1998). Almost all of the methods represent a word by a feature vector, where each feature corresponds to a type of context in which the word appeared. They differ in how the feature vectors are constructed and how the similarity between two feature vectors is computed.

In our approach, we define the features of a word w to be the set of words that occurred within a small window of w in a large corpus. The context window of an instance of w consists of the closest non-stopword on each side of w and the stopwords in between. The value of a feature w' is defined as the pointwise mutual information between w' and w : $\text{PMI}(w', w) = -\log(\frac{P(w, w')}{P(w)P(w')})$. The similarity between two different words w_1 and w_2 , $S(w_1, w_2)$, is then computed as the cosine of the angle between their feature vectors.

While the previous approaches to distributional similarity have only applied to words, we applied the same technique to proper names as well as words. The following are some example similar words and phrases with their similarities, as obtained from the Google News corpus:

- Carey: Higgins 0.39, Lambert 0.39, Payne 0.38, Kelley 0.38, Hayes 0.38, Goodwin 0.38, Griffin 0.38, Cummings 0.38, Hansen 0.38, Williamson 0.38, Peters 0.38, Walsh 0.38, Burke 0.38, Boyd 0.38, Andrews 0.38, Cunningham 0.38, Freeman 0.37, Stephens 0.37, Flynn 0.37, Ellis 0.37, Bowers 0.37, Bennett 0.37, Matthews 0.37, Johnston 0.37, Richards 0.37, Hoffman 0.37, Schultz 0.37, Steele 0.37, Dunn 0.37, Rowe 0.37, Swanson 0.37, Hawkins 0.37, Wheeler 0.37, Porter 0.37, Watkins 0.37, Meyer 0.37 [...];

- Mariah Carey: Shania Twain 0.38, Christina Aguilera 0.35, Sheryl Crow 0.35, Britney Spears 0.33, Celine Dion 0.33, Whitney Houston 0.32, Justin Timberlake 0.32, Beyonce Knowles 0.32, Bruce Springsteen 0.30, Faith Hill 0.30, LeAnn Rimes 0.30, Missy Elliott 0.30, Aretha Franklin 0.29, Jennifer Lopez 0.29, Gloria Estefan 0.29, Elton John 0.29, Norah Jones 0.29, Missy Elliot 0.29, Alicia Keys 0.29, Avril Lavigne 0.29, Kid Rock 0.28, Janet Jackson 0.28, Kylie Minogue 0.28, Beyonce 0.27, Enrique Iglesias 0.27, Michelle Branch 0.27 [...];

- Jethro Tull: Motley Crue 0.28, Black Crowes 0.26, Pearl Jam 0.26, Silverchair 0.26, Black Sabbath 0.26, Doobie Brothers 0.26, Judas Priest 0.26, Van Halen 0.25, Midnight Oil 0.25, Pere Ubu 0.24, Black Flag 0.24, Godsmack 0.24, Grateful Dead 0.24, Grand Funk Railroad 0.24, Smashing Pumpkins 0.24, Led Zeppelin 0.24, Aerosmith 0.24, Limp Bizkit 0.24, Counting Crows 0.24, Echo And The Bunnymen 0.24, Cold Chisel 0.24, Thin Lizzy 0.24 [...].

To our knowledge, the only previous study that embeds similarities into the acquisition of extraction patterns is (Stevenson and Greenwood, 2005). The authors present a method for computing pairwise similarity scores among large sets of potential syntactic (subject-verb-object) patterns, to detect centroids of mutually similar patterns. By assuming the syntactic parsing of the underlying text collection to generate the potential patterns in the first place, the method is impractical on Web-scale collections. Two patterns, e.g. *chairman-resign* and *CEO-quit*, are similar to each other if their components are present in an external hand-built ontology (i.e., WordNet), and the similarity among the components is high over the ontology. Since general-purpose ontologies, and WordNet in particular, contain many classes (e.g., *chairman* and *CEO*) but very few instances such as *Osasuna*, *Crewe* etc., the patterns containing an instance rather than a class will not be found to be similar to one another. In comparison, the classes and instances are equally useful in our method for generalizing patterns for fact extraction. We merge basic patterns into generalized patterns, regardless of whether the similar words belong, as classes or instances, in any external ontology.

2.2 Generalization via Infix-Only Patterns

By giving up the contextual constraints imposed by the prefix and postfix, infix-only patterns represent the most aggressive type of extraction patterns that still use contiguous sequences of terms. In the absence of the prefix and postfix, the outer boundaries of the fact are computed separately for the beginning of the first (left) and end of the second (right) phrases of the candidate fact. For generality, the computation relies only on the part-of-speech tags of the current seed set. Starting forward from the right extremity of the infix, we collect a growing sequence of terms whose part-of-speech tags are $[P_1+ P_2+ \dots P_n+]$, where the

notation P_i+ represents one or more consecutive occurrences of the part-of-speech tag P_i . The sequence $[P_1 P_2 \dots P_n]$ must be exactly the sequence of part of speech tags from the right side of one of the seed facts. The point where the sequence cannot be grown anymore defines the boundary of the fact. A similar procedure is applied backwards, starting from the left extremity of the infix. An infix-only pattern produces a candidate fact from a sentence only if an acceptable sequence is found to the left and also to the right of the infix.

Figure 2 illustrates the process on the infix-only pattern mentioned earlier, and one seed fact. The part-of-speech tags for the seed fact are [NNP NNP] and [CD] for the left and right sides respectively. The infix occurs in all sentences. However, the matching of the part-of-speech tags of the sentence sequences to the left and right of the infix, against the part-of-speech tags of the seed fact, only succeeds for the last three sentences. It fails for the first sentence S_1 to the left of the infix, because [$\dots$ NNP] (for *Vega*) does not match [NNP NNP]. It also fails for the second sentence S_2 to both the left and the right side of the infix, since [$\dots$ NN] (for *poet*) does not match [NNP NNP], and [JJ $\dots$] (for *several*) does not match [CD].

3 Similarities for Validation and Ranking

3.1 Revisiting Standard Ranking Criteria

Because some of the acquired extraction patterns are too generic or wrong, all approaches to iterative acquisition place a strong emphasis on the choice of criteria for ranking. Previous literature quasi-unanimously assesses the quality of each candidate fact based on the number and quality of the patterns that extract the candidate fact (more is better); and the number of seed facts extracted by the same patterns (again, more is better) (Agichtein and Gravano, 2000; Thelen and Riloff, 2002; Lita and Carbonell, 2004). However, our experiments using many variations of previously proposed scoring functions suggest that they have limited applicability in large-scale fact extraction, for two main reasons. The first is that it is impractical to perform hundreds of acquisition iterations on terabytes of text. Instead, one needs to grow the seed set aggressively in each iteration. Previous scoring functions were implicitly designed for cautious acquisition strategies (Collins and Singer, 1999), which expand the seed set very slowly across consecutive iterations.

In that case, it makes sense to single out a small number of best candidates, among the other available candidates. Comparatively, when 10,000 candidate facts or more need to be added to a seed set of 10 seeds as early as after the first iteration, it is difficult to distinguish the quality of extraction patterns based, for instance, only on the percentage of the seed set that they extract. The second reason is the noisy nature of the Web. A substantial number of factors can and will concur towards the worst-case extraction scenarios on the Web. Patterns of apparently high quality turn out to produce a large quantity of erroneous “facts” such as (*A-League*, 1997), but also the more interesting (*Jethro Tull*, 1947) as shown earlier in Figure 2, or (*Web Site David*, 1960) or (*New York*, 1831). As for extraction patterns of average or lower quality, they will naturally lead to even more spurious extractions.

3.2 Ranking of Extraction Patterns

The intuition behind our criteria for ranking generalized pattern is that patterns of higher precision tend to contain words that are indicative of the relation being mined. Thus, a pattern is more likely to produce good candidate facts if its infix contains the words *language* or *spoken* if extracting Language-SpokenIn-Country facts, or the word *capital* if extracting City-CapitalOf-Country relations. In each acquisition iteration, the scoring of patterns is a two-pass procedure. The first pass computes the normalized frequencies of all words excluding stopwords, over the entire set of extraction patterns. The computation applies separately to the prefix, infix and postfix of the patterns. In the second pass, the score of an extraction pattern is determined by the words with the highest frequency score in its prefix, infix and postfix, as computed in the first pass and adjusted for the relative distance to the start and end of the infix.

3.3 Ranking of Candidate Facts

Figure 3 introduces a new scheme for assessing the quality of the candidate facts, based on the computation of similarity scores for each candidate relative to the set of seed facts. A candidate fact, e.g., (*Richard Steele*, 1672), is similar to the seed set if both its phrases, i.e., *Richard Steele* and 1672, are similar to the corresponding phrases (*John Lennon* or *Stephen Foster* in the case of *Richard Steele*) from the seed facts. For a phrase of a candidate fact to be assigned a non-default (non-minimum)

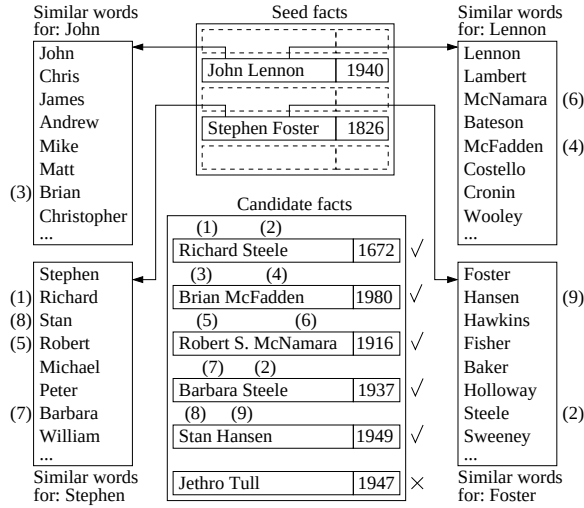


Figure 3: The role of similarities in estimating the quality of candidate facts

similarity score, the words at its extremities must be similar to one or more words situated at the same positions in the seed facts. This is the case for the first five candidate facts in Figure 3. For example, the first word *Richard* from one of the candidate facts is similar to the first word *John* from one of the seed facts. Concurrently, the last word *Steele* from the same phrase is similar to *Foster* from another seed fact. Therefore *Robert Foster* is similar to the seed facts. The score of a phrase containing N words is:

$$\begin{cases} C_1 + \sum_{i=1}^N \log(1 + Sim_i) & , \text{if } Sim_{1,N} > 0 \\ C_2 & , \text{otherwise.} \end{cases}$$

where Sim_i is the similarity of the component word at position i in the phrase, and C_1 and C_2 are scaling constants such that $C_2 \ll C_1$. Thus, the similarity score of a candidate fact aggregates individual word-to-word similarity scores, for the left side and then for the right side of a candidate fact. In turn, the similarity score of a component word Sim_i is higher if: a) the computed word-to-word similarity scores are higher relative to words at the same position i in the seeds; and b) the component word is similar to words from more than one seed fact.

The similarity scores are one of a linear combination of features that induce a ranking over the candidate facts. Three other domain-independent features contribute to the final ranking: a) a phrase completeness score computed statistically over the entire set of candidate facts, which demotes candidate facts if any of their two sides is likely to be incomplete (e.g., *Mary Lou* vs. *Mary Lou Retton*,

or *John F.* vs. *John F. Kennedy*); b) the average PageRank value over all documents from which the candidate fact is extracted; and c) the pattern-based scores of the candidate fact. The latter feature converts the scores of the patterns extracting the candidate fact into a score for the candidate fact. For this purpose, it considers a fixed-length window of words around each match of a candidate fact in some sentence from the text collection. This is equivalent to analyzing all sentence contexts from which a candidate fact can be extracted. For each window, the word with the highest frequency score, as computed in the first pass of the procedure for scoring the patterns, determines the score of the candidate fact in that context. The overall pattern-based score of a candidate fact is the sum of the scores over all its contexts of occurrence, normalized by the frequency of occurrence of the candidate over all sentences.

Besides inducing a ranking over the candidate facts, the similarity scores also serve as a validation filter over the candidate facts. Indeed, any candidates that are not similar to the seed set can be filtered out. For instance, the elimination of (*Jethro Tull*, 1947) is a side effect of verifying that *Tull* is not similar to any of the last-position words from phrases in the seed set.

4 Evaluation

4.1 Data

The source text collection consists of three chunks W_1 , W_2 , W_3 of approximately 100 million documents each. The documents are part of a larger snapshot of the Web taken in 2003 by the Google search engine. All documents are in English. The textual portion of the documents is cleaned of Html, tokenized, split into sentences and part-of-speech tagged using the TnT tagger (Brants, 2000).

The evaluation involves facts of type Person-BornIn-Year. The reasons behind the choice of this particular type are threefold. First, many Person-BornIn-Year facts are probably available on the Web (as opposed to, e.g., City-CapitalOf-Country facts), to allow for a good stress test for large-scale extraction. Second, either side of the facts (Person and Year) may be involved in many other types of facts, such that the extraction would easily diverge unless it performs correctly. Third, the phrases from one side (Person) have an utility in their own right, for lexicon

Table 1: Set of seed Person-BornIn-Year facts

Name	Year	Name	Year
Paul McCartney	1942	John Lennon	1940
Vincenzo Bellini	1801	Stephen Foster	1826
Hoagy Carmichael	1899	Irving Berlin	1888
Johann Sebastian Bach	1685	Bela Bartok	1881
Ludwig van Beethoven	1770	Bob Dylan	1941

construction or detection of person names.

The Person-BornIn-Year type is specified through an initial set of 10 seed facts shown in Table 1. Similarly to source documents, the facts are also part-of-speech tagged.

4.2 System Settings

In each iteration, the case-insensitive matching of the current set of seed facts onto the sentences produces basic patterns. The patterns are converted into generalized patterns. The length of the infix may vary between 1 and 6 words. Potential patterns are discarded if the infix contains only stop-words.

When a pattern is retained, it is used as an infix-only pattern, and allowed to generate at most 600,000 candidate facts. At the end of an iteration, approximately one third of the validated candidate facts are added to the current seed set. Consequently, the acquisition expands the initial seed set of 10 facts to 100,000 facts (after iteration 1) and then to one million facts (after iteration 2) using chunk W_1 .

4.3 Precision

A separate baseline run extracts candidate facts from the text collection following the traditional iterative acquisition approach. Pattern generalization is disabled, and the ranking of patterns and facts follows strictly the criteria and scoring functions from (Thelen and Riloff, 2002), which are also used in slightly different form in (Lita and Carbonell, 2004) and (Agichtein and Gravano, 2000). The theoretical option of running thousands of iterations over the text collection is not viable, since it would imply a non-justifiable expense of our computational resources. As a more realistic compromise over overly-cautious acquisition, the baseline run retains as many of the top candidate facts as the size of the current seed, whereas (Thelen and Riloff, 2002) only add the top five candidate facts to the seed set after each iteration. The evaluation considers all 80, a sample of the 320, and another sample of the 10,240 facts

retained after iterations 3, 5 and 10 respectively. The correctness assessment of each fact consists in manually finding some Web page that contains clear evidence that the fact is correct. If no such page exists, the fact is marked as incorrect. The corresponding precision values after the three iterations are 91.2%, 83.8% and 72.9%.

For the purpose of evaluating the precision of our system, we select a sample of facts from the entire list of one million facts extracted from chunk W_1 , ranked in decreasing order of their computed scores. The sample is generated automatically from the top of the list to the bottom, by retaining a fact and skipping the following consecutive N facts, where N is incremented at each step. The resulting list, which preserves the relative order of the facts, contains 1414 facts. The 115 facts for which a Web search engine does not return any documents, when the name (as a phrase) and the year are submitted together in a conjunctive query, are discarded from the sample of 1414 facts. In those cases, the facts were acquired from the 2003 snapshot of the Web, but queries are submitted to a search engine with access to current Web documents, hence the difference when some of the 2003 documents are no longer available or indexable.

Based on the sample set, the average precision of the list of one million facts extracted from chunk W_1 is 98.5% over the top 1/100 of the list, 93.1% over the top half of the list, and 88.3% over the entire list of one million facts. Table 2 shows examples of erroneous facts extracted from chunk W_1 . Causes of errors include incorrect approximations of the name boundaries (e.g., *Alma* in *Alma Theresa Rausch* is incorrectly tagged as an adjective), and selection of the wrong year as birth year (e.g., for *Henry Lumbar*).

In the case of famous people, the extracted facts tend to capture the correct birth year for several variations of the names, as shown in Table 3. Conversely, it is not necessary that a fact occur with high frequency in order for it to be extracted, which is an advantage over previous approaches that rely strongly on redundancy (cf. (Cafarella et al., 2005)). Table 4 illustrates a few of the correctly extracted facts that occur rarely on the Web.

4.4 Recall

In contrast to the assessment of precision, recall can be evaluated automatically, based on external

Table 2: Incorrect facts extracted from the Web

Spurious Fact	Context in Source Sentence
(Theresa Rausch, 1912)	Alma Theresa Rausch was born on 9 March 1912
(Henry Lumbar, 1937)	Henry Lumbar was born 1861 and died 1937
(Concepcion Paxety, 1817)	Maria de la Concepcion Paxety b. 08 Dec. 1817 St. Aug., FL.
(Mae Yaeger, 1872)	Ella May/Mae Yaeger was born 20 May 1872 in Mt.
(Charles Whatley, 1821)	Long, Charles Whatley b. 16 FEB 1821 d. 29 AUG
(HOLT George W. Holt, 1845)	HOLT (new line) George W. Holt was born in Alabama in 1845
(David Morrish Canadian, 1953)	David Morrish (new line) Canadian, b. 1953
(Mary Ann, 1838)	had a daughter, Mary Ann, who was born in Tennessee in 1838
(Mrs. Blackmore, 1918)	Mrs. Blackmore was born April 28, 1918, in Labaddiey

Table 3: Birth years extracted for both pseudonyms and corresponding real names

Pseudonym	Real Name	Year
Gloria Estefan	Gloria Fajardo	1957
Nicolas Cage	Nicolas Kim Coppola	1964
Ozzy Osbourne	John Osbourne	1948
Ringo Starr	Richard Starkey	1940
Tina Turner	Anna Bullock	1939
Tom Cruise	Thomas Cruise Mapother IV	1962
Woody Allen	Allen Stewart Konigsberg	1935

lists of birth dates of various people. We start by collecting two gold standard sets of facts. The first set is a random set of 609 actors and their birth years from a Web compilation (Gold_A). The second set is derived from the set of questions used in the Question Answering track (Voorhees and Tice, 2000) of the Text REtrieval Conference from 1999 through 2002. Each question asking for the birth date of a person (e.g., “*What year was Robert Frost born?*”) results in a pair containing the person’s name and the birth year specified in the answer keys. Thus, the second gold standard set contains 17 pairs of people and their birth years (Gold_T). Table 5 shows examples of facts in each of the gold standard sets.

Table 6 shows two types of recall scores computed against the gold standard sets. The recall scores over $\cap$ Gold take into consideration only the set of person names from the gold standard with some extracted year(s). More precisely, given that some years were extracted for a person name, it verifies whether they include the year specified in the gold standard for that person name. Comparatively, the recall score denoted *AllGold* is com-

Table 4: Extracted facts that occur infrequently

Fact	Source Domain
(Irvine J Forcier, 1912)	geocities.com
(Marie Louise Azelie Chabert, 1861)	vienici.com
(Jacob Shalles, 1750)	selfhost.com
(Robert Chester Claggett, 1898)	rootsweb.com
(Charlotte Mollett, 1843)	rootsweb.com
(Nora Elizabeth Curran, 1979)	jimtravis.com

Table 5: Composition of gold standard sets

Gold Set	Composition and <i>Examples of Facts</i>	
Gold _A	Actors (Web compilation)	Nr. facts: 609
	(Andie MacDowell, 1958), (Doris Day, 1924), (Diahann Carroll, 1935)	
Gold _T	People (TREC QA track)	Nr. facts: 17
	(Davy Crockett, 1786), (Julius Caesar, 100 B.C.), (King Louis XIV, 1638)	

puted over the entire set of names from the gold standard.

For the Gold_A set, the size of the $\cap$ Gold set of person names changes little when the facts are extracted from chunk W_1 vs. W_2 vs. W_3 . The recall scores over $\cap$ Gold exhibit little variation from one Web chunk to another, whereas the *AllGold* score is slightly higher on the W_3 chunk, probably due to a higher number of documents that are relevant to the extraction task. When the facts are extracted from a combination of two or three of the available Web chunks, the recall scores computed over *AllGold* are significantly higher as the size of the $\cap$ Gold set increases. In comparison, the recall scores over the growing $\cap$ Gold set increases slightly with larger evaluation sets. The highest value of the recall score for Gold_A is 89.9% over the $\cap$ Gold set, and 70.7% over *AllGold*. The smaller size of the second gold standard set, Gold_T, explains the higher variation of the values shown in the lower portion of Table 6.

4.5 Comparison to Previous Results

Another recent approach specifically addresses the problem of extracting facts from a similarly-sized collection of Web documents. In (Cafarella et al., 2005), manually-prepared extraction rules are applied to a collection of 60 million Web documents to extract entities of types Company and Country, as well as facts of type Person-CeoOf-Company and City-CapitalOf-Country. Based on manual evaluation of precision and recall, a total of 23,128 company names are extracted at precision of 80%; the number decreases to 1,116 at precision of 90%. In addition, 2,402 Person-CeoOf-Company facts

Table 6: Automatic evaluation of recall, over two gold standard sets Gold_A (609 person names) and Gold_T (17 person names)

Gold Set	Input Data (Web Chunk)	Recall (%)	
		$\cap$ Gold	AllGold
Gold _A	W ₁	86.4	49.4
	W ₂	85.0	50.5
	W ₃	86.3	54.1
	W ₁ +W ₂	88.5	64.5
	W ₁ +W ₂ +W ₃	89.9	70.7
Gold _T	W ₁	81.8	52.9
	W ₂	90.0	52.9
	W ₃	100.0	64.7
	W ₁ +W ₂	81.8	52.9
	W ₁ +W ₂ +W ₃	91.6	64.7

are extracted at precision 80%. The recall value is 80% at precision 90%. Recall is evaluated against the set of company names extracted by the system, rather than an external gold standard with pairs of a CEO and a company name. As such, the resulting metric for evaluating recall used in (Cafarella et al., 2005) is somewhat similar to, though more relaxed than, the recall score over the $\cap$ Gold set introduced in the previous section.

5 Conclusion

The combination of generalized extraction patterns and similarity-driven ranking criteria results in a fast-growth iterative approach for large-scale fact extraction. From 10 Person-BornIn-Year facts and no additional knowledge, a set of one million facts of the same type is extracted from a collection of 100 million Web documents of arbitrary quality, with a precision around 90%. This corresponds to a growth ratio of 100,000:1 between the size of the extracted set of facts and the size of the initial set of seed facts. To our knowledge, the growth ratio and the number of extracted facts are several orders of magnitude higher than in any of the previous studies on fact extraction based on either hand-written extraction rules (Cafarella et al., 2005), or bootstrapping for relation and information extraction (Agichtein and Gravano, 2000; Lita and Carbonell, 2004). The next research steps converge towards the automatic construction of a searchable repository containing billions of facts regarding people.

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plaintext collections. In *Proceedings*

of the 5th ACM International Conference on Digital Libraries (DL-00), pages 85–94, San Antonio, Texas.

- T. Brants. 2000. TnT - a statistical part of speech tagger. In *Proceedings of the 6th Conference on Applied Natural Language Processing (ANLP-00)*, pages 224–231, Seattle, Washington.
- M. Cafarella, D. Downey, S. Soderland, and O. Etzioni. 2005. KnowItNow: Fast, scalable information extraction from the web. In *Proceedings of the Human Language Technology Conference (HLT-EMNLP-05)*, pages 563–570, Vancouver, Canada.
- M. Collins and Y. Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of the 1999 Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, pages 189–196, College Park, Maryland.
- M. Fleischman, E. Hovy, and A. Echihiabi. 2003. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 1–7, Sapporo, Japan.
- G. Grefenstette. 1994. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, Boston, Massachusetts.
- T. Hasegawa, S. Sekine, and R. Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 415–422, Barcelona, Spain.
- D. Hindle. 1990. Noun classification from predicate-argument structures. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics (ACL-90)*, pages 268–275, Pittsburgh, Pennsylvania.
- D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics and the 36th Annual Meeting of the Association for Computational Linguistics (COLING-ACL-98)*, pages 768–774, Montreal, Quebec.
- L. Lita and J. Carbonell. 2004. Instance-based question answering: A data driven approach. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-04)*, pages 396–403, Barcelona, Spain.
- F. Pereira, N. Tishby, and L. Lee. 1993. Distributional clustering of english words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL-93)*, pages 183–190, Columbus, Ohio.
- 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 (AAAI-99)*, pages 474–479, Orlando, Florida.
- M. Stevenson and M. Greenwood. 2005. A semantic approach to IE pattern induction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 379–386, Ann Arbor, Michigan.
- M. Thelen and E. Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-02)*, pages 214–221, Philadelphia, Pennsylvania.
- E.M. Voorhees and D.M. Tice. 2000. Building a question-answering test collection. In *Proceedings of the 23rd International Conference on Research and Development in Information Retrieval (SIGIR-00)*, pages 200–207, Athens, Greece.

Weakly Supervised Named Entity Transliteration and Discovery from Multilingual Comparable Corpora

Alexandre Klementiev Dan Roth

Dept. of Computer Science
University of Illinois
Urbana, IL 61801
{klementi,danr}@uiuc.edu

Abstract

Named Entity recognition (NER) is an important part of many natural language processing tasks. Current approaches often employ machine learning techniques and require supervised data. However, many languages lack such resources. This paper presents an (almost) unsupervised learning algorithm for automatic discovery of Named Entities (NEs) in a resource free language, given a bilingual corpora in which it is weakly temporally aligned with a resource rich language. NEs have similar time distributions across such corpora, and often some of the tokens in a multi-word NE are transliterated. We develop an algorithm that exploits both observations iteratively. The algorithm makes use of a new, frequency based, metric for time distributions and a resource free discriminative approach to transliteration. Seeded with a small number of transliteration pairs, our algorithm discovers multi-word NEs, and takes advantage of a dictionary (if one exists) to account for translated or partially translated NEs. We evaluate the algorithm on an English-Russian corpus, and show high level of NEs discovery in Russian.

1 Introduction

Named Entity recognition has been getting much attention in NLP research in recent years, since it is seen as significant component of higher level NLP tasks such as information distillation and question answering. Most successful approaches to NER employ machine learning techniques, which require supervised training data. However,

for many languages, these resources do not exist. Moreover, it is often difficult to find experts in these languages both for the expensive annotation effort and even for language specific clues. On the other hand, comparable multilingual data (such as multilingual news streams) are becoming increasingly available (see section 4).

In this work, we make two independent observations about Named Entities encountered in such corpora, and use them to develop an algorithm that extracts pairs of NEs across languages. Specifically, given a bilingual corpora that is weakly temporally aligned, and a capability to annotate the text in one of the languages with NEs, our algorithm identifies the corresponding NEs in the second language text, and annotates them with the appropriate type, as in the source text.

The first observation is that NEs in one language in such corpora tend to co-occur with their counterparts in the other. E.g., Figure 1 shows a histogram of the number of occurrences of the word *Hussein* and its Russian transliteration in our bilingual news corpus spanning years 2001 through late 2005. One can see several common peaks in the two histograms, largest one being around the time of the beginning of the war in Iraq. The word *Russia*, on the other hand, has a distinctly different temporal signature. We can exploit such weak synchronicity of NEs across languages to associate them. In order to score a pair of entities across languages, we compute the similarity of their time distributions.

The second observation is that NEs often contain or are entirely made up of words that are phonetically transliterated or have a common etymological origin across languages (e.g. *parliament* in English and *парламент*, its Russian translation), and thus are phonetically similar. Figure 2 shows

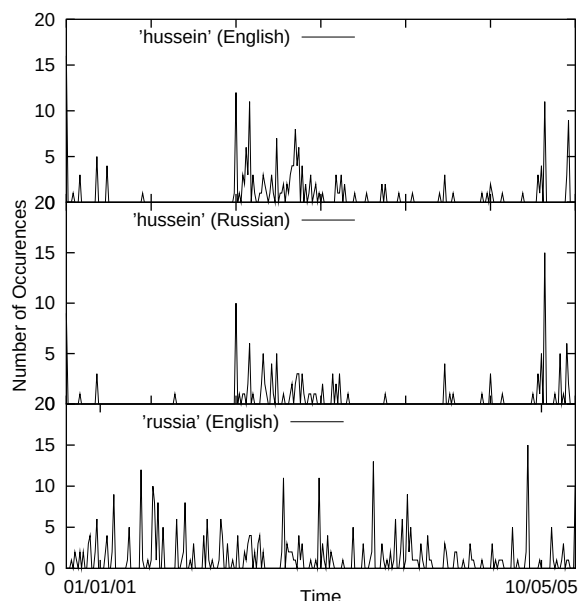


Figure 1: Temporal histograms for *Hussein* (top), its Russian transliteration (middle), and of the word *Russia* (bottom).

an example list of NEs and their possible Russian transliterations.

Approaches that attempt to use these two characteristics separately to identify NEs across languages would have significant shortcomings. Transliteration based approaches require a good model, typically handcrafted or trained on a clean set of transliteration pairs. On the other hand, time sequence similarity based approaches would incorrectly match words which happen to have similar time signatures (e.g., *Taliban* and *Afghanistan* in recent news).

We introduce an algorithm we call *co-ranking* which exploits these observations simultaneously to match NEs on one side of the bilingual corpus to their counterparts on the other. We use a Discrete Fourier Transform (Arfken, 1985) based metric for computing similarity of time distributions, and show that it has significant advantages over other metrics traditionally used. We score NEs similarity with a linear transliteration model.

We first train a transliteration model on single-word NEs. During training, for a given NE in one language, the current model chooses a list of top ranked transliteration candidates in another language. Time sequence scoring is then used to re-rank the list and choose the candidate best temporally aligned with the NE. Pairs of NEs and the best candidates are then used to iteratively train the

English NE	Russian NE
lilic	лилич
fletcher	флетчер
bradford	брэдфорд
isabel	изабель
hoffmann	гофман
kathmandu	катманду

Figure 2: Example English NEs and their transliterated Russian counterparts.

transliteration model.

Once the model is trained, NE discovery proceeds as follows. For a given NE, transliteration model selects a candidate list for each constituent word. If a dictionary is available, each candidate list is augmented with translations (if they exist). Translations will be the correct choice for some NE words (e.g. for *queen* in *Queen Victoria*), and transliterations for others (e.g. *Bush* in *Steven Bush*). We expect temporal sequence alignment to resolve many of such ambiguities. It is used to select the best translation/transliteration candidate from each word's candidate set, which are then merged into a possible NE in the other language. Finally, we verify that the NE is actually contained in the target corpus.

A major challenge inherent in discovering transliterated NEs is the fact that a single entity may be represented by multiple transliteration strings. One reason is language morphology. For example, in Russian, depending on a case being used, the same noun may appear with various endings. Another reason is the lack of transliteration standards. Again, in Russian, several possible transliterations of an English entity may be acceptable, as long as they are phonetically similar to the source.

Thus, in order to rely on the time sequences we obtain, we need to be able to group variants of the same NE into an equivalence class, and collect their aggregate mention counts. We would then score time sequences of these equivalence classes. For instance, we would like to count the aggregate number of occurrences of $\{Herzegovina, Hercegovina\}$ on the English side in order to map it accurately to the equivalence class of that NE's variants we may see on the Russian side of our corpus (e.g. $\{Герцеговина, Герцеговину, Герцеговины, Герцеговиной\}$).

One of the objectives for this work was to use as

little of the knowledge of both languages as possible. In order to effectively rely on the quality of time sequence scoring, we used a simple, knowledge poor approach to group NE variants for the languages of our corpus (see 3.2.1).

In the rest of the paper, whenever we refer to a Named Entity or an NE constituent word, we imply its equivalence class. Note that although we expect that better use of language specific knowledge would improve the results, it would defeat one of the goals of this work.

2 Previous work

There has been other work to automatically discover NE with minimal supervision. Both (Cucerzan and Yarowsky, 1999) and (Collins and Singer, 1999) present algorithms to obtain NEs from untagged corpora. However, they focus on the *classification* stage of *already segmented entities*, and make use of contextual and morphological clues that require knowledge of the language beyond the level we want to assume with respect to the target language.

The use of similarity of time distributions for information extraction, in general, and NE extraction, in particular, is not new. (Hetland, 2004) surveys recent methods for scoring time sequences for similarity. (Shinyama and Sekine, 2004) used the idea to discover NEs, but in a single language, English, across two news sources.

A large amount of previous work exists on transliteration models. Most are *generative* and consider the task of *producing* an appropriate transliteration for a given word, and thus require considerable knowledge of the languages. For example, (AbdulJaleel and Larkey, 2003; Jung et al., 2000) train English-Arabic and English-Korean generative transliteration models, respectively. (Knight and Graehl, 1997) build a generative model for backward transliteration from Japanese to English.

While generative models are often robust, they tend to make independence assumptions that do not hold in data. The discriminative learning framework argued for in (Roth, 1998; Roth, 1999) as an alternative to generative models is now used widely in NLP, even in the context of word alignment (Taskar et al., 2005; Moore, 2005). We make use of it here too, to learn a discriminative transliteration model that requires little knowledge of the target language.

We extend our preliminary work in (Klementiev and Roth, 2006) to discover multi-word Named Entities and to take advantage of a dictionary (if one exists) to handle NEs which are partially or entirely translated. We take advantage of dynamically growing feature space to reduce the number of supervised training examples.

3 Co-Ranking: An Algorithm for NE Discovery

3.1 The algorithm

In essence, the algorithm we present uses temporal alignment as a supervision signal to iteratively train a transliteration model. On each iteration, it selects a list of top ranked transliteration candidates for each NE according to the current model (line 6). It then uses temporal alignment (with thresholding) to re-rank the list and select the best transliteration candidate for the next round of training (lines 8, and 9).

Once the training is complete, lines 4 through 10 are executed without thresholding for each constituent NE word. If a dictionary is available, transliteration candidate lists $\mathcal{NE}_{\mathcal{T}}$ on line 6 are augmented with translations. We then combine the best candidates (as chosen on line 8, without thresholding) into complete target language NE. Finally, we discard transliterations which do not actually appear in the target corpus.

Input: Bilingual, comparable corpus $(\mathcal{S}, \mathcal{T})$, set of named entities $\mathcal{NE}_{\mathcal{S}}$ from $\mathcal{S}$, threshold θ Output: Transliteration model $\mathcal{M}$ <pre> 1 Initialize $\mathcal{M}$; 2 $\forall \mathcal{E} \in \mathcal{NE}_{\mathcal{S}}$, collect time distribution $\mathcal{Q}_{\mathcal{ES}}$; 3 repeat 4 $\mathcal{D} \leftarrow \emptyset$; 5 for each $\mathcal{E}_{\mathcal{S}} \in \mathcal{NE}_{\mathcal{S}}$ do 6 Use $\mathcal{M}$ to collect a list of candidates $\mathcal{NE}_{\mathcal{ET}} \in \mathcal{T}$ with high transliteration scores; 7 $\forall \mathcal{E} \in \mathcal{NE}_{\mathcal{ET}}$ collect time distribution $\mathcal{Q}_{\mathcal{ET}}$; 8 Select candidate $\mathcal{E}_{\mathcal{T}} \in \mathcal{NE}_{\mathcal{ET}}$ with the best $\omega = \text{score}(\mathcal{Q}_{\mathcal{ES}}, \mathcal{Q}_{\mathcal{ET}})$; 9 if ω exceeds θ, add tuple $(\mathcal{E}_{\mathcal{S}}, \mathcal{E}_{\mathcal{T}})$ to $\mathcal{D}$; 10 end 11 Use $\mathcal{D}$ to train $\mathcal{M}$; 12 until $\mathcal{D}$ stops changing between iterations ; </pre>

Algorithm 1: Iterative transliteration model training.

3.2 Time sequence generation and matching

In order to generate time sequence for a word, we divide the corpus into a sequence of temporal bins, and count the number of occurrences of the word in each bin. We then normalize the sequence.

We use a method called the F-index (Hetland, 2004) to implement the *score* similarity function on line 8 of the algorithm. We first run a Discrete Fourier Transform on a time sequence to extract its Fourier expansion coefficients. The score of a pair of time sequences is then computed as a Euclidean distance between their expansion coefficient vectors.

3.2.1 Equivalence Classes

As we mentioned in the introduction, an NE may map to more than one transliteration in another language. Identification of the entity's equivalence class of transliterations is important for obtaining its accurate time sequence.

In order to keep to our objective of requiring as little language knowledge as possible, we took a rather simplistic approach for both languages of our corpus. For Russian, two words were considered variants of the same NE if they share a prefix of size five or longer. Each unique word had its own equivalence class for the English side of the corpus, although, in principal, ideas such as in (Li et al., 2004) could be incorporated.

A cumulative distribution was then collected for such equivalence classes.

3.3 Transliteration model

Unlike most of the previous work considering *generative* transliteration models, we take the *discriminative* approach. We train a linear model to decide whether a word $\mathcal{E}_{\mathcal{T}} \in \mathcal{T}$ is a transliteration of an NE $\mathcal{E}_{\mathcal{S}} \in \mathcal{S}$. The words in the pair are partitioned into a set of substrings $s_{\mathcal{S}}$ and $s_{\mathcal{T}}$ up to a particular length (including the empty string $\_$). Couplings of the substrings $(s_{\mathcal{S}}, s_{\mathcal{T}})$ from both sets produce features we use for training. Note that couplings with the empty string represent insertions/omissions.

Consider the following example: $(\mathcal{E}_{\mathcal{S}}, \mathcal{E}_{\mathcal{T}}) = (\text{powell}, \text{pauel})$. We build a feature vector from this example in the following manner:

- First, we split both words into all possible substrings of up to size two:

$$\mathcal{E}_{\mathcal{S}} \rightarrow \{ \_, p, o, w, e, l, l, po, ow, we, el, ll \}$$

$$\mathcal{E}_{\mathcal{T}} \rightarrow \{ \_, p, a, u, e, l, pa, au, ue, el \}$$

- We build a feature vector by coupling substrings from the two sets:

$$((p, \_), (p, a), \dots (w, au), \dots (el, el), \dots (ll, el))$$

We use the observation that transliteration tends to preserve phonetic sequence to limit the number of couplings. For example, we can disallow the coupling of substrings whose starting positions are too far apart: thus, we might not consider a pairing (po, ue) in the above example. In our experiments, we paired substrings if their positions in their respective words differed by -1, 0, or 1.

We use the perceptron (Rosenblatt, 1958) algorithm to train the model. The model activation provides the score we use to select best transliterations on line 6. Our version of perceptron takes variable number of features in its examples; each example is a subset of all features seen so far that are active in the input. As the iterative algorithm observes more data, it discovers and makes use of more features. This model is called the *infinite attribute model* (Blum, 1992) and it follows the perceptron version of SNoW (Roth, 1998).

Positive examples used for iterative training are pairs of NEs and their best temporally aligned (thresholded) transliteration candidates. Negative examples are English non-NEs paired with random Russian words.

4 Experimental Study

We ran experiments using a bilingual comparable English-Russian news corpus we built by crawling a Russian news web site (www.lenta.ru). The site provides loose translations of (and pointers to) the original English texts. We collected pairs of articles spanning from 1/1/2001 through 10/05/2005. The corpus consists of 2,327 documents, with 0-8 documents per day. The corpus is available on our web page at <http://L2R.cs.uiuc.edu/~cogcomp/>. The English side was tagged with a publicly available NER system based on the SNoW learning architecture (Roth, 1998), that is available on the same site. This set of English NEs was hand-pruned to remove incorrectly classified words to obtain 978 single word NEs.

In order to reduce running time, some limited pre-processing was done on the Russian side. All classes, whose temporal distributions were close to uniform (i.e. words with a similar likelihood of occurrence throughout the corpus) were

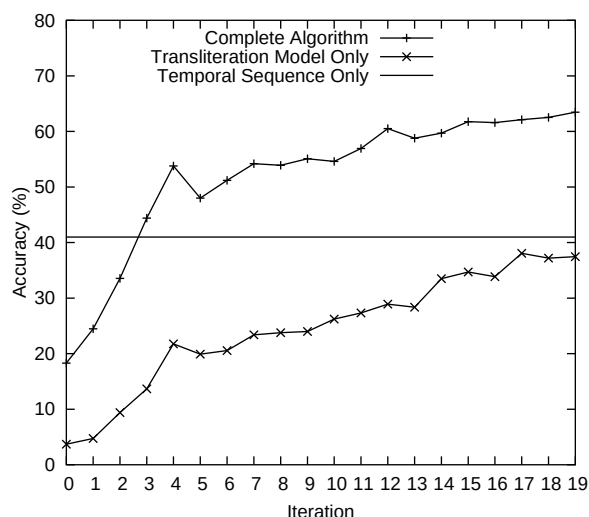


Figure 3: Proportion of correctly discovered NE pairs vs. training iteration. Complete algorithm outperforms both transliteration model and temporal sequence matching when used on their own.

deemed common and not considered as NE candidates. Unique words were thus grouped into 14,781 equivalence classes.

Unless mentioned otherwise, the transliteration model was initialized with a set of 20 pairs of English NEs and their Russian transliterations. Negative examples here and during the rest of the training were pairs of randomly selected non-NE English and Russian words.

New features were discovered throughout training; all but top 3000 features from positive and 3000 from negative examples were pruned based on the number of their occurrences so far. Features remaining at the end of training were used for NE discovery.

Insertions/omissions features were not used in the experiments as they provided no tangible benefit for the languages of our corpus.

In each iteration, we used the current transliteration model to find a list of 30 best transliteration equivalence classes for each NE. We then computed time sequence similarity score between NE and each class from its list to find the one with the best matching time sequence. If its similarity score surpassed a set threshold, it was added to the list of positive examples for the next round of training. Positive examples were constructed by pairing an NE with the common stem of its transliteration equivalence class. We used the same number of positive and negative examples.

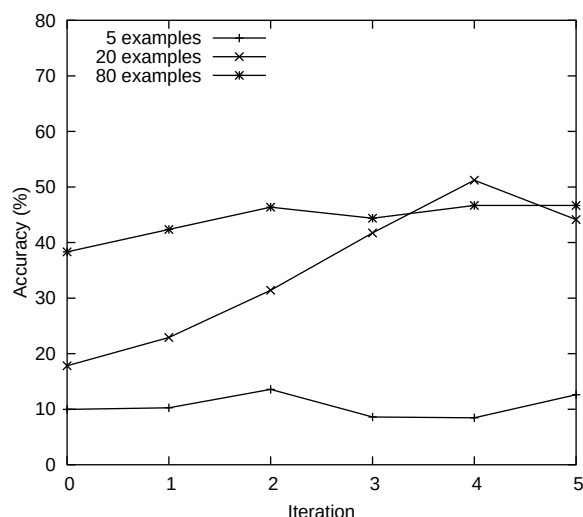


Figure 4: Proportion of correctly discovered NE pairs vs. the initial example set size. As long as the size is large enough, decreasing the number of examples does not have a significant impact on the performance of the later iterations.

We used the Mueller English-Russian dictionary to obtain translations in our multi-word NE experiments. We only considered the first dictionary definition as a candidate.

For evaluation, random 727 of the total of 978 NEs were matched to correct transliterations by a language expert (partly due to the fact that some of the English NEs were not mentioned in the Russian side of the corpus). Accuracy was computed as the percentage of NEs correctly identified by the algorithm.

In the multi-word NE experiment, 282 random multi-word (2 or more) NEs and their transliterations/translations discovered by the algorithm were verified by a language expert.

4.1 NE discovery

Figure 3 shows the proportion of correctly discovered NE transliteration equivalence classes throughout the training stage. The figure also shows the accuracy if transliterations are selected according to the current transliteration model (top scoring candidate) and temporal sequence matching alone.

The transliteration model alone achieves an accuracy of about 38%, while the time sequence alone gets about 41%. The combined algorithm achieves about 63%, giving a significant improvement.

	$k = 1$	$k = 3$	$k = 5$
Cosine	41.3	5.8	1.7
Pearson	41.1	5.8	1.7
DFT	41.0	12.4	4.8

Table 1: Proportion of correctly discovered NEs vs. corpus misalignment (k) for each of the three measures. DFT based measure provides significant advantages over commonly used metrics for weakly aligned corpora.

	$w = 1$	$w = 2$	$w = 3$
Cosine	5.8	13.5	18.4
Pearson	5.8	13.5	18.2
DFT	12.4	20.6	27.9

Table 2: Proportion of correctly discovered NEs vs. sliding window size (w) for each of the three measures.

In order to understand what happens to the transliteration model as the training proceeds, let us consider the following example. Figure 5 shows parts of transliteration lists for NE *forsyth* for two iterations of the algorithm. The weak transliteration model selects the correct transliteration (italicized) as the 24th best transliteration in the first iteration. Time sequence scoring function chooses it to be one of the training examples for the next round of training of the model. By the eighth iteration, the model has improved to select it as a best transliteration.

Not all correct transliterations make it to the top of the candidates list (transliteration model by itself is never as accurate as the complete algorithm on Figure 3). That is not required, however, as the model only needs to be good enough to place the correct transliteration anywhere in the candidate list.

Not surprisingly, some of the top transliteration candidates start sounding like the NE itself, as training progresses. On Figure 5, candidates for *forsyth* on iteration 7 include *fross* and *fossett*.

Once the transliteration model was trained, we ran the algorithm to discover multi-word NEs, augmenting candidate sets of dictionary words with their translations as described in Section 3.1. We achieved the accuracy of about 66%. The correctly discovered Russian NEs included entirely transliterated, partially translated, and entirely translated NEs. Some of them are shown on Figure 6.

4.2 Initial example set size

We ran a series of experiments to see how the size of the initial training set affects the accuracy of the model as training progresses (Figure 4). Although the performance of the early iterations is significantly affected by the size of the initial training example set, the algorithm quickly improves its performance. As we decrease the size from 80 to 20, the accuracy of the first iteration drops by over 20%, but a few iterations later the two have similar performance. However, when initialized with the set of size 5, the algorithm never manages to improve.

The intuition is the following. The few examples in the initial training set produce features corresponding to substring pairs characteristic for English-Russian transliterations. Model trained on these (few) examples chooses other transliterations containing these same substring pairs. In turn, the chosen positive examples contain other characteristic substring pairs, which will be used by the model to select more positive examples on the next round, and so on. On the other hand, if the initial set is too small, too few of the characteristic transliteration features are extracted to select a clean enough training set on the next round of training.

In general, one would expect the size of the training set necessary for the algorithm to improve to depend on the level of temporal alignment of the two sides of the corpus. Indeed, the weaker the temporal supervision the more we need to endow the model so that it can select cleaner candidates in the early iterations.

4.3 Comparison of time sequence scoring functions

We compared the performance of the DFT-based time sequence similarity scoring function we use in this paper to the commonly used *cosine* (Salton and McGill, 1986) and *Pearson's* correlation measures.

We perturbed the Russian side of the corpus in the following way. Articles from each day were randomly moved (with uniform probability) within a k -day window. We ran single word NE temporal sequence matching alone on the perturbed corpora using each of the three measures (Table 1).

Some accuracy drop due to misalignment could be accommodated for by using a larger temporal

Iteration 0		Iteration 7	
1	скоре {-е, -й, -йшего, -йший}	1	<i>форсайт</i> {-а, -, -у}
2	оформ {-лено, -лении, -ил, -ить}	2	оформ {-лено, -лении, -ил, -ить}
3	кокрэйн {-а, -}	3	проры {-вом, -ва, -ли, -тых, -вы, ...}
4	флоре {-нс, -нц, -, -нции}	4	фросс
•		5	фоссет {-т, -та, -ту, -а, -у}
•		•	
24	<i>форсайт</i> {-а, -, -у}	•	
•		•	

Figure 5: Transliteration lists for *forsyth* for two iterations of the algorithm. As transliteration model improves, the correct transliteration moves up the list.

bin for collecting occurrence counts. We tried various (sliding) window size w for a perturbed corpus with $k = 3$ (Table 2).

DFT metric outperforms the other measures significantly in most cases. NEs tend to have distributions with few pronounced peaks. If two such distributions are not well aligned, we expect both Pearson and Cosine measures to produce low scores, whereas the DFT metric should catch their similarities in the frequency domain.

5 Conclusions

We have proposed a novel algorithm for cross lingual multi-word NE discovery in a bilingual weakly temporally aligned corpus. We have demonstrated that using two independent sources of information (transliteration and temporal similarity) together to guide NE extraction gives better performance than using either of them alone (see Figure 3).

We developed a linear discriminative transliteration model, and presented a method to automatically generate features. For time sequence matching, we used a scoring metric novel in this domain. We provided experimental evidence that this metric outperforms other scoring metrics traditionally used.

In keeping with our objective to provide as little language knowledge as possible, we introduced a simplistic approach to identifying transliteration equivalence classes, which sometimes produced erroneous groupings (e.g. an equivalence class for NE *congolese* in Russian included both *congo* and *congolese* on Figure 6). We expect that more language specific knowledge used to discover accurate equivalence classes would result in performance improvements.

Other type of supervision was in the form of a

Eng. NE	Rus. NE equiv. class
carla del ponte	карла{-, -йл} дель понте
marc dutroux	марк дютру
pangbourne	пангбурн
supreme council	верхо{-вный} совет{...}
congolese	конго{-, -лезской}
north carolina	север{...} карол{-ина,...}
junichiro koizumi	дзюнитиро коидзуми
rehman	реман{-, -а}

Figure 6: Example of correct transliterations discovered by the algorithm.

very small bootstrapping transliteration set.

6 Future Work

The algorithm can be naturally extended to comparable corpora of more than two languages. Pair-wise time sequence scoring and transliteration models should give better confidence in NE matches.

The ultimate goal of this work is to automatically tag NEs so that they can be used for training of an NER system for a new language. To this end, we would like to compare the performance of an NER system trained on a corpus tagged using this approach to one trained on a hand-tagged corpus.

7 Acknowledgments

We thank Richard Sproat, ChengXiang Zhai, and Kevin Small for their useful feedback during this work, and the anonymous referees for their helpful comments. This research is supported by the Advanced Research and Development Activity (ARDA)’s Advanced Question Answering for Intelligence (AQUAINT) Program and a DOI grant under the Reflex program.

References

- Nasreen AbdulJaleel and Leah S. Larkey. 2003. Statistical transliteration for english-arabic cross language information retrieval. In *Proceedings of CIKM*, pages 139–146, New York, NY, USA.
- George Arfken. 1985. *Mathematical Methods for Physicists*. Academic Press.
- Avrim Blum. 1992. Learning boolean functions in an infinite attribute space. *Machine Learning*, 9(4):373–386.
- Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*.
- Silviu Cucerzan and David Yarowsky. 1999. Language independent named entity recognition combining morphological and contextual evidence. In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*.
- Magnus Lie Hetland, 2004. *Data Mining in Time Series Databases*, chapter A Survey of Recent Methods for Efficient Retrieval of Similar Time Sequences. World Scientific.
- Sung Young Jung, SungLim Hong, and Eunok Paek. 2000. An english to korean transliteration model of extended markov window. In *Proc. the International Conference on Computational Linguistics (COLING)*, pages 383–389.
- Alexandre Klementiev and Dan Roth. 2006. Named entity transliteration and discovery from multilingual comparable corpora. In *Proc. of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*.
- Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. In *Proc. of the Meeting of the European Association of Computational Linguistics*, pages 128–135.
- Xin Li, Paul Morie, and Dan Roth. 2004. Identification and tracing of ambiguous names: Discriminative and generative approaches. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 419–424.
- Robert C. Moore. 2005. A discriminative framework for bilingual word alignment. In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 81–88.
- Frank Rosenblatt. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65.
- Dan Roth. 1998. Learning to resolve natural language ambiguities: A unified approach. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 806–813.
- Dan Roth. 1999. Learning in natural language. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 898–904.
- Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.
- Yusuke Shinyama and Satoshi Sekine. 2004. Named entity discovery using comparable news articles. In *Proc. the International Conference on Computational Linguistics (COLING)*, pages 848–853.
- Ben Taskar, Simon Lacoste-Julien, and Michael Jordan. 2005. Structured prediction via the extragradient method. In *The Conference on Advances in Neural Information Processing Systems (NIPS)*. MIT Press.

A Composite Kernel to Extract Relations between Entities with both Flat and Structured Features

Min Zhang Jie Zhang Jian Su Guodong Zhou

Institute for Infocomm Research

21 Heng Mui Keng Terrace, Singapore 119613

{mzhang, zhangjie, sujian, zhougd}@i2r.a-star.edu.sg

Abstract

This paper proposes a novel composite kernel for relation extraction. The composite kernel consists of two individual kernels: an entity kernel that allows for entity-related features and a convolution parse tree kernel that models syntactic information of relation examples. The motivation of our method is to fully utilize the nice properties of kernel methods to explore diverse knowledge for relation extraction. Our study illustrates that the composite kernel can effectively capture both flat and structured features without the need for extensive feature engineering, and can also easily scale to include more features. Evaluation on the ACE corpus shows that our method outperforms the previous best-reported methods and significantly outperforms previous two dependency tree kernels for relation extraction.

1 Introduction

The goal of *relation extraction* is to find various predefined semantic *relations* between pairs of *entities* in text. The research on relation extraction has been promoted by the Message Understanding Conferences (MUCs) (MUC, 1987-1998) and Automatic Content Extraction (ACE) program (ACE, 2002-2005). According to the ACE Program, an *entity* is an object or set of objects in the world and a *relation* is an explicitly or implicitly stated relationship among entities. For example, the sentence “Bill Gates is chairman and chief software architect of Microsoft Corporation.” conveys the ACE-style relation “EMPLOYMENT.exec” between the entities “Bill Gates” (PERSON.Name) and “Microsoft Corporation” (ORGANIZATION.Commercial).

In this paper, we address the problem of relation extraction using kernel methods (Schölkopf and Smola, 2001). Many feature-based learning algorithms involve only the dot-product between feature vectors. Kernel methods can be regarded

as a generalization of the feature-based methods by replacing the dot-product with a kernel function between two vectors, or even between two objects. A kernel function is a similarity function satisfying the properties of being symmetric and positive-definite. Recently, kernel methods are attracting more interests in the NLP study due to their ability of implicitly exploring huge amounts of structured features using the original representation of objects. For example, the kernels for structured natural language data, such as *parse tree kernel* (Collins and Duffy, 2001), *string kernel* (Lodhi et al., 2002) and *graph kernel* (Suzuki et al., 2003) are example instances of the well-known *convolution kernels*¹ in NLP. In relation extraction, typical work on kernel methods includes: Zelenko et al. (2003), Culotta and Sorensen (2004) and Bunescu and Mooney (2005).

This paper presents a novel composite kernel to explore diverse knowledge for relation extraction. The composite kernel consists of an entity kernel and a convolution parse tree kernel. Our study demonstrates that the composite kernel is very effective for relation extraction. It also shows without the need for extensive feature engineering the composite kernel can not only capture most of the flat features used in the previous work but also exploit the useful syntactic structure features effectively. An advantage of our method is that the composite kernel can easily cover more knowledge by introducing more kernels. Evaluation on the ACE corpus shows that our method outperforms the previous best-reported methods and significantly outperforms the previous kernel methods due to its effective exploration of various syntactic features.

The rest of the paper is organized as follows. In Section 2, we review the previous work. Section 3 discusses our composite kernel. Section 4 reports the experimental results and our observations. Section 5 compares our method with the

¹ Convolution kernels were proposed for a discrete structure by Haussler (1999) in the machine learning field. This framework defines a kernel between input objects by applying convolution “sub-kernels” that are the kernels for the decompositions (parts) of the objects.

previous work from the viewpoint of feature exploration. We conclude our work and indicate the future work in Section 6.

2 Related Work

Many techniques on relation extraction, such as rule-based (MUC, 1987-1998; Miller et al., 2000), feature-based (Kambhatla 2004; Zhou et al., 2005) and kernel-based (Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005), have been proposed in the literature.

Rule-based methods for this task employ a number of linguistic rules to capture various relation patterns. Miller et al. (2000) addressed the task from the syntactic parsing viewpoint and integrated various tasks such as POS tagging, NE tagging, syntactic parsing, template extraction and relation extraction using a generative model.

Feature-based methods (Kambhatla, 2004; Zhou et al., 2005; Zhao and Grishman, 2005²) for this task employ a large amount of diverse linguistic features, such as lexical, syntactic and semantic features. These methods are very effective for relation extraction and show the best-reported performance on the ACE corpus. However, the problems are that these diverse features have to be manually calibrated and the hierarchical structured information in a parse tree is not well preserved in their parse tree-related features, which only represent simple flat *path* information connecting two entities in the parse tree through a *path* of non-terminals and a *list* of base phrase chunks.

Prior kernel-based methods for this task focus on using individual tree kernels to exploit tree structure-related features. Zelenko et al. (2003) developed a kernel over parse trees for relation extraction. The kernel matches nodes from roots to leaf nodes recursively layer by layer in a top-down manner. Culotta and Sorensen (2004) generalized it to estimate similarity between dependency trees. Their tree kernels require the matchable nodes to be at the same layer counting from the root and to have an identical path of ascending nodes from the roots to the current nodes. The two constraints make their kernel high precision but very low recall on the ACE 2003 corpus. Bunescu and Mooney (2005) proposed another dependency tree kernel for relation extraction.

Their kernel simply counts the number of common word classes at each position in the shortest paths between two entities in dependency trees. The kernel requires the two paths to have the same length; otherwise the kernel value is zero. Therefore, although this kernel shows performance improvement over the previous one (Culotta and Sorensen, 2004), the constraint makes the two dependency kernels share the similar behavior: good precision but much lower recall on the ACE corpus.

The above discussion shows that, although kernel methods can explore the huge amounts of implicit (structured) features, until now the feature-based methods enjoy more success. One may ask: how can we make full use of the nice properties of kernel methods and define an effective kernel for relation extraction?

In this paper, we study how relation extraction can benefit from the elegant properties of kernel methods: 1) implicitly exploring (structured) features in a high dimensional space; and 2) the nice mathematical properties, for example, the sum, product, normalization and polynomial expansion of existing kernels is a valid kernel (Schölkopf and Smola, 2001). We also demonstrate how our composite kernel effectively captures the diverse knowledge for relation extraction.

3 Composite Kernel for Relation Extraction

In this section, we define the composite kernel and study the effective representation of a relation instance.

3.1 Composite Kernel

Our composite kernel consists of an entity kernel and a convolution parse tree kernel. To our knowledge, convolution kernels have not been explored for relation extraction.

(1) Entity Kernel: The ACE 2003 data defines four entity features: entity headword, entity type and subtype (only for GPE), and mention type while the ACE 2004 data makes some modifications and introduces a new feature “LDC mention type”. Our statistics on the ACE data reveals that the entity features impose a strong constraint on relation types. Therefore, we design a linear kernel to explicitly capture such features:

$$K_L(R_1, R_2) = \sum_{i=1,2} K_E(R_1.E_i, R_2.E_i) \quad (1)$$

where R_1 and R_2 stands for two relation instances, E_i means the i^{th} entity of a relation instance, and

² We classify the feature-based kernel defined in (Zhao and Grishman, 2005) into the feature-based methods since their kernels can be easily represented by the dot-products between explicit feature vectors.

$K_E(\cdot, \cdot)$ is a simple kernel function over the features of entities:

$$K_E(E_1, E_2) = \sum_i C(E_1 \cdot f_i, E_2 \cdot f_i) \quad (2)$$

where f_i represents the i^{th} entity feature, and the function $C(\cdot, \cdot)$ returns 1 if the two feature values are identical and 0 otherwise. $K_E(\cdot, \cdot)$ returns the number of feature values in common of two entities.

(2) Convolution Parse Tree Kernel: A convolution kernel aims to capture structured information in terms of substructures. Here we use the same convolution parse tree kernel as described in Collins and Duffy (2001) for syntactic parsing and Moschitti (2004) for semantic role labeling. Generally, we can represent a parse tree T by a vector of integer counts of each sub-tree type (regardless of its ancestors):

$$\phi(T) = (\# subtree_1(T), \dots, \# subtree_i(T), \dots, \# subtree_n(T))$$

where $\# subtree_i(T)$ is the occurrence number of the i^{th} sub-tree type ($subtree_i$) in T . Since the number of different sub-trees is exponential with the parse tree size, it is computationally infeasible to directly use the feature vector $\phi(T)$. To solve this computational issue, Collins and Duffy (2001) proposed the following parse tree kernel to calculate the dot product between the above high dimensional vectors implicitly.

$$\begin{aligned} K(T_1, T_2) &= \langle \phi(T_1), \phi(T_2) \rangle \\ &= \sum_i \# subtree_i(T_1) \cdot \# subtree_i(T_2) \\ &= \sum_i \left(\sum_{n_1 \in N_1} I_{subtree_i}(n_1) \right) \cdot \left(\sum_{n_2 \in N_2} I_{subtree_i}(n_2) \right) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \Delta(n_1, n_2) \end{aligned} \quad (3)$$

where N_1 and N_2 are the sets of nodes in trees T_1 and T_2 , respectively, and $I_{subtree_i}(n)$ is a function that is 1 iff the $subtree_i$ occurs with root at node n and zero otherwise, and $\Delta(n_1, n_2)$ is the number of the common *subtrees* rooted at n_1 and n_2 , i.e.

$$\Delta(n_1, n_2) = \sum_i I_{subtree_i}(n_1) \cdot I_{subtree_i}(n_2)$$

$\Delta(n_1, n_2)$ can be computed by the following recursive rules:

- (1) if the productions (CFP rules) at n_1 and n_2 are different, $\Delta(n_1, n_2) = 0$;
- (2) else if both n_1 and n_2 are pre-terminals (POS tags), $\Delta(n_1, n_2) = 1 \times \lambda$;
- (3) else, $\Delta(n_1, n_2) = \lambda \prod_{j=1}^{nc(n_1)} (1 + \Delta(ch(n_1, j), ch(n_2, j)))$,

where $nc(n_1)$ is the child number of n_1 , $ch(n, j)$ is the j^{th} child of node n and λ ($0 < \lambda < 1$) is the decay factor in order to make the kernel value less variable with respect to the *subtree* sizes. In addition, the recursive rule (3) holds because given two nodes with the same children, one can construct common sub-trees using these children and common sub-trees of further offspring.

The parse tree kernel counts the number of common sub-trees as the syntactic similarity measure between two relation instances. The time complexity for computing this kernel is $O(|N_1| \cdot |N_2|)$.

In this paper, two composite kernels are defined by combining the above two individual kernels in the following ways:

1) Linear combination:

$$K_1(R_1, R_2) = \alpha \cdot \hat{K}_L(R_1, R_2) + (1 - \alpha) \cdot \hat{K}(T_1, T_2) \quad (4)$$

Here, $\hat{K}(\cdot, \cdot)$ is the normalized³ $K(\cdot, \cdot)$ and α is the coefficient. Evaluation on the development set shows that this composite kernel yields the best performance when α is set to 0.4.

2) Polynomial expansion:

$$K_2(R_1, R_2) = \alpha \cdot \hat{K}_L^P(R_1, R_2) + (1 - \alpha) \cdot \hat{K}(T_1, T_2) \quad (5)$$

Here, $\hat{K}(\cdot, \cdot)$ is the normalized $K(\cdot, \cdot)$, $K^P(\cdot, \cdot)$ is the polynomial expansion of $K(\cdot, \cdot)$ with degree $d=2$, i.e. $K^P(\cdot, \cdot) = (K(\cdot, \cdot) + 1)^2$, and α is the coefficient. Evaluation on the development set shows that this composite kernel yields the best performance when α is set to 0.23.

The polynomial expansion aims to explore the entity bi-gram features, esp. the combined features from the first and second entities, respectively. In addition, due to the different scales of the values of the two individual kernels, they are normalized before combination. This can avoid one kernel value being overwhelmed by that of another one.

The entity kernel formulated by eqn. (1) is a proper kernel since it simply calculates the dot product of the entity feature vectors. The tree kernel formulated by eqn. (3) is proven to be a proper kernel (Collins and Duffy, 2001). Since kernel function set is closed under normalization, polynomial expansion and linear combination (Schölkopf and Smola, 2001), the two composite kernels are also proper kernels.

³ A kernel $K(x, y)$ can be normalized by dividing it by $\sqrt{K(x, x) \cdot K(y, y)}$.

3.2 Relation Instance Spaces

A relation instance is encapsulated by a parse tree. Thus, it is critical to understand which portion of a parse tree is important in the kernel calculation. We study five cases as shown in Fig.1.

(1) **Minimum Complete Tree (MCT)**: the complete sub-tree rooted by the nearest common ancestor of the two entities under consideration.

(2) **Path-enclosed Tree (PT)**: the smallest common sub-tree including the two entities. In other words, the sub-tree is enclosed by the shortest path linking the two entities in the parse tree (this path is also commonly-used as the path tree feature in the feature-based methods).

(3) **Context-Sensitive Path Tree (CPT)**: the PT extended with the 1st left word of entity 1 and the 1st right word of entity 2.

(4) **Flattened Path-enclosed Tree (FPT)**: the PT with the single in and out arcs of non-terminal nodes (except POS nodes) removed.

(5) **Flattened CPT (FCPT)**: the CPT with the single in and out arcs of non-terminal nodes (except POS nodes) removed.

Fig. 1 illustrates different representations of an example relation instance. T_1 is **MCT** for the relation instance, where the sub-tree circled by a dashed line is **PT**, which is also shown in T_2 for clarity. The only difference between **MCT** and **PT** lies in that **MCT** does not allow partial production rules (for example, $NP \rightarrow PP$ is a partial production rule while $NP \rightarrow NP+PP$ is an entire production rule in the top of T_2). For instance, only the most-right child in the most-left sub-tree [NP [CD 200] [JJ domestic] [E1-PER ...]] of T_1 is kept in T_2 . By comparing the performance of T_1 and T_2 , we can evaluate the effect of sub-trees with partial production rules as shown in T_2 and the necessity of keeping the whole left and right context sub-trees as shown in T_1 in relation extraction. T_3 is **CPT**, where the two sub-trees circled by dashed lines are included as the context to T_2 and make T_3 context-sensitive. This is to evaluate whether the limited context information in **CPT** can boost performance. **FPT** in T_4 is formed by removing the two circled nodes in T_2 . This is to study whether and how the elimination of single non-terminal nodes affects the performance of relation extraction.

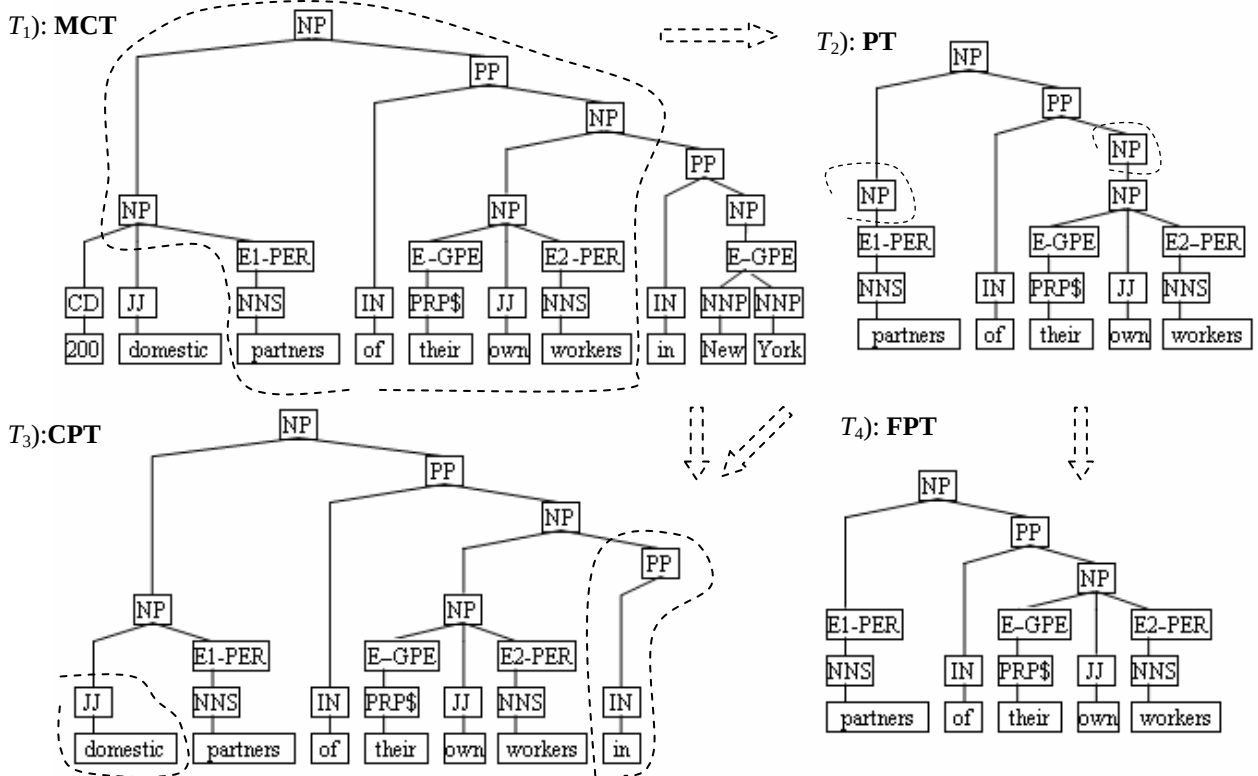


Figure 1. Different representations of a relation instance in the example sentence "...provide benefits to 200 domestic **partners** of their own **workers** in New York", where the phrase type "E1-PER" denotes that the current node is the 1st entity with type "PERSON", and likewise for the others. The relation instance is excerpted from the ACE 2003 corpus, where a relation "SOCIAL.Other-Personal" exists between entities "**partners**" (PER) and "**workers**" (PER). We use Charniak's parser (Charniak, 2001) to parse the example sentence. To save space, the **FCPT** is not shown here.

4 Experiments

4.1 Experimental Setting

Data: We use the English portion of both the ACE 2003 and 2004 corpora from LDC in our experiments. In the ACE 2003 data, the training set consists of 674 documents and 9683 relation instances while the test set consists of 97 documents and 1386 relation instances. The ACE 2003 data defines 5 entity types, 5 major relation types and 24 relation subtypes. The ACE 2004 data contains 451 documents and 5702 relation instances. It redefines 7 entity types, 7 major relation types and 23 subtypes. Since Zhao and Grishman (2005) use a 5-fold cross-validation on a subset of the 2004 data (newswire and broadcast news domains, containing 348 documents and 4400 relation instances), for comparison, we use the same setting (5-fold cross-validation on the same subset of the 2004 data, but the 5 partitions may not be the same) for the ACE 2004 data. Both corpora are parsed using Charniak’s parser (Charniak, 2001). We iterate over all pairs of entity mentions occurring in the same sentence to generate potential relation instances. In this paper, we only measure the performance of relation extraction models on “true” mentions with “true” chaining of coreference (i.e. as annotated by LDC annotators).

Implementation: We formalize relation extraction as a multi-class classification problem. SVM is selected as our classifier. We adopt the *one vs. others* strategy and select the one with the largest margin as the final answer. The training parameters are chosen using cross-validation ($C=2.4$ (SVM); $\lambda=0.4$ (tree kernel)). In our implementation, we use the binary SVMlight (Joachims, 1998) and Tree Kernel Tools (Moschitti, 2004). Precision (**P**), Recall (**R**) and F-measure (**F**) are adopted to measure the performance.

4.2 Experimental Results

In this subsection, we report the experiments of different kernel setups for different purposes.

(1) Tree Kernel only over Different Relation Instance Spaces: In order to better study the impact of the syntactic structure information in a parse tree on relation extraction, we remove the entity-related information from parse trees by replacing the entity-related phrase types (“E1-PER” and so on as shown in Fig. 1) with “NP”. Table 1 compares the performance of 5 tree kernel setups on the ACE 2003 data using the tree structure information only. It shows that:

- Overall the five different relation instance spaces are all somewhat effective for relation extraction. This suggests that structured syntactic information has good predication power for relation extraction and the structured syntactic information can be well captured by the tree kernel.
- **MCT** performs much worse than the others. The reasons may be that **MCT** includes too much left and right context information, which may introduce many noisy features and cause over-fitting (high precision and very low recall as shown in Table 1). This suggests that only keeping the complete (not partial) production rules in **MCT** does harm performance.
- **PT** achieves the best performance. This means that only keeping the portion of a parse tree enclosed by the shortest path between entities can model relations better than all others. This may be due to that most significant information is with **PT** and including context information may introduce too much noise. Although context may include some useful information, it is still a problem to correctly utilize such useful information in the tree kernel for relation extraction.
- **CPT** performs a bit worse than **PT**. In some cases (e.g. in sentence “the merge of company A and company B...”, “merge” is a critical context word), the context information is helpful. However, the effective scope of context is hard to determine given the complexity and variability of natural languages.
- The two flattened trees perform worse than the original trees. This suggests that the single non-terminal nodes are useful for relation extraction.

Evaluation on the ACE 2004 data also shows that **PT** achieves the best performance (72.5/56.7/63.6 in **P/R/F**). More evaluations with the entity type and order information incorporated into tree nodes (“E1-PER”, “E2-PER” and “E-GPE” as shown in Fig. 1) also show that **PT** performs best with 76.1/62.6/68.7 in **P/R/F** on the 2003 data and 74.1/62.4/67.7 in **P/R/F** on the 2004 data.

Instance Spaces	P(%)	R(%)	F
Minimum Complete Tree (MCT)	77.5	38.4	51.3
Path-enclosed Tree (PT)	72.8	53.8	61.9
Context-Sensitive PT(CPT)	75.9	48.6	59.2
Flattened PT	72.7	51.7	60.4
Flattened CPT	76.1	47.2	58.2

Table 1. five different tree kernel setups on the ACE 2003 five major types using the parse tree structure information only (regardless of any entity-related information)

PTs (with Tree Structure Information only)	P(%)	R(%)	F
Entity kernel only	75.1 (79.5)	42.7 (34.6)	54.4 (48.2)
Tree kernel only	72.5 (72.8)	56.7 (53.8)	63.6 (61.9)
Composite kernel 1 (linear combination)	73.5 (76.3)	67.0 (63.0)	70.1 (69.1)
Composite kernel 2 (polynomial expansion)	76.1 (77.3)	68.4 (65.6)	72.1 (70.9)

Table 2. Performance comparison of different kernel setups over the ACE major types of both the 2003 data (the numbers in parentheses) and the 2004 data (the numbers outside parentheses)

(2) Composite Kernels: Table 2 compares the performance of different kernel setups on the ACE major types. It clearly shows that:

- The composite kernels achieve significant performance improvement over the two individual kernels. This indicates that the flat and the structured features are complementary and the composite kernels can well integrate them: 1) the flat entity information captured by the entity kernel; 2) the structured syntactic connection information between the two entities captured by the tree kernel.
- The composite kernel via the polynomial expansion outperforms the one via the linear combination by ~ 2 in F-measure. It suggests that the bi-gram entity features are very useful.
- The entity features are quite useful, which can achieve F-measures of 54.4/48.2 alone and can boost the performance largely by ~ 7 (70.1-63.2/69.1-61.9) in F-measure when combining with the tree kernel.
- It is interesting that the ACE 2004 data shows consistent better performance on all setups than the 2003 data although the ACE 2003 data is two times larger than the ACE 2004 data. This may be due to two reasons: 1) The ACE 2004 data defines two new entity types and re-defines the relation types and subtypes in order to reduce the inconsistency between LDC annotators. 2) More importantly, the ACE 2004 data defines 43 entity subtypes while there are only 3 subtypes in the 2003 data. The detailed classification in the 2004 data leads to significant performance improvement of 6.2 (54.4-48.2) in F-measure over that on the 2003 data.

Our composite kernel can achieve 77.3/65.6/70.9 and 76.1/68.4/72.1 in **P/R/F** over the ACE 2003/2004 major types, respectively.

Methods (2002/2003 data)	P(%)	R(%)	F
Ours: composite kernel 2 (polynomial expansion)	77.3 (64.9)	65.6 (51.2)	70.9 (57.2)
Zhou et al. (2005): feature-based SVM	77.2 (63.1)	60.7 (49.5)	68.0 (55.5)
Kambhatla (2004): feature-based ME	(-) (63.5)	(-) (45.2)	(-) (52.8)
Ours: tree kernel with en- tity information at node	76.1 (62.4)	62.6 (48.5)	68.7 (54.6)
Bunescu and Mooney (2005): shortest path de- pendency kernel	65.5 (-)	43.8 (-)	52.5 (-)
Culotta and Sorensen (2004): dependency kernel	67.1 (-)	35.0 (-)	45.8 (-)

Table 3. Performance comparison on the ACE 2003/2003 data over both 5 major types (the numbers outside parentheses) and 24 subtypes (the numbers in parentheses)

Methods (2004 data)	P(%)	R(%)	F
Ours: composite kernel 2 (polynomial expansion)	76.1 (68.6)	68.4 (59.3)	72.1 (63.6)
Zhao and Grishman (2005): feature-based kernel	69.2 (-)	70.5 (-)	70.4 (-)

Table 4. Performance comparison on the ACE 2004 data over both 7 major types (the numbers outside parentheses) and 23 subtypes (the numbers in parentheses)

(3) Performance Comparison: Tables 3 and 4 compare our method with previous work on the ACE 2002/2003/2004 data, respectively. They show that our method outperforms the previous methods and significantly outperforms the previous two dependency kernels⁴. This may be due to two reasons: 1) the dependency tree (Culotta and Sorensen, 2004) and the shortest path (Bunescu and Mooney, 2005) lack the internal hierarchical phrase structure information, so their corresponding kernels can only carry out node-matching directly over the nodes with word tokens; 2) the parse tree kernel has less constraints. That is, it is

⁴ Bunescu and Mooney (2005) used the ACE 2002 corpus, including 422 documents, which is known to have many inconsistencies than the 2003 version. Culotta and Sorensen (2004) used a generic ACE corpus including about 800 documents (no corpus version is specified). Since the testing corpora are in different sizes and versions, strictly speaking, it is not ready to compare these methods exactly and fairly. Therefore Table 3 is only for reference purpose. We just hope that we can get a few clues from this table.

not restricted by the two constraints of the two dependency kernels (identical layer and ancestors for the matchable nodes and identical length of two shortest paths, as discussed in Section 2).

The above experiments verify the effectiveness of our composite kernels for relation extraction. They suggest that the parse tree kernel can effectively explore the syntactic features which are critical for relation extraction.

Error Type	# of error instances	
	2004 data	2003 data
False Negative	198	416
False Positive	115	171
Cross Type	62	96

Table 5. Error distribution of major types on both the 2003 and 2004 data for the composite kernel by polynomial expansion

(4) Error Analysis: Table 5 reports the error distribution of the polynomial composite kernel over the major types on the ACE data. It shows that $83.5\%(198+115/198+115+62) / 85.8\%(416+171/416+171+96)$ of the errors result from relation detection and only $16.5\%/14.2\%$ of the errors result from relation characterization. This may be due to data imbalance and sparseness issues since we find that the negative samples are 8 times more than the positive samples in the training set. Nevertheless, it clearly directs our future work.

5 Discussion

In this section, we compare our method with the previous work from the feature engineering viewpoint and report some other observations and issues in our experiments.

5.1 Comparison with Previous Work

This is to explain more about why our method performs better and significantly outperforms the previous two dependency tree kernels from the theoretical viewpoint.

(1) Compared with Feature-based Methods: The basic difference lies in the relation instance representation (parse tree vs. feature vector) and the similarity calculation mechanism (kernel function vs. dot-product). The main difference is the different feature spaces. Regarding the parse tree features, our method implicitly represents a parse tree by a vector of integer counts of each sub-tree type, i.e., we consider the entire sub-tree types and their occurring frequencies. In this way, the parse tree-related features (the *path* features

and the *chunking* features) used in the feature-based methods are embedded (as a subset) in our feature space. Moreover, the in-between word features and the entity-related features used in the feature-based methods are also captured by the tree kernel and the entity kernel, respectively. Therefore our method has the potential of effectively capturing not only most of the previous flat features but also the useful syntactic structure features.

(2) Compared with Previous Kernels: Since our method only counts the occurrence of each sub-tree without considering the layer and the ancestors of the root node of the sub-tree, our method is not limited by the constraints (identical layer and ancestors for the matchable nodes, as discussed in Section 2) in Culotta and Sorensen (2004). Moreover, the difference between our method and Bunescu and Mooney (2005) is that their kernel is defined on the shortest *path* between two entities instead of the entire sub-trees. However, the *path* does not maintain the tree structure information. In addition, their kernel requires the two paths to have the same length. Such constraint is too strict.

5.2 Other Issues

(1) Speed Issue: The recursively-defined convolution kernel is much slower compared to feature-based classifiers. In this paper, the speed issue is solved in three ways. First, the inclusion of the entity kernel makes the composite kernel converge fast. Furthermore, we find that the small portion (**PT**) of a full parse tree can effectively represent a relation instance. This significantly improves the speed. Finally, the parse tree kernel requires exact match between two sub-trees, which normally does not occur very frequently. Collins and Duffy (2001) report that in practice, running time for the parse tree kernel is more close to linear ($O(|N_1|+|N_2|)$), rather than $O(|N_1|*|N_2|)$. As a result, using the PC with Intel P4 3.0G CPU and 2G RAM, our system only takes about 110 minutes and 30 minutes to do training on the ACE 2003 (~77k training instances) and 2004 (~33k training instances) data, respectively.

(2) Further Improvement: One of the potential problems in the parse tree kernel is that it carries out exact matches between sub-trees, so that this kernel fails to handle sparse phrases (i.e. “a car” vs. “a red car”) and near-synonymic grammar tags (for example, the variations of a verb (i.e. go, went, gone)). To some degree, it could possibly lead to over-fitting and compromise the per-

formance. However, the above issues can be handled by allowing grammar-driven partial rule matching and other approximate matching mechanisms in the parse tree kernel calculation.

Finally, it is worth noting that by introducing more individual kernels our method can easily scale to cover more features from a multitude of sources (e.g. Wordnet, gazetteers, etc) that can be brought to bear on the task of relation extraction. In addition, we can also easily implement the feature weighting scheme by adjusting the $eqn.(2)$ and the rule (2) in calculating $\Delta(n_1, n_2)$ (see subsection 3.1).

6 Conclusion and Future Work

Kernel functions have nice properties. In this paper, we have designed a composite kernel for relation extraction. Benefiting from the nice properties of the kernel methods, the composite kernel could well explore and combine the flat entity features and the structured syntactic features, and therefore outperforms previous best-reported feature-based methods on the ACE corpus. To our knowledge, this is the first research to demonstrate that, without the need for extensive feature engineering, an individual tree kernel achieves comparable performance with the feature-based methods. This shows that the syntactic features embedded in a parse tree are particularly useful for relation extraction and which can be well captured by the parse tree kernel. In addition, we find that the relation instance representation (selecting effective portions of parse trees for kernel calculations) is very important for relation extraction.

The most immediate extension of our work is to improve the accuracy of relation detection. This can be done by capturing more features by including more individual kernels, such as the WordNet-based semantic kernel (Basili et al., 2005) and other feature-based kernels. We can also benefit from machine learning algorithms to study how to solve the data imbalance and sparseness issues from the learning algorithm viewpoint. In the future work, we will design a more flexible tree kernel for more accurate similarity measure.

Acknowledgements: We would like to thank Dr. Alessandro Moschitti for his great help in using his Tree Kernel Toolkits and fine-tuning the system. We also would like to thank the three anonymous reviewers for their invaluable suggestions.

References

- ACE. 2002-2005. *The Automatic Content Extraction Projects*. <http://www ldc.upenn.edu/Projects /ACE/>
- Basili R., Cammisa M. and Moschitti A. 2005. *A Semantic Kernel to classify text with very few training examples*. ICML-2005
- Bunescu R. C. and Mooney R. J. 2005. *A Shortest Path Dependency Kernel for Relation Extraction*. EMNLP-2005
- Charniak E. 2001. *Immediate-head Parsing for Language Models*. ACL-2001
- Collins M. and Duffy N. 2001. *Convolution Kernels for Natural Language*. NIPS-2001
- Culotta A. and Sorensen J. 2004. *Dependency Tree Kernel for Relation Extraction*. ACL-2004
- Haussler D. 1999. *Convolution Kernels on Discrete Structures*. Technical Report UCS-CRL-99-10, University of California, Santa Cruz.
- Joachims T. 1998. *Text Categorization with Support Vector Machine: learning with many relevant features*. ECML-1998
- Kambhatla N. 2004. *Combining lexical, syntactic and semantic features with Maximum Entropy models for extracting relations*. ACL-2004 (poster)
- Lodhi H., Saunders C., Shawe-Taylor J., Cristianini N. and Watkins C. 2002. *Text classification using string kernel*. Journal of Machine Learning Research, 2002(2):419-444
- Miller S., Fox H., Ramshaw L. and Weischedel R. 2000. *A novel use of statistical parsing to extract information from text*. NAACL-2000
- Moschitti A. 2004. *A Study on Convolution Kernels for Shallow Semantic Parsing*. ACL-2004
- MUC. 1987-1998. http://www.itl.nist.gov/iaui/894.02/related_projects/muc/
- Schölkopf B. and Smola A. J. 2001. *Learning with Kernels: SVM, Regularization, Optimization and Beyond*. MIT Press, Cambridge, MA 407-423
- Suzuki J., Hirao T., Sasaki Y. and Maeda E. 2003. *Hierarchical Directed Acyclic Graph Kernel: Methods for Structured Natural Language Data*. ACL-2003
- Zelenko D., Aone C. and Richardella A. 2003. *Kernel Methods for Relation Extraction*. Journal of Machine Learning Research. 2003(2):1083-1106
- Zhao S.B. and Grishman R. 2005. *Extracting Relations with Integrated Information Using Kernel Methods*. ACL-2005
- Zhou G.D., Su J, Zhang J. and Zhang M. 2005. *Exploring Various Knowledge in Relation Extraction*. ACL-2005

Japanese Dependency Parsing Using Co-occurrence Information and a Combination of Case Elements

Takeshi Abekawa

Graduate School of Education
University of Tokyo
abekawa@p.u-tokyo.ac.jp

Manabu Okumura

Precision and Intelligence Laboratory
Tokyo Institute of Technology
oku@pi.titech.ac.jp

Abstract

In this paper, we present a method that improves Japanese dependency parsing by using large-scale statistical information. It takes into account two kinds of information not considered in previous statistical (machine learning based) parsing methods: information about dependency relations among the case elements of a verb, and information about co-occurrence relations between a verb and its case element. This information can be collected from the results of automatic dependency parsing of large-scale corpora. The results of an experiment in which our method was used to rerank the results obtained using an existing machine learning based parsing method showed that our method can improve the accuracy of the results obtained using the existing method.

1 Introduction

Dependency parsing is a basic technology for processing Japanese and has been the subject of much research. The Japanese dependency structure is usually represented by the relationship between phrasal units called *bunsetsu*, each of which consists of one or more content words that may be followed by any number of function words. The dependency between two *bunsetsus* is direct from a dependent to its head.

Manually written rules have usually been used to determine which *bunsetsu* another *bunsetsu* tends to modify, but this method poses problems in terms of the coverage and consistency of the rules. The recent availability of larger-scale corpora annotated with dependency information has thus resulted in more work on statistical dependency analysis technologies that use machine learning algorithms (Kudo and Matsumoto, 2002; Sassano,

2004; Uchimoto et al., 1999; Uchimoto et al., 2000).

Work on statistical Japanese dependency analysis has usually assumed that all the dependency relations in a sentence are independent of each other, and has considered the *bunsetsus* in a sentence independently when judging whether or not a pair of *bunsetsus* is in a dependency relation. In judging which *bunsetsu* a *bunsetsu* modifies, this type of work has used as features the information of two *bunsetsus*, such as the head words of the two *bunsetsus*, and the morphemes at the ends of the *bunsetsus* (Uchimoto et al., 1999). It is necessary, however, to also consider features for the contextual information of the two *bunsetsus*. One such feature is the constraint that two case elements with the same case do not modify a verb.

Statistical Japanese dependency analysis takes into account syntactic information but tends not to take into account lexical information, such as co-occurrence between a case element and a verb.

The recent availability of more corpora has enabled much information about dependency relations to be obtained by using a Japanese dependency analyzer such as KNP (Kurohashi and Nagao, 1994) or CaboCha (Kudo and Matsumoto, 2002). Although this information is less accurate than manually annotated information, these automatic analyzers provide a large amount of co-occurrence information as well as information about combinations of multiple cases that tend to modify a verb.

In this paper, we present a method for improving the accuracy of Japanese dependency analysis by representing the lexical information of co-occurrence and dependency relations of multiple cases as statistical models. We also show the results of experiments demonstrating the effectiveness of our method.

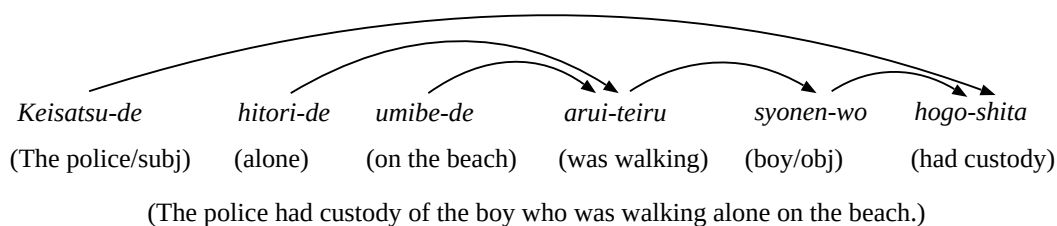


Figure 1: Example of a Japanese sentence, *bunsetsu* and dependencies

2 Parsing Japanese

The Japanese language is basically an SOV language, but word order is relatively free. In English the syntactic function of each word is represented by word order, while in Japanese it is represented by postpositions. For example, one or more postpositions following a noun play a role similar to the declension of nouns in German, which indicates grammatical case.

The syntax of a Japanese sentence is analyzed by using segments, called *bunsetsu*, that usually contain one or more content words like a noun, verb, or adjective, and zero or more function words like a particle (case marker) or verb/noun suffix. By defining a *bunsetsu* in this manner, we can analyze a sentence in a way similar to that used when analyzing the grammatical roles of words in inflected languages like German.

Japanese dependencies have the following characteristics:

- Each *bunsetsu* except the rightmost one has only one head.
- Each head *bunsetsu* is always placed to the right of (*i.e.* after) its modifier.
- Dependencies do not cross one another.

Statistical Japanese dependency analyzers (Kudo and Matsumoto, 2005; Kudo and Matsumoto, 2002; Sassano, 2004; Uchimoto et al., 1999; Uchimoto et al., 2000) automatically learn the likelihood of dependencies from a tagged corpus and calculate the best dependencies for an input sentence. These likelihoods are learned by considering the features of *bunsetsus* such as their character strings, parts of speech, and inflection types, as well as information between *bunsetsus* such as punctuation and the distance between *bunsetsus*. The weight of given features is learned

from a training corpus by calculating the weights from the frequencies of the features in the training data.

3 Japanese dependency analysis taking account of co-occurrence information and a combination of multiple cases

One constraint in Japanese is that multiple nouns of the same case do not modify a verb. Previous work on Japanese dependency analysis has assumed that all the dependency relations are independent of one another. It is therefore necessary to also consider such a constraint as a feature for contextual information. Uchimoto et al., for example, used as such a feature whether a particular type of *bunsetsu* is between two *bunsetsus* in a dependency relation (Uchimoto et al., 1999), and Sassano used information about what is just before and after the modifying *bunsetsu* and modifyee *bunsetsu* (Sassano, 2004).

In the artificial example shown in Figure 1, it is natural to consider that “*keisatsu-de*” will modify “*hogo-shita*”. Statistical Japanese dependency analyzers (Uchimoto et al., 2000; Kudo and Matsumoto, 2002), however, will output the result where “*keisatsu-de*” modifies “*arui-teiru*”. This is because in sentences without internal punctuation a noun tends to modify the nearest verb, and these analyzers do not take into account a combination of multiple cases.

Another kind of information useful in dependency analysis is the co-occurrence of a noun and a verb, which indicates to what degree the noun tends to modify the verb. In the above example, the possible modifyees of “*keisatsu-de*” are “*arui-teiru*” and “*hogo-shita*”. Taking into account information about the co-occurrence of “*keisatsu-de*” and “*arui-teiru*” and of “*keisatsu-de*” and “*hogo-shita*” makes it obvious that “*keisatsu-de*” is more likely to modify “*hogo-shita*”.

In summary, we think that statistical Japanese dependency analysis needs to take into account at least two more kinds of information: the dependency relation between multiple cases where multiple nouns of the same case do not modify a verb, and the co-occurrence of nouns and verbs. One way to use such information in statistical dependency analysis is to directly use it as features. However, Kehler et al. pointed out that this does not make the analysis more accurate (Kehler et al., 2004). This paper therefore presents a model that uses the co-occurrence information separately and reranks the analysis candidates generated by the existing machine learning model.

4 Our proposed model

We first introduce the notation for the explanation of the dependency structure T :

- $m(T)$: the number of verbs in T
- $v_i(T)$: the i -th verb in T
- $c_i(T)$: the number of case elements that modify the i -th verb in T
- $es_i(T)$: the set of case elements that modify the i -th verb in T
- $rs_i(T)$: the set of particles in the set of case elements that modify the i -th verb in T
- $ns_i(T)$: the set of nouns in the set of case elements that modify the i -th verb in T
- $r_{i,j}(T)$: the j -th particle that modifies the i -th verb in T
- $n_{i,j}(T)$: the j -th noun that modifies the i -th verb in T

We defined case element as a pair of a noun and following particles. For the dependency structure we assume the conditional probability $P(es_i(T)|v_i(T))$ that the set of case elements $es_i(T)$ depends on the $v_i(T)$, and assume the set of case elements $es_i(T)$ is composed of the set of noun $ns_i(T)$ and particles $rs_i(T)$.

$$P(es_i(T)|v_i(T)) \stackrel{def}{=} P(rs_i(T), ns_i(T)|v_i(T)) \quad (1)$$

$$= P(rs_i(T)|v_i(T)) \times P(ns_i(T)|rs_i(T), v_i(T)) \quad (2)$$

$$\simeq P(rs_i(T)|v_i(T)) \times \prod_{j=1}^{c_i(T)} P(n_{i,j}(T)|rs_i(T), v_i(T)) \quad (3)$$

$$\simeq P(rs_i(T)|v_i(T)) \times \prod_{j=1}^{c_i(T)} P(n_{i,j}(T)|r_{i,j}(T), v_i(T)) \quad (4)$$

In the transformation from Equation (2) to Equation (3), we assume that the set of noun $ns_i(T)$ is independent of the verb $v_i(T)$. And in the transformation from Equation (3) to Equation (4), we assume that the noun $n_{i,j}(T)$ is dependent on only its following particle $r_{i,j}(T)$.

Now we assume the dependency structure T of the whole sentence is composed of only the dependency relation between case elements and verbs, and propose the sentence probability defined by Equation (5).

$$P(T) = \prod_{i=1}^{m(T)} P(rs_i(T)|v_i(T)) \times \prod_{j=1}^{c_i(T)} P(n_{i,j}(T)|r_{i,j}(T), v_i(T)) \quad (5)$$

We call $P(rs_i(T)|v_i(T))$ the co-occurrence probability of the particle set and the verb, and we call $P(n_{i,j}(T)|r_{i,j}(T), v_i(T))$ the co-occurrence probability of the case element set and the verb.

In the actual dependency analysis, we try to select the dependency structure $\hat{T}$ that maximizes the Equation (5) from the possible parses T for the inputted sentence:

$$\hat{T} = \underset{T}{\operatorname{argmax}} \prod_{i=1}^{m(T)} P(rs_i(T)|v_i(T)) \times \prod_{j=1}^{c_i(T)} P(n_{i,j}(T)|r_{i,j}(T), v_i(T)). \quad (6)$$

The proposed model is inspired by the semantic role labeling method (Gildea and Jurafsky, 2002), which uses the frame element group in place of the particle set.

It differs from the previous parsing models in that we take into account the dependency relations among particles in the set of case elements that modify a verb. This information can constrain the combination of particles (cases) among *bunsetsus* that modify a verb. Assuming the independence among particles, we can rewrite Equation (5) as

$$P(T) = \prod_{i=1}^{m(T)} \prod_{j=1}^{c_i(T)} P(n_{i,j}(T), r_{i,j}(T)|v_i(T)). \quad (7)$$

4.1 Syntactic property of a verb

In Japanese, the “*ha*” case that indicates a topic tends to modify the main verb in a sentence and tends not to modify a verb in a relative clause. The

	verb: ‘ <i>aru-ku</i> ’		verb: ‘ <i>hogo-suru</i> ’	
	case elements	particle set	case elements	particle set
a	<i>keisatsu-de umibe-de hitori-de</i>	$\{de, de, de\}$	<i>syonen-wo</i>	$\{wo\}$
b	<i>umibe-de hitori-de</i>	$\{de, de\}$	<i>keisatsu-de syonen-wo</i>	$\{de, wo\}$
c	<i>hitori-de</i>	$\{de\}$	<i>keisatsu-de umibe-de syonen-wo</i>	$\{de, de, wo\}$
d		$\{none\}$	<i>keisatsu-de umibe-de hitori-de syonen-wo</i>	$\{de, de, de, wo\}$

Table 1: Analytical process of the example sentence

co-occurrence probability of the particle set therefore tends to be different for verbs with different syntactic properties.

Like (Shirai, 1998), to take into account the reliance of the co-occurrence probability of the particle set on the syntactic property of a verb, instead of using $P(rs_i(T)|v_i(T))$ in Equation (5), we use $P(rs_i(T)|syn_i(T), v_i(T))$, where $syn_i(T)$ is the syntactic property of the i -th verb in T and takes one of the following three values:

‘**verb**’ when v modifies another verb

‘**noun**’ when v modifies a noun

‘**main**’ when v modifies nothing (when it is at the end of the sentence, and is the main verb)

4.2 Illustration of model application

Here, we illustrate the process of applying our proposed model to the example sentence in Figure 1, for which there are four possible combinations of dependency relations. The *bunsetsu* combinations and corresponding sets of particles are listed in Table 1. In the analytical process, we calculate for all the combinations the co-occurrence probability of the case element set (*bunsetsu* set) and the co-occurrence probability of the particle set, and we select the $\hat{T}$ that maximizes the probability.

Some of the co-occurrence probabilities of the particle sets for the verbs “*aru-ku*” and “*hogo-suru*” in the sentence are listed in Table 2. How to estimate these probabilities is described in section 5.3. Basically, the larger the number of particles, the lower the probability is. As you can see in the comparison between $\{de, wo\}$ and $\{de, de\}$, the probability becomes lower when multiple same cases are included. Therefore, the probability can reflect the constraint that multiple case elements of the same particle tend not to modify a verb.

5 Experiments

We evaluated the effectiveness of our model experimentally. Since our model treats only the de-

rs_i	$P(rs_i noun, v_1)$ $v_1 = \text{“aru-ku”}$	$P(rs_i main, v_2)$ $v_2 = \text{“hogo-suru”}$
$\{none\}$	0.29	0.35
$\{wo\}$	0.30	0.24
$\{ga\}$	0.056	0.072
$\{ni\}$	0.040	0.041
$\{de\}$	0.032	0.033
$\{ha\}$	0.035	0.041
$\{de, wo\}$	0.022	0.018
$\{de, de\}$	0.00038	0.00038
$\{de, de, wo\}$	0.00022	0.00018
$\{de, de, de\}$	0.0000019	0.0000018
$\{de, de, de, wo\}$	0.00000085	0.00000070

Table 2: Example of the co-occurrence probabilities of particle sets

pendency relations between a noun and a verb, we cannot determine all the dependency relations in a sentence. We therefore use one of the currently available dependency analyzers to generate an ordered list of n -best possible parses for the sentence and then use our proposed model to rerank them and select the best parse.

5.1 Dependency analyzer for outputting n -best parses

We generated the n -best parses by using the “posterior context model” (Uchimoto et al., 2000). The features we used were those in (Uchimoto et al., 1999) and their combinations. We also added our original features and their combinations, with reference to (Sassano, 2004; Kudo and Matsumoto, 2002), but we removed the features that had a frequency of less than 30 in our training data. The total number of features is thus 105,608.

5.2 Reranking method

Because our model considers only the dependency relations between a noun and a verb, and thus cannot determine all the dependency relations in a sentence, we restricted the possible parses for

reranking as illustrated in Figure 2. The possible parses for reranking were the first-ranked parse and those of the next-best parses in which the verb to modify was different from that in the first-ranked one. For example, parses 1 and 3 in Figure 2 are the only candidates for reranking. In our experiments, n is set to 50.

The score we used for reranking the parses was the product of the probability of the posterior context model and the probability of our proposed model:

$$score = P_{context}(T)^\alpha \times P(T), \quad (8)$$

where $P_{context}(T)$ is the probability of the posterior context model. The α here is a parameter with which we can adjust the balance of the two probabilities, and is fixed to the best value by considering development data (different from the training data)¹.

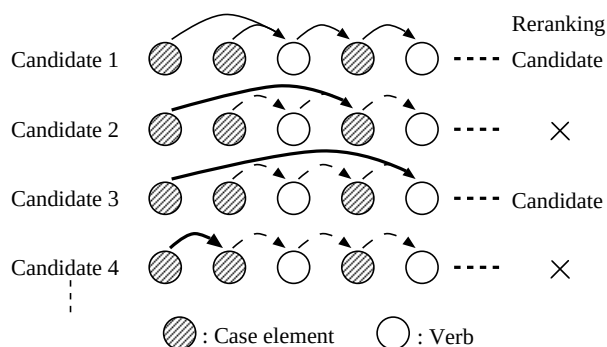


Figure 2: Selection of possible parses for reranking

Many methods for reranking the parsing of English sentences have been proposed (Charniak and Johnson, 2005; Collins and Koo, 2005; Henderson and Titov, 2005), all of which are discriminative methods which learn the difference between the best parse and next-best parses. While our reranking model using generation probability is quite simple, we can easily verify our hypothesis that the two proposed probabilities have an effect on improving the parsing accuracy. We can also verify that the parsing accuracy improves by using imprecise information obtained from an automatically parsed corpus.

Klein and Manning proposed a generative model in which syntactic (PCFG) and semantic (lexical dependency) structures are scored with separate models (Klein and Manning, 2002), but

¹In our experiments, α is set to 2.0 using development data.

they do not take into account the combination of dependencies. Shirai et al. also proposed a statistical model of Japanese language which integrates lexical association statistics with syntactic preference (Shirai et al., 1998). Our proposed model differs from their method in that it explicitly uses the combination of multiple cases.

5.3 Estimation of co-occurrence probability

We estimated the co-occurrence probability of the particle set and the co-occurrence probability of the case element set used in our model by analyzing a large-scale corpus. We collected a 30-year newspaper corpus², applied the morphological analyzer JUMAN (Kurohashi and Nagao, 1998b), and then applied the dependency analyzer with a posterior context model³. To ensure that we collected reliable co-occurrence information, we removed the information for the *bunsetsus* with punctuation⁴.

Like (Torisawa, 2001), we estimated the co-occurrence probability $P(\langle n, r, v \rangle)$ of the case element set (noun n , particle r , and verb v) by using probabilistic latent semantic indexing (PLSI) (Hofmann, 1999)⁵. If $\langle n, r, v \rangle$ is the co-occurrence of n and $\langle r, v \rangle$, we can calculate $P(\langle n, r, v \rangle)$ by using the following equation:

$$P(\langle n, r, v \rangle) = \sum_{z \in Z} P(n|z)P(\langle r, v \rangle|z)P(z), \quad (9)$$

where z indicates a latent semantic class of co-occurrence (hidden class). Probabilistic parameters $P(n|z)$, $P(\langle r, v \rangle|z)$, and $P(z)$ in Equation (9) can be estimated by using the EM algorithm. In our experiments, the dimension of the hidden class z was set to 300. As a result, the collected $\langle n, r, v \rangle$ total 102,581,924 pairs. The number of n and v is 57,315 and 15,098, respectively.

The particles for which the co-occurrence probability was estimated were the set of case particles, the “*ha*” case particle, and a class of “*fukujoshi*”

²13 years’ worth of articles from the Mainichi Shimbun, 14 years’ worth from the Yomiuri Shimbun, and 3 years’ worth from the Asahi Shimbun.

³We used the following package for calculation of Maximum Entropy:

http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.

⁴The result of dependency analysis with a posterior context model for the Kyodai Corpus showed that the accuracy for the *bunsetsu* without punctuation is 90.6%, while the accuracy is only 76.4% for those with punctuation.

⁵We used the following package for calculation of PLSI: <http://chasen.org/~taku/software/plsi/>.

		<i>Bunsetsu</i> accuracy	Sentence accuracy
Whole data	Context model	90.95% (73,390/80,695)	54.40% (5,052/9,287)
	Our model	91.21% (73,603/80,695)	55.17% (5,124/9,287)
Only for reranked sentences	Context model	90.72% (68,971/76,026)	48.33% (3,813/7,889)
	Our model	91.00% (69,184/76,026)	49.25% (3,885/7,889)
Only for case elements	Context model	91.80% (28,849/31,427)	–
	Our model	92.47% (29,062/31,427)	–

Table 3: Accuracy before/after reranking

particles. Therefore, the total number of particles was 10.

We also estimated the co-occurrence probability of the particle set $P(rs|syn, v)$ by using PLSI. We regarded the triple $\langle rs, syn, v \rangle$ (the co-occurrence of particle set rs , verb v , and the syntactic property syn) as the co-occurrence of rs and $\langle syn, v \rangle$. The dimension of the hidden class was 100. The total number of $\langle rs, syn, v \rangle$ pairs was 1,016,508, v was 18,423, and rs was 1,490. The particle set should be treated not as a non-ordered set but as an occurrence ordered set. However, we think correct probability estimation using an occurrence ordered set is difficult, because it gives rise to an explosion in the number of combination,

5.4 Experimental environment

The evaluation data we used was Kyodai Corpus 3.0, a corpus manually annotated with dependency relations (Kurohashi and Nagao, 1998a). The statistics of the data are as follows:

- Training data: 24,263 sentences, 234,474 *bunsetsu*s
- Development data: 4,833 sentences, 47,580 *bunsetsu*s
- Test data: 9,287 sentences, 89,982 *bunsetsu*s

The test data contained 31,427 case elements, and 28,801 verbs.

The evaluation measures we used were *bunsetsu* accuracy (the percentage of *bunsetsu* for which the correct modifyee was identified) and sentence accuracy (the percentage of sentences for which the correct dependency structure was identified).

5.5 Experimental results

5.5.1 Evaluation of our model

Our first experiment evaluated the effectiveness of reranking with our proposed model. *Bunsetsu*

		Our reranking model	
		correct	incorrect
Context model	correct	73,119	271
	incorrect	484	6,821

Table 4: 2×2 contingency table of the number of correct *bunsetsu* (posterior context model $\times$ our model)

and sentence accuracies before and after reranking, for the entire set of test data as well as for only those sentences whose parse was actually reranked, are listed in Table 3.

The results showed that the accuracy could be improved by using our proposed model to rerank the results obtained with the posterior context model. McNemar testing showed that the null hypothesis that there is no difference between the accuracy of the results obtained with the posterior context model and those obtained with our model could be rejected with a p value < 0.01 . The difference in accuracy is therefore significant.

5.5.2 Comparing variant models

We next experimentally compare the following variations of the proposed model:

- one in which the case element set is assumed to be independent [Equation (7)]
- one using the co-occurrence probability of the particle set, $P(rs|syn, v)$, in our model
- one using only the co-occurrence probability of the case element, $P(n|r, v)$, in our model
- one not taking into account the syntactic property of a verb (*i.e.* a model in which the co-occurrence probability is defined as $P(r|v)$, without the syntactic property syn)
- one in which the co-occurrence probability of the case element, $P(n|r, v)$, is simply added

	<i>Bunsetsu</i> accuracy	Sentence accuracy
Context model	90.95%	54.40%
Our model	91.21%	55.17%
model (a)	91.12%	54.90%
model (b)	91.10%	54.69%
model (c)	91.11%	54.91%
model (d)	91.15%	54.82%
model (e)	90.96%	54.33%
model (f)	89.50%	48.33%
Kudo et al 2005	91.37%	56.00%

Table 5: Comparison of various models

to a feature set used in the posterior context model

- (f) one using only our proposed probabilities without the probability of the posterior context model

The accuracies obtained with each of these models are listed in Table 5, from which we can conclude that it is effective to take into account the dependency between case elements because model (a) is less accurate than our model.

Since the accuracy of model (d) is comparable to that of our model, we can conclude that the consideration of the syntactic property of a verb does not necessarily improve dependency analysis.

The accuracy of model (e), which uses the co-occurrence probability of the case element set as features in the posterior context model, is comparable to that of the posterior context model. This result is similar to the one obtained by (Kehler et al., 2004), where the task was anaphora resolution. Although we think the co-occurrence probability is useful information for dependency analysis, this result shows that simply adding it as a feature does not improve the accuracy.

5.5.3 Changing the amount of training data

Changing the size of the training data set, we investigated whether the degree of accuracy improvement due to reranking depends on the accuracy of the existing dependency analyzer.

Figure 3 shows that the accuracy improvement is constant even if the accuracy of the dependency analyzer is varied.

5.6 Discussion

The score used in reranking is the product of the probability of the posterior context model and the

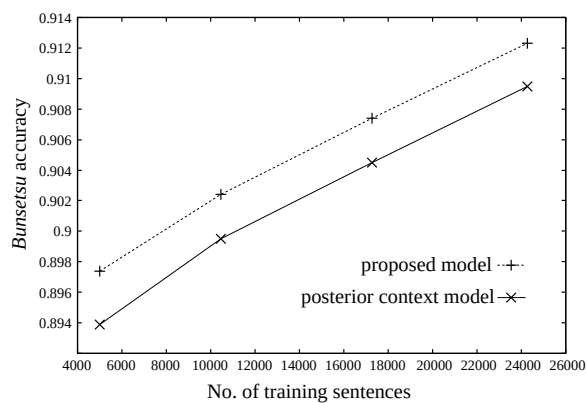


Figure 3: *Bunsetsu* accuracy when the size of the training data is changed

probability of our proposed model. The results in Table 5 show that the parsing accuracy of model (f), which uses only the probabilities obtained with our proposed model, is quite low. We think the reason for this is that our two co-occurrence probabilities cannot take account of syntactic properties, such as punctuation and the distance between two *bunsetsus*, which improve dependency analysis.

Furthermore, when the sentence has multiple verbs and case elements, the constraint of our proposed model tends to distribute case elements to each verb equally. To investigate such bias, we calculated the variance of the number of case elements per verb.

Table 6 shows that the variance for our proposed model (Equation [5]) is the lowest, and this model distributes case elements to each verb equally. The variance of the posterior context model is higher than that of the test data, probably because the syntactic constraint in this model affects parsing too much. Therefore the variance of the reranking model (Equation [8]), which is the combination of our proposed model and the posterior context model, is close to that of the test data.

The best parse which uses this data set is (Kudo and Matsumoto, 2005), and their parsing accuracy is 91.37%. The features and the parsing method used by their model are almost equal to the posterior context model, but they use a different method of probability estimation. If their model could generate *n*-best parsing and attach some kind of score to each parse tree, we would combine their model in place of the posterior context model.

At the stage of incorporating the proposed approach to a parser, the consistency with other pos-

	context model	test data	Equation [8]	Equation [5]
variance (σ^2)	0.724	0.702	0.696	0.666

*The average number of elements per verb is 1.078.

Table 6: The variance of the number of elements per verb

sible methods that deal with other relations should be taken into account. This will be one of our future tasks.

6 Conclusion

We presented a method of improving Japanese dependency parsing by using large-scale statistical information. Our method takes into account two types of information, not considered in previous statistical (machine learning based) parsing methods. One is information about the dependency relations among the case elements of a verb, and the other is information about co-occurrence relations between a verb and its case element. Experimental results showed that our method can improve the accuracy of the existing method.

References

- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n -best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 173–180.
- Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–69.
- Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.
- James Henderson and Ivan Titov. 2005. Data-defined kernels for parse reranking derived from probabilistic models. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 181–188.
- Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57.
- Andrew Kehler, Douglas Appelt, Lara Taylor, and Aleksandr Simma. 2004. The (non)utility of predicate-argument frequencies for pronoun interpretation. In *Proceedings of the HLT/NAACL 2004*, pages 289–296.
- Dan Klein and Christopher D. Manning. 2002. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pages 3–10.
- Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *CoNLL 2002: Proceedings of the 6th Conference on Natural Language Learning 2002 (COLING 2002 Post-Conference Workshops)*, pages 63–69.
- Taku Kudo and Yuji Matsumoto. 2005. Japanese dependency parsing using relative preference of dependency. *Transactions of Information Processing Society of Japan*, 46(4):1082–1092. (in Japanese).
- Sadao Kurohashi and Makoto Nagao. 1994. Kn parser: Japanese dependency/case structure analyzer. In *Proceedings of the Workshop on Sharable Natural Language Resources*, pages 48–55.
- Sadao Kurohashi and Makoto Nagao. 1998a. Building a Japanese parsed corpus while improving the parsing system. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, pages 719–724.
- Sadao Kurohashi and Makoto Nagao. 1998b. *Japanese Morphological Analysis System JUMAN version 3.5*. Department of Informatics, Kyoto University. (in Japanese).
- Manabu Sassano. 2004. Linear-time dependency analysis for Japanese. In *Proceedings of the COLING 2004*, pages 8–14.
- Kiyoaki Shirai, Kentaro Inui, Takenobu Tokunaga, and Hozumi Tanaka. 1998. An empirical evaluation on statistical parsing of Japanese sentences using lexical association statistics. In *Proceedings of the 3rd Conference on EMNLP*, pages 80–87.
- Kiyoaki Shirai. 1998. The integrated natural language processing using statistical information. Technical Report TR98-0004, Department of Computer Science, Tokyo Institute of Technology. (in Japanese).
- Kentaro Torisawa. 2001. An unsupervised method for canonicalization of Japanese postpositions. In *Proceedings of the 6th Natural Language Processing Pacific Rim Symposium (NLPRS)*, pages 211–218.
- Kiyotaka Uchimoto, Satoshi Sekine, and Hitoshi Isahara. 1999. Japanese dependency structure analysis based on maximum entropy models. *Transactions of Information Processing Society of Japan*, 40(9):3397–3407. (in Japanese).
- Kiyotaka Uchimoto, Masaki Murata, Satoshi Sekine, and Hitoshi Isahara. 2000. Dependency model using posterior context. In *Proceedings of the Sixth International Workshop on Parsing Technology (IWPT2000)*, pages 321–322.

Answer Extraction, Semantic Clustering, and Extractive Summarization for Clinical Question Answering

Dina Demner-Fushman^{1,3} and Jimmy Lin^{1,2,3}

¹Department of Computer Science

²College of Information Studies

³Institute for Advanced Computer Studies

University of Maryland

College Park, MD 20742, USA

demner@cs.umd.edu, jimmylin@umd.edu

Abstract

This paper presents a hybrid approach to question answering in the clinical domain that combines techniques from summarization and information retrieval. We tackle a frequently-occurring class of questions that takes the form “What is the best drug treatment for X?” Starting from an initial set of MEDLINE citations, our system first identifies the drugs under study. Abstracts are then clustered using semantic classes from the UMLS ontology. Finally, a short extractive summary is generated for each abstract to populate the clusters. Two evaluations—a manual one focused on short answers and an automatic one focused on the supporting abstracts—demonstrate that our system compares favorably to PubMed, the search system most widely used by physicians today.

1 Introduction

Complex information needs can rarely be addressed by single documents, but rather require the integration of knowledge from multiple sources. This suggests that modern information retrieval systems, which excel at producing ranked lists of documents sorted by relevance, may not be sufficient to provide users with a good overview of the “information landscape”.

Current question answering systems aspire to address this shortcoming by gathering relevant “facts” from multiple documents in response to information needs. The so-called “definition” or “other” questions at recent TREC evaluations (Voorhees, 2005) serve as good examples:

“good answers” to these questions include interesting “nuggets” about a particular person, organization, entity, or event.

The importance of cross-document information synthesis has not escaped the attention of other researchers. The last few years have seen a convergence between the question answering and summarization communities (Amigó et al., 2004), as highlighted by the shift from generic to query-focused summaries in the 2005 DUC evaluation (Dang, 2005). Despite a focus on document ranking, different techniques for organizing search results have been explored by information retrieval researchers, as exemplified by techniques based on clustering (Hearst and Pedersen, 1996; Dumais et al., 2001; Lawrie and Croft, 2003).

Our work, which is situated in the domain of clinical medicine, lies at the intersection of question answering, information retrieval, and summarization. We employ answer extraction to identify short answers, semantic clustering to group similar results, and extractive summarization to produce supporting evidence. This paper describes how each of these capabilities contributes to an information system tailored to the requirements of physicians. Two separate evaluations demonstrate the effectiveness of our approach.

2 Clinical Information Needs

Although the need to answer questions related to patient care has been well documented (Covell et al., 1985; Gorman et al., 1994; Ely et al., 1999), studies have shown that existing search systems, e.g., PubMed, the U.S. National Library of Medicine’s search engine, are often unable to supply physicians with clinically-relevant answers in a timely manner (Gorman et al., 1994; Chambliss and Conley, 1996). Clinical information

Disease: Chronic Prostatitis ► anti-microbial 1. [temafloxacin] Treatment of chronic bacterial prostatitis with temafloxacin. Temafloxacin 400 mg b.i.d. administered orally for 28 days represents a safe and effective treatment for chronic bacterial prostatitis. 2. [ofloxacin] Ofloxacin in the management of complicated urinary tract infections, including prostatitis. In chronic bacterial prostatitis, results to date suggest that ofloxacin may be more effective clinically and as effective microbiologically as carbenicillin. 3. ... ► Alpha-adrenergic blocking agent 1. [terazosine] Terazosin therapy for chronic prostatitis/chronic pelvic pain syndrome: a randomized, placebo controlled trial. CONCLUSIONS: Terazosin proved superior to placebo for patients with chronic prostatitis/chronic pelvic pain syndrome who had not received alpha-blockers previously. 2. ...

Table 1: System response to the question “What is the best drug treatment for chronic prostatitis?”

systems for decision support represent a potentially high-impact application. From a research perspective, the clinical domain is attractive because substantial knowledge has already been codified in the Unified Medical Language System (UMLS) (Lindberg et al., 1993). The 2004 version of the UMLS Metathesaurus contains information about over 1 million biomedical concepts and 5 million concept names. This and related resources allow us to explore knowledge-based techniques with substantially less upfront investment.

Naturally, physicians have a wide spectrum of information needs, ranging from questions about the selection of treatment options to questions about legal issues. To make the retrieval problem more tractable, we focus on a subset of therapy questions taking the form “What is the best drug treatment for X ?”, where X can be any number of diseases. We have chosen to tackle this class of questions because studies of physicians’ behavior in natural settings have revealed that such questions occur quite frequently (Ely et al., 1999). By leveraging the natural distribution of clinical information needs, we can make the greatest impact with the least effort.

Our research follows the principles of evidence-based medicine (EBM) (Sackett et al., 2000), which provides a well-defined model to guide the process of clinical question answering. EBM is a widely-accepted paradigm for medical practice that involves the explicit use of current best evidence, i.e., high-quality patient-centered clinical research reported in the primary medical literature,

to make decisions about patient care. As shown by previous work (Cogdill and Moore, 1997; De Groote and Dorsch, 2003), citations from the MEDLINE database (maintained by the U.S. National Library of Medicine) serve as a good source of clinical evidence. As a result of these findings, our work focuses on MEDLINE abstracts as the source for answers.

3 Question Answering Approach

Conflicting desiderata shape the characteristics of “answers” to clinical questions. On the one hand, conciseness is paramount. Physicians are always under time pressure when making decisions, and information overload is a serious concern. Furthermore, we ultimately envision deploying advanced retrieval systems in portable packages such as PDAs to serve as tools in bedside interactions (Hauser et al., 2004). The small form factor of such devices limits the amount of text that can be displayed. However, conciseness exists in tension with completeness. For physicians, the implications of making potentially life-altering decisions mean that all evidence must be carefully examined in context. For example, the efficacy of a drug is always framed in the context of a specific sample population, over a set duration, at some fixed dosage, etc. A physician simply cannot recommend a particular course of action without considering all these factors.

Our approach seeks to balance conciseness and completeness by providing hierarchical and inter-

active “answers” that support multiple levels of drill-down. A partial example is shown in Figure 1. Top-level answers to “What is the best drug treatment for X?” consist of categories of drugs that may be of interest to the physician. Each category is associated with a cluster of abstracts from MEDLINE about that particular treatment option. Drilling down into a cluster, the physician is presented with extractive summaries of abstracts that outline the clinical findings. To obtain more detail, the physician can pull up the complete abstract text, and finally the electronic version of the entire article (if available). In the example shown in Figure 1, the physician can see that two classes of drugs (anti-microbial and alpha-adrenergic blocking agent) are relevant for the disease “chronic prostatitis”. Drilling down into the first cluster, the physician can see summarized evidence for two specific types of anti-microbials (temafloxacin and ofloxacin) extracted from MEDLINE abstracts.

Three major capabilities are required to produce the “answers” described above. First, the system must accurately identify the drugs under study in an abstract. Second, the system must group abstracts based on these substances in a meaningful way. Third, the system must generate short summaries of the clinical findings. We describe a clinical question answering system that implements exactly these capabilities (answer extraction, semantic clustering, and extractive summarization).

4 System Implementation

Our work is primarily concerned with synthesizing coherent answers from a set of search results—the actual source of these results is not important. For convenience, we employ MEDLINE citations retrieved by the PubMed search engine (which also serves as a baseline for comparison). Given an initial set of citations, answer generation proceeds in three phases, described below.

4.1 Answer Extraction

Given a set of abstracts, our system first identifies the drugs under study; these later become the short answers. In the parlance of evidence-based medicine, drugs fall into the category of “interventions”, which encompasses everything from surgical procedures to diagnostic tests.

Our extractor for interventions relies on MetaMap (Aronson, 2001), a program that automatically identifies entities corresponding to

UMLS concepts. UMLS has an extensive coverage of drugs, falling under the semantic type PHARMACOLOGICAL SUBSTANCE and a few others. All such entities are identified as candidates and each is scored based on a number of features: its position in the abstract, its frequency of occurrence, etc. A separate evaluation on a blind test set demonstrates that our extractor is able to accurately recognize the interventions in a MEDLINE abstract; see details in (Demner-Fushman and Lin, 2005; Demner-Fushman and Lin, 2006 in press).

4.2 Semantic Clustering

Retrieved MEDLINE citations are organized into semantic clusters based on the main interventions identified in the abstract text. We employed a variant of the hierarchical agglomerative clustering algorithm (Zhao and Karypis, 2002) that utilizes semantic relationships within UMLS to compute similarities between interventions.

Iteratively, we group abstracts whose interventions fall under a common ancestor, i.e., a hypernym. The more generic ancestor concept (i.e., the class of drugs) is then used as the cluster label. The process repeats until no new clusters can be formed. In order to preserve granularity at the level of practical clinical interest, the tops of the UMLS hierarchy were truncated; for example, the MeSH category “Chemical and Drugs” is too general to be useful. This process was manually performed during system development. We decided to allow an abstract to appear in multiple clusters if more than one intervention was identified, e.g., if the abstract compared the efficacy of two treatments. Once the clusters have been formed, all citations are then sorted in the order of the original PubMed results, with the most abstract UMLS concept as the cluster label. Clusters themselves are sorted in decreasing size under the assumption that more clinical research is devoted to more pertinent types of drugs.

Returning to the example in Figure 1, the abstracts about temafloxacin and ofloxacin were clustered together because both drugs are hyponyms of anti-microbials within the UMLS ontology. As can be seen, this semantic resource provides a powerful tool for organizing search results.

4.3 Extractive Summarization

For each MEDLINE citation, our system generates a short extractive summary consisting of three elements: the main intervention (which is usu-

ally more specific than the cluster label); the title of the abstract; and the top-scoring outcome sentence. The “outcome”, another term from evidence-based medicine, asserts the clinical findings of a study, and is typically found towards the end of a MEDLINE abstract. In our case, outcome sentences state the efficacy of a drug in treating a particular disease. Previously, we have built an outcome extractor capable of identifying such sentences in MEDLINE abstracts using supervised machine learning techniques (Demner-Fushman and Lin, 2005; Demner-Fushman and Lin, 2006 in press). Evaluation on a blind held-out test set shows high classification accuracy.

5 Evaluation Methodology

Given that our work draws from QA, IR, and summarization, a proper evaluation that captures the salient characteristics of our system proved to be quite challenging. Overall, evaluation can be decomposed into two separate components: locating a suitable resource to serve as ground truth and leveraging it to assess system responses.

It is not difficult to find disease-specific pharmacology resources. We employed *Clinical Evidence* (CE), a periodic report created by the British Medical Journal (BMJ) Publishing Group that summarizes the best known drugs for a few dozen diseases. Note that the existence of such secondary sources does not obviate the need for automated systems because they are perpetually falling out of date due to rapid advances in medicine. Furthermore, such reports are currently created by highly-experienced physicians, which is an expensive and time-consuming process.

For each disease, CE classifies drugs into one of six categories: beneficial, likely beneficial, trade-offs (i.e., may have adverse side effects), unknown, unlikely beneficial, and harmful. Included with each entry is a list of references—citations consulted by the editors in compiling the resource. Although the completeness of the drugs enumerated in CE is questionable, it nevertheless can be viewed as “authoritative”.

5.1 Previous Work

How can we leverage a resource such as CE to assess the responses generated by our system? A survey of evaluation methodologies reveals shortcomings in existing techniques.

Answers to factoid questions are automatically

scored using regular expression patterns (Lin, 2005). In our application, this is inadequate for many reasons: there is rarely an exact string match between system output and drugs mentioned in CE, primarily due to synonymy (for example, alpha-adrenergic blocking agent and α -blocker refer to the same class of drugs) and ontological mismatch (for example, CE might mention beta-agonists, while a retrieved abstract discusses formoterol, which is a specific representative of beta-agonists). Furthermore, while this evaluation method can tell us if the drugs proposed by the system are “good”, it cannot measure how well the answer is supported by MEDLINE citations; recall that answer justification is important for physicians.

The nugget evaluation methodology (Voorhees, 2005) developed for scoring answers to complex questions is not suitable for our task, since there is no coherent notion of an “answer text” that the user reads end-to-end. Furthermore, it is unclear what exactly a “nugget” in this case would be. For similar reasons, methodologies for summarization evaluation are also of little help. Typically, system-generated summaries are either evaluated manually by humans (which is expensive and time-consuming) or automatically using a metric such as ROUGE, which compares system output against a number of reference summaries. The interactive nature of our answers violates the assumption that systems’ responses are static text segments. Furthermore, it is unclear what exactly should go into a reference summary, because physicians may want varying amounts of detail depending on familiarity with the disease and patient-specific factors.

Evaluation methodologies from information retrieval are also inappropriate. User studies have previously been employed to examine the effect of categorized search results. However, they often conflate the effectiveness of the interface with that of the underlying algorithms. For example, Dumais et al. (2001) found significant differences in task performance based on different ways of using purely presentational devices such as mouseovers, expandable lists, etc. While interface design is clearly important, it is not the focus of our work.

Clustering techniques have also been evaluated in the same manner as text classification algorithms, in terms of precision, recall, etc. based on some ground truth (Zhao and Karypis, 2002).

This, however, assumes the existence of stable, invariant categories, which is not the case since our output clusters are query-specific. Although it may be possible to manually create “reference clusters”, we lack sufficient resources to develop such a data set. Furthermore, it is unclear if sufficient interannotator agreement can be obtained to support meaningful evaluation.

Ultimately, we devised two separate evaluations to assess the quality of our system output based on the techniques discussed above. The first is a manual evaluation focused on the cluster labels (i.e., drug categories), based on a factoid QA evaluation methodology. The second is an automatic evaluation of the retrieved abstracts using ROUGE, drawing elements from summarization evaluation. Details of the evaluation setup and results are preceded by a description of the test collection we created from *CE*.

5.2 Test Collection

We were able to mine the June 2004 edition of *Clinical Evidence* to create a test collection for system evaluation. We randomly selected thirty diseases, generating a development set of five questions and a test set of twenty-five questions. Some examples include: acute asthma, chronic prostatitis, community acquired pneumonia, and erectile dysfunction. *CE* listed an average of 11.3 interventions per disease; of those, 2.3 on average were marked as beneficial and 1.9 as likely beneficial. On average, there were 48.4 references associated with each disease, representing the articles consulted during the compilation of *CE* itself. Of those, 34.7 citations on average appeared in MEDLINE; we gathered all these abstracts, which serve as the reference summaries for our ROUGE-based automatic evaluation.

Since the focus of our work is not on retrieval algorithms per se, we employed PubMed to fetch an initial set of MEDLINE citations and performed answer synthesis using those results. The PubMed citations also serve as a baseline, since it represents a system commonly used by physicians.

In order to obtain the best possible set of citations, the first author (an experienced PubMed searcher), manually formulated queries, taking advantage of MeSH (Medical Subject Headings) terms when available. MeSH terms are controlled vocabulary concepts assigned manually by trained medical indexers (based on the full text of the ar-

ticles), and encode a substantial amount of knowledge about the contents of the citation. PubMed allows searches on MeSH terms, which usually yield accurate results. In addition, we limited retrieved citations to those that have the MeSH heading “drug therapy” and those that describe a clinical trial (another metadata field). Finally, we restricted the date range of the queries so that abstracts published after our version of *CE* were excluded. Although the query formulation process currently requires a human, we envision automating this step using a template-based approach in the future.

6 System Evaluation

We adapted existing techniques to evaluate our system in two separate ways: a factoid-style manual evaluation focused on short answers and an automatic evaluation with ROUGE using *CE*-cited abstracts as the reference summaries. The setup and results for both are detailed below.

6.1 Manual Evaluation of Short Answers

In our manual evaluation, system outputs were assessed as if they were answers to factoid questions. We gathered three different sets of answers. For the baseline, we used the main intervention from each of the first three PubMed citations. For our test condition, we considered the three largest clusters, taking the main intervention from the first abstract in each cluster. This yields three drugs that are at the same level of ontological granularity as those extracted from the unclustered PubMed citations. For our third condition, we assumed the existence of an oracle which selects the three best clusters (as determined by the first author, a medical doctor). From each of these three clusters, we extracted the main intervention of the first abstracts. This oracle condition represents an achievable upper bound with a human in the loop. Physicians are highly-trained professionals that already have significant domain knowledge. Faced with a small number of choices, it is likely that they will be able to select the most promising cluster, even if they did not previously know it.

This preparation yielded up to nine drug names, three from each experimental condition. For short, we refer to these as PubMed, Cluster, and Oracle, respectively. After blinding the source of the drugs and removing duplicates, each short answer was presented to the first author for evaluation. Since

	<i>Clinical Evidence</i>							Physician		
	B	LB	T	U	UB	H	N	Good	Okay	Bad
PubMed	0.200	0.213	0.160	0.053	0.000	0.013	0.360	0.600	0.227	0.173
Cluster	0.387	0.173	0.173	0.027	0.000	0.000	0.240	0.827	0.133	0.040
Oracle	0.400	0.200	0.133	0.093	0.013	0.000	0.160	0.893	0.093	0.013

Table 2: Manual evaluation of short answers: distribution of system answers with respect to *CE* categories (left side) and with respect to the assessor’s own expertise (right side). (Key: B=beneficial, LB=likely beneficial, T=tradeoffs, U=unknown, UB=unlikely beneficial, H=harmful, N=not in *CE*)

the assessor had no idea from which condition an answer came, this process guarded against assessor bias.

Each answer was evaluated in two different ways: first, with respect to the ground truth in *CE*, and second, using the assessor’s own medical expertise. In the first set of judgments, the assessor determined which of the six categories (beneficial, likely beneficial, tradeoffs, unknown, unlikely beneficial, harmful) the system answer belonged to, based on the *CE* recommendations. As we have discussed previously, a human (with sufficient domain knowledge) is required to perform this matching due to synonymy and differences in ontological granularity. However, note that the assessor only considered the drug name when making this categorization. In the second set of judgments, the assessor separately determined if the short answer was “good”, “okay” (marginal), or “bad” based both on *CE* and her own experience, taking into account the abstract title and the top-scoring outcome sentence (and if necessary, the entire abstract text).

Results of this manual evaluation are presented in Table 2, which shows the distribution of judgments for the three experimental conditions. For baseline PubMed, 20% of the examined drugs fell in the beneficial category; the values are 39% for the Cluster condition and 40% for the Oracle condition. In terms of short answers, our system returns approximately twice as many beneficial drugs as the baseline, a marked increase in answer accuracy. Note that a large fraction of the drugs evaluated were not found in *CE* at all, which provides an estimate of its coverage. In terms of the assessor’s own judgments, 60% of PubMed short answers were found to be “good”, compared to 83% and 89% for the Cluster and Oracle conditions, respectively. From a factoid QA point of view, we can conclude that our system outperforms the PubMed baseline.

6.2 Automatic Evaluation of Abstracts

A major limitation of the factoid-based evaluation methodology is that it does not measure the quality of the abstracts from which the short answers were extracted. Since we lacked the necessary resources to manually gather abstract-level judgments for evaluation, we sought an alternative.

Fortunately, *CE* can be leveraged to assess the “goodness” of abstracts automatically. We assume that references cited in *CE* are examples of high quality abstracts, since they were used in generating the drug recommendations. Following standard assumptions made in summarization evaluation, we considered abstracts that are similar in content with these “reference abstracts” to also be “good” (i.e., relevant). Similarity in content can be quantified with ROUGE.

Since physicians demand high precision, we assess the cumulative relevance after the first, second, and third abstract that the clinician is likely to have examined (where the relevance for each individual abstract is given by its ROUGE-1 precision score). For the baseline PubMed condition, the examined abstracts simply correspond to the first three hits in the result set. For our test system, we developed three different orderings. The first, which we term cluster round-robin, selects the first abstract from the top three clusters (by size). The second, which we term oracle cluster order, selects three abstracts from the best cluster, assuming the existence of an oracle that informs the system. The third, which we term oracle round-robin, selects the first abstract from each of the three best clusters (also determined by an oracle).

Results of this evaluation are shown in Table 3. The columns show the cumulative relevance (i.e., ROUGE score) after examining the first, second, and third abstract, under the different ordering conditions. To determine statistical significance, we applied the Wilcoxon signed-rank test, the

	Rank 1	Rank 2	Rank 3
PubMed Ranked List	0.170	0.349	0.523
Cluster Round-Robin	0.181 (+6.3%) [°]	0.356 (+2.1%) [°]	0.526 (+0.5%) [°]
Oracle Cluster Order	0.206 (+21.5%) ^Δ	0.392 (+12.6%) ^Δ	0.597 (+14.0%) [▲]
Oracle Round-Robin	0.206 (+21.5%) ^Δ	0.396 (+13.6%) ^Δ	0.586 (+11.9%) [▲]

Table 3: Cumulative relevance after examining the first, second, and third abstracts, according to different orderings. (° denotes n.s., Δ denotes sig. at 0.90, ▲ denotes sig. at 0.95)

standard non-parametric test for applications of this type. Due to the relatively small test set (only 25 questions), the increase in cumulative relevance exhibited by the cluster round-robin condition is not statistically significant. However, differences for the oracle conditions were significant.

7 Discussion and Related Work

According to two separate evaluations, it appears that our system outperforms the PubMed baseline. However, our approach provides more advantages over a linear result set that are not highlighted in these evaluations. Although difficult to quantify, categorized results provide an overview of the information landscape that is difficult to acquire by simply browsing a ranked list—user studies of categorized search have affirmed its value (Hearst and Pedersen, 1996; Dumais et al., 2001). One main advantage we see in our application is better “redundancy management”. With a ranked list, the physician may be forced to browse through multiple redundant abstracts that discuss the same or similar drugs to get a sense of the different treatment options. With our cluster-based approach, however, potentially redundant information is grouped together, since interventions discussed in a particular cluster are ontologically related through UMLS. The physician can examine different clusters for a broad overview, or peruse multiple abstracts within a cluster for a more thorough review of the evidence. Our cluster-based system is able to support both types of behaviors.

This work demonstrates the value of semantic resources in the question answering process, since our approach makes extensive use of the UMLS ontology in all phases of answer synthesis. The coverage of individual drugs, as well as the relationship between different types of drugs within UMLS enables both answer extraction and semantic clustering. As detailed in (Demner-Fushman and Lin, 2006 in press), UMLS-based features are also critical in the identification of clinical out-

comes, on which our extractive summaries are based. As a point of comparison, we also implemented a purely term-based approach to clustering PubMed citations. The results are so incoherent that a formal evaluation would prove to be meaningless. Semantic relations between drugs, as captured in UMLS, provide an effective method for organizing results—these relations cannot be captured by keyword content alone. Furthermore, term-based approaches suffer from the cluster labeling problem: it is difficult to automatically generate a short heading that describes cluster content.

Nevertheless, there are a number of assumptions behind our work that are worth pointing out. First, we assume a high quality initial result set. Since the class of questions we examine translates naturally into accurate PubMed queries that can make full use of human-assigned MeSH terms, the overall quality of the initial citations can be assured. Related work in retrieval algorithms (Demner-Fushman and Lin, 2006 in press) shows that accurate relevance scoring of MEDLINE citations in response to more general clinical questions is possible.

Second, our system does not actually perform semantic processing to determine the efficacy of a drug: it only recognizes “topics” and outcome sentences that state clinical findings. Since the system by default orders the clusters based on size, it implicitly equates “most popular drug” with “best drug”. Although this assumption is false, we have observed in practice that more-studied drugs are more likely to be beneficial.

In contrast with the genomics domain, which has received much attention from both the IR and NLP communities, retrieval systems for the clinical domain represent an underexplored area of research. Although individual components that attempt to operationalize principles of evidence-based medicine do exist (Mendonça and Cimino, 2001; Niu and Hirst, 2004), complete end-to-end clinical question answering systems are dif-

difficult to find. Within the context of the PERSIVAL project (McKeown et al., 2003), researchers at Columbia have developed a system that leverages patient records to rerank search results. Since the focus is on personalized summaries, this work can be viewed as complementary to our own.

8 Conclusion

The primary contribution of this work is the development of a clinical question answering system that caters to the unique requirements of physicians, who demand both conciseness and completeness. These competing factors can be balanced in a system's response by providing multiple levels of drill-down that allow the information space to be viewed at different levels of granularity. We have chosen to implement these capabilities through answer extraction, semantic clustering, and extractive summarization. Two separate evaluations demonstrate that our system outperforms the PubMed baseline, illustrating the effectiveness of a hybrid approach that leverages semantic resources.

9 Acknowledgments

This work was supported in part by the U.S. National Library of Medicine. The second author thanks Esther and Kiri for their loving support.

References

- E. Amigó, J. Gonzalo, V. Peinado, A. Peñas, and F. Verdejo. 2004. An empirical study of information synthesis task. In *ACL 2004*.
- A. Aronson. 2001. Effective mapping of biomedical text to the UMLS Metathesaurus: The MetaMap program. In *AMIA 2001*.
- M. Chambliss and J. Conley. 1996. Answering clinical questions. *The Journal of Family Practice*, 43:140–144.
- K. Cogdill and M. Moore. 1997. First-year medical students' information needs and resource selection: Responses to a clinical scenario. *Bulletin of the Medical Library Association*, 85(1):51–54.
- D. Covell, G. Uman, and P. Manning. 1985. Information needs in office practice: Are they being met? *Annals of Internal Medicine*, 103(4):596–599.
- H. Dang. 2005. Overview of DUC 2005. In *DUC 2005 Workshop at HLT/EMNLP 2005*.
- S. De Groote and J. Dorsch. 2003. Measuring use patterns of online journals and databases. *Journal of the Medical Library Association*, 91(2):231–240.
- D. Demner-Fushman and J. Lin. 2005. Knowledge extraction for clinical question answering: Preliminary results. In *AAAI 2005 Workshop on QA in Restricted Domains*.
- D. Demner-Fushman and J. Lin. 2006, in press. Answering clinical questions with knowledge-based and statistical techniques. *Comp. Ling.*
- S. Dumais, E. Cutrell, and H. Chen. 2001. Optimizing search by showing results in context. In *CHI 2001*.
- J. Ely, J. Osherooff, M. Ebell, G. Bergus, B. Levy, M. Chambliss, and E. Evans. 1999. Analysis of questions asked by family doctors regarding patient care. *BMJ*, 319:358–361.
- P. Gorman, J. Ash, and L. Wykoff. 1994. Can primary care physicians' questions be answered using the medical journal literature? *Bulletin of the Medical Library Association*, 82(2):140–146, April.
- S. Hauser, D. Demner-Fushman, G. Ford, and G. Thoma. 2004. PubMed on Tap: Discovering design principles for online information delivery to handheld computers. In *MEDINFO 2004*.
- M. Hearst and J. Pedersen. 1996. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *SIGIR 1996*.
- D. Lawrie and W. Croft. 2003. Generating hierarchical summaries for Web searches. In *SIGIR 2003*.
- J. Lin. 2005. Evaluation of resources for question answering evaluation. In *SIGIR 2005*.
- D. Lindberg, B. Humphreys, and A. McCray. 1993. The Unified Medical Language System. *Methods of Information in Medicine*, 32(4):281–291.
- K. McKeown, N. Elhadad, and V. Hatzivassiloglou. 2003. Leveraging a common representation for personalized search and summarization in a medical digital library. In *JCDL 2003*.
- E. Mendonça and J. Cimino. 2001. Building a knowledge base to support a digital library. In *MEDINFO 2001*.
- Y. Niu and G. Hirst. 2004. Analysis of semantic classes in medical text for question answering. In *ACL 2004 Workshop on QA in Restricted Domains*.
- David Sackett, Sharon Straus, W. Richardson, William Rosenberg, and R. Haynes. 2000. *Evidence-Based Medicine: How to Practice and Teach EBM*. Churchill Livingstone, second edition.
- E. Voorhees. 2005. Using question series to evaluate question answering system effectiveness. In *HLT/EMNLP 2005*.
- Y. Zhao and G. Karypis. 2002. Evaluation of hierarchical clustering algorithms for document datasets. In *CIKM 2002*.

Discovering *asymmetric* entailment relations between verbs using selectional preferences

Fabio Massimo Zanzotto

DISCo

University of Milano-Bicocca

Via Bicocca degli Arcimboldi 8, Milano, Italy

zanzotto@disco.unimib.it

Marco Pennacchiotti, Maria Teresa Pazienza

ART Group - DISP

University of Rome “Tor Vergata”

Viale del Politecnico 1, Roma, Italy

{pennacchiotti, pazienza}@info.uniroma2.it

Abstract

In this paper we investigate a novel method to detect *asymmetric* entailment relations between verbs. Our starting point is the idea that some point-wise verb selectional preferences carry relevant semantic information. Experiments using WordNet as a gold standard show promising results. Where applicable, our method, used in combination with other approaches, significantly increases the performance of entailment detection. A combined approach including our model improves the AROC of 5% absolute points with respect to standard models.

1 Introduction

Natural Language Processing applications often need to rely on large amount of lexical semantic knowledge to achieve good performances. Asymmetric verb relations are part of it. Consider for example the question “*What college did Marcus Camby play for?*”. A question answering (QA) system could find the answer in the snippet “*Marcus Camby won for Massachusetts*” as the question verb *play* is related to the verb *win*. The vice-versa is not true. If the question is “*What college did Marcus Camby won for?*”, the snippet “*Marcus Camby played for Massachusetts*” cannot be used. *Winnig* entails *playing* but not vice-versa, as the relation between *win* and *play* is asymmetric.

Recently, many automatically built verb lexical-semantic resources have been proposed to support lexical inferences, such as (Resnik and Diab, 2000; Lin and Pantel, 2001; Glickman and Dagan, 2003). All these resources focus on *symmetric* semantic relations, such as *verb similarity*. Yet,

not enough attention has been paid so far to the study of *asymmetric* verb relations, that are often the only way to produce correct inferences, as the example above shows.

In this paper we propose a novel approach to identify asymmetric relations between verbs. The main idea is that asymmetric entailment relations between verbs can be analysed in the context of class-level and word-level selectional preferences (Resnik, 1993). Selectional preferences indicate an entailment relation between a verb and its arguments. For example, the selectional preference $\{human\}$ *win* may be read as a *smooth* constraint: *if x is the subject of win then it is likely that x is a human*, i.e. $win(x) \rightarrow human(x)$. It follows that selectional preferences like $\{player\}$ *win* may be read as suggesting the entailment relation $win(x) \rightarrow play(x)$.

Selectional preferences have been often used to infer semantic relations among verbs and to build symmetric semantic resources as in (Resnik and Diab, 2000; Lin and Pantel, 2001; Glickman and Dagan, 2003). However, in those cases these are exploited in a different way. The assumption is that verbs are semantically related if they share similar selectional preferences. Then, according to the Distributional Hypothesis (Harris, 1964), verbs occurring in similar sentences are likely to be semantically related.

The Distributional Hypothesis suggests a generic *equivalence* between words. Related methods can then only discover symmetric relations. These methods can incidentally find verb pairs as (*win,play*) where an asymmetric entailment relation holds, but they cannot state the direction of entailment (e.g., $win \rightarrow play$).

As we investigate the idea that a single relevant verb selectional preference (as $\{player\}$

win) could produce an entailment relation between verbs, our starting point can not be the Distributional Hypothesis. Our assumption is that some *point-wise assertions* carry relevant semantic information (as in (Robison, 1970)). We do not derive a semantic relation between verbs by comparing their selectional preferences, but we use point-wise corpus-induced selectional preferences.

The rest of the paper is organised as follows. In Sec. 2 we discuss the intuition behind our research. In Sec. 3 we describe different types of verb entailment. In Sec. 4 we introduce our model for detecting entailment relations among verbs. In Sec. 5 we review related works that are used both for comparison and for building combined methods. Finally, in Sec. 6 we present the results of our experiments.

2 Selectional Preferences and Verb Entailment

Selectional restrictions are strictly related to entailment. When a verb or a noun *expects* a modifier having a predefined property it means that the truth value of the related sentences strongly depends on the satisfiability of these expectations. For example, “*X is blue*” implies the expectation that *X* has a colour. This expectation may be seen as a sort of entailment between “being a modifier of that verb or noun” and “having a property”. If the sentence is “*The number three is blue*”, then the sentence is false as the underlying entailment $blue(x) \rightarrow has\_colour(x)$ does not hold (cf. (Resnik, 1993)). In particular, this rule applies to verb *logical* subjects: if a verb *v* has a selectional restriction requiring its logical subjects to satisfy a property *c*, it follows that the implication:

$$v(x) \rightarrow c(x)$$

should be verified for each logical subject *x* of the verb *v*. The implication can also be read as: if *x* has the property of doing the action *v* this implies that *x* has the property *c*. For example, if the verb is *to eat*, the selectional restrictions of *to eat* would imply that its subjects have the property of being *animate*.

Resnik (1993) introduced a smoothed version of selectional restrictions called *selectional preferences*. These preferences describe the *desired properties* a modifier should have. The claim is that if a selectional preference *holds*, it is more probable that *x* has the property *c* given that it

modifies *v* rather than *x* has this property in the general case, i.e.:

$$p(c(x)|v(x)) > p(c(x)) \quad (1)$$

The probabilistic setting of selectional preferences also suggests an entailment: the implication $v(x) \rightarrow c(x)$ holds with a given degree of certainty. This definition is strictly related to the probabilistic textual entailment setting in (Glickman et al., 2005).

We can use selectional preferences, intended as probabilistic entailment rules, to induce entailment relations among verbs. In our case, if a verb v_t expects that the subject “has the property of doing an action v_h ”, this may be used to induce that the verb v_t probably entails the verb v_h , i.e.:

$$v_t(x) \rightarrow v_h(x) \quad (2)$$

As for class-based selectional preference acquisition, corpora can be used to estimate these particular kinds of preferences. For example, the sentence “*John McEnroe won the match...*” contributes to probability estimation of the class-based selectional preference $win(x) \rightarrow human(x)$ (since *John McEnroe* is a *human*). In particular contexts, it contributes also to the induction of the entailment relation between *win* and *play*, as *John McEnroe* has the *property of playing*. However, as the example shows, classes relevant for acquiring selectional preferences (such as *human*) are explicit, as they do not depend from the context. On the contrary, properties such as “having the property of doing an action” are less explicit, as they depend more strongly on the context of sentences. Thus, properties useful to derive entailment relations among verbs are more difficult to find. For example, it is easier to derive that *John McEnroe* is a *human* (as it is a stable property) than that he has the property of *playing*. Indeed, this latter property may be relevant only in the context of the previous sentence.

However, there is a way to overcome this limitation: *agentive nouns* such as *runner* make explicit this kind of property and often play subject roles in sentences. Agentive nouns usually denote the “*doer*” or “*performer*” of some action. This is exactly what is needed to make clearer the relevant property $v_h(x)$ of the noun playing the logical subject role. The action v_h will be the one entailed by the verb v_t heading the sentence. As an example in the sentence “*the player wins*”, the action *play*

evocated by the agentive noun *player* is entailed by *win*.

3 Verb entailment: a classification

The focus of our study is on verb entailment. A brief review of the WordNet (Miller, 1995) verb hierarchy (one of the main existing resources on verb entailment relations) is useful to better explain the problem and to better understand the applicability of our hypothesis.

In WordNet, verbs are organized in synonymy sets (*synsets*) and different kinds of semantic relations can hold between two verbs (i.e. two *synsets*): *troponymy*, *causation*, *backward-presupposition*, and *temporal inclusion*. All these relations are intended as specific types of *lexical entailment*. According to the definition in (Miller, 1995) lexical entailment holds between two verbs v_t and v_h when the sentence *Someone v_t* entails the sentence *Someone v_h* (e.g. “*Someone wins*” entails “*Someone plays*”). Lexical entailment is then an asymmetric relation. The four types of WordNet lexical entailment can be classified looking at the temporal relation between the entailing verb v_t and the entailed verb v_h .

Troponymy represents the hyponymy relation between verbs. It stands when v_t and v_h are temporally co-extensive, that is, when the actions described by v_t and v_h begin and end at the same times (e.g. *limp*→*walk*). The relation of *temporal inclusion* captures those entailment pairs in which the action of one verb is temporally included in the action of the other (e.g. *snore*→*sleep*). *Backward-presupposition* stands when the entailed verb v_h happens before the entailing verb v_t and it is necessary for v_t . For example, *win* entails *play* via backward-presupposition as it temporally follows and presupposes *play*. Finally, in *causation* the entailing verb v_t necessarily causes v_h . In this case, the temporal relation is thus inverted with respect to backward-presupposition, since v_t precedes v_h . In causation, v_t is always a causative verb of change, while v_h is a resultative stative verb (e.g. *buy*→*own*, and *give*→*have*).

As a final note, it is interesting to notice that the Subject-Verb structure of v_t is generally preserved in v_h for all forms of lexical entailment. The two verbs have the same subject. The only exception is *causation*: in this case the subject of the entailed verb v_h is usually the object of v_t (e.g., *X give Y* → *Y have*). In most cases the subject of v_t carries

out an action that changes the state of the object of v_t , that is then described by v_h .

The intuition described in Sec. 2 is then applicable only for some kinds of verb entailments. First, the *causation* relation can not be captured since the two verbs should have the same subject (cf. eq. (2)). Secondly, troponymy seems to be less interesting than the other relations, since our focus is more on a *logic* type of entailment (i.e., v_t and v_h express two different actions one depending from the other). We then focus our study and our experiments on *backward-presupposition* and *temporal inclusion*. These two relations are organized in WordNet in a single set (called *ent*) parted from troponymy and causation pairs.

4 The method

Our method needs two steps. Firstly (Sec. 4.1), we translate the verb selectional expectations in specific Subject-Verb lexico-syntactic patterns $\mathcal{P}(v_t, v_h)$. Secondly (Sec. 4.2), we define a statistical measure $\mathcal{S}(v_t, v_h)$ that captures the verb preferences. This measure describes how much the relations between target verbs (v_t, v_h) are *stable* and commonly agreed.

Our method to detect verb entailment relations is based on the idea that some point-wise assertions carry relevant semantic information. This idea has been firstly used in (Robison, 1970) and it has been explored for extracting semantic relations between nouns in (Hearst, 1992), where lexico-syntactic patterns are induced by corpora. More recently this method has been applied for structuring terminology in *isa* hierarchies (Morin, 1999) and for learning question-answering patterns (Ravichandran and Hovy, 2002).

4.1 Nominalized textual entailment lexico-syntactic patterns

The idea described in Sec. 2 can be applied to generate Subject-Verb textual entailment lexico-syntactic patterns. It often happens that verbs can undergo an *agentive nominalization*, e.g., *play* vs. *player*. The overall procedure to verify if an entailment between two verbs (v_t, v_h) holds in a point-wise assertion is: *whenever it is possible to apply the agentive nominalization to the hypothesis v_h , scan the corpus to detect those expressions in which the agentified hypothesis verb is the subject of a clause governed by the text verb v_t .*

Given a verb pair (v_t, v_h) the assertion is for-

Lexico-syntactic patterns	
<i>nominalization</i>	$\mathcal{P}_{nom}(v_t, v_h) = \{$ $\text{"agent}(v_h) _{num:sing} \ v_t _{person:third,t:pres},$ $\text{"agent}(v_h) _{num:plur} \ v_t _{person:nothird,t:pres},$ $\text{"agent}(v_h) _{num:sing} \ v_t _{t:past},$ $\text{"agent}(v_h) _{num:plur} \ v_t _{t:past}\}$
<i>happens-before</i> (Chklovski and Pantel, 2004)	$\mathcal{P}_{hb}(v_t, v_h) = \{$ $\text{"}v_h _{t:inf} \text{ and then } v_t _{t:pres},$ $\text{"}v_h _{t:inf} \text{ * and then } v_t _{t:pres},$ $\text{"}v_h _{t:past} \text{ and then } v_t _{t:pres},$ $\text{"}v_h _{t:past} \text{ * and then } v_t _{t:pres},$ $\text{"}v_h _{t:inf} \text{ and later } v_t _{t:pres},$ $\text{"}v_h _{t:past} \text{ and later } v_t _{t:pres},$ $\text{"}v_h _{t:inf} \text{ and subsequently } v_t _{t:pres},$ $\text{"}v_h _{t:past} \text{ and subsequently } v_t _{t:pres},$ $\text{"}v_h _{t:inf} \text{ and eventually } v_t _{t:pres},$ $\text{"}v_h _{t:past} \text{ and eventually } v_t _{t:pres}\}$
<i>probabilistic entailment</i> (Glickman et al., 2005)	$\mathcal{P}_{pe}(v_t, v_h) = \{$ $\text{"}v_h _{person:third,t:pres} \wedge \text{"}v_t _{person:third,t:pres},$ $\text{"}v_h _{t:past} \wedge \text{"}v_t _{t:past},$ $\text{"}v_h _{t:pres,cont} \wedge \text{"}v_t _{t:pres,cont},$ $\text{"}v_h _{person:nothird,t:pres} \wedge \text{"}v_t _{person:nothird,t:pres}\}$
<i>additional sets</i>	$\mathcal{F}_{agent}(v) = \{\text{"agent}(v) _{num:sing}, \text{"agent}(v) _{num:plur}\}$ $\mathcal{F}(v) = \{\text{"}v _{person:third,t:present},$ $\text{"}v _{person:nothird,t:present}, \text{"}v _{t:past}\}$ $\mathcal{F}_{all}(v) = \{\text{"}v _{person:third,t:pres}, \text{"}v _{t:pres,cont},$ $\text{"}v _{person:nothird,t:present}, \text{"}v _{t:past}\}$

Table 1: Nominalization and related textual entailment lexico-syntactic patterns

malized in a set of textual entailment lexico-syntactic patterns, that we call *nominalized patterns* $\mathcal{P}_{nom}(v_t, v_h)$. This set is described in Tab. 1. $agent(v)$ is the noun deriving from the agentification of the verb v . Elements such as $l|_{f_1, \dots, f_N}$ are the tokens generated from lemmas l by applying constraints expressed via the feature-value pairs $f_1, \dots, f_N$. For example, in the case of the verbs *play* and *win*, the related set of textual entailment expressions derived from the patterns are $\mathcal{P}_{nom}(win, play) = \{\text{"player wins", "players win", "player won", "players won"}\}$. In the experiments hereafter described, the required verbal forms have been obtained using the publicly available morphological tools described in (Minnen et al., 2001). Simple heuristics have been used to produce the agentive nominalizations of verbs¹.

Two more sets of expressions, $\mathcal{F}_{agent}(v)$ and $\mathcal{F}(v)$ representing the single events in the pair, are needed for the second step (Sec. 4.2). This two *additional sets* are described in Tab. 1. In the example, the derived expressions are $\mathcal{F}_{agent}(play) = \{\text{"player", "players"}\}$ and $\mathcal{F}(win) = \{\text{"wins", "won"}\}$.

4.2 Measures to estimate the entailment strength

The above textual entailment patterns define *point-wise* entailment assertions. If pattern instances are found in texts, the related verb-subject pairs suggest but not confirm a verb selectional preference.

¹Agentive nominalization has been obtained adding “-er” to the verb root taking into account possible special cases such as verbs ending in “-y”. A form is retained as a correct nominalization if it is in WordNet.

The related entailment can not be considered commonly agreed. For example, the sentence “*Like a writer composes a story, an artist must tell a good story through their work.*” suggests that *compose* entails *write*. However, it may happen that these correctly detected entailments are accidental, that is, the detected relation is only valid for the given text. For example, if the text fragment “*The writers take a simple idea and apply it to this task*” is taken in isolation, it suggests that *take* entails *write*, but this could be questionable.

In order to get rid of these wrong verb pairs, we perform a statistical analysis of the verb selectional preferences over a corpus. This assessment will validate point-wise entailment assertions.

Before introducing the statistical entailment indicator, we provide some definitions. Given a corpus C containing samples, we will refer to the absolute frequency of a textual expression t in the corpus C with $f_C(t)$. The definition can be easily extended to a set of expressions T .

Given a pair v_t and v_h we define the following *entailment strength indicator* $\mathcal{S}(v_t, v_h)$. Specifically, the measure $\mathcal{S}_{nom}(v_t, v_h)$ is derived from point-wise mutual information (Church and Hanks, 1989):

$$\mathcal{S}_{nom}(v_t, v_h) = \log \frac{p(v_t, v_h | nom)}{p(v_t)p(v_h | pers)} \quad (3)$$

where *nom* is the event of having a nominalized textual entailment pattern and *pers* is the event of having an agentive nominalization of verbs. Probabilities are estimated using maximum-likelihood:

$$p(v_t, v_h | nom) \approx \frac{f_C(\mathcal{P}_{nom}(v_t, v_h))}{f_C(\bigcup \mathcal{P}_{nom}(v'_t, v'_h))},$$

$p(v_t) \approx f_C(\mathcal{F}(v_t))/f_C(\bigcup \mathcal{F}(v))$, and $p(v_h|pers) \approx f_C(\mathcal{F}_{agent}(v_h))/f_C(\bigcup \mathcal{F}_{agent}(v))$. Counts are considered useful when they are greater or equal to 3.

The measure $\mathcal{S}_{nom}(v_t, v_h)$ indicates the relatedness between two elements composing a pair, in line with (Chklovski and Pantel, 2004; Glickman et al., 2005) (see Sec. 5). Moreover, if $\mathcal{S}_{nom}(v_t, v_h) > 0$ the verb selectional preference property described in eq. (1) is satisfied.

5 Related “non-distributional” methods and integrated approaches

Our method is a “non-distributional” approach for detecting semantic relations between verbs. We are interested in comparing and integrating our method with similar approaches. We focus on two methods proposed in (Chklovski and Pantel, 2004) and (Glickman et al., 2005). We will shortly review these approaches in light of what introduced in the previous sections. We also present a simple way to combine these different approaches.

The lexico-syntactic patterns introduced in (Chklovski and Pantel, 2004) have been developed to detect six kinds of verb relations: *similarity*, *strength*, *antonymy*, *enablement*, and *happens-before*. Even if, as discussed in (Chklovski and Pantel, 2004), these patterns are not specifically defined as entailment detectors, they can be useful for this purpose. In particular, some of these patterns can be used to investigate the *backward-presupposition* entailment. Verb pairs related by backward-presupposition are not completely temporally included one in the other (cf. Sec. 3): the entailed verb v_h precedes the entailing verb v_t . One set of lexical patterns in (Chklovski and Pantel, 2004) seems to capture the same idea: the *happens-before* (*hb*) patterns. These patterns are used to detect not temporally overlapping verbs, whose relation is semantically very similar to entailment. As we will see in the experimental section (Sec. 6), these patterns show a positive relation with the entailment relation. Tab. 1 reports the *happens-before* lexico-syntactic patterns ($\mathcal{P}_{hb}$) as proposed in (Chklovski and Pantel, 2004). In contrast to what is done in (Chklovski and Pantel, 2004) we decided to directly count patterns derived from different verbal forms and not to use an estimation factor. As in our work, also in (Chklovski and Pantel, 2004), a mutual-information-related measure is used as statistical

indicator. The two methods are then fairly in line.

The other approach we experiment is the “quasi-pattern” used in (Glickman et al., 2005) to capture lexical entailment between two sentences. The pattern has to be discussed in the more general setting of the probabilistic entailment between texts: the *text* T and the *hypothesis* H . The idea is that the implication $T \rightarrow H$ holds (with a degree of truth) if the probability that H holds knowing that T holds is higher than the probability that H holds alone, i.e.:

$$p(H|T) > p(H) \quad (4)$$

This equation is similar to equation (1) in Sec. 2. In (Glickman et al., 2005), words in H and T are supposed to be mutually independent. The previous relation between H and T probabilities then holds also for word pairs. A special case can be applied to verb pairs:

$$p(v_h|v_t) > p(v_h) \quad (5)$$

Equation (5) can be interpreted as the result of the following “quasi-pattern”: the verbs v_h and v_t should co-occur in the same document. It is possible to formalize this idea in the *probabilistic entailment* “quasi-patterns” reported in Tab. 1 as $\mathcal{P}_{pe}$, where verb form variability is taken into consideration. In (Glickman et al., 2005) point-wise mutual information is also a relevant statistical indicator for entailment, as it is strictly related to eq. (5).

For both approaches, the strength indicator $\mathcal{S}_{hb}(v_t, v_h)$ and $\mathcal{S}_{pe}(v_t, v_h)$ are computed as follows:

$$\mathcal{S}_y(v_t, v_h) = \log \frac{p(v_t, v_h|y)}{p(v_t)p(v_h)} \quad (6)$$

where y is *hb* for the *happens-before* patterns and *pe* for the *probabilistic entailment* patterns. Probabilities are estimated as in the previous section.

Considering independent the probability spaces where the three patterns lay (i.e., the space of subject-verb pairs for *nom*, the space of coordinated sentences for *hb*, and the space of documents for *pe*), the combined approaches are obtained summing up $\mathcal{S}_{nom}$, $\mathcal{S}_{hb}$, and $\mathcal{S}_{pe}$. We will then experiment with these combined approaches: *nom + pe*, *nom + hb*, *nom + hb + pe*, and *hb + pe*.

6 Experimental Evaluation

The aim of the experimental evaluation is to establish if the nominalized pattern is useful to help

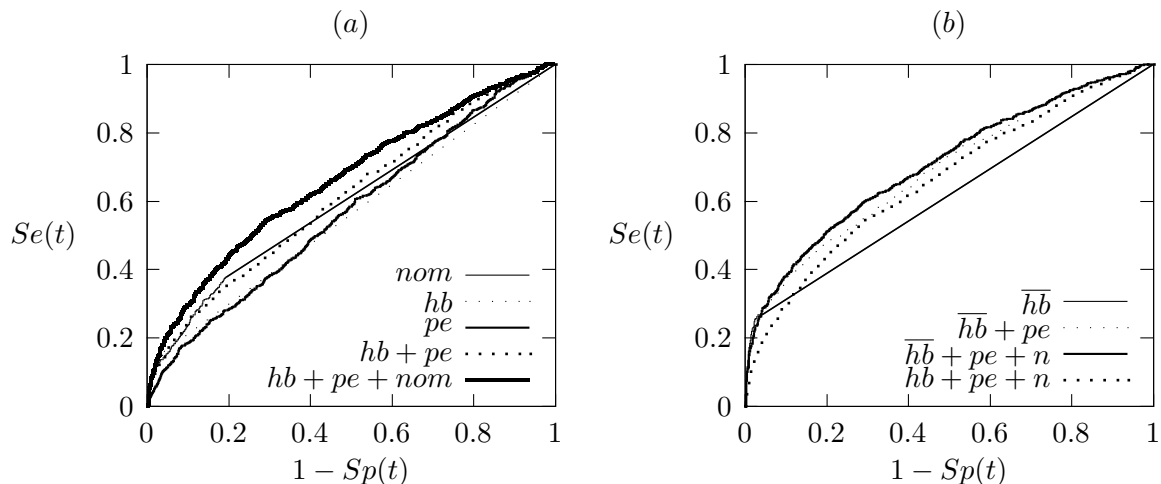


Figure 1: ROC curves of the different methods

in detecting verb entailment. We experiment with the method by itself or in combination with other sets of patterns. We are then interested only in verb pairs where the nominalized pattern is applicable. The best pattern or the best combined method should be the one that gives the highest values of $\mathcal{S}$ to verb pairs in entailment relation, and the lowest value to other pairs.

We need a corpus C over which to estimate probabilities, and two dataset, one of verb entailment pairs, the *True Set* (TS), and another with verbs not in entailment, the *Control Set* (CS). We use the web as corpus C where to estimate $\mathcal{S}_{mi}$ and GoogleTM as a count estimator. The web has been largely employed as a corpus (e.g., (Turney, 2001)). The findings described in (Keller and Lapata, 2003) suggest that the count estimations we need in our study over *Subject-Verb* bigrams are highly correlated to corpus counts.

6.1 Experimental settings

Since we have a predefined (but not exhaustive) set of verb pairs in entailment, i.e. *ent* in WordNet, we cannot replicate a natural distribution of verb pairs that are or are not in entailment. Recall and precision lose sense. Then, the best way to compare the patterns is to use the ROC curve (Green and Swets, 1996) mixing *sensitivity* and *specificity*. ROC analysis provides a natural means to check and estimate how a statistical measure is able to distinguish positive examples, the *True Set* (TS), and negative examples, the *Control Set* (CS). Given a threshold t , $Se(t)$ is the probability of a candidate pair (v_h, v_t) to belong to *True Set* if

the test is positive, while $Sp(t)$ is the probability of belonging to *Control Set* if the test is negative, i.e.:

$$Se(t) = p((v_h, v_t) \in TS | \mathcal{S}(v_h, v_t) > t)$$

$$Sp(t) = p((v_h, v_t) \in CS | \mathcal{S}(v_h, v_t) < t)$$

The ROC curve ($Se(t)$ vs. $1 - Sp(t)$) naturally follows (see Fig. 1). Better methods will have ROC curves more similar to the step function $f(1 - Sp(t)) = 0$ when $1 - Sp(t) = 0$ and $f(1 - Sp(t)) = 1$ when $0 < 1 - Sp(t) \leq 1$.

The ROC analysis provides another useful evaluation tool: the AROC, i.e. the total area under the ROC curve. Statistically, AROC represents the probability that the method in evaluation will rank a chosen positive example higher than a randomly chosen negative instance. AROC is usually used to better compare two methods that have similar ROC curves. Better methods will have higher AROCs.

As *True Set* (TS) we use the controlled verb entailment pairs *ent* contained in WordNet. As described in Sec. 3, the entailment relation is a semantic relation defined at the synset level, standing in the verb sub-hierarchy. That is, each pair of synsets (S_t, S_h) is an oriented entailment relation between S_t and S_h . WordNet contains 409 entailed synsets. These entailment relations are consequently stated also at the lexical level. The pair (S_t, S_h) naturally implies that v_t entails v_h for each possible $v_t \in S_t$ and $v_h \in S_h$. It is possible to derive from the 409 entailment synset a test set of 2,233 verb pairs. As *Control Set* we use two sets: *random* and $\overline{ent}$. The *random* set

is randomly generated using verb in *ent*, taking care of avoiding to capture pairs in entailment relation. A pair is considered a control pair if it is not in the *True Set* (the intersection between the *True Set* and the *Control Set* is empty). The $\overline{ent}$ is the set of pairs in *ent* with pairs in the reverse order. These two *Control Sets* will give two possible ways of evaluating the methods: a *general* and a more *complex* task.

As a pre-processing step, we have to *clean* the two sets from pairs in which the hypotheses can not be nominalized, as our pattern $\mathcal{P}_{nom}$ is applicable only in these cases. The pre-processing step retains 1,323 entailment verb pairs. For comparative purposes the *random Control Set* is kept with the same cardinality of the *True Set* (in all, 1400 verb pairs).

$\mathcal{S}$ is then evaluated for each pattern over the *True Set* and the *Control Set*, using equation (3) for $\mathcal{P}_{nom}$, and equation (6) for $\mathcal{P}_{pe}$ and $\mathcal{P}_{hb}$. The best pattern or combined method is the one that is able to most neatly split entailment pairs from random pairs. That is, it should in average assign higher $\mathcal{S}$ values to pairs in the *True Set*.

6.2 Results and analysis

In the first experiment we compared the performances of the methods in dividing the *ent* test set and the *random* control set. The compared methods are: (1) the set of patterns taken alone, i.e. *nom*, *hb*, and *pe*; (2) some combined methods, i.e. *nom + pe*, *hb + pe*, and *nom + hb + pe*. Results of this first experiment are reported in Tab. 2 and Fig. 1.(a). As Figure 1.(a) shows, our nominalization pattern $\mathcal{P}_{nom}$ performs better than the others. Only $\mathcal{P}_{hb}$ seems to outperform nominalization in some point of the ROC curve, where $\mathcal{P}_{nom}$ presents a slight concavity, maybe due to a consistent overlap between positive and negative examples at specific values of the $\mathcal{S}$ threshold t . In order to understand which of the two patterns has the best discrimination power a comparison of the AROC values is needed. As Table 2 shows, $\mathcal{P}_{nom}$ has the best AROC value (59.94%) indicating a more interesting behaviour with respect to $\mathcal{P}_{hb}$ and $\mathcal{P}_{pe}$. It is respectively 2 and 3 absolute percent point higher. Moreover, the combinations *nom + hb + pe* and *nom + pe* that includes the $\mathcal{P}_{nom}$ pattern have a very high performance considering the difficulty of the task, i.e. 66% and 64%. If compared with the combina-

	AROC	best accuracy
<i>hb</i>	56.00	57.11
<i>pe</i>	57.00	55.75
<i>nom</i>	59.94	59.86
<i>nom + pe</i>	64.40	61.33
<i>hb + pe</i>	61.44	58.98
<i>hb + nom + pe</i>	66.44	63.09
$\overline{hb}$	61.64	62.73
$\overline{hb} + pe$	69.03	64.71
$\overline{hb} + nom + pe$	70.82	66.07

Table 2: Performances in the general case: *ent* vs. *random*

	AROC	best accuracy
<i>hb</i>	43.82	50.11
<i>nom</i>	54.91	54.94
$\overline{hb}$	56.18	57.16
<i>hb + nom</i>	49.35	51.73
$\overline{hb} + nom$	57.67	57.22

Table 3: Performances in the *complex* case: *ent* vs. $\overline{ent}$

tion *hb + pe* that excludes the $\mathcal{P}_{nom}$ pattern (61%), the improvement in the AROC is of 5% and 3%. Moreover, the shape of the *nom + hb + pe* ROC curve in Fig. 1.(a) is above all the other in all the points.

In the second experiment we compared methods in the more *complex* task of dividing the *ent* set from the $\overline{ent}$ set. In this case methods are asked to determine if *win* $\rightarrow$ *play* is a correct entailment and *play* $\rightarrow$ *win* is not. Results of these set of experiments is presented in Tab. 3. The nominalized pattern *nom* preserves its discriminative power. Its AROC is over the chance line even if, as expected, it is worse than the one obtained in the *general* case. Surprisingly, the *happens-before* (*hb*) set of patterns seems to be not correlated the entailment relation. The temporal relation v_h -happens-before- v_t does not seem to be captured by those patterns. But, if this evidence is seen in a positive way, it seems that the patterns are better capturing the entailment when used in the reversed way ($\overline{hb}$). This is confirmed by its AROC value. If we observe for example one of the implications in the *True Set*, *reach* $\rightarrow$ *go* what is happening may become clearer. Sample sentences respectively for the *hb* case and the $\overline{hb}$ case are “*The group therefore elected to go to Tyso and then reach Anskaven*” and “*striving to reach personal goals and then go beyond them*”. It seems that in the second case *then* assumes an *enabling* role more than only a *temporal* role. After this sur-

prising result, as we expected, in this experiment even the combined approach $\overline{hb} + nom$ behaves better than $hb + nom$ and better than $\overline{hb}$, respectively around 8% and 1.5% absolute points higher (see Tab. 3).

The above results imposed the running of a third experiment over the *general* case. We need to compare the entailment indicators derived exploiting the new use of hb , i.e. $\overline{hb}$, with respect to the methods used in the first experiment. Results are reported in Tab. 2 and Fig. 1.(b). As Fig. 1.(b) shows, the $\overline{hb}$ has a very interesting behaviour for small values of $1 - Sp(t)$. In this area it behaves extremely better than the combined method $nom + hb + pe$. This is an advantage and the combined method $nom + \overline{hb} + pe$ exploit it as both the AROC and the shape of the ROC curve demonstrate. Again the method $nom + \overline{hb} + pe$ that includes the P_{nom} pattern has 1,5% absolute points with respect to the combined method $\overline{hb} + pe$ that does not include this information.

7 Conclusions

In this paper we presented a method to discover asymmetric entailment relations between verbs and we empirically demonstrated interesting improvements when used in combination with similar approaches. The method is promising and there is still some space for improvements. As implicitly experimented in (Chklovski and Pantel, 2004), some beneficial effect can be obtained combining these “non-distributional” methods with the methods based on the Distributional Hypothesis.

References

- Timoty Chklovski and Patrick Pantel. 2004. VerbOCEAN: Mining the web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain.
- Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information and lexicography. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (ACL)*, Vancouver, Canada.
- Oren Glickman and Ido Dagan. 2003. Identifying lexical paraphrases from a single corpus: A case study for verbs. In *Proceedings of the International Conference Recent Advances of Natural Language Processing (RANLP-2003)*, Borovets, Bulgaria.
- Oren Glickman, Ido Dagan, and Moshe Koppel. 2005. Web based probabilistic textual entailment. In *Proceedings of the 1st Pascal Challenge Workshop*, Southampton, UK.
- David M. Green and John A. Swets. 1996. *Signal Detection Theory and Psychophysics*. John Wiley and Sons, New York, USA.
- Zellig Harris. 1964. Distributional structure. In Jerrold J. Katz and Jerry A. Fodor, editors, *The Philosophy of Linguistics*, New York. Oxford University Press.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 15th International Conference on Computational Linguistics (CoLing-92)*, Nantes, France.
- Frank Keller and Mirella Lapata. 2003. Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29(3), September.
- Dekan Lin and Patrick Pantel. 2001. DIRT-discovery of inference rules from text. In *Proc. of the ACM Conference on Knowledge Discovery and Data Mining (KDD-01)*, San Francisco, CA.
- George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, November.
- Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of english. *Natural Language Engineering*, 7(3):207–223.
- Emmanuel Morin. 1999. *Extraction de liens sémantiques entre termes à partir de corpus de textes techniques*. Ph.D. thesis, Université de Nantes, Faculté des Sciences et de Techniques.
- Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th ACL Meeting*, Philadelphia, Pennsylvania.
- Philip Resnik and Mona Diab. 2000. Measuring verb similarity. In *Twenty Second Annual Meeting of the Cognitive Science Society (COGSCI2000)*, Philadelphia.
- Philip Resnik. 1993. *Selection and Information: A Class-Based Approach to Lexical Relationships*. Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania.
- Harold R. Robison. 1970. Computer-detectable semantic structures. *Information Storage and Retrieval*, 6(3):273–288.
- Peter D. Turney. 2001. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *Proc. of the 12th European Conference on Machine Learning*, Freiburg, Germany.

Event Extraction in a Plot Advice Agent

Harry Halpin

School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh, EH8 9LW
Scotland, UK
H.Halpin@ed.ac.uk

Johanna D. Moore

School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh, EH8 9LW
Scotland, UK
J.Moore@ed.ac.uk

Abstract

In this paper we present how the automatic extraction of events from text can be used to both classify narrative texts according to plot quality and produce advice in an interactive learning environment intended to help students with story writing. We focus on the story rewriting task, in which an exemplar story is read to the students and the students rewrite the story in their own words. The system automatically extracts events from the raw text, formalized as a sequence of temporally ordered predicate-arguments. These events are given to a machine-learner that produces a coarse-grained rating of the story. The results of the machine-learner and the extracted events are then used to generate fine-grained advice for the students.

1 Introduction

In this paper we investigate how features of a text discovered via automatic event extraction can be used in both natural language understanding and advice generation in the domain of narrative instruction. The background application is a fully automated plot analysis agent to improve the writing of students could be used by current narrative tutoring systems (Robertson and Wiemer-Hastings, 2002). As shown by participatory design studies, teachers are interested in a plot analysis agent that can give online natural language advice and many students enjoy feedback from an automated agent (Robertson and Cross, 2003). We use automatic event extraction to create a story-independent automated agent that can both analyze the plot of a story and generate appropriate advice.

1.1 The Story Rewriting Task

A task used in schools is the *story rewriting task*, where a story, the *exemplar story*, is read to the students, and afterwards the story is rewritten by each student, providing a corpus of *rewritten stories*. This task tests the students ability to both listen and write, while removing from the student the cognitive load needed to generate a new plot. This task is reminiscent of the well-known “War of the Ghosts” experiment used in psychology for studying memory (Bartlett, 1932) and related to work in fields such as summarization (Lemaire et al., 2005) and narration (Halpin et al., 2004).

1.2 Agent Design

The goal of the agent is to classify each of the *rewritten stories* for overall plot quality. This rating can be used to give “coarse-grained” *general advice*. The agent should then provide “fine-grained” *specific advice* to the student on how their plot could be improved. The agent should be able to detect if the story should be re-read or a human teacher summoned to help the student.

To accomplish this task, we extract events that represent the entities and their actions in the plot from both the exemplar and the rewritten stories. A plot comparison algorithm checks for the presence or absence of events from the exemplar story in each rewritten story. The results of this algorithm will be used by a machine-learner to classify each story for overall plot quality and provide general “canned” advice to the student. The features statistically shared by “excellent” stories represent the important events of the exemplar story. The results of a search for these important events in a rewritten story provides the input needed by templates to generate specific advice for a student.

2 Corpus

In order to train our agent, we collected a corpus of 290 stories from primary schools based on two different exemplar stories. The first is an episode of “The Wonderful Adventures of Nils” by Selma Lagerlof (160 stories) and the second a re-telling of “The Treasure Thief” by Herodotus (130 stories). These will be referred to as the “Adventure” and “Thief” corpora.

2.1 Rating

An experienced teacher, Rater *A*, designed a rating scheme equivalent to those used in schools. The scheme rates the stories as follows:

1. *Excellent*: An excellent story shows that the student has “read beyond the lines” and demonstrates a deep understanding of the story, using inference to grasp points that may not have been explicit in the story. The student should be able to retrieve *all* the important links, and not all the details, but the right details.
2. *Good*: A good story shows that the student understood the story and has “read between the lines.” The student recalls the main events and links in the plot. However, the student shows no deep understanding of the plot and does not make use of inference. This can often be detected by the student leaving out an important link or emphasizing the wrong details.
3. *Fair*: A fair story shows that student has listened to the story but not understood the story, and so is only trying to repeat what they have heard. This is shown by the fact that the fair story is missing multiple important links in the story, including a possibly vital part of the story.
4. *Poor*: A poor story shows the student has had trouble listening to the story. The poor story is missing a substantial amount of the plot, with characters left out and events confused. The student has trouble connecting the parts of the story.

To check the reliability of the rating scheme, two other teachers (Rater *B* and Rater *C*) rated subsets (82 and 68 respectively) of each of the corpora. While their absolute agreement with Rater *A*

Class	Adventure	Thief
1 (Excellent)	.231	.146
2 (Good)	.300	.377
3 (Fair)	.156	.292
4 (Poor)	.313	.185

Table 1: **Probability Distribution of Ratings**

makes the task appear subjective (58% for *B* and 53% for *C*), their relative agreement was high, as almost all disagreements were by one level in the rating scheme. Therefore we use Cronbach’s α and τ_b instead of Cohen’s or Fleiss’ κ to take into account the fact that our scale is ordinal. Between Rater *A* and *B* there was a Cronbach’s α statistic of .90 and a Kendall’s τ_b statistic of .74. Between Rater *B* and *C* there was a Cronbach’s α statistic of .87 and Kendall’s τ_b statistic of .67. These statistics show the rating scheme to be reliable and the distribution of plot ratings are given in Table 1.

2.2 Linguistic Issues

One challenge facing this task is the ungrammatical and highly irregular text produced by the students. Many stories consist of one long run-on sentence. This leads a traditional parsing system with a direct mapping from the parse tree to a semantic representation to fail to achieve a parse on 35% percent of the stories, and as such could not be used (Bos et al., 2004). The stories exhibit frequent use of reported speech and the switching from first-person to third-person within a single sentence. Lastly, the use of incorrect spelling e.g., “stalk” for “stork” appearing in multiple stories in the corpus, the consistent usage of homonyms such as “there” for “their,” and the invention of words (“torlix”), all prove to be frequent.

3 Plot Analysis

To automatically rate student writing many tutoring systems use Latent Semantic Analysis, a variation on the “bag-of-words” technique that uses dimensionality reduction (Graesser et al., 2000). We hypothesize that better results can be achieved using a “representational” account that explicitly represents each event in the plot. These semantic relationships are important in stories, e.g., “The thief jumped on the donkey” being distinctly different from “The donkey jumped on the thief.” What characters participate in an action matter, since “The king stole the treasure” reveals a major

misunderstanding while “The thief stole the treasure” shows a correct interpretation by the student.

3.1 Stories as Events

We represent a story as a sequence of *events*, $p_1 \dots p_h$, represented as a list of predicate-arguments, similar to the event calculus (Mueller, 2003). Our predicate-argument structure is a minimal subset of first-order logic (no quantifiers), and so is compatible with case-frame and dependency representations. Every *event* has a *predicate* (function) p that has one or more *arguments*, $n_1 \dots n_a$. In the tradition of Discourse Representation Theory (Kamp and Reyle, 1993), our current predicate argument structure could be converted automatically to first order logic by using a default existential quantification over the predicates and joining them conjunctively. Predicate names are often verbs, while their arguments are usually, although not exclusively, nouns or adjectives. When describing a set of events in the story, a superscript is used to keep the arguments in an event distinct, as n_5^2 is argument 2 in event 5. The same argument name may appear in multiple events. The plot of any given story is formalized as an *event structure* composed of h events in a partial order, with the partial order denoting their temporal order:

$$p_1(n_1^1, n_1^2, \dots, n_1^a), \dots, p_h(n_h^2, n_h^4 \dots n_h^c)$$

An example from the “Thief” exemplar story is “The Queen nagged the king to build a treasure chamber. The king decided to have a treasure chamber.” This can be represented by an event structure as:

nag(king, queen)
build(chamber)
decide(king)
have(chamber)

Note due the ungrammatical corpus we cannot at this time extract neo-Davidsonian events. A sentence maps onto one, multiple, or no events. A unique name and closed-world assumption is enforced, although for purposes of comparing event we compare membership of argument and predicate names in WordNet synsets in addition to exact name matches (Fellbaum, 1998).

4 Extracting Events

Paralleling work in summarization, it is hypothesized that the quality of a rewritten story can be

defined by the presence or absence of “semantic content units” that are crucial details of the text that may have a variety of syntactic forms (Nenkova and Passonneau, 2004). We further hypothesize these can be found in chunks of the text automatically identified by a chunker, and we can represent these units as predicate-arguments in our event structure. The event structure of each story is automatically extracted using an XML-based pipeline composed of NLP processing modules, and unlike other story systems, extract full events instead of filling in a frame of a story script (Riloff, 1999). Using the latest version of the Language Technology Text Tokenization Toolkit (Grover et al., 2000), words are tokenized and sentence boundaries detected. Words are given part-of-speech tags by a maximum entropy tagger from the toolkit. We do not attempt to obtain a full parse of the sentence due to the highly irregular nature of the sentences. Pronouns are resolved using a rule-based reimplement of the CogNIAC algorithm (Baldwin, 1997) and sentences are lemmatized and chunked using the Cass Chunker (Abney, 1995). It was felt the chunking method would be the only feasible way to retrieve portions of the sentences that may contain complete “semantic content units” from the ungrammatical and irregular text. The application of a series of rules, mainly mapping verbs to predicate names and nouns to arguments, to the results of the chunker produces events from chunks as described in our previous work (McNeill et al., 2006). The accuracy of our rule-set was developed by using the grammatical exemplar stories as a testbed, and a blind judge found they produced 68% interpretable or “sensible” events given the ungrammatical text. Students usually use the present or past tense exclusively throughout the story and events are usually presented in order of occurrence. An inspection of our corpus showed 3% of stories in our corpus seemed to get the order of events wrong (Hickmann, 2003).

4.1 Comparing Stories

Since the student is rewriting the story using their own words, a certain variance from the plot of the exemplar story should be expected and even rewarded. Extra statements that may be true, but are not explicitly stated in the story, can be inferred by the students. Statements that are true but are not highly relevant to the course of the

plot can likewise be left out. Word similarity must be taken into account, so that “The king is protecting his gold” can be recognized as “The pharaoh guarded the treasure.” Characters change in context, as one character that is described as the “younger brother” is from the viewpoint of his mother “the younger son.” So, building a model from the events of two stories and simply checking equivalence can not be used for comparison, since a wide variety of partial equivalence must be taken into account.

Instead of using absolute measures of equivalence based on model checking or measures based on word distribution, we compare each story on the basis of the presence or absence of events. This approach takes advantage of WordNet to define synonym matching and uses the relational structure of the events to allow partial matching of predicate functions and arguments. The events of the exemplar story are assumed to be correct, and they are searched for in the rewritten story in the order in which they occur in the exemplar. If an event is matched (including using WordNet), then in turn each of the arguments attempts to be matched.

This algorithm is given more formally in Figure 1. The complete event structure from the exemplar story, E , and the complete event structure from the rewritten story R , with each individual event predicate name labelled as e and r respectively, and their arguments labelled as n in either N_e and N_r . $\text{SYN}(x)$ is the synset of the term x , including hypernyms and hyponyms except upper ontology ones. The results of the algorithm are stored in binary vector F with index i . 1 denotes an exact match or WordNet synset match, and 0 a failure to find any match.

4.2 Results

As a baseline system LSA produces a similarity score for each rewritten story by comparing it to the exemplar, this score is used as a distance metric for a k -Nearest Neighbor classifier (Deerwester et al., 1990). The parameters for LSA were empirically determined to be a dimensionality of 200 over the semantic space given by the recommended reading list for American 6th graders (Landauer and Dumais, 1997). These parameters resulted in the LSA similarity score having a Pearson’s correlation of -.520 with Rater A . k was found to be optimal at 9.

Algorithm 4.1: PLOTCOMPARE(E, R)

```

 $i \leftarrow 0$ 
 $f \leftarrow \emptyset$ 
for  $e \in E$ 
  do for  $r \in R$ 
    if  $e = \text{SYN}(r)$ 
      then  $f_i \leftarrow 1$ 
      else  $f_i \leftarrow 0$ 
      for  $n_e \in N_e$ 
        do for  $n_r \in N_r$ 
          if  $n_e = \text{SYN}(n_r)$ 
            then  $f_i \leftarrow 1$ 
            else  $f_i \leftarrow 0$ 
           $i = i + 1$ 

```

Figure 1: **Plot Comparison Algorithm**

Classifier	Corpus	Features	% Correct
k-NN	Adventure	LSA	47.5
Naive Bayes	Adventure	PLOT	55.6
k-NN	Thief	LSA	41.2
Naive Bayes	Thief	PLOT	45.4

Table 2: **Machine-Learning Results**

The results of the plot comparison algorithm were given as features to machine-learners, with results produced using ten-fold cross-validation. A Naive Bayes learner discovers the different statistical distributions of events for each rating. The results for both the “Adventure” and “Thief” stories are displayed in Table 2. “PLOT” means the results of the Plot Comparison Algorithm were used as features for the machine-learner while “LSA” means the similarity scores for Latent Semantic Analysis were used instead. Note that the same machine-learner could not be used to judge the effect of LSA and PLOT since LSA scores are real numbers and PLOT a set of features encoded as binary vectors.

The results do not seem remarkable at first glance. However, recall that the human raters had an average of 56% agreement on story ratings, and in that light the Naive Bayes learner approaches the performance of human raters. Surprisingly, when the LSA score is used as a feature in addition to the results of the plot comparison algorithm for the Naive Bayes learners, there is no further improvement. This shows features given by the event

Class	1	2	3	4
1 (Excellent)	14	22	0	1
2 (Good)	5	36	0	7
3 (Fair)	3	20	0	2
4 (Poor)	0	11	0	39

Table 3: **Naive Bayes Confusion Matrix: “Adventure”**

Class	Precision	Recall
Excellent	.64	.38
Good	.40	.75
Fair	.00	.00
Poor	.80	.78

Table 4: **Naive Bayes Results: “Adventure”**

structure better characterize plot structure than the word distribution. Unlike previous work, the use of both the plot comparison results and LSA did not improve performance for Naive Bayes, so the results of using Naive Bayes with both are not reported (Halpin et al., 2004).

The results for the “Adventure” corpus are in general better than the results for the “Thief” corpus. However, this is due to the “Thief” corpus being smaller and having an infrequent number of “Excellent” and “Poor” stories, as shown in Table 1. In the “Thief” corpus the learner simply collapses most stories into “Good,” resulting in very poor performance. Another factor may be that the “Thief” story was more complex than the “Adventure” story, featuring 9 characters over 5 scenes, as opposed to the “Adventure” corpus that featured 4 characters over 2 scenes.

For the “Adventure” corpus, the Naive Bayes classifier produces the best results, as detailed in Table 4 and the confusion matrix in Figure 3. A close inspection of the results shows that in the “Adventure Corpus” the “Poor” and “Good” stories are classified in general fairly well by the Naive Bayes learner, while some of the “Excellent” stories are classified as correctly. A significant number of both “Excellent” and most “Fair” stories are classified as “Good.” The “Fair” category, due to its small size in the training corpus, has disappeared. No “Poor” stories are classified as “Excellent,” and no “Excellent” stories are classified as “Poor.” The increased difficulty in distinguishing “Excellent” stories from “Good” stories is likely due to the use of inference by “Excellent”

stories, which our system does not use. An inspection of the rating scale’s wording reveals the similarity in wording between the “Fair” and “Good” ratings. This may explain the lack of “Fair” stories in the corpus and therefore the inability of machine-learners to recognize them. As given by a survey of five teachers experienced in using the story rewriting task in schools, this level of performance is not ideal but acceptable to teachers.

Our technique is also shown to be easily portable over different domains where a teacher can annotate around one hundred sample stories using our scale, although performance seems to suffer the more complex a story is. Since the Naive Bayes classifier is fast (able to classify stories in only a few seconds) and the entire algorithm from training to advice generation (as detailed below) is fully automatic once a small training corpus has been produced, this technique can be used in real-life tutoring systems and easily ported to other stories.

5 Automated Advice

The plot analysis agent is not meant to give the students grades for their stories, but instead use the automatic ratings as an intermediate step to produce advice, like other hybrid tutoring systems (Rose et al., 2002). The advice that the agent can generate from the automatic rating classification is limited to coarse-grained *general advice*. However, by inspecting the results of the plot comparison algorithm, our agent is capable of giving detailed fine-grained *specific advice* from the relationships of the events in the story. One tutoring system resembling ours is the WRITE system, but we differ from it by using event structure to represent the information in the system, instead of using rhetorical features (Burstein et al., 2003). In this regards it more closely resembles the physics tutoring system WHY-ATLAS, although we deal with narrative stories of a longer length than physics essays. The WHY-ATLAS physics tutor identifies missing information in the explanations of students using theorem-proving (Rose et al., 2002).

5.1 Advice Generation Algorithm

Different types of stories need different amounts of advice. An “Excellent” story needs less advice than a “Good” story. One advice statement is “general,” while the rest are specific. The system

produces a total of seven advice statements for a “Poor” story, and two less statements for each rating level above “Poor.”

With the aid of a teacher, a number of “canned” text statements offering general advice were created for each rating class. These include statements such as “It’s very good! I only have a few pointers” for a “Good” story and “Let’s get help from the teacher” for “Poor” story. The advice generation begins by randomly selecting a statement suitable for the rating of the story. Those students whose stories are rated “Poor” are asked if they would like to re-read the story and ask a teacher for help.

The generation of specific advice uses the results of the plot-comparison algorithm to produce specific advice. A number of advice templates were produced, and the results of the *Advice Generation Algorithm* fill in the needed values of the template. The ϕ most frequent events in “Excellent” stories are called the *Important Event Structure*, which represents the “important” events in the story in temporal order. Empirical experiments led us $\phi = 10$ for the “Adventure” story, but for longer stories like the “Thief” story a larger ϕ would be appropriate. These events correspond to the ones given the highest weights by the Naive Bayes algorithm. For each event in the event structure of a rewritten story, a search for a match in the important event structure is taken. If a predicate name match is found in the important event structure, the search continues to attempt to match the arguments. If the event and the arguments do not match, advice is generated using the structure of the “important” event that it cannot find in the rewritten story.

This advice may use both the predicate name and its arguments, such as “Did the stork fly?” from *fly(stork)*. If an argument is missing, the advice may be about only the argument(s), like “Can you tell me more about the stork?” If the event is out of order, advice is given to the student to correct the order, as in “I think something with the stork happened earlier in the story.”

This algorithm is formalized in Figure 2, with all variables being the same as in the Plot Analysis Algorithm, except that W is the Important Event Structure composed of events w with the set of arguments N_w . M is a binary vector used to store the success of a match with index i . The ADV function, given an event, generates one ad-

Algorithm

5.1: ADVISEGENERATE(W, R)

```

for  $w \in W$ 
  {
     $M = \emptyset$ 
     $i = 0$ 
    for  $r \in R$ 
      {
        if  $w = r$  or  $SYN(r)$ 
          then  $m_i = 1$ 
          else  $m_i = 0$ 
         $i = i + 1$ 
        do for  $n_w \in N_w$ 
          {
            for  $n_r \in N_r$ 
              {
                if  $n_w = SYN(n_r)$  or  $n_r$ 
                  then  $m_i \leftarrow 1$ 
                  else  $m_i \leftarrow 0$ 
                 $i = i + 1$ 
              }
            do
          }
         $ADV(w, M)$ 
      }
  }

```

Figure 2: Advice Generation Algorithm

vice statement to be given to the student.

An element of randomization was used to generate a diversity of types of answers. An advice generation function (ADV) takes an important event (w) and its binary matching vector (M) and generates an advice statement for w . Per important event this advice generation function is parameterized so that it has a 10% chance of delivering advice based on the entire event, 20% chance of producing advice that dealt with temporal order (these being parameters being found ideal after testing the algorithm), and otherwise produces advice based on the arguments.

5.2 Advice Evaluation

The plot advice algorithm is run using a randomly selected corpus of 20 stories, 5 from each plot rating level using the “Adventure Corpus.” This produced matching advice for each story, for a total of 80 advice statements.

5.3 Advice Rating

An advice rating scheme was developed to rate the advice produced in consultation with a teacher.

1. *Excellent*: The advice was suitable for the story, and helped the student gain insight into the story.
2. *Good*: The advice was suitable for the story,

Rating	% Given
Excellent	0
Good	35
Fair	60
Poor	5

Table 5: **Advice Rating Results**

and would help the student.

3. *Fair*: The advice was suitable, but should have been phrased differently.
4. *Poor*: The advice really didn't make sense and would only confuse the student further.

Before testing the system on students, it was decided to have teachers evaluate how well the advice given by the system corresponded to the advice they would give in response to a story. A teacher read each story and the advice. They then rated the advice using the advice rating scheme. Each story was rated for its overall advice quality, and then each advice statement was given comments by the teacher, such that we could derive how each individual piece of advice contributed to the global rating. Some of the general "coarse-grained" advice was "Good! You got all the main parts of the story" for an "Excellent" story, "Let's make it even better!" for a "Good" story, and "Reading the story again with a teacher would be help!" for a "Poor" story. Sometimes the advice generation algorithm was remarkably accurate. In one story the connection between a curse being lifted by the possession of a coin by the character Nils was left out by a student. The advice generation algorithm produced the following useful advice statement: "Tell me more about the curse and Nils." Occasionally an automatically extracted event that is difficult to interpret by a human or simply incorrectly is extracted. This in turn can cause advice that does not make any sense can be produced, such as "Tell me more about a spot?". Qualitative analysis showed that "missing important advice" to be the most significant problem, followed by "nonsensical advice."

5.4 Results

The results are given in Table 5. The majority of the advice was rated overall as "fair." Only one story was given "poor" advice, and a few were given "good" advice. However, most advice rated

as "good" was the advice generated by "excellent" stories, which generate less advice than other types of stories. "Poor" stories were given almost entirely "fair" advice, although once "poor" advice was generated. In general, the teacher found "coarse-grained" advice to be very useful, and was very pleased that the agent could detect when the student needed to re-read the story and when a student did not need to write any more. In some cases the specific advice was shown to help provide a "crucial detail" and help "elicit a fact." The advice was often "repetitive" and "badly phrased." The specific advice came under criticism for often not "being directed enough" and for being "too literal" and not "inferential enough." The rater noticed that "The program can not differentiate between an unfinished story...and one that is confused." and that "Some why, where and how questions could be used" in the advice.

6 Conclusion and Future Work

Since the task involved a fine-grained analysis of the rewritten story, the use of events that take plot structure into account made sense regardless of its performance. The use of events as structured features in a machine-learning classifier outperformed a classifier that relied on a unstructured "bag-of-words" as features. The system achieved close to human performance on rating the stories. Since each of the events used as a feature in the machine-learner corresponds to a particular event in the story, the features are easily interpretable by other components in the system and interpretable by humans. This allows these events to be used in a template-driven system to generate advice for students based on the structure of their plot.

Extracting events from text is fraught with error, particularly in the ungrammatical and informal domain used in this experiment. This is often a failure of our system to detect semantic content units through either not including them in chunks or only partially including a single unit in a chunk. Chunking also has difficulty dealing with prepositions, embedded speech, semantic role labels, and complex sentences correctly. Improvement in our ability to retrieve semantics would help both story classification and advice generation.

Advice generation was impaired by the ability to produce directed questions from the events using templates. This is because while our system could detect important events and their or-

der, it could not make explicit their connection through inference. Given the lack of a large-scale open-source accessible “common-sense” knowledge base and the difficulty in extracting inferential statements from raw text, further progress using inference will be difficult. Progress in either making it easier for a teacher to make explicit the important inferences in the text or improved methodology to learn inferential knowledge from the text would allow further progress. Tantalizingly, this ability for a reader to use “inference to grasp points that may not have been explicit in the story” is given as the hallmark of truly understanding a story by teachers.

References

- Steven Abney. 1995. Chunks and dependencies: Bringing processing evidence to bear on syntax. In Jennifer Cole, Georgia Green, and Jerry Morgan, editors, *Computational Linguistics and the Foundations of Linguistic Theory*, pages 145–164.
- Breck Baldwin. 1997. CogNIAC : A High Precision Pronoun Resolution Engine.
- F.C. Bartlett. 1932. *Remembering*. Cambridge University Press, Cambridge.
- Johan Bos, Stephen Clark, Mark Steedman, James Curran, and Julia Hockenmaier. 2004. Wide-coverage semantic representations from a CCG parser. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*. Geneva, Switzerland.
- Jill Burstein, Daniel Marcu, and Kevin Knight. 2003. Finding the WRITE Stuff: Automatic Identification of Discourse Structure in Student Essays. *IEEE Intelligent Systems*, pages 32–39.
- S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society For Information Science*, (41):391–407.
- Christine Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.
- A. Graesser, P. Wiemer-Hastings, K. Wiemer-Hastings, D. Harter, and N. Person. 2000. Using latent semantic analysis to evaluate the contributions of students in autotutor. *Interactive Learning Environments*, 8:149–169.
- Claire Grover, Colin Matheson, Andrei Mikheev, and Marc Moens. 2000. LT TTT - A Flexible Tokenisation Tool. In *Proceedings of the Second Language Resources and Evaluation Conference*.
- Harry Halpin, Johanna Moore, and Judy Robertson. 2004. Automatic analysis of plot for story rewriting. In *Proceedings of Empirical Methods in Natural Language Processing*, Barcelona, Spain.
- Maya Hickmann. 2003. *Children’s Discourse: person, space and time across language*. Cambridge University Press, Cambridge, UK.
- Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic*. Kluwer Academic.
- Thomas Landauer and Susan Dumais. 1997. A solution to Plato’s problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*.
- B. Lemaire, S. Mandin, P. Dessus, and G. Denhire. 2005. Computational cognitive models of summarization assessment skills. In *Proceedings of the 27th Annual Meeting of the Cognitive Science Society*, Stresa, Italy.
- Fiona McNeill, Harry Halpin, Ewan Klein, and Alan Bundy. 2006. Merging stories with shallow semantics. In *Proceedings of the Knowledge Representation and Reasoning for Language Processing Workshop at the European Association for Computational Linguistics*, Genoa, Italy.
- Erik T. Mueller. 2003. Story understanding through multi-representation model construction. In Graeme Hirst and Sergei Nirenburg, editors, *Text Meaning: Proceedings of the HLT-NAACL 2003 Workshop*, pages 46–53, East Stroudsburg, PA. Association for Computational Linguistics.
- Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Joint Conference of the North American Association for Computational Linguistics and Human Language Technologies*. Boston, USA.
- E. Riloff. 1999. Information extraction as a stepping stone toward story understanding. In Ashwin Ram and Kenneth Moorman, editors, *Computational Models of Reading and Understanding*. MIT Press.
- Judy Robertson and Beth Cross. 2003. Children’s perceptions about writing with their teacher and the StoryStation learning environment. *Narrative and Interactive Learning Environments: Special Issue of International Journal of Continuing Engineering Education and Life-long Learning*.
- Judy Robertson and Peter Wiemer-Hastings. 2002. Feedback on children’s stories via multiple interface agents. In *International Conference on Intelligent Tutoring Systems*, Biarritz, France.
- C. Rose, D. Bhembé, A. Roque, S. Siler, R. Srivastava, and K. VanLehn. 2002. A hybrid language understanding approach for robust selection of tutoring goals. In *International Conference on Intelligent Tutoring Systems*, Biarritz, France.

An All-Subtrees Approach to Unsupervised Parsing

Rens Bod

School of Computer Science
University of St Andrews
North Haugh, St Andrews
KY16 9SX Scotland, UK
rb@dc.s-st-and.ac.uk

Abstract

We investigate generalizations of the all-subtrees "DOP" approach to unsupervised parsing. Unsupervised DOP models assign all possible binary trees to a set of sentences and next use (a large random subset of) all subtrees from these binary trees to compute the most probable parse trees. We will test both a relative frequency estimator for unsupervised DOP and a maximum likelihood estimator which is known to be statistically consistent. We report state-of-the-art results on English (WSJ), German (NEGRA) and Chinese (CTB) data. To the best of our knowledge this is the first paper which tests a maximum likelihood estimator for DOP on the Wall Street Journal, leading to the surprising result that *an unsupervised parsing model beats a widely used supervised model (a treebank PCFG)*.

1 Introduction

The problem of bootstrapping syntactic structure from unlabeled data has regained considerable interest. While supervised parsers suffer from shortage of hand-annotated data, unsupervised parsers operate with unlabeled raw data of which unlimited quantities are available. During the last few years there has been steady progress in the field. Where van Zaanen (2000) achieved 39.2% unlabeled f-score on ATIS word strings, Clark (2001) reports 42.0% on the same data, and Klein and Manning (2002) obtain 51.2% f-score on ATIS part-of-speech strings using a constituent-context

model called CCM. On Penn Wall Street Journal p-o-s-strings ≤ 10 (WSJ10), Klein and Manning (2002) report 71.1% unlabeled f-score with CCM. And the hybrid approach of Klein and Manning (2004), which combines constituency and dependency models, yields 77.6% f-score.

Bod (2006) shows that a further improvement on the WSJ10 can be achieved by an unsupervised generalization of the all-subtrees approach known as Data-Oriented Parsing (DOP). This unsupervised DOP model, coined U-DOP, first assigns all possible unlabeled binary trees to a set of sentences and next uses all subtrees from (a large subset of) these trees to compute the most probable parse trees. Bod (2006) reports that U-DOP not only outperforms previous unsupervised parsers but that its performance is as good as a binarized *supervised* parser (i.e. a treebank PCFG) on the WSJ.

A possible drawback of U-DOP, however, is the statistical inconsistency of its estimator (Johnson 2002) which is inherited from the DOP1 model (Bod 1998). That is, even with unlimited training data, U-DOP's estimator is not guaranteed to converge to the correct weight distribution. Johnson (2002: 76) argues in favor of a maximum likelihood estimator for DOP which is statistically consistent. As it happens, in Bod (2000) we already developed such a DOP model, termed ML-DOP, which reestimates the subtree probabilities by a maximum likelihood procedure based on Expectation-Maximization. Although cross-validation is needed to avoid overlearning, ML-DOP outperforms DOP1 on the OVIS corpus (Bod 2000).

This raises the question whether we can create an *unsupervised* DOP model which is also

statistically consistent. In this paper we will show that an unsupervised version of ML-DOP can be constructed along the lines of U-DOP. We will start out by summarizing DOP, U-DOP and ML-DOP, and next create a new unsupervised model called UML-DOP. We report that UML-DOP not only obtains higher parse accuracy than U-DOP on three different domains, but that it also achieves this with *fewer* subtrees than U-DOP. To the best of our knowledge, this paper presents the first *unsupervised* parser that outperforms a widely used *supervised* parser on the WSJ, i.e. a treebank PCFG. We will raise the question whether the end of supervised parsing is in sight.

2 DOP

The key idea of DOP is this: given an annotated corpus, use all subtrees, regardless of size, to parse new sentences. The DOP1 model in Bod (1998) computes the probabilities of parse trees and sentences from the relative frequencies of the subtrees. Although it is now known that DOP1's relative frequency estimator is statistically inconsistent (Johnson 2002), the model yields excellent empirical results and has been used in state-of-the-art systems. Let's illustrate DOP1 with a simple example. Assume a corpus consisting of only two trees, as given in figure 1.

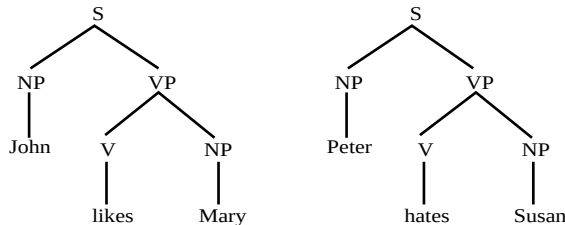


Figure 1. A corpus of two trees

New sentences may be derived by combining fragments, i.e. subtrees, from this corpus, by means of a node-substitution operation indicated as $\circ$. Node-substitution identifies the leftmost nonterminal frontier node of one subtree with the root node of a second subtree (i.e., the second subtree is *substituted* on the leftmost nonterminal frontier node of the first subtree). Thus a new sentence such as *Mary likes Susan* can be derived by

combining subtrees from this corpus, shown in figure 2.

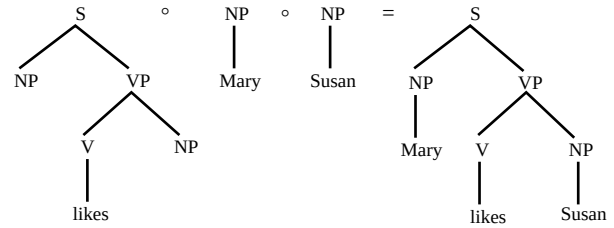


Figure 2. A derivation for *Mary likes Susan*

Other derivations may yield the same tree, e.g.:

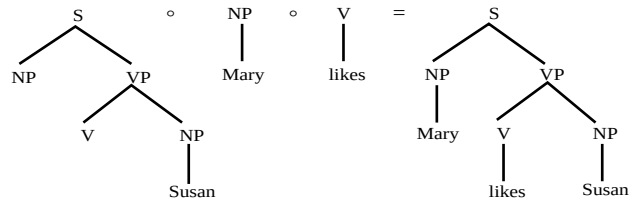


Figure 3. Another derivation yielding same tree

DOP1 computes the probability of a subtree t as the probability of selecting t among all corpus subtrees that can be substituted on the same node as t . This probability is computed as the number of occurrences of t in the corpus, $|t|$, divided by the total number of occurrences of all subtrees t' with the same root label as t .¹ Let $r(t)$ return the root label of t . Then we may write:

$$P(t) = \frac{|t|}{\sum_{t': r(t')=r(t)} |t'|}$$

The probability of a derivation $t_1 \circ \dots \circ t_n$ is computed by the product of the probabilities of its subtrees t_i :

$$P(t_1 \circ \dots \circ t_n) = \prod_i P(t_i)$$

As we have seen, there may be several distinct derivations that generate the same parse tree. The probability of a parse tree T is the sum of the

¹ This subtree probability is redressed by a simple correction factor discussed in Goodman (2003: 136) and Bod (2003).

probabilities of its distinct derivations. Let t_{id} be the i -th subtree in the derivation d that produces tree T , then the probability of T is given by

$$P(T) = \sum_d \prod_i P(t_{id})$$

Thus DOP1 considers counts of subtrees of a wide range of sizes: everything from counts of single-level rules to entire trees is taken into account to compute the most probable parse tree of a sentence. A disadvantage of the approach may be that an extremely large number of subtrees (and derivations) must be considered. Fortunately there exists a compact isomorphic PCFG-reduction of DOP1 whose size is linear rather than exponential in the size of the training set (Goodman 2003). Moreover, Collins and Duffy (2002) show how a tree kernel can be applied to DOP1's all-subtrees representation. The currently most successful version of DOP1 uses a PCFG-reduction of the model with an n -best parsing algorithm (Bod 2003).

3 U-DOP

U-DOP extends DOP1 to unsupervised parsing (Bod 2006). Its key idea is to assign all unlabeled binary trees to a set of sentences and to next use (in principle) all subtrees from these binary trees to parse new sentences. U-DOP thus proposes one of the richest possible models in bootstrapping trees. Previous models like Klein and Manning's (2002, 2005) CCM model limit the dependencies to "contiguous subsequences of a sentence". This means that CCM neglects dependencies that are *non*-contiguous such as between *more* and *than* in "*BA carried more people than cargo*". Instead, U-DOP's all-subtrees approach captures both contiguous and non-contiguous lexical dependencies.

As with most other unsupervised parsing models, U-DOP induces trees for p-o-s strings rather than for word strings. The extension to word strings is straightforward as there exist highly accurate unsupervised part-of-speech taggers (e.g. Schütze 1995) which can be directly combined with unsupervised parsers.

To give an illustration of U-DOP, consider the WSJ p-o-s string NNS VBD JJ NNS which may correspond for instance to the sentence *Investors suffered heavy losses*. U-DOP starts by

assigning all possible binary trees to this string, where each root node is labeled S and each internal node is labeled X. Thus NNS VBD JJ NNS has a total of five binary trees shown in figure 4 -- where for readability we add words as well.

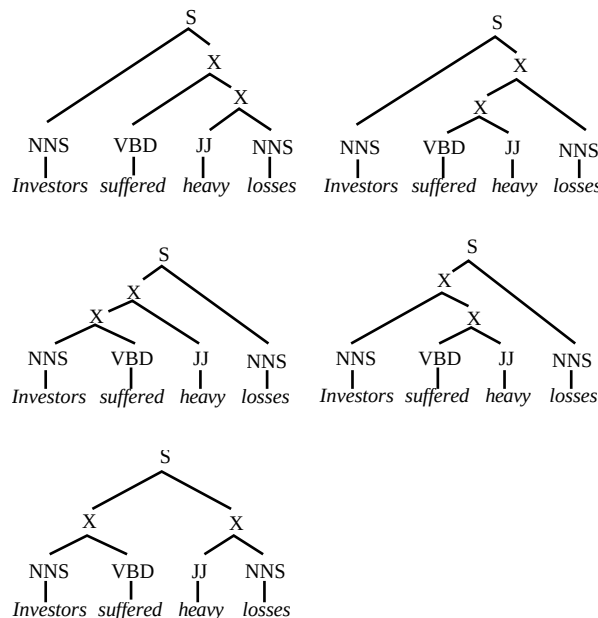


Figure 4. All binary trees for NNS VBD JJ NNS
(*Investors suffered heavy losses*)

While we can efficiently represent the set of all binary trees of a string by means of a chart, we need to unpack the chart if we want to extract subtrees from this set of binary trees. And since the total number of binary trees for the small WSJ10 is almost 12 million, it is doubtful whether we can apply the unrestricted U-DOP model to such a corpus. U-DOP therefore randomly samples a large subset from the total number of parse trees from the chart (see Bod 2006) and next converts the subtrees from these parse trees into a PCFG-reduction (Goodman 2003). Since the computation of the most probable parse tree is NP-complete (Sima'an 1996), U-DOP estimates the most probable tree from the 100 most probable derivations using Viterbi n -best parsing. We could also have used the more efficient k -best hypergraph parsing technique by Huang and Chiang (2005), but we have not yet incorporated this into our implementation.

To give an example of the dependencies that U-DOP can take into account, consider the following subtrees in figure 5 from the trees in

figure 4 (where we again add words for readability). These subtrees show that U-DOP takes into account both contiguous and non-contiguous substrings.

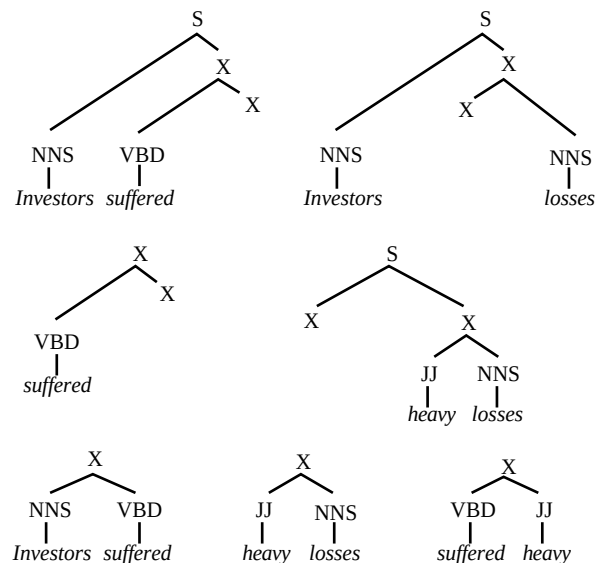


Figure 5. Some subtrees from trees in figure 4

Of course, if we only had the sentence *Investors suffered heavy losses* in our corpus, there would be no difference in probability between the five parse trees in figure 4. However, if we also have a different sentence where JJ NNS (*heavy losses*) appears in a different context, e.g. in *Heavy losses were reported*, its covering subtree gets a relatively higher frequency and the parse tree where *heavy losses* occurs as a constituent gets a higher total probability.

4 ML-DOP

ML-DOP (Bod 2000) extends DOP with a maximum likelihood reestimation technique based on the expectation-maximization (EM) algorithm (Dempster et al. 1977) which is known to be statistically consistent (Shao 1999). ML-DOP reestimates DOP's subtree probabilities in an iterative way until the changes become negligible. The following exposition of ML-DOP is heavily based on previous work by Bod (2000) and Magerman (1993).

It is important to realize that there is an implicit assumption in DOP that all possible derivations of a parse tree contribute equally to the

total probability of the parse tree. This is equivalent to saying that there is a hidden component to the model, and that DOP can be trained using an EM algorithm to determine the maximum likelihood estimate for the training data. The EM algorithm for this ML-DOP model is related to the Inside-Outside algorithm for context-free grammars, but the reestimation formula is complicated by the presence of subtrees of depth greater than 1. To derive the reestimation formula, it is useful to consider the state space of all possible derivations of a tree.

The derivations of a parse tree T can be viewed as a state trellis, where each state contains a partially constructed tree in the course of a leftmost derivation of T . s_t denotes a state containing the tree t which is a subtree of T . The state trellis is defined as follows.

The initial state, s_0 , is a tree with depth zero, consisting of simply a root node labeled with S . The final state, s_T , is the given parse tree T .

A state s_t is connected forward to all states s_{t_f} such that $t_f = t \circ t'$, for some t' . Here the appropriate t' is defined to be $t_f - t$.

A state s_t is connected backward to all states s_{t_b} such that $t = t_b \circ t'$, for some t' . Again, t' is defined to be $t - t_b$.

The construction of the state lattice and assignment of transition probabilities according to the ML-DOP model is called the forward pass. The probability of a given state, $P(s)$, is referred to as $\alpha(s)$. The forward probability of a state s_t is computed recursively

$$\alpha(s_t) = \sum_{s_{t_b}} \alpha(s_{t_b}) P(t - t_b).$$

The backward probability of a state, referred to as $\beta(s)$, is calculated according to the following recursive formula:

$$\beta(s_t) = \sum_{s_{t_f}} \beta(s_{t_f}) P(t_f - t)$$

where the backward probability of the goal state is set equal to the forward probability of the goal state, $\beta(s_T) = \alpha(s_T)$.

The update formula for the count of a subtree t is (where $r(t)$ is the root label of t):

$$ct(t) = \sum_{s_{t_b} : \exists s_{t_f}, t_b \circ t = t_f} \frac{\beta(s_{t_f})\alpha(s_{t_b})P(t | r(t))}{\alpha(s_{goal})}$$

The updated probability distribution, $P'(t | r(t))$, is defined to be

$$P'(t | r(t)) = \frac{ct(t)}{ct(r(t))}$$

where $ct(r(t))$ is defined as

$$ct(r(t)) = \sum_{t' : r(t')=r(t)} ct(t')$$

In practice, ML-DOP starts out by assigning the same relative frequencies to the subtrees as DOP1, which are next reestimated by the formulas above. We may in principle start out with any initial parameters, including random initializations, but since ML estimation is known to be very sensitive to the initialization of the parameters, it is convenient to start with parameters that are known to perform well.

To avoid overtraining, ML-DOP uses the subtrees from one half of the training set to be trained on the other half, and vice versa. This cross-training is important since otherwise UML-DOP would assign the training set trees their empirical frequencies and assign zero weight to all other subtrees (cf. Prescher et al. 2004). The updated probabilities are iteratively reestimated until the decrease in cross-entropy becomes negligible. Unfortunately, no compact PCFG-reduction of ML-DOP is known. As a consequence, parsing with ML-DOP is very costly and the model has hitherto never been tested on corpora larger than OVIS (Bonnema et al. 1997). Yet, we will show that by clever pruning we can extend our experiments not only to the WSJ, but also to the German NEGRA and the Chinese CTB. (Zollmann and Sima'an 2005 propose a different consistent estimator for DOP, which we cannot go into here).

5 UML-DOP

Analogous to U-DOP, UML-DOP is an unsupervised generalization of ML-DOP: it first assigns all unlabeled binary trees to a set of

sentences and next extracts a large (random) set of subtrees from this tree set. It then reestimates the initial probabilities of these subtrees by ML-DOP on the sentences from a held-out part of the tree set. The training is carried out by dividing the tree set into two equal parts, and reestimating the subtrees from one part on the other. As initial probabilities we use the subtrees' relative frequencies as described in section 2 (smoothed by Good-Turing -- see Bod 1998), though it would also be interesting to see how the model works with other initial parameters, in particular with the usage frequencies proposed by Zuidema (2006).

As with U-DOP, the total number of subtrees that can be extracted from the binary tree set is too large to be fully taken into account. Together with the high computational cost of reestimation we propose even more drastic pruning than we did in Bod (2006) for U-DOP. That is, while for sentences ≤ 7 words we use all binary trees, for each sentence ≥ 8 words we randomly sample a fixed number of 128 trees (which effectively favors more frequent trees). The resulting set of trees is referred to as the binary tree set. Next, we randomly extract for each subtree-depth a fixed number of subtrees, where the depth of subtree is the longest path from root to any leaf. This has roughly the same effect as the correction factor used in Bod (2003, 2006). That is, for each particular depth we sample subtrees by first randomly selecting a node in a random tree from the binary tree set after which we select random expansions from that node until a subtree of the particular depth is obtained. For our experiments in section 6, we repeated this procedure 200,000 times for each depth. The resulting subtrees are then given to ML-DOP's reestimation procedure. Finally, the reestimated subtrees are used to compute the most probable parse trees for all sentences using Viterbi n -best, as described in section 3, where the most probable parse is estimated from the 100 most probable derivations.

A potential criticism of (U)ML-DOP is that since we use DOP1's relative frequencies as initial parameters, ML-DOP may only find a local maximum nearest to DOP1's estimator. But this is of course a criticism against any iterative ML approach: it is not guaranteed that the global maximum is found (cf. Manning and Schütze 1999: 401). Nevertheless we will see that our reestimation

procedure leads to significantly better accuracy compared to U-DOP (the latter would be equal to UML-DOP under 0 iterations). Moreover, in contrast to U-DOP, UML-DOP *can* be theoretically motivated: it maximizes the likelihood of the data using the statistically consistent EM algorithm.

6 Experiments: Can we beat supervised by unsupervised parsing?

To compare UML-DOP to U-DOP, we started out with the WSJ10 corpus, which contains 7422 sentences ≤ 10 words after removing empty elements and punctuation. We used the same evaluation metrics for unlabeled precision (UP) and unlabeled recall (UR) as defined in Klein (2005: 21-22). Klein's definitions differ slightly from the standard PARSEVAL metrics: multiplicity of brackets is ignored, brackets of span one are ignored and the bracket labels are ignored. The two metrics of UP and UR are combined by the unlabeled f-score F1 which is defined as the harmonic mean of UP and UR: $F1 = 2 * UP * UR / (UP + UR)$.

For the WSJ10, we obtained a binary tree set of $5.68 * 10^5$ trees, by extracting the binary trees as described in section 5. From this binary tree set we sampled 200,000 subtrees for each subtree-depth. This resulted in a total set of roughly $1.7 * 10^6$ subtrees that were reestimated by our maximum-likelihood procedure. The decrease in cross-entropy became negligible after 14 iterations (for both halves of WSJ10). After computing the most probable parse trees, UML-DOP achieved an f-score of 82.9% which is a 20.5% error reduction compared to U-DOP's f-score of 78.5% on the same data (Bod 2006).

We next tested UML-DOP on two additional domains which were also used in Klein and Manning (2004) and Bod (2006): the German NEGRA10 (Skut et al. 1997) and the Chinese CTB10 (Xue et al. 2002) both containing 2200+ sentences ≤ 10 words after removing punctuation. Table 1 shows the results of UML-DOP compared to U-DOP, the CCM model by Klein and Manning (2002), the DMV dependency learning model by Klein and Manning (2004) as well as their combined model DMV+CCM.

Table 1 shows that UML-DOP scores better than U-DOP and Klein and Manning's models in all cases. It thus pays off to not only use subtrees rather

than substrings (as in CCM) but to also reestimate the subtrees' probabilities by a maximum-likelihood procedure rather than using their (smoothed) relative frequencies (as in U-DOP). Note that UML-DOP achieves these improved results with *fewer* subtrees than U-DOP, due to UML-DOP's more drastic pruning of subtrees. It is also noteworthy that UML-DOP, like U-DOP, does not employ a separate class for non-constituents, so-called distituents, while CCM and CCM+DMV do. (Interestingly, the top 10 most frequently learned constituents by UML-DOP were exactly the same as by U-DOP -- see the relevant table in Bod 2006).

Model	English (WSJ10)	German (NEGRA10)	Chinese (CTB10)
CCM	71.9	61.6	45.0
DMV	52.1	49.5	46.7
DMV+CCM	77.6	63.9	43.3
U-DOP	78.5	65.4	46.6
UML-DOP	82.9	67.0	47.2

Table 1. F-scores of UML-DOP compared to previous models on the same data

We were also interested in testing UML-DOP on longer sentences. We therefore included all WSJ sentences up to 40 words after removing empty elements and punctuation (WSJ40) and again sampled 200,000 subtrees for each depth, using the same method as before. Furthermore, we compared UML-DOP against a supervised binarized PCFG, i.e. a treebank PCFG whose simple relative frequency estimator corresponds to maximum likelihood (Chi and Geman 1998), and which we shall refer to as "ML-PCFG". To this end, we used a random 90%/10% division of WSJ40 into a training set and a test set. The ML-PCFG had thus access to the Penn WSJ trees in the training set, while UML-DOP had to bootstrap all structure from the *flat* strings from the training set to next parse the 10% test set -- clearly a much more challenging task. Table 2 gives the results in terms of f-scores.

The table shows that UML-DOP scores better than U-DOP, also for WSJ40. Our results on WSJ10 are somewhat lower than in table 1 due to the use of a smaller training set of 90% of the data. But the most surprising result is that UML-DOP's f-

score is higher than the *supervised* binarized treebank PCFG (ML-PCFG) for both WSJ10 and WSJ40. In order to check whether this difference is statistically significant, we additionally tested on 10 different 90/10 divisions of the WSJ40 (which were the same splits as in Bod 2006). For these splits, UML-DOP achieved an average f-score of 66.9%, while ML-PCFG obtained an average f-score of 64.7%. The difference in accuracy between UML-DOP and ML-PCFG was statistically significant according to paired *t*-testing ($p \leq 0.05$). To the best of our knowledge this means that we have shown for the first time that *an unsupervised parsing model (UML-DOP) outperforms a widely used supervised parsing model (a treebank PCFG) on the WSJ40*.

Model	WSJ10	WSJ40
U-DOP	78.1	63.9
UML-DOP	82.5	66.4
ML-PCFG	81.5	64.6

Table 2. F-scores of U-DOP, UML-DOP and a supervised treebank PCFG (ML-PCFG) for a random 90/10 split of WSJ10 and WSJ40.

We should keep in mind that (1) a treebank PCFG is not state-of-the-art: its performance is mediocre compared to e.g. Bod (2003) or McClosky et al. (2006), and (2) that our treebank PCFG is binarized as in Klein and Manning (2005) to make results comparable. To be sure, the unbinarized version of the treebank PCFG obtains 89.0% average f-score on WSJ10 and 72.3% average f-score on WSJ40. Remember that the Penn Treebank annotations are often exceedingly flat, and many branches have arity larger than two. It would be interesting to see how UML-DOP performs if we also accept ternary (and wider) branches -- though the total number of possible trees that can be assigned to strings would then further explode.

UML-DOP's performance still remains behind that of *supervised* (binarized) DOP parsers, such as DOP1, which achieved 81.9% average f-score on the 10 WSJ40 splits, and ML-DOP, which performed slightly better with 82.1% average f-score. And if DOP1 and ML-DOP are not binarized, their average f-scores are respectively 90.1% and 90.5% on WSJ40. However, DOP1 and

ML-DOP heavily depend on annotated data whereas UML-DOP only needs unannotated data. It would thus be interesting to see how close UML-DOP can get to ML-DOP's performance if we enlarge the amount of training data.

7 Conclusion: Is the end of supervised parsing in sight?

Now that we have outperformed a well-known supervised parser by an unsupervised one, we may raise the question as to whether the end of supervised NLP comes in sight. All supervised parsers are reaching an asymptote and further improvement does not seem to come from more hand-annotated data but by adding unsupervised or semi-supervised techniques (cf. McClosky et al. 2006). Thus if we modify our question as: does the *exclusively* supervised approach to parsing come to an end, we believe that the answer is certainly yes.

Yet we should neither rule out the possibility that entirely unsupervised methods will in fact surpass semi-supervised methods. The main problem is how to quantitatively compare these different parsers, as any evaluation on hand-annotated data (like the Penn treebank) will unreasonably favor semi-supervised parsers. There is thus a quest for designing an annotation-independent evaluation scheme. Since parsers are becoming increasingly important in applications like syntax-based machine translation and structural language models for speech recognition, one way to go would be to compare these different parsing methods by isolating their contribution in improving a concrete NLP system, rather than by testing them against gold standard annotations which are inherently theory-dependent.

The initially disappointing results of inducing trees entirely from raw text was not so much due to the difficulty of the bootstrapping problem *per se*, but to (1) the poverty of the initial models and (2) the difficulty of finding theory-independent evaluation criteria. The time has come to fully reappraise unsupervised parsing models which should be trained on massive amounts of data, and be evaluated in a concrete application.

There is a final question as to how far the DOP approach to unsupervised parsing can be stretched. In principle we can assign all possible syntactic categories, semantic roles, argument

structures etc. to a set of given sentences and let the statistics decide which assignments are most useful in parsing new sentences. Whether such a massively maximalist approach is feasible can only be answered by empirical investigation in due time.

Acknowledgements

Thanks to Willem Zuidema, David Tugwell and especially to three anonymous reviewers whose unanonymous suggestions on DOP and EM considerably improved the original paper. A substantial part of this research was carried out in the context of the NWO Exact project "Unsupervised Stochastic Grammar Induction from Unlabeled Data", project number 612.066.405.

References

- Bod, R. 1998. *Beyond Grammar: An Experience-Based Theory of Language*, CSLI Publications, distributed by Cambridge University Press.
- Bod, R. 2000. Combining semantic and syntactic structure for language modeling. *Proceedings ICSLP 2000*, Beijing.
- Bod, R. 2003. An efficient implementation of a new DOP model. *Proceedings EACL 2003*, Budapest.
- Bod, R. 2006. Unsupervised Parsing with U-DOP. *Proceedings CONLL 2006*, New York.
- Bonnema, R., R. Bod and R. Scha, 1997. A DOP model for semantic interpretation, *Proceedings ACL/EACL 1997*, Madrid.
- Chi, Z. and S. Geman 1998. Estimation of Probabilistic Context-Free Grammars. *Computational Linguistics* 24, 299-305.
- Clark, A. 2001. Unsupervised induction of stochastic context-free grammars using distributional clustering. *Proceedings CONLL 2001*.
- Collins, M. and N. Duffy 2002. New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron. *Proceedings ACL 2002*, Philadelphia.
- Dempster, A., N. Laird and D. Rubin, 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm, *Journal of the Royal Statistical Society* 39, 1-38.
- Goodman, J. 2003. Efficient algorithms for the DOP model. In R. Bod, R. Scha and K. Sima'an (eds.). *Data-Oriented Parsing*, University of Chicago Press.
- Huang, L. and D. Chiang 2005. Better *k*-best parsing. *Proceedings IWPT 2005*, Vancouver.
- Johnson, M. 2002. The DOP estimation method is biased and inconsistent. *Computational Linguistics* 28, 71-76.
- Klein, D. 2005. *The Unsupervised Learning of Natural Language Structure*. PhD thesis, Stanford University.
- Klein, D. and C. Manning 2002. A general constituent-context model for improved grammar induction. *Proceedings ACL 2002*, Philadelphia.
- Klein, D. and C. Manning 2004. Corpus-based induction of syntactic structure: models of dependency and constituency. *Proceedings ACL 2004*, Barcelona.
- Klein, D. and C. Manning 2005. Natural language grammar induction with a generative constituent-context model. *Pattern Recognition* 38, 1407-1419.
- Magerman, D. 1993. *Expectation-Maximization for Data-Oriented Parsing*, IBM Technical Report, Yorktown Heights, NY.
- McClosky, D., E. Charniak and M. Johnson 2006. Effective self-training for parsing. *Proceedings HLT-NAACL 2006*, New York.
- Manning, C. and H. Schütze 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press.
- Prescher, D., R. Scha, K. Sima'an and A. Zollmann 2004. On the statistical consistency of DOP estimators. *Proceedings CLIN 2004*, Leiden.
- Schütze, H. 1995. Distributional part-of-speech tagging. *Proceedings ACL 1995*, Dublin.
- Shao, J. 1999. *Mathematical Statistics*. Springer Verlag, New York.
- Sima'an, K. 1996. Computational complexity of probabilistic disambiguation by means of tree grammars. *Proceedings COLING 1996*, Copenhagen.
- Skut, W., B. Krenn, T. Brants and H. Uszkoreit 1997. An annotation scheme for free word order languages. *Proceedings ANLP 1997*.
- Xue, N., F. Chiou and M. Palmer 2002. Building a large-scale annotated Chinese corpus. *Proceedings COLING 2002*, Taipei.
- van Zaanen, M. 2000. ABL: Alignment-Based Learning. *Proceedings COLING 2000*, Saarbrücken.
- Zollmann, A. and K. Sima'an 2005. A consistent and efficient estimator for data-oriented parsing. *Journal of Automata, Languages and Combinatorics*, in press.
- Zuidema, W. 2006. What are the productive units of natural language grammar? A DOP approach to the automatic identification of constructions. *Proceedings CONLL 2006*, New York.

Advances in Discriminative Parsing

Joseph Turian and I. Dan Melamed

{lastname}@cs.nyu.edu

Computer Science Department

New York University

New York, New York 10003

Abstract

The present work advances the accuracy and training speed of discriminative parsing. Our discriminative parsing method has no generative component, yet surpasses a generative baseline on constituent parsing, and does so with minimal linguistic cleverness. Our model can incorporate arbitrary features of the input and parse state, and performs feature selection incrementally over an exponential feature space during training. We demonstrate the flexibility of our approach by testing it with several parsing strategies and various feature sets. Our implementation is freely available at: <http://nlp.cs.nyu.edu/parser/>.

1 Introduction

Discriminative machine learning methods have improved accuracy on many NLP tasks, including POS-tagging, shallow parsing, relation extraction, and machine translation. Some advances have also been made on full syntactic constituent parsing. Successful discriminative parsers have relied on generative models to reduce training time and raise accuracy above generative baselines (Collins & Roark, 2004; Henderson, 2004; Taskar et al., 2004). However, relying on information from a generative model might prevent these approaches from realizing the accuracy gains achieved by discriminative methods on other NLP tasks. Another problem is training speed: Discriminative parsers are notoriously slow to train.

In the present work, we make progress towards overcoming these obstacles. We propose a flexible, end-to-end discriminative method for training

parsers, demonstrating techniques that might also be useful for other structured prediction problems. The proposed method does model selection without ad-hoc smoothing or frequency-based feature cutoffs. It requires no heuristics or human effort to optimize the single important hyper-parameter. The training regime can use all available information from the entire parse history. The learning algorithm projects the hand-provided features into a compound feature space and performs incremental feature selection over this large feature space. The resulting parser achieves higher accuracy than a generative baseline, despite not using a generative model as a feature.

Section 2 describes the parsing algorithm. Section 3 presents the learning method. Section 4 presents experiments with discriminative parsers built using these methods. Section 5 compares our approach to related work.

2 Parsing Algorithm

The following terms will help to explain our work. A **span** is a range over contiguous words in the input. Spans **cross** if they overlap but neither contains the other. An **item** is a (span, label) pair. A **state** is a partial parse, i.e. a set of items, none of whose spans may cross. A parse **inference** is a (state, item) pair, i.e. a state and an item to be added to it. The **frontier** of a state consists of the items with no parents yet. The **children** of a candidate inference are the frontier items below the item to be inferred, and the **head** of a candidate inference is the child item chosen by English head rules (Collins, 1999, pp. 238–240). A parse **path** is a sequence of parse inferences. For some input sentence and training parse tree, a state is **correct** if the parser can infer zero or more additional items to obtain the training parse tree, and an inference

is correct if it leads to a correct state.

Given input sentence s , the parser searches for parse $\hat{p}$ out of the possible parses $P(s)$:

$$\hat{p} = \arg \min_{p \in P(s)} C_{\Theta}(p) \quad (1)$$

where $C_{\Theta}(p)$ is the **cost** of parse p under model Θ :

$$C_{\Theta}(p) = \sum_{i \in p} c_{\Theta}(i) \quad (2)$$

Section 3.1 describes how to compute $c_{\Theta}(i)$. Because $c_{\Theta}(i) \in \mathbb{R}^+$, the cost of a partial parse monotonically increases as we add items to it.

The parsing algorithm considers a succession of states. The initial state contains terminal items, whose labels are the POS tags given by the tagger of Ratnaparkhi (1996). Each time we pop a state from the agenda, c_{Θ} computes the costs for the candidate bottom-up inferences generated from that state. Each candidate inference results in a successor state to be placed on the agenda.

The cost function c_{Θ} can consider arbitrary properties of the input and parse state. We are not aware of any tractable solution to Equation 1, such as dynamic programming. Therefore, the parser finds $\hat{p}$ using a variant of uniform-cost search. The parser implements the search using an agenda that stores entire states instead of single items. Each time a state is popped from the agenda, the parser uses depth-first search starting from the state that was popped until it (greedily) finds a complete parse. In preliminary experiments, this search strategy was faster than standard uniform-cost search (Russell & Norvig, 1995).

3 Training Method

3.1 General Setting

Our training set I consists of candidate inferences from the parse trees in the training data. From each training inference $i \in I$ we generate the tuple $\langle X(i), y(i), b(i) \rangle$. $X(i)$ is a feature vector describing i , with each element in $\{0, 1\}$. We will use $X_f(i)$ to refer to the element of $X(i)$ that pertains to feature f . $y(i) = +1$ if i is correct, and $y(i) = -1$ if not. Some training examples might be more important than others, so each is given a **bias** $b(i) \in \mathbb{R}^+$, as detailed in Section 3.3.

The goal during training is to induce a hypothesis $h_{\Theta}(i)$, which is a real-valued inference scoring function. In the present work, h_{Θ} is a linear model parameterized by a real vector Θ , which has one

entry for each feature f :

$$h_{\Theta}(i) = \Theta \cdot X(i) = \sum_f \Theta_f \cdot X_f(i) \quad (3)$$

The sign of $h_{\Theta}(i)$ predicts the y -value of i and the magnitude gives the confidence in this prediction.

The training procedure optimizes Θ to minimize the expected risk R_{Θ} over training set I . R_{Θ} is the **objective** function, a combination of **loss** function L_{Θ} and **regularization** term Ω_{Θ} :

$$R_{\Theta}(I) = L_{\Theta}(I) + \Omega_{\Theta} \quad (4)$$

The loss of the inference set decomposes into the loss of individual inferences:

$$L_{\Theta}(I) = \sum_{i \in I} l_{\Theta}(i) \quad (5)$$

In principle, l_{Θ} can be any loss function, but in the present work we use the log-loss (Collins et al., 2002):

$$l_{\Theta}(i) = b(i) \cdot \ln(1 + \exp(-\mu_{\Theta}(i))) \quad (6)$$

and $\mu_{\Theta}(i)$ is the **margin** of inference i :

$$\mu_{\Theta}(i) = y(i) \cdot h_{\Theta}(i) \quad (7)$$

Inference cost $c_{\Theta}(i)$ in Equation 2 is $l_{\Theta}(i)$ computed using $y(i) = +1$ and $b(i) = 1$, i.e.:

$$c_{\Theta}(i) = \ln(1 + \exp(-h_{\Theta}(i))) \quad (8)$$

Ω_{Θ} in Equation 4 is a regularizer, which penalizes complex models to reduce overfitting and generalization error. We use the ℓ_1 penalty:

$$\Omega_{\Theta} = \sum_f \lambda \cdot |\Theta_f| \quad (9)$$

where λ is a parameter that controls the strength of the regularizer. This choice of objective R_{Θ} is motivated by Ng (2004), who suggests that, given a learning setting where the number of irrelevant features is exponential in the number of training examples, we can nonetheless learn effectively by building decision trees to minimize the ℓ_1 -regularized log-loss. On the other hand, Ng (2004) suggests that most of the learning algorithms commonly used by discriminative parsers *will* overfit when exponentially many irrelevant features are present.¹

Learning over an exponential feature space is the very setting we have in mind. *A priori*, we define only a set A of simple **atomic** features (given

¹including the following learning algorithms:

- unregularized logistic regression
- logistic regression with an ℓ_2 penalty (i.e. a Gaussian prior)
- SVMs using most kernels
- multilayer neural nets trained by backpropagation
- the perceptron algorithm

in Section 4). The learner then induces **compound** features, each of which is a conjunction of possibly negated atomic features. Each atomic feature can have one of three values (yes/no/don't care), so the size of the compound feature space is $3^{|A|}$, exponential in the number of atomic features. It was also exponential in the number of training examples in our experiments ($|A| \approx |I|$).

3.2 Boosting ℓ_1 -Regularized Decision Trees

We use an ensemble of confidence-rated decision trees (Schapire & Singer, 1999) to represent h_Θ .² The path from the root to each node n in a decision tree corresponds to some compound feature f , and we write $\varphi(n) = f$. To score an inference i using a decision tree, we percolate the inference's features $X(i)$ down to a leaf n and return confidence $\Theta_{\varphi(n)}$. An inference i percolates down to node n iff $X_{\varphi(n)} = 1$. Each leaf node n keeps track of the parameter value $\Theta_{\varphi(n)}$.³ The score $h_\Theta(i)$ given to an inference i by the whole ensemble is the sum of the confidences returned by the trees in the ensemble.

Listing 1 Outline of training algorithm.

```

1: procedure TRAIN( $I$ )
2:   ensemble  $\leftarrow \emptyset$ 
3:    $\lambda \leftarrow \infty$ 
4:   while dev set accuracy is increasing do
5:      $t \leftarrow$  tree with one (root) node
6:     while the root node cannot be split do
7:       decay  $\ell_1$  parameter  $\lambda$ 
8:     while some leaf in  $t$  can be split do
9:       split the leaf to maximize gain
10:    percolate every  $i \in I$  to a leaf node
11:    for each leaf  $n$  in  $t$  do
12:      update  $\Theta_{\varphi(n)}$  to minimize  $R_\Theta$ 
13:    append  $t$  to ensemble

```

Listing 1 presents our training algorithm. At the beginning of training, the ensemble is empty, $\Theta = \mathbf{0}$, and the ℓ_1 parameter λ is set to ∞ (Steps 1.2 and 1.3). We train until the objective cannot be further reduced for the current choice of λ . We then determine the accuracy of the parser on a held-out development set using the previous λ value (before it was decreased), and stop training when this

²Turian and Melamed (2005) reported that decision trees applied to parsing have higher accuracy and training speed than decision stumps, so we build full decision trees rather than stumps.

³Any given compound feature can appear in more than one tree, but each leaf node has a distinct confidence value. For simplicity, we ignore this possibility in our discussion.

accuracy reaches a plateau (Step 1.4). Otherwise, we relax the regularization penalty by decreasing λ (Steps 1.6 and 1.7) and continue training. In this way, instead of choosing the best λ heuristically, we can optimize it during a single training run (Turian & Melamed, 2005).

Each training iteration (Steps 1.5–1.13) has several steps. First, we choose some compound features that have high magnitude gradient with respect to the objective function. We do this by building a new decision tree, whose leaves represent the chosen compound features (Steps 1.5–1.9). Second, we confidence-rate each leaf to minimize the objective over the examples that percolate down to that leaf (Steps 1.10–1.12). Finally, we append the decision tree to the ensemble and update parameter vector Θ accordingly (Step 1.13). In this manner, compound feature selection is performed incrementally *during* training, as opposed to *a priori*.

Our strategy minimizing the objective $R_\Theta(I)$ (Equation 4) is a variant of steepest descent (Perkins et al., 2003). To compute the gradient of the unpenalized loss L_Θ with respect to the parameter Θ_f of feature f , we have:

$$\frac{\partial L_\Theta(I)}{\partial \Theta_f} = \sum_{i \in I} \frac{\partial l_\Theta(i)}{\partial \mu_\Theta(i)} \cdot \frac{\partial \mu_\Theta(i)}{\partial \Theta_f} \quad (10)$$

where:

$$\frac{\partial \mu_\Theta(i)}{\partial \Theta_f} = y(i) \cdot X_f(i) \quad (11)$$

Using Equation 6, we define the **weight** of an example i under the current model as the rate at which loss decreases as the margin of i increases:

$$w_\Theta(i) = -\frac{\partial l_\Theta(i)}{\partial \mu_\Theta(i)} = b(i) \cdot \frac{1}{1 + \exp(\mu_\Theta(i))} \quad (12)$$

Recall that $X_f(i)$ is either 0 or 1. Combining Equations 10–12 gives:

$$\frac{\partial L_\Theta(I)}{\partial \Theta_f} = - \sum_{\substack{i \in I \\ X_f(i)=1}} y(i) \cdot w_\Theta(i) \quad (13)$$

We define the **gain** of feature f as:

$$G_\Theta(I; f) = \max\left(0, \left| \frac{\partial L_\Theta(I)}{\partial \Theta_f} \right| - \lambda\right) \quad (14)$$

Equation 14 has this form because the gradient of the penalty term is undefined at $\Theta_f = 0$. This discontinuity is why ℓ_1 regularization tends to produce sparse models. If $G_\Theta(I; f) = 0$, then the objective $R_\Theta(I)$ is at its minimum with respect to parameter Θ_f . Otherwise, $G_\Theta(I; f)$ is the magnitude

of the gradient of the objective as we adjust Θ_f in the appropriate direction.

To build each decision tree, we begin with a root node. The root node corresponds to a dummy “always true” feature. We recursively split nodes by choosing a splitting feature that will allow us to increase the gain. Node n with corresponding compound feature $\varphi(n) = f$ can be split by atomic feature a if:

$$G_{\Theta}(I; f \wedge a) + G_{\Theta}(I; f \wedge \neg a) > G_{\Theta}(I; f) \quad (15)$$

If no atomic feature satisfies the splitting criterion in Equation 15, then n becomes a leaf node of the decision tree and $\Theta_{\varphi(n)}$ becomes one of the values to be optimized during the parameter update step. Otherwise, we choose atomic feature $\hat{a}$ to split node n :

$$\hat{a} = \arg \max_{a \in A} (G_{\Theta}(I; f \wedge a) + G_{\Theta}(I; f \wedge \neg a)) \quad (16)$$

This split creates child nodes n_1 and n_2 , with $\varphi(n_1) = f \wedge \hat{a}$ and $\varphi(n_2) = f \wedge \neg \hat{a}$.

Parameter update is done sequentially on only the most recently added compound features, which correspond to the leaves of the new decision tree. After the entire tree is built, we percolate examples down to their appropriate leaf nodes. We then choose for each leaf node n the parameter $\Theta_{\varphi(n)}$ that minimizes the objective over the examples in that leaf. A convenient property of decision trees is that the leaves’ compound features are mutually exclusive. Their parameters can be directly optimized independently of each other using a line search over the objective.

3.3 The Training Set

We choose a single correct path from each training parse tree, and the training examples correspond to all candidate inferences considered in every state along this path.⁴ In the deterministic setting there is only one correct path, so example generation is identical to that of Sagae and Lavie (2005). If parsing proceeds non-deterministically then there might be multiple paths that lead to the same final parse, so we choose one randomly. This method of generating training examples does not require a working parser and can be run prior to any training. The disadvantage of this approach is that it minimizes the error of the parser at *correct* states only. It does not account for compounded error or

⁴Nearly all of the examples generated are negative ($y = -1$).

teach the parser to recover from mistakes gracefully.

Turian and Melamed (2005) observed that uniform example biases $b(i)$ produced lower accuracy as training progressed, because the induced classifiers minimized the error per *example*. To minimize the error per *state*, we assign every training state equal value and share half the value uniformly among the negative examples for the examples generated from that state and the other half uniformly among the positive examples.

We parallelize training by inducing 26 label classifiers (one for each non-terminal label in the Penn Treebank). Parallelization might not uniformly reduce training time because different label classifiers train at different rates. However, parallelization uniformly reduces *memory* usage because each label classifier trains only on inferences whose consequent item has that label.

4 Experiments

Discriminative parsers are notoriously slow to train. For example, Taskar et al. (2004) took several months to train on the ≤ 15 word sentences in the English Penn Treebank (Dan Klein, p.c.). The present work makes progress towards faster discriminative parser training: our slowest classifier took fewer than 5 days to train. Even so, it would have taken much longer to train on the entire treebank. We follow Taskar et al. (2004) in training and testing on ≤ 15 word sentences in the English Penn Treebank (Taylor et al., 2003). We used sections 02–21 for training, section 22 for development, and section 23 for testing, preprocessed as per Table 1. We evaluated our parser using the standard PARSEVAL measures (Black et al., 1991): labelled precision, labelled recall, and labelled F-measure (Prec., Rec., and F_1 , respectively), which are based on the number of non-terminal items in the parser’s output that match those in the gold-standard parse.⁵

As mentioned in Section 2, items are inferred bottom-up and the parser cannot infer any item that crosses an item already in the state. Although there are $O(n^2)$ possible (span, label) pairs over a frontier containing n items, we reduce this to the $\approx 5 \cdot n$ inferences that have at most five children.⁶

⁵The correctness of a stratified shuffling test has been called into question (Michael Collins, p.c.), so we are not aware of any valid significance tests for observed differences in PARSEVAL scores.

⁶Only 0.57% of non-terminals in the preprocessed develop-

Table 1 Steps for preprocessing the data. Starred steps are performed only when parse trees are available in the data (e.g. not on test data).

1. * Strip functional tags and trace indices, and remove traces.
2. * Convert PRT to ADVP. (This convention was established by Magerman (1995).)
3. Remove quotation marks (i.e. terminal items tagged ‘ ‘ or ’ ’). (Bikel, 2004)
4. * Raise punctuation. (Bikel, 2004)
5. Remove outermost punctuation.^a
6. * Remove unary projections to self (i.e. duplicate items with the same span and label).
7. POS tag the text using the tagger of Ratnaparkhi (1996).
8. Lowercase headwords.

^aAs pointed out by an anonymous reviewer of Collins (2003), removing outermost punctuation might discard useful information. Collins and Roark (2004) saw a LFMS improvement of 0.8% over their baseline discriminative parser after adding punctuation features, one of which encoded the sentence-final punctuation.

To ensure the parser does not enter an infinite loop, no two items in a state can have both the same span and the same label. Given these restrictions on candidate inferences, there were roughly 40 million training examples generated in the training set. These were partitioned among the 26 constituent label classifiers. Building a decision tree (Steps 1.5–1.9 in Listing 1) using the entire example set I can be very expensive. We estimate loss gradients (Equation 13) using a sample of the inference set, which gives a 100-fold increase in training speed (Turian & Melamed, 2006).

Our atomic feature set A contains 300K features, each of the form “is there an item in group J whose label/headword/headtag/headtagclass is ‘X’?”.⁷ Possible values of ‘X’ for each predicate are collected from the training data. For $1 \leq n \leq 3$, possible values for J are:

- the first/last n child items
- the first n left/right context items
- the n children items left/right of the head
- the head item.

The left and right **context** items are the frontier items to the left and right of the children of the candidate inference, respectively.

4.1 Different Parsing Strategies

To demonstrate the flexibility of our learning procedure, we trained three different parsers: left-to-right (l2r), right-to-left (r2l),

ment set have more than five children.

⁷The predicate headtagclass is a supertype of the headtag. Given our compound features, these are not strictly necessary, but they accelerate training. An example is “proper noun,” which contains the POS tags given to singular and plural proper nouns. Space constraints prevent enumeration of the headtagclasses, which are instead provided at the URL given in the abstract.

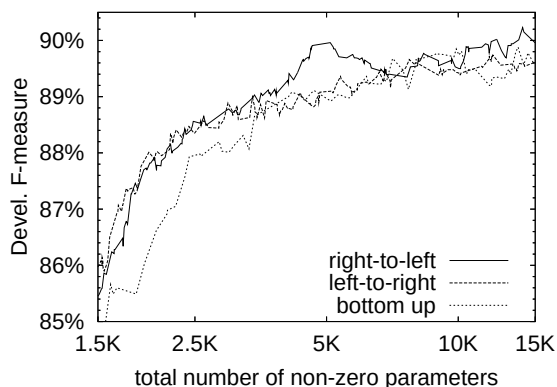
Table 2 Results on the development set, training and testing using only ≤ 15 word sentences.

	active		% Rec.	% Prec.	F ₁
	λ	features			
l2r	0.040	11.9K	89.86	89.63	89.74
b.u.	0.020	13.7K	89.92	89.84	89.88
r2l	0.014	14.0K	90.66	89.81	90.23

and non-deterministic bottom-up (b.u.). The non-deterministic parser was allowed to choose any bottom-up inference. The other two parsers were deterministic: bottom-up inferences had to be performed strictly left-to-right or right-to-left, respectively. We stopped training when each parser had 15K active features. Figure 1 shows the accuracy of the different runs over the development set as training progressed. Table 2 gives the PARSEVAL scores of these parsers at their optimal ℓ_1 penalty setting. We found that the perplexity of the r2l model was low so that, in 85% of the sentences, its greedy parse was the optimal one. The l2r parser does poorly because its decisions were more difficult than those of the other parsers. If it inferred far-right items, it was more likely to prevent correct subsequent inferences that were to the left. But if it inferred far-left items, then it went against the right-branching tendency of English sentences. The left-to-right parser would likely improve if we were to use a left-corner transform (Collins & Roark, 2004).

Parsers in the literature typically choose some local threshold on the amount of search, such as a maximum beam width. With an accurate scoring function, restricting the search space using a fixed beam width might be unnecessary. Instead, we imposed a *global* threshold on exploration of the search space. Specifically, if the

Figure 1 F_1 scores on the development set of the Penn Treebank, using only ≤ 15 word sentences. The x-axis shows the number of non-zero parameters in each parser, summed over all classifiers.



parser has found some complete parse and has explored at least 100K states (i.e. scored at least 100K inferences), search stopped prematurely and the parser would return the (possibly sub-optimal) current best complete parse. The l2r and r2l parsers never exceeded this threshold, and always found the optimal complete parse. However, the non-deterministic bottom-up parser’s search was cut-short in 28% of the sentences. The non-deterministic parser can reach each parse state through many different paths, so it searches a larger space than a deterministic parser, with more redundancy.

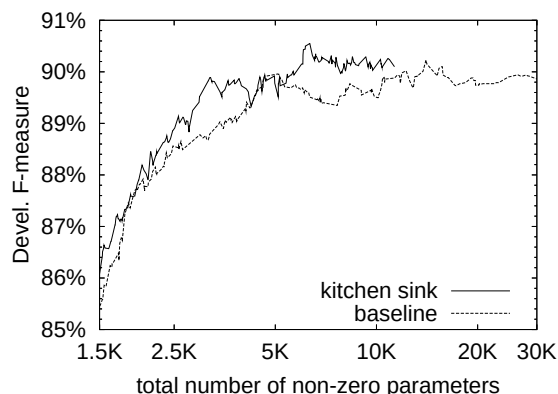
To gain a better understanding of the weaknesses of our parser, we examined a sample of 50 development sentences that the r2l parser did not get entirely correct. Roughly half the errors were due to noise and genuine ambiguity. The remaining errors fell into three types, occurring with roughly the same frequency:

- **ADVPs and ADJPs** The r2l parser had $F_1 = 81.1\%$ on ADVPs, and $F_1 = 71.3\%$ on ADJPs. Annotation of ADJP and ADVP in the PTB is inconsistent, particularly for *unary* projections.

- **POS Tagging Errors** Many of the parser’s errors were due to incorrect POS tags. In future work we will integrate POS-tagging as inferences of the parser, allowing it to entertain competing hypotheses about the correct tagging.

- **Bilexical dependencies** Although compound features exist to detect affinities between words, the parser had difficulties with bilexical dependency decisions that were unobserved in the training data. The classifier would need more training data to learn these affinities.

Figure 2 F_1 scores of right-to-left parsers with different atomic feature sets on the development set of the Penn Treebank, using only ≤ 15 word sentences.



4.2 More Atomic Features

We compared our right-to-left parser with the baseline set of atomic features to one with a far richer atomic feature set, including *unbounded* context features, length features, and features of the terminal items. This “kitchen sink” parser merely has access to many more item groups J , described in Table 3. All features are all of the form given earlier, except for length features (Eisner & Smith, 2005). Length features compute the size of one of the groups of items in the indented list in Table 3. The feature determines if this length is equal to/greater than to n , $0 \leq n \leq 15$. The kitchen sink parser had 1.1 million atomic features, 3.7 times the number available in the baseline. In future work, we plan to try linguistically more sophisticated features (Charniak & Johnson, 2005) as well as sub-tree features (Bod, 2003; Kudo et al., 2005).

Figure 2 shows the accuracy of the right-to-left parsers with different atomic feature sets over the development set as training progressed. Even though the baseline training made progress more quickly than the kitchen sink, the kitchen sink’s F_1 surpassed the baseline’s F_1 early in training, and at 6.3K active parameters it achieved a development set F_1 of 90.55%.

4.3 Test Set Results

To situate our results in the literature, we compare our results to those reported by Taskar et al. (2004) and Turian and Melamed (2005) for their discriminative parsers, which were also trained and tested on ≤ 15 word sentences. We also compare our parser to a representative non-discriminative

Table 3 Item groups available in the kitchen sink run.

- the first/last n child items, $1 \leq n \leq 4$
- the first n left/right context items, $1 \leq n \leq 4$
- the n children items left/right of the head, $1 \leq n \leq 4$
- the n th frontier item left/right of the leftmost/head/rightmost child item, $1 \leq n \leq 3$
- the n th terminal item left/right of the leftmost/head/rightmost terminal item dominated by the item being inferred, $1 \leq n \leq 3$
- the leftmost/head/rightmost child item of the leftmost/head/rightmost child item
- the following groups of frontier items:
 - all items
 - left/right context items
 - non-leftmost/non-head/non-rightmost child items
 - child items left/right of the head item, inclusive/exclusive
- the terminal items dominated by one of the item groups in the indented list above

Table 4 Results of parsers on the test set, training and testing using only ≤ 15 word sentences.

	% Rec.	% Prec.	F ₁
Turian and Melamed (2005)	86.47	87.80	87.13
Bikel (2004)	87.85	88.75	88.30
Taskar et al. (2004)	89.10	89.14	89.12
kitchen sink	89.26	89.55	89.40

parser (Bikel, 2004)⁸, the only one that we were able to train and test under exactly the same experimental conditions (including the use of POS tags from the tagger of Ratnaparkhi (1996)). Table 4 shows the PARSEVAL results of these four parsers on the test set.

5 Comparison with Related Work

Our parsing approach is based upon a single end-to-end discriminative learning machine. Collins and Roark (2004) and Taskar et al. (2004) beat the generative baseline only after using the standard trick of using the output from a generative model as a feature. Henderson (2004) finds that discriminative training was too slow, and reports accuracy higher than generative models by discriminatively reranking the output of his generative model. Unlike these state-of-the-art discriminative parsers, our method does not (yet) use any information from a generative model to improve training speed or accuracy. As far as we know, we present the first discriminative parser that does not use information from a generative model to beat a

generative baseline (the Collins model).

The main limitation of our work is that we can do training reasonably quickly only on short sentences because a sentence with n words generates $O(n^2)$ training inferences in total. Although generating training examples in advance without a working parser (Turian & Melamed, 2005) is much faster than using inference (Collins & Roark, 2004; Henderson, 2004; Taskar et al., 2004), our training time can probably be decreased further by choosing a parsing strategy with a lower branching factor. Like our work, Ratnaparkhi (1999) and Sagae and Lavie (2005) generate examples off-line, but their parsing strategies are essentially shift-reduce so each sentence generates only $O(n)$ training examples.

An advantage of our approach is its flexibility. As our experiments showed, it is quite simple to substitute in different parsing strategies. Although we used very little linguistic information (the head rules and the POS tag classes), our model could also start with more sophisticated task-specific features in its atomic feature set. Atomic features that access arbitrary information are represented directly without the need for an induced intermediate representation (cf. Henderson, 2004).

Other papers (Clark & Curran, 2004; Kaplan et al., 2004, e.g.) have applied log-linear models to parsing. These works are based upon conditional models, which include a normalization term. However, our loss function forgoes normalization, which means that it is easily decomposed into the loss of individual inferences (Equation 5).

⁸Bikel (2004) is a “clean room” reimplementation of the Collins (1999) model with comparable accuracy.

Decomposition of the loss allows the objective to be optimized in parallel. This might be an advantage for larger structured prediction problems where there are more opportunities for parallelization, for example machine translation.

The only important hyper-parameter in our method is the ℓ_1 penalty factor. We optimize it as part of the training process, choosing the value that maximizes accuracy on a held-out development set. This technique stands in contrast to more ad-hoc methods for choosing hyper-parameters, which may require prior knowledge or additional experimentation.

6 Conclusion

Our work has made advances in both accuracy and training speed of discriminative parsing. As far as we know, we present the first discriminative parser that surpasses a generative baseline on constituent parsing without using a generative component, and it does so with minimal linguistic cleverness. Our approach performs feature selection incrementally over an exponential feature space during training. Our experiments suggest that the learning algorithm is overfitting-resistant, as hypothesized by Ng (2004). If this is the case, it would reduce the effort required for feature engineering. An engineer can merely design a set of atomic features whose powerset contains the requisite information. Then, the learning algorithm can perform feature selection over the compound feature space, avoiding irrelevant compound features.

In future work, we shall make some standard improvements. Our parser should infer its own POS tags to improve accuracy. A shift-reduce parsing strategy will generate fewer training inferences, and might lead to shorter training times. Lastly, we plan to give the model linguistically more sophisticated features. We also hope to apply the model to other structured prediction tasks, such as syntax-driven machine translation.

Acknowledgments

The authors would like to thank Chris Pike, Cynthia Rudin, and Ben Wellington, as well as the anonymous reviewers, for their helpful comments and constructive criticism. This research was sponsored by NSF grants #0238406 and #0415933.

References

- Bikel, D. M. (2004). Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4).
- Black, E., Abney, S., Flickenger, D., Gdaniec, C., Grishman, R., Harrison, P., et al. (1991). A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Speech and Natural Language*.
- Bod, R. (2003). An efficient implementation of a new DOP model. In *EACL*.
- Charniak, E., & Johnson, M. (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *ACL*.
- Clark, S., & Curran, J. R. (2004). Parsing the WSJ using CCG and log-linear models. In *ACL*.
- Collins, M. (1999). *Head-driven statistical models for natural language parsing*. Doctoral dissertation.
- Collins, M. (2003). Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4).
- Collins, M., & Roark, B. (2004). Incremental parsing with the perceptron algorithm. In *ACL*.
- Collins, M., Schapire, R. E., & Singer, Y. (2002). Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48(1-3).
- Eisner, J., & Smith, N. A. (2005). Parsing with soft and hard constraints on dependency length. In *IWPT*.
- Henderson, J. (2004). Discriminative training of a neural network statistical parser. In *ACL*.
- Kaplan, R. M., Riezler, S., King, T. H., Maxwell, III, J. T., Vasserman, A., & Crouch, R. (2004). Speed and accuracy in shallow and deep stochastic parsing. In *HLT/NAACL*.
- Kudo, T., Suzuki, J., & Isozaki, H. (2005). Boosting-based parse reranking with subtree features. In *ACL*.
- Magerman, D. M. (1995). Statistical decision-tree models for parsing. In *ACL*.
- Ng, A. Y. (2004). Feature selection, ℓ_1 vs. ℓ_2 regularization, and rotational invariance. In *ICML*.
- Perkins, S., Lacker, K., & Theiler, J. (2003). Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3.
- Ratnaparkhi, A. (1996). A maximum entropy part-of-speech tagger. In *EMNLP*.
- Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine Learning*, 34(1-3).
- Russell, S., & Norvig, P. (1995). *Artificial intelligence: A modern approach*.
- Sagae, K., & Lavie, A. (2005). A classifier-based parser with linear run-time complexity. In *IWPT*.
- Schapire, R. E., & Singer, Y. (1999). Improved boosting using confidence-rated predictions. *Machine Learning*, 37(3).
- Taskar, B., Klein, D., Collins, M., Koller, D., & Manning, C. (2004). Max-margin parsing. In *EMNLP*.
- Taylor, A., Marcus, M., & Santorini, B. (2003). The Penn Treebank: an overview. In A. Abeillé (Ed.), *Treebanks: Building and using parsed corpora* (chap. 1).
- Turian, J., & Melamed, I. D. (2005). Constituent parsing by classification. In *IWPT*.
- Turian, J., & Melamed, I. D. (2006). Computational challenges in parsing by classification. In *HLT-NAACL workshop on computationally hard problems and joint inference in speech and language processing*.

Prototype-Driven Grammar Induction

Aria Haghighi

Computer Science Division
University of California Berkeley
aria42@cs.berkeley.edu

Dan Klein

Computer Science Division
University of California Berkeley
klein@cs.berkeley.edu

Abstract

We investigate *prototype-driven* learning for primarily unsupervised grammar induction. Prior knowledge is specified declaratively, by providing a few canonical examples of each target phrase type. This sparse prototype information is then propagated across a corpus using distributional similarity features, which augment an otherwise standard PCFG model. We show that distributional features are effective at distinguishing bracket *labels*, but not determining bracket *locations*. To improve the quality of the induced trees, we combine our PCFG induction with the CCM model of Klein and Manning (2002), which has complementary strengths: it identifies brackets but does not label them. Using only a handful of prototypes, we show substantial improvements over naive PCFG induction for English and Chinese grammar induction.

1 Introduction

There has been a great deal of work on unsupervised grammar induction, with motivations ranging from scientific interest in language acquisition to engineering interest in parser construction (Carroll and Charniak, 1992; Clark, 2001). Recent work has successfully induced unlabeled grammatical structure, but has not successfully learned labeled tree structure (Klein and Manning, 2002; Klein and Manning, 2004; Smith and Eisner, 2004).

In this paper, our goal is to build a system capable of producing labeled parses in a target grammar with as little total effort as possible. We investigate a *prototype-driven* approach to grammar induction, in which one supplies canonical examples of each target concept. For example, we might specify that we are interested in trees which use the symbol NP and then list several examples of prototypical NPs (determiner noun, pronouns, etc., see figure 1 for a sample prototype list). This prototype information is similar to specifying an *annotation scheme*, which even human annotators

must be provided before they can begin the construction of a treebank. In principle, prototype-driven learning is just a kind of semi-supervised learning. However, in practice, the information we provide is on the order of dozens of total seed instances, instead of a handful of fully parsed trees, and is of a different nature.

The prototype-driven approach has three strengths. First, since we provide a set of target symbols, we can evaluate induced trees using standard labeled parsing metrics, rather than the far more forgiving unlabeled metrics described in, for example, Klein and Manning (2004). Second, knowledge is declaratively specified in an interpretable way (see figure 1). If a user of the system is unhappy with its systematic behavior, they can alter it by altering the prototype information (see section 7.1 for examples). Third, and related to the first two, one does not confuse the ability of the system to learn a consistent grammar with its ability to learn the grammar a user has in mind.

In this paper, we present a series of experiments in the induction of labeled context-free trees using a combination of unlabeled data and sparse prototypes. We first affirm the well-known result that simple, unconstrained PCFG induction produces grammars of poor quality as measured against treebank structures. We then augment a PCFG with prototype features, and show that these features, when propagated to non-prototype sequences using distributional similarity, are effective at learning bracket labels on fixed unlabeled trees, but are still not enough to learn good tree structures without bracketing information. Finally, we intersect the feature-augmented PCFG with the CCM model of Klein and Manning (2002), a high-quality bracketing model. The intersected model is able to learn trees with higher unlabeled F_1 than those in Klein and Manning (2004). More impor-

tantly, its trees are labeled and can be evaluated according to labeled metrics. Against the English Penn Treebank, our final trees achieve a labeled F_1 of 65.1 on short sentences, a 51.7% error reduction over naive PCFG induction.

2 Experimental Setup

The majority of our experiments induced tree structures from the WSJ section of the English Penn treebank (Marcus et al., 1994), though see section 7.4 for an experiment on Chinese. To facilitate comparison with previous work, we extracted WSJ-10, the 7,422 sentences which contain 10 or fewer words after the removal of punctuation and null elements according to the scheme detailed in Klein (2005). We learned models on all or part of this data and compared their predictions to the manually annotated treebank trees for the sentences on which the model was trained. As in previous work, we begin with the part-of-speech (POS) tag sequences for each sentence rather than lexical sequences (Carroll and Charniak, 1992; Klein and Manning, 2002).

Following Klein and Manning (2004), we report unlabeled bracket precision, recall, and F_1 . Note that according to their metric, brackets of size 1 are omitted from the evaluation. Unlike that work, all of our induction methods produce trees labeled with symbols which are identified with treebank categories. Therefore, we also report labeled precision, recall, and F_1 , still ignoring brackets of size 1.¹

3 Experiments in PCFG induction

As an initial experiment, we used the inside-outside algorithm to induce a PCFG in the straightforward way (Lari and Young, 1990; Manning and Schütze, 1999). For all the experiments in this paper, we considered binary PCFGs over the nonterminals and terminals occurring in WSJ-10. The PCFG rules were of the following forms:

- $X \rightarrow Y Z$, for nonterminal types X, Y , and Z , with $Y \neq X$ or $Z \neq X$
- $X \rightarrow t Y$, $X \rightarrow Y t$, for each terminal t
- $X \rightarrow t t'$, for terminals t and t'

For a given sentence S , our CFG generates labeled trees T over S .² Each tree consists of binary

productions $X(i, j) \rightarrow \alpha$ over constituent spans (i, j) , where α is a pair of non-terminal and/or terminal symbols in the grammar. The generative probability of a tree T for S is:

$$P_{CFG}(T, S) = \prod_{X(i,j) \rightarrow \alpha \in T} P(\alpha|X)$$

In the inside-outside algorithm, we iteratively compute posterior expectations over production occurrences at each training span, then use those expectations to re-estimate production probabilities. This process is guaranteed to converge to a local extremum of the data likelihood, but initial production probability estimates greatly influence the final grammar (Carroll and Charniak, 1992). In particular, uniform initial estimates are an (unstable) fixed point. The classic approach is to add a small amount of random noise to the initial probabilities in order to break the symmetry between grammar symbols.

We randomly initialized 5 grammars using treebank non-terminals and trained each to convergence on the first 2000 sentences of WSJ-10. Viterbi parses were extracted for each of these 2000 sentences according to each grammar. Of course, the parses' symbols have nothing to anchor them to our intended treebank symbols. That is, an NP in one of these grammars may correspond to the target symbol VP, or may not correspond well to any target symbol. To evaluate these learned grammars, we must map the models' phrase types to target phrase types. For each grammar, we followed the common approach of greedily mapping model symbols to target symbols in the way which maximizes the labeled F_1 . Note that this can, and does, result in mapping multiple model symbols to the most frequent target symbols. This experiment, labeled PCFG $\times$ NONE in figure 4, resulted in an average labeled F_1 of 26.3 and an unlabeled F_1 of 45.7. The unlabeled F_1 is better than randomly choosing a tree (34.7), but not better than always choosing a right branching structure (61.7).

Klein and Manning (2002) suggest that the task of labeling constituents is significantly easier than identifying them. Perhaps it is too much to ask a PCFG induction algorithm to perform both of these tasks simultaneously. Along the lines of Pereira and Schabes (1992), we reran the inside-outside algorithm, but this time placed zero mass on all trees which did not respect the bracketing of the gold trees. This constraint does not fully

¹In cases where multiple gold labels exist in the gold trees, precision and recall were calculated as in Collins (1999).

²Restricting our CFG to a binary branching grammar results in an upper bound of 88.1% on unlabeled F_1 .

Phrase	Prototypes	Phrase	Prototypes
NP	DT NN JJ NNS NNP NNP	VP	VBN IN NN VBD DT NN MD VB CD
S	PRP VBD DT NN DT NN VBD IN DT NN DT VBZ DT JJ NN	QP	CD CD RB CD DT CD CD
PP	IN NN TO CD CD	ADJP	RB JJ JJ JJ CC JJ
ADVP	IN PRP RB RB RB CD RB CC RB		
VP-INF	VB NN	NP-INF	NN POS

Figure 1: English phrase type prototype list manually specified (The entire supervision for our system). The second part of the table is additional prototypes discussed in section 7.1.

eliminate the structural uncertainty since we are inducing binary trees and the gold trees are flatter than binary in many cases. This approach of course achieved the upper bound on unlabeled F_1 , because of the gold bracket constraints. However, it only resulted in an average labeled F_1 of 52.6 (experiment PCFG $\times$ GOLD in figure 4). While this labeled score is an improvement over the PCFG $\times$ NONE experiment, it is still relatively disappointing.

3.1 Encoding Prior Knowledge with Prototypes

Clearly, we need to do something more than adding structural bias (e.g. bracketing information) if we are to learn a PCFG in which the symbols have the meaning and behaviour we intend. How might we encode information about our prior knowledge or intentions?

Providing labeled trees is clearly an option. This approach tells the learner how symbols should recursively relate to each other. Another option is to provide fully linearized yields as prototypes. We take this approach here, manually creating a list of POS sequences typical of the 7 most frequent categories in the Penn Treebank (see figure 1).³ Our grammar is limited to these 7 phrase types plus an additional type which has no prototypes and is unconstrained.⁴ This list grounds each sym-

³A possible objection to this approach is the introduction of improper researcher bias via specifying prototypes. See section 7.3 for an experiment utilizing an automatically generated prototype list with comparable results.

⁴In our experiments we found that adding prototypes for more categories did not improve performance and took more

bol in terms of an observable portion of the data, rather than attempting to relate unknown symbols to other unknown symbols.

Broadly, we would like to learn a grammar which explains the observed data (EM’s objective) but also meets our prior expectations or requirements of the target grammar. How might we use such a list to constrain the learning of a PCFG with the inside-outside algorithm? We might require that all occurrences of a prototype sequence, say DT NN, be constituents of the corresponding type (NP). However, human-elicited prototypes are not likely to have the property that, when they occur, they are (nearly) always constituents. For example, DT NN is a perfectly reasonable example of a noun phrase, but is not a constituent when it is part of a longer DT NN NN constituent. Therefore, when summing over trees with the inside-outside algorithm, we could require a weaker property: whenever a prototype sequence is a constituent it must be given the label specified in the prototype file.⁵ This constraint is enough to break the symmetry between the model labels, and therefore requires neither random initialization for training, nor post-hoc mapping of labels for evaluation. Adding prototypes in this way and keeping the gold bracket constraint gave 59.9 labeled F_1 . The labeled F_1 measure is again an improvement over naive PCFG induction, but is perhaps less than we might expect given that the model has been given bracketing information and has prototypes as a form of supervision to direct it.

In response to a prototype, however, we may wish to conclude something stronger than a constraint on that *particular* POS sequence. We might hope that sequences which are similar to a prototype in some sense are generally given the same label as that prototype. For example, DT NN is a noun phrase prototype, the sequence DT JJ NN is another good candidate for being a noun phrase. This kind of propagation of constraints requires that we have a good way of defining and detecting similarity between POS sequences.

3.2 Phrasal Distributional Similarity

A central linguistic argument for constituent types is *substitutability*: phrases of the same type appear

time. We note that we still evaluate against all phrase types regardless of whether or not they are modeled by our grammar.

⁵Even this property is likely too strong: prototypes may have multiple possible labels, for example DT NN may also be a QP in the English treebank.

Yield	Prototype	Skew KL	Phrase Type	Skew KL
DT JJ NN	DT NN	0.10	NP	0.39
IN DT VBG NN	IN NN	0.24	PP	0.45
DT NN MD VB DT NNS	PRP VBD DT NN	0.54	S	0.58
CC NN	IN NN	0.43	PP	0.71
MD NNS	PRP VBD DT NN	1.43	NONE	-

Figure 2: Yields along with most similar prototypes and phrase types, guessed according to (3).

in similar contexts and are mutually substitutable (Harris, 1954; Radford, 1988). For instance, DT JJ NN and DT NN occur in similar contexts, and are indeed both common NPs. This idea has been repeatedly and successfully operationalized using various kinds of *distributional clustering*, where we define a similarity measure between two items on the basis of their immediate left and right contexts (Schütze, 1995; Clark, 2000; Klein and Manning, 2002).

As in Clark (2001), we characterize the distribution of a sequence by the distribution of POS tags occurring to the left and right of that sequence in a corpus. Each occurrence of a POS sequence α falls in a context $x \alpha y$, where x and y are the adjacent tags. The distribution over contexts $x - y$ for a given α is called its *signature*, and is denoted by $\sigma(\alpha)$. Note that $\sigma(\alpha)$ is composed of context counts from all occurrences, constituent and disjunct, of α . Let $\sigma_c(\alpha)$ denote the context distribution for α where the context counts are taken only from constituent occurrences of α . For each phrase type in our grammar, X , define $\sigma_c(X)$ to be the context distribution obtained from the counts of all constituent occurrences of type X :

$$\sigma_c(X) = \mathbb{E}_{p(\alpha|X)} \sigma_c(\alpha) \quad (1)$$

where $p(\alpha|X)$ is the distribution of yield types for phrase type X . We compare context distributions using the skewed KL divergence:

$$D_{SKL}(p, q) = D_{KL}(p \| \gamma p + (1 - \gamma)q)$$

where γ controls how much of the source distributions is mixed in with the target distribution.

A reasonable baseline rule for classifying the phrase type of a POS yield is to assign it to the phrase from which it has minimal divergence:

$$type(\alpha) = \arg \min_X D_{SKL}(\sigma_c(\alpha), \sigma_c(X)) \quad (2)$$

However, this rule is not always accurate, and, moreover, we do not have access to $\sigma_c(\alpha)$ or $\sigma_c(X)$. We chose to approximate $\sigma_c(X)$ using the prototype yields for X as samples from

$p(\alpha|X)$. Letting $proto(X)$ denote the (few) prototype yields for phrase type X , we define $\tilde{\sigma}(X)$:

$$\tilde{\sigma}(X) = \frac{1}{|proto(X)|} \sum_{\alpha \in proto(X)} \sigma(\alpha)$$

Note $\tilde{\sigma}(X)$ is an approximation to (1) in several ways. We have replaced an expectation over $p(\alpha|X)$ with a uniform weighting of $proto(X)$, and we have replaced $\sigma_c(\alpha)$ with $\sigma(\alpha)$ for each term in that expectation. Because of this, we will rely only on high confidence guesses, and allow yields to be given a NONE type if their divergence from each $\tilde{\sigma}(X)$ exceeds a fixed threshold t . This gives the following alternative to (2):

$$type(\alpha) = \begin{cases} \text{NONE, if } \min_X D_{SKL}(\sigma(\alpha), \tilde{\sigma}(X)) < t \\ \arg \min_X D_{SKL}(\sigma(\alpha), \tilde{\sigma}(X)), \text{ otherwise} \end{cases} \quad (3)$$

We built a distributional model implementing the rule in (3) by constructing $\sigma(\alpha)$ from context counts in the WSJ portion of the Penn Treebank as well as the BLIPP corpus. Each $\tilde{\sigma}(X)$ was approximated by a uniform mixture of $\sigma(\alpha)$ for each of X 's prototypes α listed in figure 1.

This method of classifying constituents is very precise if the threshold is chosen conservatively enough. For instance, using a threshold of $t = 0.75$ and $\gamma = 0.1$, this rule correctly classifies the majority label of a constituent-type with 83% precision, and has a recall of 23% over constituent types. Figure 2 illustrates some sample yields, the prototype sequence to which it is least divergent, and the output of rule (3).

We incorporated this distributional information into our PCFG induction scheme by adding a *prototype feature* over each span (i, j) indicating the output of (3) for the yield α in that span. Associated with each sentence S is a feature map F specifying, for each (i, j) , a prototype feature p_{ij} . These features are generated using an augmented CFG model, CFG₊, given by:⁶

$$\begin{aligned} P_{CFG_+}(T, F) &= \prod_{X(i,j) \rightarrow \alpha \in T} P(p_{ij}|X) P(\alpha|X) \\ &= \prod_{X(i,j) \rightarrow \alpha \in T} \phi_{CFG_+}(X \rightarrow \alpha, p_{ij}) \end{aligned}$$

⁶Technically, all features in F must be generated for each assignment to T , which means that there should be terms in this equation for the prototype features on disjunct spans. However, we fixed the prototype distribution to be uniform for disjunct spans so that the equation is correct up to a constant depending on F .

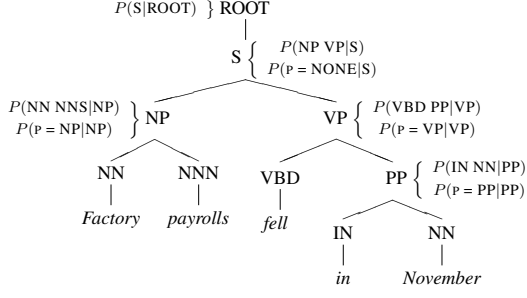


Figure 3: Illustration of PCFG augmented with prototype similarity features.

where $\phi_{CFG+}(X \rightarrow \alpha, p_{ij})$ is the local factor for placing $X \rightarrow \alpha$ on a span with prototype feature p_{ij} . An example is given in figure 3.

For our experiments, we fixed $P(p_{ij}|X)$ to be:

$$P(p_{ij}|X) = \begin{cases} 0.60, & \text{if } p_{ij} = X \\ \text{uniform,} & \text{otherwise} \end{cases}$$

Modifying the model in this way, and keeping the gold bracketing information, gave 71.1 labeled F_1 (see experiment PROTO $\times$ GOLD in figure 4), a 40.3% error reduction over naive PCFG induction in the presence of gold bracketing information. We note that the our labeled F_1 is upper-bounded by 86.0 due to unary chains and more-than-binary configurations in the treebank that cannot be obtained from our binary grammar.

We conclude that in the presence of gold bracket information, we can achieve high labeled accuracy by using a CFG augmented with distributional prototype features.

4 Constituent Context Model

So far, we have shown that, given perfect perfect bracketing information, distributional prototype features allow us to learn tree structures with fairly accurate labels. However, such bracketing information is not available in the unsupervised case.

Perhaps we don't actually need bracketing constraints in the presence of prototypes and distributional similarity features. However this experiment, labeled PROTO $\times$ NONE in figure 4, gave only 53.1 labeled F_1 (61.1 unlabeled), suggesting that some amount of bracketing constraint is necessary to achieve high performance.

Fortunately, there are unsupervised systems which can induce unlabeled bracketings with reasonably high accuracy. One such model is

the constituent-context model (CCM) of Klein and Manning (2002), a generative distributional model. For a given sentence S , the CCM generates a bracket matrix, B , which for each span (i, j) , indicates whether or not it is a constituent ($B_{ij} = c$) or a distituent ($B_{ij} = d$). In addition, it generates a feature map F' , which for each span (i, j) in S specifies a pair of features, $F'_{ij} = (y_{ij}, c_{ij})$, where y_{ij} is the POS yield of the span, and c_{ij} is the context of the span, i.e identity of the conjoined left and right POS tags:

$$P_{CCM}(B, F') = P(B) \prod_{(i,j)} P(y_{ij}|B_{ij}) P(c_{ij}|B_{ij})$$

The distribution $P(B)$ only places mass on bracketings which correspond to binary trees. We can efficiently compute $P_{CCM}(B, F')$ (up to a constant) depending on F' using local factors $\phi_{CCM}(y_{ij}, c_{ij})$ which decomposes over constituent spans:⁷

$$\begin{aligned} P_{CCM}(B, F') &\propto \prod_{(i,j):B_{ij}=c} \frac{P(y_{ij}|c)P(c_{ij}|c)}{P(y_{ij}|d)P(c_{ij}|d)} \\ &= \prod_{(i,j):B_{ij}=c} \phi_{CCM}(y_{ij}, c_{ij}) \end{aligned}$$

The CCM by itself yields an unlabeled F_1 of 71.9 on WSJ-10, which is reasonably high, but does not produce labeled trees.

5 Intersecting CCM and PCFG

The CCM and PCFG models provide complementary views of syntactic structure. The CCM explicitly learns the non-recursive contextual and yield properties of constituents and distituents. The PCFG model, on the other hand, does not explicitly model properties of distituents but instead focuses on modeling the hierarchical and recursive properties of natural language syntax. One would hope that modeling both of these aspects simultaneously would improve the overall quality of our induced grammar.

We therefore combine the CCM with our feature-augmented PCFG, denoted by PROTO in experiment names. When we run EM on either of the models alone, at each iteration and for each training example, we calculate posteriors over that

⁷Klein (2005) gives a full presentation.

model’s latent variables. For CCM, the latent variable is a bracketing matrix B (equivalent to an unlabeled binary tree), while for the CFG_+ the latent variable is a labeled tree T . While these latent variables aren’t exactly the same, there is a close relationship between them. A bracketing matrix constrains possible labeled trees, and a given labeled tree determines a bracketing matrix. One way to combine these models is to encourage both models to prefer latent variables which are compatible with each other.

Similar to the approach of Klein and Manning (2004) on a different model pair, we intersect CCM and CFG_+ by multiplying their scores for any labeled tree. For each possible labeled tree over a sentence S , our generative model for a labeled tree T is given as follows:

$$P(T, F, F') = P_{\text{CFG}_+}(T, F) P_{\text{CCM}}(B(T), F') \quad (4)$$

where $B(T)$ corresponds to the bracketing matrix determined by T . The EM algorithm for the product model will maximize:

$$\begin{aligned} P(S, F, F') &= \sum_{T \in \mathcal{T}(S)} P_{\text{CCM}}(B, F') P_{\text{CFG}_+}(T, F) \\ &= \sum_B P_{\text{CCM}}(B, F') \sum_{T \in \mathcal{T}(B, S)} P_{\text{CFG}_+}(T, F) \end{aligned}$$

where $\mathcal{T}(S)$ is the set of labeled trees consistent with the sentence S and $\mathcal{T}(B, S)$ is the set of labeled trees consistent with the bracketing matrix B and the sentence S . Notice that this quantity increases as the CCM and CFG_+ models place probability mass on compatible latent structures, giving an intuitive justification for the success of this approach.

We can compute posterior expectations over (B, T) in the combined model (4) using a variant of the inside-outside algorithm. The local factor for a binary rule $r = X \rightarrow YZ$, over span (i, j) , with CCM features $F'_{ij} = (y_{ij}, c_{ij})$ and prototype feature p_{ij} , is given by the product of local factors for the CCM and CFG_+ models:

$$\phi(r, (i, j)) = \phi_{\text{CCM}}(y_{ij}, c_{ij}) \phi_{\text{CFG}_+}(r, p_{ij})$$

From these local factors, the inside-outside algorithm produces expected counts for each binary rule, r , over each span (i, j) and split point k , denoted by $P(r, (i, j), k | S, F, F')$. These posteriors are sufficient to re-estimate all of our model parameters.

Setting	Labeled			Unlabeled		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1
No Brackets						
PCFG $\times$ NONE	23.9	29.1	26.3	40.7	52.1	45.7
PROTO $\times$ NONE	51.8	62.9	56.8	59.6	76.2	66.9
Gold Brackets						
PCFG $\times$ GOLD	47.0	57.2	51.6	78.8	100.0	88.1
PROTO $\times$ GOLD	64.8	78.7	71.1	78.8	100.0	88.1
CCM Brackets						
CCM	-	-	-	64.2	81.6	71.9
PCFG $\times$ CCM	32.3	38.9	35.3	64.1	81.4	71.8
PROTO $\times$ CCM	56.9	68.5	62.2	68.4	86.9	76.5
BEST	59.4	72.1	65.1	69.7	89.1	78.2
UBOUND	78.8	94.7	86.0	78.8	100.0	88.1

Figure 4: English grammar induction results. The upper bound on labeled recall is due to unary chains.

6 CCM as a Bracketer

We tested the product model described in section 5 on WSJ-10 under the same conditions as in section 3. Our initial experiment utilizes no prototype information, random initialization, and greedy remapping of its labels. This experiment, PCFG $\times$ CCM in figure 4, gave 35.3 labeled F_1 , compared to the 51.6 labeled F_1 with gold bracketing information (PCFG $\times$ GOLD in figure 4).

Next we added the manually specified prototypes in figure 1, and constrained the model to give these yields their labels if chosen as constituents. This experiment gave 48.9 labeled F_1 (73.3 unlabeled). The error reduction is 21.0% labeled (5.3% unlabeled) over PCFG $\times$ CCM.

We then experimented with adding distributional prototype features as discussed in section 3.2 using a threshold of 0.75 and $\gamma = 0.1$. This experiment, PROTO $\times$ CCM in figure 4, gave 62.2 labeled F_1 (76.5 unlabeled). The error reduction is 26.0% labeled (12.0% unlabeled) over the experiment using prototypes without the similarity features. The overall error reduction from PCFG $\times$ CCM is 41.6% (16.7%) in labeled (unlabeled) F_1 .

7 Error Analysis

The most common type of error by our PROTO $\times$ CCM system was due to the binary grammar restriction. For instance common NPs, such as DT JJ NN, analyzed as $[\text{NP DT } [\text{NP JJ NN}]]$, which proposes additional $\bar{\text{N}}$ constituents compared to the flatter treebank analysis. This discrepancy greatly, and perhaps unfairly, damages NP precision (see figure 6). However, this error is unavoidable

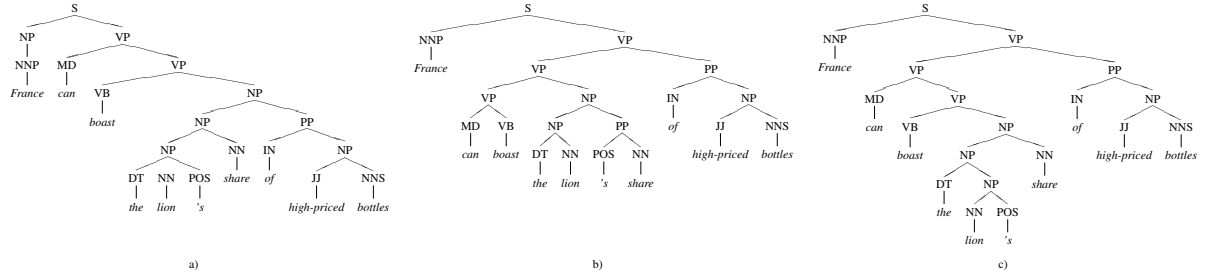


Figure 5: Examples of corrections from adding VP-INF and NP-POS prototype categories. The tree in (a) is the Treebank parse, (b) is the parse with $\text{PROTO} \times \text{CCM}$ model, and (c) is the parse with the BEST model (added prototype categories), which fixes the possessive NP and infinitival VP problems, but not the PP attachment.

given our grammar restriction.

Figure 5(b) demonstrates three other errors. Possessive NPs are analyzed as $[_{NP} \text{ NN } [_{PP} \text{ POS NN }]]$, with the POS element treated as a preposition and the possessed NP as its complement. While labeling the POS NN as a PP is clearly incorrect, placing a constituent over these elements is not unreasonable and in fact has been proposed by some linguists (Abney, 1987). Another type of error also reported by Klein and Manning (2002) is MD VB groupings in infinitival VPs also sometimes argued by linguists (Halliday, 2004). More seriously, prepositional phrases are almost always attached “high” to the verb for longer NPs.

7.1 Augmenting Prototypes

One of the advantages of the prototype driven approach, over a fully unsupervised approach, is the ability to refine or add to the annotation specification if we are not happy with the output of our system. We demonstrate this flexibility by augmenting the prototypes in figure 1 with two new categories NP-POS and VP-INF, meant to model possessive noun phrases and infinitival verb phrases, which tend to have slightly different distributional properties from normal NPs and VPs. These new sub-categories are used during training and then stripped in post-processing. This prototype list gave 65.1 labeled F_1 (78.2 unlabeled). This experiment is labeled BEST in figure 4. Looking at the CFG-learned rules in figure 7, we see that the basic structure of the treebank grammar is captured.

7.2 Parsing with only the PCFG

In order to judge how well the PCFG component of our model did in isolation, we experimented with training our BEST model with the CCM component, but dropping it at test time. This experi-

Label	Prec.	Rec.	F_1
S	79.3	80.0	79.7
NP	49.0	74.4	59.1
VP	80.4	73.3	76.7
PP	45.6	78.6	57.8
QP	36.2	78.8	49.6
ADJP	29.4	33.3	31.2
ADVP	25.0	12.2	16.4

Figure 6: Precision, recall, and F_1 for individual phrase types in the BEST model

Rule	Probability	Rule	Probability
$S \rightarrow NP VP$	0.51	$VP \rightarrow VBZ NP$	0.20
$S \rightarrow PRP VP$	0.13	$VP \rightarrow VBD NP$	0.15
$S \rightarrow NNP VP$	0.06	$VP \rightarrow VBP NP$	0.09
$S \rightarrow NNS VP$	0.05	$VP \rightarrow VB NP$	0.08
$NP \rightarrow DT NN$	0.12	$ROOT \rightarrow S$	0.95
$NP \rightarrow NP PP$	0.09	$ROOT \rightarrow NP$	0.05
$NP \rightarrow NNP NNP$	0.09		
$NP \rightarrow JJ NN$	0.07	$QP \rightarrow CD CD$	0.35
$PP \rightarrow IN NP$	0.37	$QP \rightarrow CD NN$	0.30
$PP \rightarrow CC NP$	0.06	$QP \rightarrow QP PP$	0.10
$PP \rightarrow TO VP$	0.05	$QP \rightarrow QP NNS$	0.05
$PP \rightarrow TO QP$	0.04		
$ADJP \rightarrow RB VBN$	0.37	$ADVP \rightarrow RB RB$	0.25
$ADJP \rightarrow RB JJ$	0.31	$ADVP \rightarrow ADJP PRP$	0.15
$ADJP \rightarrow RBR JJ$	0.09	$ADVP \rightarrow RB CD$	0.10

Figure 7: Top PCFG Rules learned by BEST model

ment gave 65.1 labeled F_1 (76.8 unlabeled). This demonstrates that while our PCFG performance degrades without the CCM, it can be used on its own with reasonable accuracy.

7.3 Automatically Generated Prototypes

There are two types of bias which enter into the creation of prototypes lists. One of them is the bias to choose examples which reflect the annotation semantics we wish our model to have. The second is the iterative change of prototypes in order to maximize F_1 . Whereas the first is appro-

appropriate, indeed the point, the latter is not. In order to guard against the second type of bias, we experimented with automatically extracted generated prototype lists which would not be possible without labeled data. For each phrase type category, we extracted the three most common yield associated with that category that differed in either first or last POS tag. Repeating our $\text{PROTO} \times \text{CCM}$ experiment with this list yielded 60.9 labeled F_1 (76.5 unlabeled), comparable to the performance of our manual prototype list.

7.4 Chinese Grammar Induction

In order to demonstrate that our system is somewhat language independent, we tested our model on CTB-10, the 2,437 sentences of the Chinese Treebank (Ircs, 2002) of length at most 10 after punctuation is stripped. Since the authors have no expertise in Chinese, we automatically extracted prototypes in the same way described in section 7.3. Since we did not have access to a large auxiliary POS tagged Chinese corpus, our distributional model was built only from the treebank text, and the distributional similarities are presumably degraded relative to the English. Our PCFG $\times$ CCM experiment gave 18.0 labeled F_1 (43.4 unlabeled). The $\text{PROTO} \times \text{CCM}$ model gave 39.0 labeled F_1 (53.2 unlabeled). Presumably with access to more POS tagged data, and the expertise of a Chinese speaker, our system would see increased performance. It is worth noting that our unlabeled F_1 of 53.2 is the best reported from a primarily unsupervised system, with the next highest figure being 46.7 reported by Klein and Manning (2004).

8 Conclusion

We have shown that distributional prototype features can allow one to specify a target labeling scheme in a compact and declarative way. These features give substantial error reduction in labeled F_1 measure for English and Chinese grammar induction. They also achieve the best reported unlabeled F_1 measure.⁸ Another positive property of this approach is that it tries to reconcile the success of distributional clustering approaches to grammar induction (Clark, 2001; Klein and Manning, 2002), with the CFG tree models in the supervised literature (Collins, 1999). Most importantly, this is the first work, to the authors' knowl-

edge, which has learned CFGs in an unsupervised or semi-supervised setting and can parse natural language text with any reasonable accuracy.

Acknowledgments We would like to thank the anonymous reviewers for their comments. This work is supported by a Microsoft / CITRIS grant and by an equipment donation from Intel.

References

- Stephen P. Abney. 1987. *The English Noun Phrase in its Sentential Aspect*. Ph.D. thesis, MIT.
- Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. Technical Report CS-92-16.
- Alexander Clark. 2000. Inducing syntactic categories by context distribution clustering. In *CoNLL*, pages 91–94, Lisbon, Portugal.
- Alexander Clark. 2001. The unsupervised induction of stochastic context-free grammars using distributional clustering. In *CoNLL*.
- Michael Collins. 1999. *The Unsupervised learning of Natural Language Structure*. Ph.D. thesis, University of Rochester.
- M.A.K Halliday. 2004. An introduction to functional grammar. Edward Arnold, 2nd edition.
- Zellig Harris. 1954. *Distributional Structure*. University of Chicago Press, Chicago.
- Nianwen Xue Ircs. 2002. Building a large-scale annotated chinese corpus.
- Dan Klein and Christopher Manning. 2002. A generative constituent-context model for improved grammar induction. In *ACL*.
- Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *ACL*.
- Dan Klein. 2005. *The unsupervised learning of Natural Language Structure*. Ph.D. thesis, Stanford University.
- Karim Lari and Steve Young. 1990. The estimation of stochastic context-free grammars using the inside/outside algorithm. *Computer Speech and Language*, 2(4):35–56.
- Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.
- Fernando C. N. Pereira and Yves Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Meeting of the Association for Computational Linguistics*, pages 128–135.
- Andrew Radford. 1988. *Transformational Grammar*. Cambridge University Press, Cambridge.
- Hinrich Schütze. 1995. Distributional part-of-speech tagging. In *EACL*.
- Noah A. Smith and Jason Eisner. 2004. Guiding unsupervised grammar induction using contrastive estimation. In *Working notes of the IJCAI workshop on Grammatical Inference Applications*.

⁸The next highest results being 77.1 and 46.7 for English and Chinese respectively from Klein and Manning (2004).

Exploring Correlation of Dependency Relation Paths for Answer Extraction

Dan Shen

Department of Computational Linguistics
Saarland University
Saarbruecken, Germany
dshen@coli.uni-sb.de

Dietrich Klakow

Spoken Language Systems
Saarland University
Saarbruecken, Germany
klakow@lsv.uni-saarland.de

Abstract

In this paper, we explore correlation of dependency relation paths to rank candidate answers in answer extraction. Using the correlation measure, we compare dependency relations of a candidate answer and mapped question phrases in sentence with the corresponding relations in question. Different from previous studies, we propose an approximate phrase mapping algorithm and incorporate the mapping score into the correlation measure. The correlations are further incorporated into a Maximum Entropy-based ranking model which estimates path weights from training. Experimental results show that our method significantly outperforms state-of-the-art syntactic relation-based methods by up to 20% in MRR.

1 Introduction

Answer Extraction is one of basic modules in open domain Question Answering (QA). It is to further process relevant sentences extracted with Passage / Sentence Retrieval and pinpoint exact answers using more linguistic-motivated analysis. Since QA turns to find exact answers rather than text snippets in recent years, answer extraction becomes more and more crucial.

Typically, answer extraction works in the following steps:

- Recognize expected answer type of a question.
- Annotate relevant sentences with various types of named entities.

- Regard the phrases annotated with the expected answer type as candidate answers.
- Rank candidate answers.

In the above work flow, answer extraction heavily relies on named entity recognition (NER). On one hand, NER reduces the number of candidate answers and eases answer ranking. On the other hand, the errors from NER directly degrade answer extraction performance. To our knowledge, most top ranked QA systems in TREC are supported by effective NER modules which may identify and classify more than 20 types of named entities (NE), such as *abbreviation*, *music*, *movie*, etc. However, developing such named entity recognizer is not trivial. Up to now, we haven't found any paper relevant to QA-specific NER development. So, it is hard to follow their work. In this paper, we just use a general MUC-based NER, which makes our results reproducible.

A general MUC-based NER can't annotate a large number of NE classes. In this case, all noun phrases in sentences are regarded as candidate answers, which makes candidate answer sets much larger than those filtered by a well developed NER. The larger candidate answer sets result in the more difficult answer extraction. Previous methods working on surface word level, such as density-based ranking and pattern matching, may not perform well. Deeper linguistic analysis has to be conducted. This paper proposes a statistical method which exploring correlation of dependency relation paths to rank candidate answers. It is motivated by the observation that relations between proper answers and question phrases in candidate sentences are always similar to the corresponding relations in question. For example, the question "What did Alfred Nobel invent?" and the

candidate sentence "... in the will of Swedish industrialist Alfred Nobel, who invented *dynamite*."

For each question, firstly, dependency relation paths are defined and extracted from the question and each of its candidate sentences. Secondly, the paths from the question and the candidate sentence are paired according to question phrase mapping score. Thirdly, correlation between two paths of each pair is calculated by employing Dynamic Time Warping algorithm. The input of the calculation is correlations between dependency relations, which are estimated from a set of training path pairs. Lastly, a Maximum Entropy-based ranking model is proposed to incorporate the path correlations and rank candidate answers. Furthermore, sentence supportive measure are presented according to correlations of relation paths among question phrases. It is applied to re-rank the candidate answers extracted from the different candidate sentences. Considering phrases may provide more accurate information than individual words, we extract dependency relations on phrase level instead of word level.

The experiment on TREC questions shows that our method significantly outperforms a density-based method by 50% in MRR and three state-of-the-art syntactic-based methods by up to 20% in MRR. Furthermore, we classify questions by judging whether NER is used. We investigate how these methods perform on the two question sets. The results indicate that our method achieves better performance than the other syntactic-based methods on both question sets. Especially for more difficult questions, for which NER may not help, our method improves MRR by up to 31%.

The paper is organized as follows. Section 2 discusses related work and clarifies what is new in this paper. Section 3 presents relation path correlation in detail. Section 4 and 5 discuss how to incorporate the correlations for answer ranking and re-ranking. Section 6 reports experiment and results.

2 Related Work

In recent years' TREC Evaluation, most top ranked QA systems use syntactic information in answer extraction. Next, we will briefly discuss the main usages.

(Kaisser and Becker, 2004) match a question into one of predefined patterns, such as "*When did Jack Welch retire from GE?*" to the pattern

"*When+did+NP+Verb+NPorPP*". For each question pattern, there is a set of syntactic structures for potential answer. Candidate answers are ranked by matching the syntactic structures. This method worked well on TREC questions. However, it is costing to manually construct question patterns and syntactic structures of the patterns.

(Shen et al., 2005) classify question words into four classes *target word*, *head word*, *subject word* and *verb*. For each class, syntactic relation patterns which contain one question word and one proper answer are automatically extracted and scored from training sentences. Then, candidate answers are ranked by partial matching to the syntactic relation patterns using tree kernel. However, the criterion to classify the question words is not clear in their paper. Proper answers may have absolutely different relations with different *subject words* in sentences. They don't consider the corresponding relations in questions.

(Tanev et al., 2004; Wu et al., 2005) compare syntactic relations in questions and those in answer sentences. (Tanev et al., 2004) reconstruct a basic syntactic template tree for a question, in which one of the nodes denotes expected answer position. Then, answer candidates for this question are ranked by matching sentence syntactic tree to the question template tree. Furthermore, the matching is weighted by lexical variations. (Wu et al., 2005) combine n-gram proximity search and syntactic relation matching. For syntactic relation matching, question tree and sentence subtree around a candidate answer are matched from node to node.

Although the above systems apply the different methods to compare relations in question and answer sentences, they follow the same hypothesis that proper answers are more likely to have same relations in question and answer sentences. For example, in question "*Who founded the Black Panthers organization?*", where, the question word "*who*" has the dependency relations "*subj*" with "*found*" and "*subj obj nn*" with "*Black Panthers organization*", in sentence "*Hilliard introduced Bobby Seale, who co-founded the Black Panther Party here ...*", the proper answer "*Bobby Seale*" has the same relations with most question phrases. These methods achieve high precision, but poor recall due to relation variations. One meaning is often represented as different relation combinations. In the above example, appositive rela-

tion frequently appears in answer sentences, such as "*Black Panther Party co-founder **Bobby Seale** is ordered bound and gagged ...*" and indicates proper answer *Bobby Seale* although it is asked in different way in the question.

(Cui et al., 2004) propose an approximate dependency relation matching method for both passage retrieval and answer extraction. The similarity between two relations is measured by their co-occurrence rather than exact matching. They state that their method effectively overcomes the limitation of the previous exact matching methods. Lastly, they use the sum of similarities of all path pairs to rank candidate answers, which is based on the assumption that all paths have equal weights. However, it might not be true. For example, in question "*What book did Rachel Carson write in 1962?*", the phrase "*Rachel Carson*" looks like more important than "*1962*" since the former is question topic and the latter is a constraint for expected answer. In addition, lexical variations are not well considered and a weak relation path alignment algorithm is used in their work.

Based on the previous works, this paper explores correlation of dependency relation paths between questions and candidate sentences. Dynamic time warping algorithm is adapted to calculate path correlations and approximate phrase mapping is proposed to cope with phrase variations. Finally, maximum entropy-based ranking model is developed to incorporate the correlations and rank candidate answers.

3 Dependency Relation Path Correlation

In this section, we discuss how the method performs in detail.

3.1 Dependency Relation Path Extraction

We parse questions and candidate sentences with MiniPar (Lin, 1994), a fast and robust parser for grammatical dependency relations. Then, we extract relation paths from dependency trees.

Dependency relation path is defined as a structure $P = \langle N_1, R, N_2 \rangle$ where, N_1, N_2 are two phrases and R is a relation sequence $R = \langle r_1, \dots, r_i \rangle$ in which r_i is one of the predefined dependency relations. Totally, there are 42 relations defined in MiniPar. A relation sequence R between two phrases N_1, N_2 is extracted by traversing from the N_1 node to the N_2 node in a dependency tree.

Q: What book did Rachel Carson write in 1962?		
Paths for Answer Ranking		
N1 (EAP)	R	N2
What	det	book
What	det obj subj	Rachel Carson
What	det obj	write
What	det obj mod pcomp-n	1962
Paths for Answer Re-ranking		
book	obj subj	Rachel Carson
book	obj	write
book	obj mod pcomp-n	1962
...		

S: Rachel Carson 's 1962 book " Silent Spring " said dieldrin causes mania.		
Paths for Answer Ranking		
N1 (CA)	R	N2
Silent Spring	title	book
Silent Spring	title gen	Rachel Carson
Silent Spring	title num	1962
Paths for Answer Re-ranking		
book	gen	Rachel Carson
book	num	1962
...		

Figure 1: Relation Paths for sample question and sentence. *EAP* indicates expected answer position; *CA* indicates candidate answer

For each question, we extract relation paths among noun phrases, main verb and question word. The question word is further replaced with "*EAP*", which indicates the expected answer position. For each candidate sentence, we firstly extract relation paths between answer candidates and mapped question phrases. These paths will be used for answer ranking (Section 4). Secondly, we extract relation paths among mapped question phrases. These paths will be used for answer re-ranking (Section 5). Question phrase mapping will be discussed in Section 3.4. Figure 1 shows some relation paths extracted for an example question and candidate sentence.

Next, the relation paths in a question and each of its candidate sentences are paired according to their phrase similarity. For any two relation path P_i and P_j which are extracted from the question and the candidate sentence respectively, if $Sim(N_{i1}, N_{j1}) > 0$ and $Sim(N_{i2}, N_{j2}) > 0$, P_i and P_j are paired as $\langle P_i, P_j \rangle$. The question phrase "*EAP*" is mapped to candidate answer phrase in the sentence. The similarity between two

Path Pairs for Answer Ranking			
N1 (EAP / CA)	Rq	Rs	N2
Silent Spring	det	title	book
Silent Spring	det obj subj	title gen	Rachel Carson
Silent Spring	det obj mod pcomp-n	title num	1962
Path Pairs for Answer Re-ranking			
N1	Rq	Rs	N2
book	obj subj	gen	Rachel Carson
book	obj mod pcomp-n	num	1962
...			

Figure 2: Paired Relation Path

phrases will be discussed in Section 3.4. Figure 2 further shows the paired relation paths which are presented in Figure 1.

3.2 Dependency Relation Path Correlation

Comparing a proper answer and other wrong candidate answers in each sentence, we assume that relation paths between the proper answer and question phrases in the sentence are more correlated to the corresponding paths in question. So, for each path pair $\langle P_1, P_2 \rangle$, we measure the correlation between its two paths P_1 and P_2 .

We derive the correlations between paths by adapting dynamic time warping (DTW) algorithm (Rabiner et al., 1978). DTW is to find an optimal alignment between two sequences which maximizes the accumulated correlation between two sequences. A sketch of the adapted algorithm is as follows.

Let $R_1 = \langle r_{11}, \dots, r_{1n} \rangle, (n = 1, \dots, N)$ and $R_2 = \langle r_{21}, \dots, r_{2m} \rangle, (m = 1, \dots, M)$ denote two relation sequences. R_1 and R_2 consist of N and M relations respectively. $R_1(n) = r_{1n}$ and $R_2(m) = r_{2m}$. $Cor(r_1, r_2)$ denotes the correlation between two individual relations r_1, r_2 , which is estimated by a statistical model during training (Section 3.3). Given the correlations $Cor(r_{1n}, r_{2m})$ for each pair of relations (r_{1n}, r_{2m}) within R_1 and R_2 , the goal of DTW is to find a path, $m = map(n)$, which map n onto the corresponding m such that the accumulated correlation Cor^* along the path is maximized.

$$Cor^* = \max_{map(n)} \left\{ \sum_{n=1}^N Cor(R_1(n), R_2(map(n))) \right\}$$

A dynamic programming method is used to determine the optimum path $map(n)$. The accumulated correlation Cor_A to any grid point (n, m) can be recursively calculated as

$$Cor_A(n, m) = Cor(r_{1n}, r_{2m}) + \max_{q \leq m} Cor_A(n-1, q)$$

$$Cor^* = Cor_A(N, M)$$

The overall correlation measure has to be normalized as longer sequences normally give higher correlation value. So, the correlation between two sequences R_1 and R_2 is calculated as

$$Cor(R_1, R_2) = Cor^* / \max(N, M)$$

Finally, we define the correlation between two relation paths P_1 and P_2 as

$$Cor(P_1, P_2) = Cor(R_1, R_2) \times Sim(N_{11}, N_{21}) \times Sim(N_{12}, N_{22})$$

Where, $Sim(N_{11}, N_{21})$ and $Sim(N_{12}, N_{22})$ are the phrase mapping score when pairing two paths, which will be described in Section 3.4. If two phrases are absolutely different $Cor(N_{11}, N_{21}) = 0$ or $Cor(N_{12}, N_{22}) = 0$, the paths may not be paired since $Cor(P_1, P_2) = 0$.

3.3 Relation Correlation Estimation

In the above section, we have described how to measure path correlations. The measure requires relation correlations $Cor(r_1, r_2)$ as inputs. We apply a statistical method to estimate the relation correlations from a set of training path pairs. The training data collecting will be described in Section 6.1.

For each question and its answer sentences in training data, we extract relation paths between "EAP" and other phrases in the question and paths between proper answer and mapped question phrases in the sentences. After pairing the question paths and the corresponding sentence paths, correlation of two relations is measured by their bipartite co-occurrence in all training path pairs. Mutual information-based measure (Cui et al., 2004) is employed to calculate the relation correlations.

$$Cor(r_i^Q, r_j^S) = \log \frac{\sum \alpha \times \delta(r_i^Q, r_j^S)}{f_Q(r_i^Q) \times f_S(r_j^S)}$$

where, r_i^Q and r_j^S are two relations in question paths and sentence paths respectively. $f_Q(r_i^Q)$ and $f_S(r_j^S)$ are the numbers of occurrences of r_i^Q in question paths and r_j^S in sentence paths respectively. $\delta(r_i^Q, r_j^S)$ is 1 when r_i^Q and r_j^S co-occur in a path pair, and 0 otherwise. α is a factor to discount the co-occurrence value for long paths. It is set to the inverse proportion of the sum of path lengths of the path pair.

3.4 Approximate Question Phrase Mapping

Basic noun phrases (BNP) and verbs in questions are mapped to their candidate sentences. A BNP is defined as the smallest noun phrase in which there are no noun phrases embedded. To address lexical and format variations between phrases, we propose an approximate phrase mapping strategy.

A BNP is separated into a set of heads $H = \{h_1, \dots, h_i\}$ and a set of modifiers $M = \{m_1, \dots, m_j\}$. Some heuristic rules are applied to judge heads and modifiers: 1. If BNP is a named entity, all words are heads. 2. The last word of BNP is head. 3. Rest words are modifiers.

The similarity between two BNPs $Sim(BNP_q, BNP_s)$ is defined as:

$$Sim(BNP_q, BNP_s) = \lambda Sim(H_q, H_s) + (1 - \lambda) Sim(M_q, M_s)$$

$$Sim(H_q, H_s) = \frac{\sum_{h_i \in H_q} \sum_{h_j \in H_s} Sim(h_i, h_j)}{|H_q \cup H_s|}$$

$$Sim(M_q, M_s) = \frac{\sum_{m_i \in M_q} \sum_{m_j \in M_s} Sim(m_i, m_j)}{|M_q \cup M_s|}$$

Furthermore, the similarity between two heads $Sim(h_i, h_j)$ are defined as:

- $Sim = 1$, if $h_i = h_j$ after stemming;
- $Sim = 1$, if $h_i = h_j$ after format alternation;
- $Sim = SemSim(h_i, h_j)$

These items consider morphological, format and semantic variations respectively. 1. The morphological variations match words after stemming, such as "*Rhodes scholars*" and "*Rhodes scholarships*". 2. The format alternations cope with special characters, such as "-" for "*Ice-T*" and "*Ice T*", "&" for "*Abercrombie and Fitch*" and "*Abercrombie & Fitch*". 3. The semantic similarity $SemSim(h_i, h_j)$ is measured using WordNet and eXtended WordNet. We use the same semantic path finding algorithm, relation weights and semantic similarity measure as (Moldovan and Novischi, 2002). For efficiency, only hypernym, hyponym and entailment relations are considered and search depth is set to 2 in our experiments.

Particularly, the semantic variations are not considered for NE heads and modifiers. Modifier similarity $Sim(m_i, m_j)$ only consider the morphological and format variations. Moreover, verb similarity measure $Sim(v_1, v_2)$ is the same as head similarity measure $Sim(h_i, h_j)$.

4 Candidate Answer Ranking

According to path correlations of candidate answers, a Maximum Entropy (ME)-based model is applied to rank candidate answers. Unlike (Cui et al., 2004), who rank candidate answers with the sum of the path correlations, ME model may estimate the optimal weights of the paths based on a training data set. (Berger et al., 1996) gave a good description of ME model. The model we use is similar to (Shen et al., 2005; Ravichandran et al., 2003), which regard answer extraction as a ranking problem instead of a classification problem. We apply Generalized Iterative Scaling for model parameter estimation and Gaussian Prior for smoothing.

If expected answer type is unknown during question processing or corresponding type of named entities isn't recognized in candidate sentences, we regard all basic noun phrases as candidate answers. Since a MUC-based NER loses many types of named entities, we have to handle larger candidate answer sets. Orthographic features, similar to (Shen et al., 2005), are extracted to capture word format information of candidate answers, such as capitalizations, digits and lengths, etc. We expect they may help to judge what proper answers look like since most NER systems work on these features.

Next, we will discuss how to incorporate path correlations. Two facts are considered to affect path weights: **question phrase type** and **path length**. For each question, we divide question phrases into four types: target, topic, constraint and verb. **Target** is a kind of word which indicates the expected answer type of the question, such as "*party*" in "*What party led Australia from 1983 to 1996?*". **Topic** is the event/person that the question talks about, such as "*Australia*". Intuitively, it is the most important phrase of the question. **Constraint** are the other phrases of the question except topic, such as "*1983*" and "*1996*". **Verb** is the main verb of the question, such as "*lead*". Furthermore, since shorter path indicates closer relation between two phrases, we discount path correlation in long question path by dividing the correlation by the length of the question path. Lastly, we sum the discounted path correlations for each type of question phrases and fire it as a feature, such as "*Target_Cor=c*", where c is the correlation value for question *target*. ME-based ranking model incorporate the orthographic and path

correlation features to rank candidate answers for each of candidate sentences.

5 Candidate Answer Re-ranking

After ranking candidate answers, we select the highest ranked one from each candidate sentence. In this section, we are to re-rank them according to sentence supportive degree. We assume that a candidate sentence supports an answer if relations between mapped question phrases in the candidate sentence are similar to the corresponding ones in question. Relation paths between any two question phrases are extracted and paired. Then, correlation of each pair is calculated. Re-rank formula is defined as follows:

$$Score(answer) = \alpha \times \sum_i Cor(P_{i1}, P_{i2})$$

where, α is answer ranking score. It is the normalized prediction value of the ME-based ranking model described in Section 4. $\sum_i Cor(P_{i1}, P_{i2})$ is the sum of correlations of all path pairs. Finally, the answer with the highest score is returned.

6 Experiments

In this section, we set up experiments on TREC factoid questions and report evaluation results.

6.1 Experiment Setup

The goal of answer extraction is to identify exact answers from given candidate sentence collections for questions. The candidate sentences are regarded as the most relevant sentences to the questions and retrieved by IR techniques. Qualities of the candidate sentences have a strong impact on answer extraction. It is meaningless to evaluate the questions of which none candidate sentences contain proper answer in answer extraction experiment. To our knowledge, most of current QA systems lose about half of questions in sentence retrieval stage. To make more questions evaluated in our experiments, for each of questions, we automatically build a candidate sentence set from TREC judgements rather than use sentence retrieval output.

We use TREC99-03 questions for training and TREC04 questions for testing. As to build training data, we retrieve all of the sentences which contain proper answers from relevant documents according to TREC judgements and answer patterns.

Then, We manually check the sentences and remove those in which answers cannot be supported. As to build candidate sentence sets for testing, we retrieve all of the sentences from relevant documents in judgements and keep those which contain at least one question key word. Therefore, each question has at least one proper candidate sentence which contains proper answer in its candidate sentence set.

There are 230 factoid questions (27 NIL questions) in TREC04. NIL questions are excluded from our test set because TREC doesn't supply relevant documents and answer patterns for them. Therefore, we will evaluate 203 TREC04 questions. Five answer extraction methods are evaluated for comparison:

- **Density:** Density-based method is used as baseline, in which we choose candidate answer with the shortest surface distance to question phrases.
- **SynPattern:** Syntactic relation patterns (Shen et al., 2005) are automatically extracted from training set and are partially matched using tree kernel.
- **StrictMatch:** Strict relation matching follows the assumption in (Tanev et al., 2004; Wu et al., 2005). We implement it by adapting relation correlation score. In stead of learning relation correlations during training, we predefine them as: $Cor(r_1, r_2) = 1$ if $r_1 = r_2$; 0, otherwise.
- **ApprMatch:** Approximate relation matching (Cui et al., 2004) aligns two relation paths using fuzzy matching and ranks candidates according to the sum of all path similarities.
- **CorME:** It is the method proposed in this paper. Different from *ApprMatch*, ME-based ranking model is implemented to incorporate path correlations which assigns different weights for different paths respectively. Furthermore, phrase mapping score is incorporated into the path correlation measure.

These methods are briefly described in Section 2. Performance is evaluated with Mean Reciprocal Rank (MRR). Furthermore, we list percentages of questions correctly answered in terms of top 5 answers and top 1 answer returned respectively. No answer validations are used to adjust answers.

Table 1: Overall performance

	Density	SynPattern	StrictMatch	ApprMatch	CorME
MRR	0.45	0.56	0.57	0.60	0.67
Top1	0.36	0.53	0.49	0.53	0.62
Top5	0.56	0.60	0.67	0.70	0.74

6.2 Results

Table 1 shows the overall performance of the five methods. The main observations from the table are as follows:

1. The methods *SynPattern*, *StrictMatch*, *ApprMatch* and *CorME* significantly improve MRR by 25.0%, 26.8%, 34.5% and 50.1% over the baseline method *Density*. The improvements may benefit from the various explorations of syntactic relations.
2. The performance of *SynPattern* (0.56MRR) and *StrictMatch* (0.57MRR) are close. *SynPattern* matches relation sequences of candidate answers with the predefined relation sequences extracted from a training data set, while *StrictMatch* matches relation sequences of candidate answers with the corresponding relation sequences in questions. But, both of them are based on the assumption that the more number of same relations between two sequences, the more similar the sequences are. Furthermore, since most TREC04 questions only have one or two phrases and many questions have similar expressions, *SynPattern* and *StrictMatch* don't make essential difference.
3. *ApprMatch* and *CorME* outperform *SynPattern* and *StrictMatch* by about 6.1% and 18.4% improvement in MRR. Strict matching often fails due to various relation representations in syntactic trees. However, such variations of syntactic relations may be captured by *ApprMatch* and *CorME* using a MI-based statistical method.
4. *CorME* achieves the better performance by 11.6% than *ApprMatch*. The improvement may benefit from two aspects: 1) *ApprMatch* assigns equal weights to the paths of a candidate answer and question phrases, while *CorME* estimate the weights according to phrase type and path length. After training a

ME model, the weights are assigned, such as 5.72 for *topic* path ; 3.44 for *constraints* path and 1.76 for *target* path. 2) *CorME* incorporates approximate phrase mapping scores into path correlation measure.

We further divide the questions into two classes according to whether NER is used in answer extraction. If the expected answer type of a question is unknown, such as "How did James Dean die?" or the type cannot be annotated by NER, such as "What ethnic group/race are Crip members?", we put the question in *Qw/oNE* set, otherwise, we put it in *QwNE*. For the questions in *Qw/oNE*, we extract all basic noun phrases and verb phrases as candidate answers. Then, answer extraction module has to work on the larger candidate sets. Using a MUC-based NER, the recognized types include person, location, organization, date, time and money. In TREC04 questions, 123 questions are put in *QwNE* and 80 questions in *Qw/oNE*.

Table 2: Performance on two question sets *QwNE* and *Qw/oNE*

	QwNE	Qw/oNE
Density	0.66	0.11
SynPattern	0.71	0.36
StrictMatch	0.70	0.36
ApprMatch	0.72	0.42
CorME	0.79	0.47

We evaluate the performance on *QwNE* and *Qw/oNE* respectively, as shown in Table 2. The density-based method *Density* (0.11MRR) loses many questions in *Qw/oNE*, which indicates that using only surface word information is not sufficient for large candidate answer sets. On the contrary, *SynPattern*(0.36MRR), *StrictPattern*(0.36MRR), *ApprMatch*(0.42MRR) and *CorME* (0.47MRR) which capture syntactic information, perform much better than *Density*. Our method *CorME* outperforms the other syntactic-based methods on both *QwNE* and *Qw/oNE*. Es-

pecially for more difficult questions *Qw/oNE*, the improvements (up to 31% in MRR) are more obvious. It indicates that our method can be used to further enhance state-of-the-art QA systems even if they have a good NER.

In addition, we evaluate component contributions of our method based on the main idea of relation path correlation. Three components are tested: 1. **Appr. Mapping** (Section 3.4). We replace approximate question phrase mapping with exact phrase mapping and withdraw the phrase mapping scores from path correlation measure. 2. **Answer Ranking** (Section 4). Instead of using ME model, we sum all of the path correlations to rank candidate answers, which is similar to (Cui et al., 2004). 3. **Answer Re-ranking** (Section 5). We disable this component and select top 5 answers according to answer ranking scores.

Table 3: Component Contributions

	MRR
Overall	0.67
- Appr. Mapping	0.63
- Answer Ranking	0.62
- Answer Re-ranking	0.66

The contribution of each component is evaluated with the overall performance degradation after it is removed or replaced. Some findings are concluded from Table 3. Performances degrade when replacing approximate phrase mapping or ME-based answer ranking, which indicates that both of them have positive effects on the systems. This may be also used to explain why *CorME* outperforms *ApprMatch* in Table 1. However, removing answer re-ranking doesn't affect much. Since short questions, such as "What does AARP stand for?", frequently occur in TREC04, exploring the phrase relations for such questions isn't helpful.

7 Conclusion

In this paper, we propose a relation path correlation-based method to rank candidate answers in answer extraction. We extract and pair relation paths from questions and candidate sentences. Next, we measure the relation path correlation in each pair based on approximate phrase mapping score and relation sequence alignment, which is calculated by DTW algorithm. Lastly, a ME-based ranking model is proposed to incorporate the path correlations and rank candidate

answers. The experiment on TREC questions shows that our method significantly outperforms a density-based method by 50% in MRR and three state-of-the-art syntactic-based methods by up to 20% in MRR. Furthermore, the method is especially effective for difficult questions, for which NER may not help. Therefore, it may be used to further enhance state-of-the-art QA systems even if they have a good NER. In the future, we are to further evaluate the method based on the overall performance of a QA system and adapt it to sentence retrieval task.

References

- Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71.
- Hang Cui, Keya Li, Renxu Sun, Tat-Seng Chua, and Min-Yen Kan. 2004. National university of singapore at the trec-13 question answering. In *Proceedings of TREC2004, NIST*.
- M. Kaisser and T. Becker. 2004. Question answering by searching large corpora with linguistic methods. In *Proceedings of TREC2004, NIST*.
- Dekang Lin. 1994. Principar—an efficient, broad-coverage, principle-based parser. In *Proceedings of COLING1994*, pages 42–488.
- Dan Moldovan and Adrian Novischi. 2002. Lexical chains for question answering. In *Proceedings of COLING2002*.
- L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson. 1978. Considerations in dynamic time warping algorithms for discrete word recognition. In *Proceedings of IEEE Transactions on acoustics, speech and signal processing*.
- Deepak Ravichandran, Eduard Hovy, and Franz Josef Och. 2003. Statistical qa - classifier vs. re-ranker: What's the difference? In *Proceedings of ACL2003 workshop on Multilingual Summarization and Question Answering*.
- Dan Shen, Geert-Jan M. Kruijff, and Dietrich Klakow. 2005. Exploring syntactic relation patterns for question answering. In *Proceedings of IJCNLP2005*.
- H. Tanev, M. Kouylekov, and B. Magnini. 2004. Combining linguistic processing and web mining for question answering: Itc-1st at trec-2004. In *Proceedings of TREC2004, NIST*.
- M. Wu, M. Y. Duan, S. Shaikh, S. Small, and T. Strzalkowski. 2005. University at albany's ilqua in trec 2005. In *Proceedings of TREC2005, NIST*.

Question Answering with Lexical Chains Propagating Verb Arguments

Adrian Novischi

Dan Moldovan

Language Computer Corp.

1701 N. Collins Blvd, Richardson, TX, 75080

{adrian,moldovan}@languagecomputer.com

Abstract

This paper describes an algorithm for propagating verb arguments along lexical chains consisting of WordNet relations. The algorithm creates verb argument structures using VerbNet syntactic patterns. In order to increase the coverage, a larger set of verb senses were automatically associated with the existing patterns from VerbNet. The algorithm is used in an in-house Question Answering system for re-ranking the set of candidate answers. Tests on factoid questions from TREC 2004 indicate that the algorithm improved the system performance by 2.4%.

1 Introduction

In Question Answering the correct answer can be formulated with different but related words than the question. Connecting the words in the question with the words in the candidate answer is not enough to recognize the correct answer. For example the following question from TREC 2004 (Voorhees, 2004):

Q: (boxer Floyd Patterson) Who did he beat to win the title?

has the following wrong answer:

WA: He saw Ingemar Johanson knock down Floyd Patterson seven times there in winning the heavyweight title.

Although the above sentence contains the words *Floyd, Patterson, win, title*, and the verb *beat* can be connected to the verb *knock_down* using lexical chains from WordNet, this sentence does not answer the question because the verb arguments are in the wrong position. The proposed answer describes Floyd Patterson as being the object/patient

of the *beating* event while in the question he is the subject/agent of the similar event. Therefore the selection of the correct answer from a list of candidate answers requires the check of additional constraints including the match of verb arguments.

Previous approaches to answer ranking, used syntactic partial matching, syntactic and semantic relations and logic forms for selecting the correct answer from a set of candidate answers. Tanev et al. (Tanev et al., 2004) used an algorithm for partial matching of syntactic structures. For lexical variations they used a dependency based thesaurus of similar words (Lin, 1998). Hang et al. (Cui et al., 2004) used an algorithm to compute the similarity between dependency relation paths from a parse tree to rank the candidate answers. In TREC 2005, Ahn et al. (Ahn et al., 2005) used Discourse Representation Structures (DRS) resembling logic forms and semantic relations to represent questions and answers and then computed a score “indicating how well DRSs match each other”. Moldovan and Rus (Moldovan and Rus, 2001) transformed the question and the candidate answers into logic forms and used a logic prover to determine if the candidate answer logic form (ALF) entails the question logic form (QLF). Continuing this work Moldovan et al. (Moldovan et al., 2003) built a logic prover for Question Answering. The logic prover uses a relaxation module that is used iteratively if the proof fails at the price of decreasing the score of the proof. This logic prover was improved with temporal context detection (Moldovan et al., 2005).

All these approaches superficially addressed verb lexical variations. Similar meanings can be expressed using different verbs that use the same arguments in different positions. For example the sentence:

John bought a cowboy hat for \$50
can be reformulated as:

John paid \$50 for a cowboy hat.

The verb *buy* entails the verb *pay* however the arguments *a cowboy hat* and *\$50* have different position around the verb.

This paper describes the approach for propagating the arguments from one verb to another using lexical chains derived using WordNet (Miller, 1995). The algorithm uses verb argument structures created from VerbNet syntactic patterns (Kipper et al., 2000b).

Section 2 presents VerbNet syntactic patterns and the machine learning approach used to increase the coverage of verb senses. Section 3 describes the algorithms for propagating verb arguments. Section 4 presents the results and the final section 5 draws the conclusions.

2 VerbNet Syntactic Patterns

The algorithm for propagating verb arguments uses structures for representing them. Several choices were considered for retrieving verbs' argument structure. Verb syntactic patterns from WordNet (called frames) could not be used because some tokens in the patterns (like "PP" or "CLAUSE") cannot be mapped to arguments. FrameNet (Baker et al., 1998) and PropBank (Kingsbury and Palmer, 2002) contain verb syntactic patterns, but they do not have a mapping to WordNet. Finally VerbNet (Kipper et al., 2000b) represents a verb lexicon with syntactic and semantic information. This resource has a mapping to WordNet and therefore was considered the most suitable for propagating predicate arguments along lexical chains.

2.1 VerbNet description

VerbNet is based on classes of verbs. Each verb entry points to a set of classes and each class represents a sense of a verb. The classes are organized hierarchically. Each class contains a set of syntactic patterns corresponding to licensed constructions. Each syntactic pattern is an ordered list of tokens and each token represents a group of words. The tokens contain various information and constraints about the word or the group of words they represent. The name of the token can represent the thematic role of an argument, the verb itself, prepositions, adjectives, adverbs or plain words. VerbNet uses 29 thematic roles (presented in ta-

Table 1: VerbNet thematic roles

Thematic Roles		
Topic	Experiencer	Stimulus
Cause	Actor	Actor1
Actor2	Agent	Asset
Attribute	Benefactor	Beneficiary
Destination	Instrument	Location
Material	Patient	Patient1
Patient2	Predicate	Product
Recipient	Source	Theme
Theme1	Theme2	Time
Extent	Value	

ble 1). VerbNet has a static aspect and a dynamic aspect. The static aspect refers to the organization of verb entries. The dynamic aspect refers to the lexicalized trees associated with syntactic patterns. A detailed description of VerbNet dynamic aspect can be found in (Kipper et al., 2000a).

The algorithm for propagating predicate arguments uses the syntactic patterns associated with each sensekey. Each class contains a set of WordNet verb sensekeys and a set of syntactic patterns. Therefore, syntactic patterns can be associated with verb sensekey from the same class. Since sensekeys represent word senses in WordNet, each verb synset can be associated with a set of VerbNet syntactic patterns. VerbNet syntactic patterns allow predicate arguments to be propagated along lexical chains. However, not all verb senses in WordNet are listed in VerbNet classes. For the remaining verb sensekeys that are not listed in VerbNet, syntactic patterns were assigned automatically using machine learning as described in the following section.

2.2 Associating syntactic patterns with new verb senses

In order to propagate predicate arguments along lexical chains, ideally every verb in every synonym set has to have a set of syntactic patterns. Only a part of verb senses are listed in VerbNet classes. WordNet 2.0 has 24,632 verb sensekeys, but only 4,983 sensekeys are listed in VerbNet classes. For the rest, syntactic patterns were assigned automatically. In order to assign these syntactic patterns to the verb senses not listed in VerbNet, training examples were needed, both positive and negative. The learning took place for one syntactic pattern at a time. A syntactic pattern can be listed in more than one class. All verb senses associated with a syntactic pattern can be considered positive examples of verbs having that syntactic pattern. For generating negative examples,

the following assumption was used: if a verb sense listed in a VerbNet class is not associated with a given syntactic pattern, then that verb sense represents a negative example for that pattern. 352 syntactic patterns were found in all VerbNet classes. A training example was generated for each pair of syntactic patterns and verb sensekeys, resulting in a total number of 1,754,016 training examples. These training examples were used to infer rules that would classify if a verb sense key can be associated with a given syntactic pattern. Training examples were created by using the following features: verb synset semantic category, verb synset position in the IS-A hierarchy, the fact that the verb synset is related to other synsets with CAUSATION relation, the semantic classes of all noun synsets derivationally related with the given verb synset and the WordNet syntactic pattern ids. A machine learning algorithm based on C5.0 (Quinlan, 1998) was run on these training examples. Table 2 presents the performance of the learning algorithm using a 10-fold cross validation for several patterns. A number of 20,759 pairs of verb senses with their syntactic patterns were added to the existing 35,618 pairs in VerbNet. In order to improve the performance of the question answering system, around 100 patterns were manually associated with some verb senses.

Table 2: Performance of learning verb senses for several syntactic patterns

Id	Pattern	Performance
0	<Agent> <VERB> <Theme>	74.2%
1	<Experiencer> <VERB> <Cause>	98.6%
2	<Experiencer> <VERB> <Oblique> for <Cause>	98.7%
3	<Experiencer> <VERB> <Cause> in <Oblique>	98.7%
4	<Agent> <VERB> <Recipient>	94.7%
5	<Agent> <VERB> <Patient>	85.6%
6	<Patient> <VERB> <ADV>	85.1%
...	...	...
348	<Agent> <VERB> <Patient> at <Cause>	99.8%
349	<Agent> <VERB> in <Theme>	99.8%
350	<Agent> <VERB> <Source> <ADJ>	99.5%
351	<Agent> <VERB> at <Source>	99.3%

3 Propagating Verb Arguments

Given the argument structure of a verb in a sentence and a lexical chain between this verb and another, the algorithm for propagating verb arguments transforms this structure step by step, for

each relation in the lexical chain. During each step the head of the structure changes its value and the arguments can change their position. The arguments change their position in a way that preserves the original meaning as much as possible. The argument structures mirror the syntactic patterns that a verb with a given sense can have. An argument structure contains the type of the pattern, the head and an array of tokens. Each token represents an argument with a thematic role or an adjective, an adverb, a preposition or just a regular word. The head and the arguments with thematic roles are represented by concepts. A concept is created from a word found in text. If the word is found in WordNet, the concept structure contains its surface form, its lemma, its part of speech and its WordNet sense. If the word is not found in WordNet, its concept structure contains only the word and the part of speech. The value of the field for an argument is represented by the concept that is the head of the phrase representing the argument. Because a synset may contain more than one verb and each verb can have different types of syntactic patterns, propagation of verb arguments along a single relation can result in more than one structure. The output of the algorithm as well as the output of the propagation of each relation in the lexical chain is the set of argument structures with the head being a verb from the set of synonyms of the target synset. For a given relation in the lexical chain, each structure coming from the previous step is transformed into a set of new structures. The relations used and the process of argument propagation is described below.

3.1 Relations used

A restricted number of WordNet relations were used for creating lexical chains. Lexical chains between verbs were used for propagating verb arguments, and lexical chains between nouns were used to link semantically related arguments expressed with different words.

Between verb synsets the following relations were used: HYPERNYM, TROPONYM, ENTAILMENT and CAUSATION. These relations were selected because they reveal patterns about how they propagate predicate arguments.

The HYPERNYMY relation links one specific verb synset to one that is more general. Most of the time, the arguments have the same thematic roles for the two verbs. Sometimes the hypernym

synset has a syntactic pattern that has more thematic roles than the syntactic pattern of the start synset. In this case the pattern of the hypernym is not considered for propagation.

The HYPONYMY relation is the reverse of HYPERNYMY and links one verb synset to a more specific one. Inference to a more specific verb requires abduction. Most of the time, the arguments have the same thematic roles for the two verbs. Usually the hyponym of the verb synset is more specific and have less syntactic patterns than the original synset. This is why a syntactic pattern of a verb can be linked with the syntactic pattern of its hyponym that has more thematic roles. These additional thematic roles in the syntactic pattern of the hyponym will receive the value ANY-CONCEPT when verb arguments are propagated along this relation.

ENTAILMENT relation links two verb synsets that express two different events that are related: the first entails the second. This is different than HYPERNYMY or HYPONYMY that links verbs that express the same event with more or less details. Most of the time the subject of these two sentences has the same thematic role. If the thematic role of subjects is different, then the syntactic pattern of the target verb is not considered for propagation. The same happens if the start pattern contains less arguments than the target pattern. Additional arguments can change the meaning of the target pattern.

A relation that is the reverse of the ENTAILMENT is not coded in WordNet but, it is used for a better connectivity. Given one sentence S_1 with a verb V_1 that is entailed by a verb V_2 , the sentence S_1 can be reformulated using the verb V_2 , and thus creating sentence S_2 . Sentence S_1 does not imply sentence S_2 but makes it plausible. Most of the time, the subject of these two sentences has the same thematic role. If the thematic role of subjects is different, then the pattern of the target verb synset is not considered for propagation. The same happens if the start pattern has less arguments than the target pattern. Additional arguments can change the meaning of the target pattern.

The CAUSATION relation puts certain restrictions on the syntactic patterns of the two verb synsets. The first restriction applies to the syntactic pattern of the start synset: its subject must be an *Agent* or an *Instrument* and its object must be a *Patient*.

The second restriction applies to the syntactic pattern of the destination synset: its subject must be a *Patient*. If the two syntactic patterns obey these restrictions then an instance of the destination synset pattern is created and its arguments will receive the value of the argument with the same thematic role in the pattern belonging to start synset.

The reverse of the CAUSATION relation is not codified in WordNet database but it is used in lexical chains to increase the connectivity between synsets. Similar to causation relation, the reverse causation imposes two restrictions on the patterns belonging to the start and destination synset. First restriction applies to the syntactic pattern of the start synset: its subject must have the thematic role of *Patient*. The second restriction applies to the syntactic pattern of the destination synset: its subject must be an *Agent* or an *Instrument* and its object must be a *Patient*. If the two syntactic patterns obey these restrictions then an instance of the destination synset pattern is created and its arguments will receive the value of the argument with the same thematic role in the pattern belonging to start synset.

When deriving lexical chains for linking words from questions and correct answers in TREC 2004, it was observed that many chains contain a pair of DERIVATION relations. Since a pair of DERIVATION relations can link either two noun synsets or two verb synsets, the pair was concatenated into a new relation called SIM-DERIV. The number of SIM-DERIV relations is presented in table 3. For example the verb synsets *emanate#2* and *emit#1* are not synonyms (not listed in the same synset) but they are linked by a SIM-DERIV relation (both have a DERIVATION relation to the noun synset (*n-emission#1*, *emanation#2*) - nominalizations of the two verbs are listed in the same synset). There are no restrictions between pairs of patterns that participate in argument propagation. The arguments in the syntactic pattern instance of the destination synset take their values from the arguments with the same thematic roles from the syntactic pattern instance of the start synset.

Table 3: The SIM-DERIV relations generated for nouns and verb .

Relation	Source	Target	Number
SIM-DERIV	noun	noun	45,178
SIM-DERIV	verb	verb	15,926

The VERBGROUP and SEE-ALSO relations were not included in the experiment because it is not clear how they propagate arguments.

A restricted set of instances of DERIVATION relation was used to link verbs to nouns that describe their action. When arguments are propagated from verb to noun, the noun synset will receive a set of syntactic patterns instances similar to the semantic instances of the verb. When arguments are propagated from noun to verb, a new created structure for the verb sense takes the values for its arguments from the arguments with similar thematic roles in the noun structure.

Between the heads of two argument structures there can exist lexical chains of size 0, meaning that the heads of the two structures are in the same synset. However, the type of the start structure can be different than the type of the target structure. In this case, the arguments still have to be propagated from one structure to another. The arguments in the target structure will take the values of the arguments with the same thematic role in the start structure or the value ANY-CONCEPT if these arguments cannot be found.

Relations between nouns were not used by the algorithm but they are used after the algorithm is applied, to link the arguments from a resulted structure to the arguments with the same semantic roles in the target structure. If such a link exists, then the arguments are considered to match. From the existing WordNet relations between noun synsets only HYPERNYM and HYPONYM were used.

3.2 Assigning weights to the relations

Two synsets can be connected by a large number of lexical chains. For efficiency, the algorithm runs only on a restricted number of lexical chains. In order to select the most likely lexical chains, they were ordered decreasingly by their weight. The weight of a lexical chain is computed using the following formula inspired by (Moldovan and Novischi, 2002):

$$W = W_1 * A_{1,2} * MG_1 * W_2 * A_{2,3} * MG_2 * \dots * A_{n-1,n} * MG_{n-1} * W_n$$

where n represents the number of relations in the lexical chain. The formula uses the weights W_i ($i = 1..n$) of the relations along the chain (presented in table 4) and coefficients for pairs of relations $A_{i,i+1}$ (some of them presented in table 5, the rest having a weight of 1.0). This formula re-

sulted from the observation that the relations are not equal (some relations like HYPERNYM are stronger than other relations) and that the order of relations in the lexical chain influences its fitness (the order of relations is approximated by the weight given to pairs of relations). The formula uses the “measure of generality” of a concept defined as:

$$MG_C = \frac{500}{500 + N_{RGLOSS}}$$

where N_{RGLOSS} represents the number of occurrences of a given concept in WordNet glosses.

Table 4: The weight assigned to each relation

Relation	Weight
HYPERNYM	0.8
HYPONYM	0.7
DERIVATION	0.6
ENTAILMENT	0.7
R-ENTAILMENT	0.6
CAUSATION	0.7
R-CAUSATION	0.6

Table 5: Some of the weights assigned to pair of relations

Relation 1	Relation 2	Coefficient Weight
HYPERNYM	HYPONYM	1.25
HYPERNYM	ENTAILMENT	1.25
HYPERNYM	R-ENTAILMENT	0.8
HYPERNYM	CAUSATION	1.25
HYPERNYM	R-CAUSATION	1.25
HYPONYM	HYPERNYM	0.8
HYPONYM	ENTAILMENT	1.25
HYPONYM	R-ENTAILMENT	0.8
HYPONYM	CAUSATION	1.25
HYPONYM	R-CAUSATION	0.8
ENTAILMENT	HYPERNYM	1.25
ENTAILMENT	HYPONYM	0.8
ENTAILMENT	CAUSATION	1.25
ENTAILMENT	R-CAUSATION	0.8
R-ENTAILMENT	HYPERNYM	0.8
R-ENTAILMENT	HYPONYM	0.8
R-ENTAILMENT	CAUSATION	0.8
R-ENTAILMENT	R-CAUSATION	1.25
CAUSATION	HYPERNYM	1.25
CAUSATION	HYPONYM	0.8
CAUSATION	ENTAILMENT	1.25
CAUSATION	R-ENTAILMENT	0.8

3.3 Example

In the test set from the QA track in TREC 2004 we found the following question with correct answer:

Q 28.2: (Abercrombie & Fitch) When was it established?

A: ... Abercrombie & Fitch began life in 1982 ...

The verb *establish* in the question has sense 2 in WordNet 2.0 and the verb *begin* in the answer

has also sense 2. The following lexical chain can be found between these two verbs:

```
(v-begin#2,start#4)
  R-CAUSATION
(v-begin#3,lead_off#2,start#2,commence#2)
  SIM-DERIV
(v-establish#2,found#1)
```

From the question, an argument structure is created for the verb *establish#2* using the following pattern:

<Agent> establish#2 <Patient>

where the argument with the thematic role of *Agent* has the value ANY-CONCEPT, and the *Patient* argument has the value *Abercrombie & Fitch*.

From the answer, an argument structure is created for verb *begin#2* using the pattern:

<Patient> begin#2 <Theme>

where the *Patient* argument has the value *Abercrombie & Fitch* and the *Theme* argument has the value *n-life#2*. This structure is propagated along the lexical chain, each relation at a time. First for the R-CAUSATION relation links the verb *begin#2* having the pattern:

<Patient> <Verb> <Theme>

with the verb *begin#3* that has the pattern:

<Agent> begin#3 <Patient>

The *Patient* keeps its value *Abercrombie & Fitch* event though it is changing its syntactic role from subject of the verb *begin#2* to the object of the verb *begin#3*. The *Theme* argument is lost along this relation, instead the new argument with the thematic role of *Agent* receives the special value ANY-CONCEPT.

The second relation in the chain, SIM-DERIV links two verbs that have the same syntactic pattern:

<Agent> <Verb> <Patient>

Therefore a new structure is created for the verb *establish#2* using this pattern and its arguments take their values from the similar arguments in the argument structure for verb *begin#3*. This new structure exactly matches the argument structure from the question therefore the answer is ranked the highest in the set of candidate answer. Figure 1 illustrates the argument propagation process for this example.

4 Experiments and Results

The algorithm for propagating verb arguments was used to improve performance of an in-house Ques-

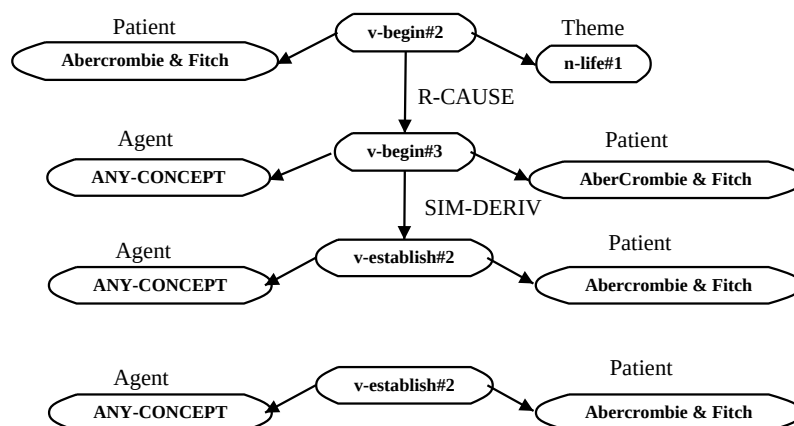
tion Answering system (Moldovan et al., 2004). This improvement comes from a better matching between a question and the sentences containing the correct answer. Integration of this algorithm into the Question Answering system requires 3 steps: (1) creation of structures containing verb arguments for the questions and its possible answers, (2) derivation of lexical chains between the two structures and propagation of the arguments along lexical chains, (3) measuring the similarity between the propagated structures and the structures from the question and re-ranking of the candidate answers based on similarity scores. Structures containing predicate arguments are created for all the verbs in the question and all verbs in each possible answer. The QA system takes care of coreference resolution.

Argument structures are created for verbs in both active and passive voice. If the verb is in passive voice, then its arguments are normalized to active voice. The subject phrase of the verb in passive voice represents its object and the noun phrase inside prepositional phrase with preposition “by” becomes its subject. Special attention is given to di-transitive verbs. If in passive voice, the subject phrase can represent either the direct object or indirect object. The distinction is made in the following way: if the verb in passive voice has a direct object then the subject represents the indirect object (beneficiary), otherwise the subject represents direct object. All the other arguments are treated in the same way as in the active voice case.

After the structures are created from a candidate answer and a question, lexical chains are created between their heads. Because lexical chains link two word senses, the heads need to be disambiguated. Before searching for lexical chains, the heads could be already partially disambiguated, because only a restricted number of senses of the head verb can have the VerbNet syntactic pattern matching the input text. An additional semantic disambiguation can take place before deriving lexical chains. The verbs from the answer and question can also be disambiguated by selecting the best lexical chain between them. This was the approach used in our experiment.

The algorithm propagating verb arguments was tested on a set of 106 pairs of phrases with similar meaning for which argument structures could be built. These phrases were selected from pairs of questions and their correct answers from the

A: ... Abercrombie & Fitch began life in 1982



Q 28.2 (Abercrombie & Fitch) When was it established?

Figure 1: Example of lexical chain that propagates syntactic constraints from answer to question.

set of factoid questions in TREC 2004 and also from the pairs of scenarios and hypotheses from first edition of PASCAL RTE Challenge (Dagan et al., 2005). Table 6 shows algorithm performance. The columns in the table correspond to the following cases:

- how many cases the algorithm propagated all the arguments;
- how many cases the algorithm propagated one argument;
- how many cases the algorithm did not propagate any argument;

using top 5, 20, 50 lexical chains.

The purpose of the algorithm for propagating predicate arguments is to measure the similarity between the sentences for which the argument structures have been built. This similarity can be computed by comparing the target argument structure with the propagated argument structure. The similarity score is computed in the following way: if N represents the number of arguments in a pattern, each argument matched is defined to have a contribution of $1/(N + 1)$, except for the subject that has a contribution if matched of $2/(N+1)$. The propagated pattern is compared with the target pattern and the score is computed by summing up the contributions of all matched arguments.

The set of factoid questions in TREC 2004 has 230 questions. Lexical chains containing the restricted set of relations that propagate verb arguments were found for 33 questions, linking verbs in those questions to verbs in their correct an-

swer. This is the maximum number of questions on which the algorithm for propagating syntactic constraints can have an impact without using other knowledge. The algorithm for propagating verb argument could be applied on 15 of these questions. Table 7 shows the improvement of the Question Answering system when the first 20 or 50 answers returned by factoid strategy are re-ranked according to similarity scores between argument structures. The performance of the question answering system was measured using Mean Reciprocal Rank (MRR).

Table 7: The impact of the algorithm for propagating predicate arguments over the question answering system

Number of answers	Performance
Top 20	1.9%
Top 50	2.4%

5 Conclusion

This paper describes the approach of propagating verb arguments along lexical chains with WordNet relations using VerbNet frames. Since VerbNet frames are not associated with all verb senses from WordNet, some verb senses were added automatically to the existing VerbNet frames. The algorithm was used to improve the performance of the answer's ranking stage in Question Answering system. Only a restricted set of WordNet semantic

Table 6: The performance of the algorithm for propagating predicate arguments with semantic constraints

	Arguments propagated	Top 5 chains	Top 10 chains	Top 20 chains
a	all arguments	23(21.6%)	28(26.4%)	32(30.2%)
b	at least one argument	73(68.8%)	81(76.4%)	89(83.9%)
c	no arguments	32(30.2%)	25(23.6%)	17(16.0%)

relations were used to propagate predicate arguments. Lexical chains were also derived between the arguments for a better match. On the set of factoid questions from TREC 2004, it was found that for 33(14.3%) questions, the words in the question and the related words in the answer could be linked using lexical chains containing only the relations from the restricted set that propagate verb arguments. Overall, the algorithm for propagating verb arguments improved the system performance with 2.4%

References

- Kisuh Ahn, Johan Bos, James R. Curran, Dave Kor, Malvina Nissim, and Bonnie Webber. 2005. Question Answering with QED at TREC-2005. In *Proceedings of TREC 2005*.
- Collin F. Baker, Charles J. Fillmore, , and John B Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the COLING-ACL*, Montreal, Canada.
- Hang Cui, Keya Li, Renxu Sun, Tat-Seng Chua, and Min-Yen-Kan. 2004. National University of Singapore at the TREC-13 Question Answering Main Task. In *Proceedings of the 13th Text Retrieval Conference (TREC-2004)*, Gaithersburg, Maryland, USA, November 16-19.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. Recognising Textual Entailment Challenge, <http://www.pascal-network.org/Challenges/RTE>, March.
- Paul Kingsbury and Martha Palmer. 2002. From Treebank to PropBank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, Las Palmas, Spain.
- K. Kipper, H. Dang, W. Schuler, and M. Palmer. 2000a. Building a class-based verb lexicon using tags. In *Proceedings of Fifth TAG+ Workshop*.
- Karin Kipper, Hoa Trang Dang, and Martha Palmer. 2000b. Class-based construction of a verb lexicon. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 691–696. AAAI Press / The MIT Press.
- D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL-98*, Montreal, Canada, August.
- G. Miller. 1995. WordNet: a lexical database. *Communications of the ACM*, 38(11):39–41, November.
- Dan Moldovan and Adrian Novischi. 2002. Lexical chains for question answering. In *Proceedings of COLING 2002*, pages 674–680.
- Dan I. Moldovan and Vasile Rus. 2001. Logic Form Transformation of WordNet and its Applicability to Question Answering. In *Proceedings of the ACL 2001*, Toulouse, France, July.
- Dan I. Moldovan, Christine Clark, Sanda M. Harabagiu, and Steven J. Maiorano. 2003. Cogex: A logic prover for question answering. In *Proceedings of HLT-NAACL 2003*, Edmonton, Canada, May-June.
- Dan Moldovan, Sanda Harabagiu, Christine Clark, and Mitchell Bowden. 2004. PowerAnswer 2: Experiments and Analysis over TREC 2004. In *Proceedings of Text Retrieval Conference 2004*.
- Dan Moldovan, Christine Clark, and Sanda Harabagiu. 2005. Temporal Context Representation and Reasoning. In *Proceedings of IJCAI-2005*, pages 1099–1104, Edinburgh, Scotland, July-August.
- R. Quinlan. 1998. C5.0: An Informal Tutorial, RuleQuest.
- H. Tanev, M. Kouylekov, and B. Magnini. 2004. Combining linguistic processing and web mining for question answering: Itc-irst at trec 2004. In *Proceedings of the 13th Text Retrieval Conference (TREC-2004)*, pages 429–438, Gaithersburg, Maryland, USA, November 16-19.
- Ellen M. Voorhees. 2004. Overview of the TREC 2004 Question Answering Track. In *Proceedings of the 13th Text Retrieval Conference (TREC-2004)*, pages 83–105, Gaithersburg, Maryland, USA, November 16-19.

Methods for Using Textual Entailment in Open-Domain Question Answering

Sanda Harabagiu and Andrew Hickl

Language Computer Corporation
1701 North Collins Boulevard
Richardson, Texas 75080 USA
sanda@languagecomputer.com

Abstract

Work on the semantics of questions has argued that the relation between a question and its answer(s) can be cast in terms of logical entailment. In this paper, we demonstrate how computational systems designed to recognize *textual entailment* can be used to enhance the accuracy of current open-domain automatic question answering (Q/A) systems. In our experiments, we show that when textual entailment information is used to either filter or rank answers returned by a Q/A system, accuracy can be increased by as much as 20% overall.

1 Introduction

Open-Domain Question Answering (Q/A) systems return a textual expression, identified from a vast document collection, as a response to a question asked in natural language. In the quest for producing accurate answers, the open-domain Q/A problem has been cast as: (1) a pipeline of linguistic processes pertaining to the processing of questions, relevant passages and candidate answers, interconnected by several types of lexico-semantic feedback (cf. (Harabagiu et al., 2001; Moldovan et al., 2002)); (2) a combination of language processes that transform questions and candidate answers in logic representations such that reasoning systems can select the correct answer based on their proofs (cf. (Moldovan et al., 2003)); (3) a noisy-channel model which selects the most likely answer to a question (cf. (Echihabi and Marcu, 2003)); or (4) a constraint satisfaction problem, where sets of auxiliary questions are used to provide more information and

better constrain the answers to individual questions (cf. (Prager et al., 2004)). While different in their approach, each of these frameworks seeks to approximate the forms of semantic inference that will allow them to identify valid textual answers to natural language questions.

Recently, the task of automatically recognizing one form of semantic inference – textual entailment – has received much attention from groups participating in the 2005 and 2006 PASCAL Recognizing Textual Entailment (RTE) Challenges (Dagan et al., 2005).¹

As currently defined, the RTE task requires systems to determine whether, given two text fragments, the meaning of one text could be reasonably inferred, or *textually entailed*, from the meaning of the other text. We believe that systems developed specifically for this task can provide current question-answering systems with valuable semantic information that can be leveraged to identify exact answers from ranked lists of candidate answers. By replacing the pairs of texts evaluated in the RTE Challenge with combinations of questions and candidate answers, we expect that textual entailment could provide yet another mechanism for approximating the types of inference needed in order answer questions accurately.

In this paper, we present three different methods for incorporating systems for textual entailment into the traditional Q/A architecture employed by many current systems. Our experimental results indicate that (even at their current level of performance) textual entailment systems can substantially improve the accuracy of Q/A, even when no other form of semantic inference is employed.

The remainder of the paper is organized as fol-

¹<http://www.pascal-network.org/Challenges/RTE>

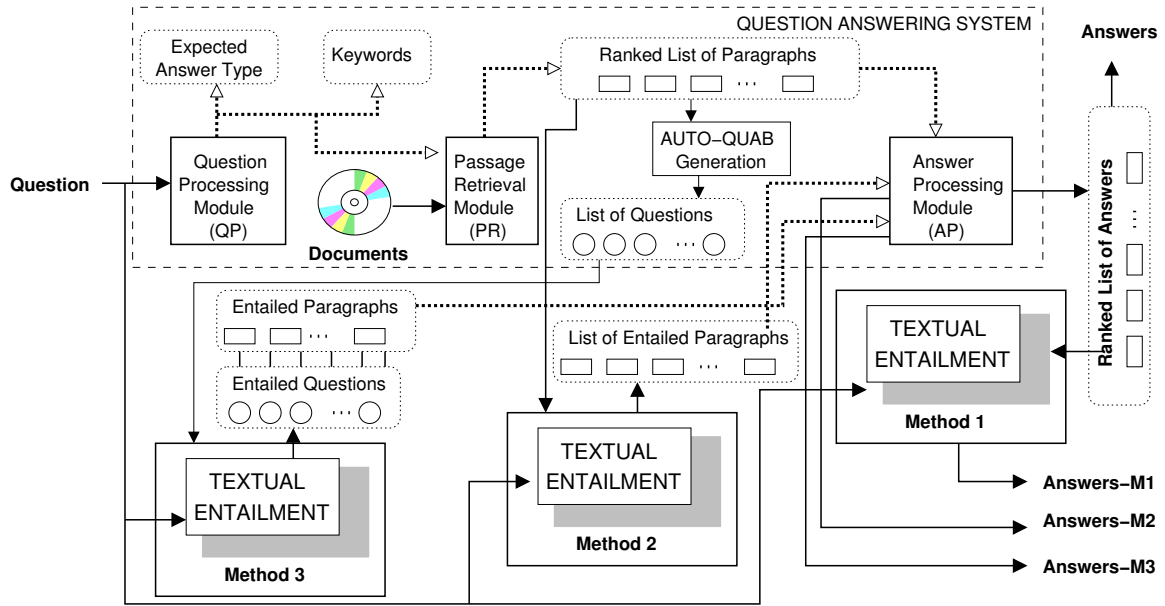


Figure 1: Integrating Textual Entailment in Q/A.

lows. Section 2 describes the three methods of using textual entailment in open-domain question answering that we have identified, while Section 3 presents the textual entailment system we have used. Section 4 details our experimental methods and our evaluation results. Finally, Section 5 provides a discussion of our findings, and Section 6 summarizes our conclusions.

2 Integrating Textual Entailment in Question Answering

In this section, we describe three different methods for integrating a textual entailment (TE) system into the architecture of an open-domain Q/A system.

Work on the semantics of questions (Groenendijk, 1999; Lewis, 1988) has argued that the formal answerhood relation found between a question and a set of (correct) answers can be cast in terms of logical entailment. Under these approaches (referred to as *licensing* by (Groenendijk, 1999) and *aboutness* by (Lewis, 1988)), p is considered to be an answer to a question $?q$ iff $?q$ logically entails the set of worlds in which p is true (i.e. $?p$). While the notion of *textual entailment* has been defined far less rigorously than logical entailment, we believe that the recognition of textual entailment between a question and a set of candidate answers – or between a question and questions generated from answers – can enable Q/A systems to identify correct answers with greater precision than current keyword- or pattern-based

techniques.

As illustrated in Figure 1, most open-domain Q/A systems generally consist of a sequence of three modules: (1) a *question processing* (QP) module; (2) a *passage retrieval* (PR) module; and (3) an *answer processing* (AP) module. Questions are first submitted to a QP module, which extracts a set of relevant keywords from the text of the question and identifies the question’s expected answer type (EAT). Keywords – along with the question’s EAT – are then used by a PR module to retrieve a ranked list of paragraphs which may contain answers to the question. These paragraphs are then sent to an AP module, which extracts an exact *candidate answer* from each passage and then ranks each candidate answer according to the likelihood that it is a correct answer to the original question.

Method 1. In Method 1, each of a ranked list of answers that do not meet the minimum conditions for TE are removed from consideration and then re-ranked based on the *entailment confidence* (a real-valued number ranging from 0 to 1) assigned by the TE system to each remaining example. The system then outputs a new set of ranked answers which do not contain any answers that are not entailed by the user’s question.

Table 1 provides an example where Method 1 could be used to make the right prediction for a set of answers. Even though A_1 was ranked in sixth position, the identification of a high-confidence positive entailment enabled it to be returned as the

top answer. In contrast, the recognition of a negative entailment for A_2 caused this answer to be dropped from consideration altogether.

Q_1 : "What did Peter Minuit buy for the equivalent of \$24.00?"				
	Rank ₁	TE	Rank ₂	Answer Text
A_1	6 th	YES (0.89)	1 st	Everyone knows that, back in 1626, Peter Minuit bought Manhattan from the Indians for \$24 worth of trinkets.
A_2	1 st	NO (0.81)	–	In 1626, an enterprising Peter Minuit flagged down some passing locals, plied them with beads, cloth and trinkets worth an estimated \$24, and walked away with the whole island.

Table 1: Re-ranking of answers by Method 1.

Method 2. Since AP is often a resource-intensive process for most Q/A systems, we expect that TE information can be used to limit the number of passages considered during AP. As illustrated in Method 2 in Figure 1, lists of passages retrieved by a PR module can either be ranked (or filtered) using TE information. Once ranking is complete, answer extraction takes place only on the set of *entailed passages* that the system considers likely to contain a correct answer to the user’s question.

Method 3. In previous work (Harabagiu et al., 2005b), we have described techniques that can be used to automatically generate well-formed natural language questions from the text of paragraphs retrieved by a PR module. In our current system, sets of automatically-generated questions (AGQ) are created using a stand-alone *AutoQUAB* generation module, which assembles question-answer pairs (known as *QUABs*) from the top-ranked passages returned in response to a question. Table 2 lists some of the questions that this module has produced for the question Q_2 : “How hot does the inside of an active volcano get?”.

Q_2 : "How hot does the inside of an active volcano get?"	
A_2	Tamagawa University volcano expert Takeyo Kosaka said lava fragments belched out of the mountain on January 31 were as hot as 300 degrees Fahrenheit . The intense heat from a second eruption on Tuesday forced rescue operations to stop after 90 minutes. Because of the high temperatures, the bodies of only five of the volcano’s initial victims were retrieved.
Positive Entailment	
AGQ ₁	What temperature were the lava fragments belched out of the mountain on January 31?
AGQ ₂	How many degrees Fahrenheit were the lava fragments belched out of the mountain on January 31?
Negative Entailment	
AGQ ₃	When did rescue operations have to stop?
AGQ ₄	How many bodies of the volcano’s initial victims were retrieved?

Table 2: TE between AGQs and user question.

Following (Groenendijk, 1999), we expect that if a question $?q$ logically entails another question $?q'$, then some subset of the answers entailed by $?q'$ should also be interpreted as valid answers to $?q$. By establishing TE between a question and AGQs derived from passages identified by the Q/A system for that question, we expect we can iden-

tify a set of answer passages that contain correct answers to the original question. For example, in Table 2, we find that entailment between questions indicates the correctness of a candidate answer: here, establishing that Q_2 entails AGQ₁ and AGQ₂ (but not AGQ₃ or AGQ₄) enables the system to select A_2 as the correct answer.

When at least one of the AGQs generated by the AutoQUAB module is entailed by the original question, all AGQs that do not reach TE are filtered from consideration; remaining passages are assigned an entailment confidence score and are sent to the AP module in order to provide an exact answer to the question. Following this process, candidate answers extracted from the AP module were then re-associated with their AGQs and resubmitted to the TE system (as in Method 1). Question-answer pairs deemed to be positive instances of entailment were then stored in a database and used as additional training data for the AutoQUAB module. When no AGQs were found to be entailed by the original question, however, passages were ranked according to their entailment confidence and sent to AP for further processing and validation.

3 The Textual Entailment System

Processing textual entailment, or recognizing whether the information expressed in a text can be inferred from the information expressed in another text, can be performed in four ways. We can try to (1) derive linguistic information from the pair of texts, and cast the inference recognition as a classification problem; or (2) evaluate *the probability* that an entailment can exist between the two texts; (3) represent the knowledge from the pair of texts in some representation language that can be associated with an inferential mechanism; or (4) use the classical AI definition of entailment and build models of the world in which the two texts are respectively true, and then check whether the models associated with one text are included in the models associated with the other text. Although we believe that each of these methods should be investigated fully, we decided to focus only on the first method, which allowed us to build the TE system illustrated in Figure 2.

Our TE system consists of (1) a *Preprocessing Module*, which derives linguistic knowledge from the text pair; (2) an *Alignment Module*, which takes advantage of the notions of lexical alignment

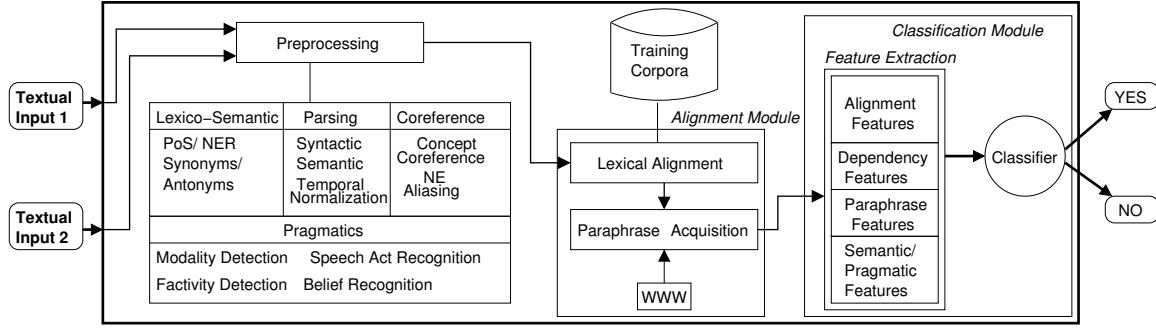


Figure 2: Textual Entailment Architecture.

and textual paraphrases; and (3) a *Classification Module*, which uses a machine learning classifier (based on decision trees) to make an entailment judgment for each pair of texts.

As described in (Hickl et al., 2006), the Preprocessing module is used to syntactically parse texts, identify the semantic dependencies of predicates, label named entities, normalize temporal and spatial expressions, resolve instances of coreference, and annotate predicates with polarity, tense, and modality information.

Following preprocessing, texts are sent to an *Alignment Module* which uses a Maximum Entropy-based classifier in order to estimate the probability that pairs of constituents selected from texts encode corresponding information that could be used to inform an entailment judgment. This module assumes that since sets of entailing texts necessarily predicate about the same set of individuals or events, systems should be able to identify elements from each text that convey similar types of presuppositions. Examples of predicates and arguments aligned by this module are presented in Figure 3.

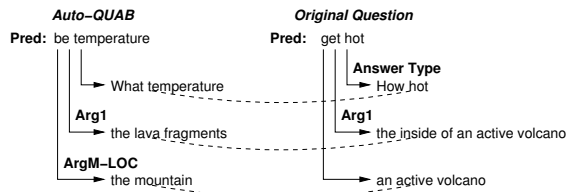


Figure 3: Alignment Graph

Aligned constituents are then used to extract sets of phrase-level alternations (or “paraphrases”) from the WWW that could be used to capture correspondences between texts longer than individual constituents. The top 8 candidate paraphrases for two of the aligned elements from Figure 3 are presented in Table 3.

Finally, the *Classification Module* employs a

Judgment	Paraphrase
YES	<i>lava fragments</i> in pyroclastic flows can reach 400 degrees
YES	<i>an active volcano</i> can get up to 2000 degrees
NO	<i>an active volcano</i> above you are slopes of 30 degrees
YES	<i>the active volcano</i> with steam reaching 80 degrees
YES	<i>lava fragments</i> such as cinders may still be as hot as 300 degrees
NO	<i>lava</i> is a liquid at high temperature: typically from 700 degrees

Table 3: Phrase-Level Alternations

decision tree classifier in order to determine whether an entailment relationship exists for each pair of texts. This classifier is learned using features extracted from the previous modules, including features derived from (1) the (lexical) alignment of the texts, (2) syntactic and semantic dependencies discovered in each text passage, (3) paraphrases derived from web documents, and (4) semantic and pragmatic annotations. (A complete list of features can be found in Figure 4.) Based on these features, the classifier outputs both an entailment judgment (either *yes* or *no*) and a confidence value, which is used to rank answers or paragraphs in the architecture illustrated in Figure 1.

3.1 Lexical Alignment

Several approaches to the RTE task have argued that the recognition of textual entailment can be enhanced when systems are able to identify – or align – corresponding entities, predicates, or phrases found in a pair of texts. In this section, we show that by using a machine learning-based classifier which combines lexico-semantic information from a wide range of sources, we are able to accurately identify aligned constituents in pairs of texts with over 90% accuracy.

We believe the alignment of corresponding entities can be cast as a classification problem which uses lexico-semantic features in order to compute an alignment probability $p(a)$, which corresponds to the likelihood that a term selected from one text entails a term from another text. We used constituency information from a chunk parser to decompose the pair of texts into a set of disjoint seg-

ALIGNMENT FEATURES: These three features are derived from the results of the lexical alignment classification. ◊1◊ LONGEST COMMON STRING: This feature represents the longest contiguous string common to both texts. ◊2◊ UNALIGNED CHUNK: This feature represents the number of chunks in one text that are not aligned with a chunk from the other ◊3◊ LEXICAL ENTAILMENT PROBABILITY: This feature is defined in (Glickman and Dagan, 2005).
DEPENDENCY FEATURES: These four features are computed from the PropBank-style annotations assigned by the semantic parser. ◊1◊ ENTITY-ARG MATCH: This is a boolean feature which fires when aligned entities were assigned the same argument role label. ◊2◊ ENTITY-NEAR-ARG MATCH: This feature is collapsing the arguments Arg_1 and Arg_2 (as well as the Arg_M subtypes) into single categories for the purpose of counting matches. ◊3◊ PREDICATE-ARG MATCH: This boolean feature is flagged when at least two aligned arguments have the same role. ◊4◊ PREDICATE-NEAR-ARG MATCH: This feature is collapsing the arguments Arg_1 and Arg_2 (as well as the Arg_M subtypes) into single categories for the purpose of counting matches.
PARAPHRASE FEATURES: These three features are derived from the paraphrases acquired for each pair. ◊1◊ SINGLE PATTERN MATCH: This is a boolean feature which fired when a paraphrase matched either of the texts. ◊2◊ BOTH PATTERN MATCH: This is a boolean feature which fired when paraphrases matched both texts. ◊3◊ CATEGORY MATCH: This is a boolean feature which fired when paraphrases could be found from the same paraphrase cluster that matched both texts.
SEMANTIC/PRAGMATIC FEATURES: These six features are extracted by the preprocessing module. ◊1◊ NAMED ENTITY CLASS: This feature has a different value for each of the 150 named entity classes. ◊2◊ TEMPORAL NORMALIZATION: This boolean feature is flagged when the temporal expressions are normalized to the same ISO 9000 equivalents. ◊3◊ MODALITY MARKER: This boolean feature is flagged when the two texts use the same modal verbs. ◊4◊ SPEECH-ACT: This boolean feature is flagged when the lexicons indicate the same speech act in both texts. ◊5◊ FACTIVITY MARKER: This boolean feature is flagged when the factivity markers indicate either TRUE or FALSE in both texts simultaneously. ◊6◊ BELIEF MARKER: This boolean feature is set when the belief markers indicate either TRUE or FALSE in both texts simultaneously.
CONTRAST FEATURES: These six features are derived from the opposing information provided by antonymy relations or chains. ◊1◊ NUMBER OF LEXICAL ANTONYMY RELATIONS: This feature counts the number of antonyms from WordNet that are discovered between the two texts. ◊2◊ NUMBER OF ANTONYMY CHAINS: This feature counts the number of antonymy chains that are discovered between the two texts. ◊3◊ CHAIN LENGTH: This feature represents a vector with the lengths of the antonymy chains discovered between the two texts. ◊4◊ NUMBER OF GLOSSES: This feature is a vector representing the number of Gloss relations used in each antonymy chain. ◊5◊ NUMBER OF MORPHOLOGICAL CHANGES: This feature is a vector representing the number of Morphological-Derivation relations found in each antonymy chain. ◊6◊ NUMBER OF NODES WITH DEPENDENCIES: This feature is a vector indexing the number of nodes in each antonymy chain that contain dependency relations. ◊7◊ TRUTH-VALUE MISMATCH: This is a boolean feature which fired when two aligned predicates differed in any truth value. ◊8◊ POLARITY MISMATCH: This is a boolean feature which fired when predicates were assigned opposite polarity values.

Figure 4: Features Used in Classifying Entailment

ments known as “alignable chunks”. Alignable chunks from one text (C_t) and the other text (C_h) are then assembled into an alignment matrix ($C_t \times C_h$). Each pair of chunks ($p \in C_t \times C_h$) is then submitted to a Maximum Entropy-based classifier which determines whether or not the pair of chunks represents a case of lexical entailment.

Three classes of features were used in the

Alignment Classifier: (1) a set of statistical features (e.g. cosine similarity), (2) a set of lexico-semantic features (including WordNet Similarity (Pedersen et al., 2004), named entity class equality, and part-of-speech equality), and (3) a set of string-based features (such as Levenshtein edit distance and morphological stem equality).

As in (Hickl et al., 2006), we used a two-step approach to obtain sufficient training data for the Alignment Classifier. First, humans were tasked with annotating a total of 10,000 alignment pairs (extracted from the 2006 PASCAL Development Set) as either positive or negative instances of alignment. These annotations were then used to train a hillclimber that was used to annotate a larger set of 450,000 alignment pairs selected at random from the training corpora described in Section 3.3. These machine-annotated examples were then used to train the Maximum Entropy-based classifier that was used in our TE system. Table 4 presents results from TE’s linear- and Maximum Entropy-based Alignment Classifiers on a sample of 1000 alignment pairs selected at random from the 2006 PASCAL Test Set.

Classifier	Training Set	Precision	Recall	F-Measure
Linear	10K pairs	0.837	0.774	0.804
Maximum Entropy	10K pairs	0.881	0.851	0.866
Maximum Entropy	450K pairs	0.902	0.944	0.922

Table 4: Performance of Alignment Classifier

3.2 Paraphrase Acquisition

Much recent work on automatic paraphrasing (Barzilay and Lee, 2003) has used relatively simple statistical techniques to identify text passages that contain the same information from parallel corpora. Since sentence-level paraphrases are generally assumed to contain information about the same event, these approaches have generally assumed that all of the available paraphrases for a given sentence will include at least one pair of entities which can be used to extract sets of paraphrases from text.

The TE system uses a similar approach to gather phrase-level alternations for each entailment pair. In our system, the two highest-confidence entity alignments returned by the Lexical Alignment module were used to construct a query which was used to retrieve the top 500 documents from *Google*, as well as all matching instances from our training corpora described in Section 3.3. This method did not always extract true paraphrases of either texts. In order increase the likelihood that

only true paraphrases were considered as phrase-level alternations for an example, extracted sentences were clustered using complete-link clustering using a technique proposed in (Barzilay and Lee, 2003).

3.3 Creating New Sources of Training Data

In order to obtain more training data for our TE system, we extracted more than 200,000 examples of textual entailment from large newswire corpora.

Positive Examples. Following an idea proposed in (Burger and Ferro, 2005), we created a corpus of approximately 101,000 textual entailment examples by pairing the headline and first sentence from newswire documents. In order to increase the likelihood of including only positive examples, pairs were filtered that did not share an entity (or an NP) in common between the headline and the first sentence

Judgment	Example
YES	Text-1: Sydney newspapers made a secret deal not to report on the fawning and spending during the city's successful bid for the 2000 Olympics, former Olympics Minister Bruce Baird said today. Text-2: Papers Said To Protect Sydney Bid
YES	Text-1: An IOC member expelled in the Olympic bribery scandal was consistently drunk as he checked out Stockholm's bid for the 2004 Games and got so offensive that he was thrown out of a dinner party, Swedish officials said. Text-2: Officials Say IOC Member Was Drunk

Table 5: Positive Examples

Negative Examples. Two approaches were used to gather negative examples for our training set. First, we extracted 98,000 pairs of sequential sentences that included mentions of the same named entity from a large newswire corpus. We also extracted 21,000 pairs of sentences linked by connectives such as *even though*, *in contrast* and *but*.

Judgment	Example
NO	Text-1: One player losing a close friend is Japanese pitcher Hideki Irabu, who was befriended by Wells during spring training last year. Text-2: Irabu said he would take Wells out to dinner when the Yankees visit Toronto.
NO	Text-1: According to the professor, present methods of cleaning up oil slicks are extremely costly and are never completely efficient. Text-2: <i>In contrast</i> , he stressed, Clean Mag has a 100 percent pollution retrieval rate, is low cost and can be recycled.

Table 6: Negative Examples

4 Experimental Results

In this section, we describe results from four sets of experiments designed to explore how textual entailment information can be used to enhance the quality of automatic Q/A systems. We show that

by incorporating features from TE into a Q/A system which employs no other form of textual inference, we can improve accuracy by more than 20% over a baseline.

We conducted our evaluations on a set of 500 factoid questions selected randomly from questions previously evaluated during the annual TREC Q/A evaluations.² Of these 500 questions, 335 (67.0%) were automatically assigned an answer type from our system's answer type hierarchy; the remaining 165 (33.0%) questions were classified as having an unknown answer type. In order to provide a baseline for our experiments, we ran a version of our Q/A system, known as FERRET (Harabagiu et al., 2005a), that does not make use of textual entailment information when identifying answers to questions. Results from this baseline are presented in Table 7.

Question Set	Questions	Correct	Accuracy	MRR
Known Answer Types	335	107	32.0%	0.3001
Unknown Answer Types	265	81	30.6%	0.2987

Table 7: Q/A Accuracy without TE

The performance of the TE system described in Section 3 was first evaluated in the 2006 PAS-CAL RTE Challenge. In this task, systems were tasked with determining whether the meaning of a sentence (referred to as a *hypothesis*) could be reasonably inferred from the meaning of another sentence (known as a *text*). Four types of sentence pairs were evaluated in the 2006 RTE Challenge, including: pairs derived from the output of (1) automatic question-answering (QA) systems, (2) information extraction systems (IE), (3) information retrieval (IR) systems, and (4) multi-document summarization (SUM) systems. The accuracy of our TE system across these four tasks is presented in Table 8.

		Training Data	
		Development Set	Additional Corpora
Number of Examples		800	201,000
Task	QA-test	0.5750	0.6950
	IE-test	0.6450	0.7300
	IR-test	0.6200	0.7450
	SUM-test	0.7700	0.8450
Overall Accuracy		0.6525	0.7538

Table 8: Accuracy on the 2006 RTE Test Set

In previous work (Hickl et al., 2006), we have found that the type and amount of training data available to our TE system significantly ($p < 0.05$) impacted its performance on the 2006 RTE Test Set. When our system was trained on the training corpora described in Section 3.3, the overall accuracy of the system increased by more than 10%,

²Text Retrieval Conference (<http://trec.nist.gov>)

from 65.25% to 75.38%. In order to provide training data that replicated the task of recognizing entailment between a question and an answer, we assembled a corpus of 5000 question-answer pairs selected from answers that our baseline Q/A system returned in response to a new set of 1000 questions selected from the TREC test sets. 2500 positive training examples were created from answers identified by human annotators to be correct answers to a question, while 2500 negative examples were created by pairing questions with incorrect answers returned by the Q/A system.

After training our TE system on this corpus, we performed the following four experiments:

Method 1. In the first experiment, the ranked lists of answers produced by the Q/A system were submitted to the TE system for validation. Under this method, answers that were not entailed by the question were removed from consideration; the top-ranked entailed answer was then returned as the system’s answer to the question. Results from this method are presented in Table 9.

Method 2. In this experiment, entailment information was used to rank passages returned by the PR module. After an initial relevance ranking was determined from the PR engine, the top 50 passages were paired with the original question and were submitted to the TE system. Passages were re-ranked using the entailment judgment and the entailment confidence computed for each pair and then submitted to the AP module. Features derived from the entailment confidence were then combined with the keyword- and relation-based features described in (Harabagiu et al., 2005a) in order to produce a final ranking of candidate answers. Results from this method are presented in Table 9.

Method 3. In the third experiment, TE was used to select AGQs that were entailed by the question submitted to the Q/A system. Here, AutoQUAB was used to generate questions for the top 50 candidate answers identified by the system. When at least one of the top 50 AGQs were entailed by the original question, the answer passage associated with the top-ranked entailed question was returned as the answer. When none of the top 50 AGQs were entailed by the question, question-answer pairs were re-ranked based on the entailment confidence, and the top-ranked answer was returned. Results for both of these conditions are presented in Table 9.

Hybrid Method. Finally, we found that the best results could be obtained by combining aspects of each of these three strategies. Under this approach, candidate answers were initially ranked using features derived from entailment classifications performed between (1) the original question and each candidate answer and (2) the original question and the AGQ generated from each candidate answer. Once a ranking was established, answers that were not judged to be entailed by the question were also removed from final ranking. Results from this hybrid method are provided in Table 9.

	Known EAT		Unknown EAT	
	Acc	MRR	Acc	MRR
Baseline	32.0%	0.3001	30.6%	0.2978
Method 1	44.1%	0.4114	39.5%	0.3833
Method 2	52.4%	0.5558	42.7%	0.4135
Method 3	41.5%	0.4257	37.5%	0.3575
Hybrid	53.9%	0.5640	41.9%	0.4010

Table 9: Q/A Performance with TE

5 Discussion

The experiments reported in this paper suggest that current TE systems may be able to provide open-domain Q/A systems with the forms of semantic inference needed to perform accurate answer validation. While probabilistic or web-based methods for answer validation have been previously explored in the literature (Magnini et al., 2002), these approaches have modeled the relationship between a question and a (correct) answer in terms of *relevance* and have not tried to approximate the deeper semantic phenomena that are involved in determining answerhood.

Our work suggests that considerable gains in performance can be obtained by incorporating TE during both answer processing and passage retrieval. While best results were obtained using the Hybrid Method (which boosted performance by nearly 28% for questions with known EATs), each of the individual methods managed to boost the overall accuracy of the Q/A system by at least 7%. When TE was used to filter non-entailed answers from consideration (Method 1), the overall accuracy of the Q/A system increased by 12% over the baseline (when an EAT could be identified) and by nearly 9% (when no EAT could be identified). In contrast, when entailment information was used to rank passages and candidate answers, performance increased by 22% and 10% respectively. Somewhat smaller performance gains were achieved when TE was used to select

amongst AGQs generated by our Q/A system's AutoQUAB module (Method 3). We expect that by adding features to TE system specifically designed to account for the semantic contributions of a question's EAT, we may be able to boost the performance of this method.

6 Conclusions

In this paper, we discussed three different ways that a state-of-the-art textual entailment system could be used to enhance the performance of an open-domain Q/A system. We have shown that when textual entailment information is used to either filter or rank candidate answers returned by a Q/A system, Q/A accuracy can be improved from 32% to 52% (when an answer type can be detected) and from 30% to 40% (when no answer type can be detected). We believe that these results suggest that current supervised machine learning approaches to the recognition of textual entailment may provide open-domain Q/A systems with the inferential information needed to develop viable answer validation systems.

7 Acknowledgments

This material is based upon work funded in whole or in part by the U.S. Government and any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Government.

References

- Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *HLT-NAACL*.
- John Burger and Lisa Ferro. 2005. Generating an Entailment Corpus from News Headlines. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 49–54.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL Recognizing Textual Entailment Challenge. In *Proceedings of the PASCAL Challenges Workshop*.
- Abdessamad Echihabi and Daniel Marcu. 2003. A noisy-channel approach to question answering. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*.
- Oren Glickman and Ido Dagan. 2005. A Probabilistic Setting and Lexical Co-occurrence Model for Textual Entailment. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, Ann Arbor, USA.
- Jeroen Groenendijk. 1999. The logic of interrogation: Classical version. In *Proceedings of the Ninth Semantics and Linguistics Theory Conference (SALT IX)*, Ithaca, NY.
- Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunsecu, Roxana Girju, Vasile Rus, and Paul Morarescu. 2001. The Role of Lexico-Semantic Feedback in Open-Domain Textual Question-Answering. In *Proceedings of the 39th Meeting of the Association for Computational Linguistics*.
- S. Harabagiu, D. Moldovan, C. Clark, M. Bowden, A. Hickl, and P. Wang. 2005a. Employing Two Question Answering Systems in TREC 2005. In *Proceedings of the Fourteenth Text REtrieval Conference*.
- Sanda Harabagiu, Andrew Hickl, John Lehmann, and Dan Moldovan. 2005b. Experiments with Interactive Question-Answering. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*.
- Andrew Hickl, John Williams, Jeremy Bensley, Kirk Roberts, Bryan Rink, and Ying Shi. 2006. Recognizing Textual Entailment with LCC's Groundhog System. In *Proceedings of the Second PASCAL Challenges Workshop*.
- David Lewis. 1988. Relevant Implication. *Theoria*, 54(3):161–174.
- Bernardo Magnini, Matteo Negri, Roberto Prevete, and Hristo Tanev. 2002. Is it the right answer? exploiting web redundancy for answer validation. In *Proceedings of the Fortieth Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA.
- Dan Moldovan, Marius Pasca, Sanda Harabagiu, and Mihai Surdeanu. 2002. Performance Issues and Error Analysis in an Open-Domain Question Answering System. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics*.
- Dan Moldovan, Christine Clark, Sanda Harabagiu, and Steve Maiorano. 2003. COGEX: A Logic Prover for Question Answering. In *Proceedings of HLT/NAACL-2003*.
- T. Pedersen, S. Patwardhan, and J. Michelizzi. 2004. WordNet::Similarity - Measuring the Relatedness of Concepts. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, San Jose, CA.
- John Prager, Jennifer Chu-Carroll, and Krzysztof Czuba. 2004. Question answering using constraint satisfaction: Qa-by-dossier-with-contraints. In *Proceedings of the ACL-2004*, pages 574–581, Barcelona, Spain, July.

Using String-Kernels for Learning Semantic Parsers

Rohit J. Kate

Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, TX 78712-0233, USA
rjkate@cs.utexas.edu

Raymond J. Mooney

Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, TX 78712-0233, USA
mooney@cs.utexas.edu

Abstract

We present a new approach for mapping natural language sentences to their formal meaning representations using string-kernel-based classifiers. Our system learns these classifiers for every production in the formal language grammar. Meaning representations for novel natural language sentences are obtained by finding the most probable semantic parse using these string classifiers. Our experiments on two real-world data sets show that this approach compares favorably to other existing systems and is particularly robust to noise.

1 Introduction

Computational systems that learn to transform natural language sentences into formal meaning representations have important practical applications in enabling user-friendly natural language communication with computers. However, most of the research in natural language processing (NLP) has been focused on lower-level tasks like syntactic parsing, word-sense disambiguation, information extraction etc. In this paper, we have considered the important task of doing deep semantic parsing to map sentences into their computer-executable meaning representations.

Previous work on learning semantic parsers either employ rule-based algorithms (Tang and Mooney, 2001; Kate et al., 2005), or use statistical feature-based methods (Ge and Mooney, 2005; Zettlemoyer and Collins, 2005; Wong and Mooney, 2006). In this paper, we present a novel kernel-based statistical method for learning semantic parsers. Kernel methods (Cristianini and Shawe-Taylor, 2000) are particularly suitable

for semantic parsing because it involves mapping phrases of natural language (NL) sentences to semantic concepts in a meaning representation language (MRL). Given that natural languages are so flexible, there are various ways in which one can express the same semantic concept. It is difficult for rule-based methods or even statistical feature-based methods to capture the full range of NL contexts which map to a semantic concept because they tend to enumerate these contexts. In contrast, kernel methods allow a convenient mechanism to implicitly work with a potentially infinite number of features which can robustly capture these range of contexts even when the data is noisy.

Our system, KRISP (Kernel-based Robust Interpretation for Semantic Parsing), takes NL sentences paired with their formal meaning representations 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 then estimates the probability of the production covering different substrings of the sentence. This information is used to compositionally build a complete meaning representation (MR) of the sentence.

Some of the previous work on semantic parsing has focused on fairly simple domains, primarily, ATIS (Air Travel Information Service) (Price, 1990) whose semantic analysis is equivalent to filling a single semantic frame (Miller et al., 1996; Popescu et al., 2004). In this paper, we have tested KRISP on two real-world domains in which meaning representations are more complex with richer predicates and nested structures. Our experiments demonstrate that KRISP compares favor-

NL: “If the ball is in our goal area then our player 1 should intercept it.”

CLANG: ((bpos (goal-area our))
(do our {1} intercept))

Figure 1: An example of an NL advice and its CLANG MR.

ably to other existing systems and is particularly robust to noise.

2 Semantic Parsing

We call the process of mapping natural language (NL) utterances into their computer-executable meaning representations (MRs) as *semantic parsing*. These MRs are expressed in formal languages which we call meaning representation languages (MRLs). We assume that all MRLs have deterministic context free grammars, which is true for almost all computer languages. This ensures that every MR will have a unique parse tree. A learning system for semantic parsing is given a training corpus of NL sentences paired with their respective MRs from which it has to induce a semantic parser which can map novel NL sentences to their correct MRs.

Figure 1 shows an example of an NL sentence and its MR from the CLANG domain. CLANG (Chen et al., 2003) is the standard formal coach language in which coaching advice is given to soccer agents which compete on a simulated soccer field in the RoboCup¹ Coach Competition. In the MR of the example, `bpos` stands for “ball position”.

The second domain we have considered is the GEOQUERY domain (Zelle and Mooney, 1996) which is a query language for a small database of about 800 U.S. geographical facts. Figure 2 shows an NL query and its MR form in a functional query language. The parse of the functional query language is also shown along with the involved productions. This example is also used later to illustrate how our system does semantic parsing. The MR in the functional query language can be read as if processing a list which gets modified by various functions. From the innermost expression going outwards it means: the state of Texas, the list containing all the states next to the state of Texas and the list of all the rivers which flow through these states. This list is finally returned as the answer.

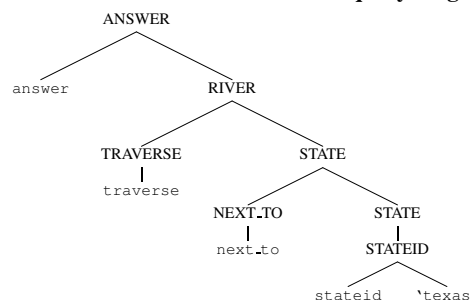
¹<http://www.robocup.org/>

NL: “Which rivers run through the states bordering Texas?”

Functional query language:

`answer(traverse(next_to(stateid('texas'))))`

Parse tree of the MR in functional query language:



Productions:

ANSWER → answer(RIVER)	RIVER → TRAVERSE(STATE)
STATE → NEXT_TO(STATE)	STATE → STATEID
TRAVERSE → traverse	NEXT_TO → next_to
STATEID → stateid('texas')	

Figure 2: An example of an NL query and its MR in a functional query language with its parse tree.

KRISP does semantic parsing using the notion of a *semantic derivation* of an NL sentence. In the following subsections, we define the semantic derivation of an NL sentence and its probability. The task of semantic parsing then is to find the most probable semantic derivation of an NL sentence. In section 3, we describe how KRISP learns the string classifiers that are used to obtain the probabilities needed in finding the most probable semantic derivation.

2.1 Semantic Derivation

We define a *semantic derivation*, D , of an NL sentence, s , as a parse tree of an MR (not necessarily the correct MR) such that each node of the parse tree also contains a substring of the sentence in addition to a production. We denote nodes of the derivation tree by tuples $(\pi, [i..j])$, where π is its production and $[i..j]$ stands for the substring $s[i..j]$ of s (i.e. the substring from the i th word to the j th word), and we say that the node or its production *covers* the substring $s[i..j]$. The substrings covered by the children of a node are not allowed to overlap, and the substring covered by the parent must be the concatenation of the substrings covered by its children. Figure 3 shows a semantic derivation of the NL sentence and the MR parse which were shown in figure 2. The words are numbered according to their position in the sentence. Instead of non-terminals, productions are shown in the nodes to emphasize the role of productions in semantic derivations.

Sometimes, the children of an MR parse tree

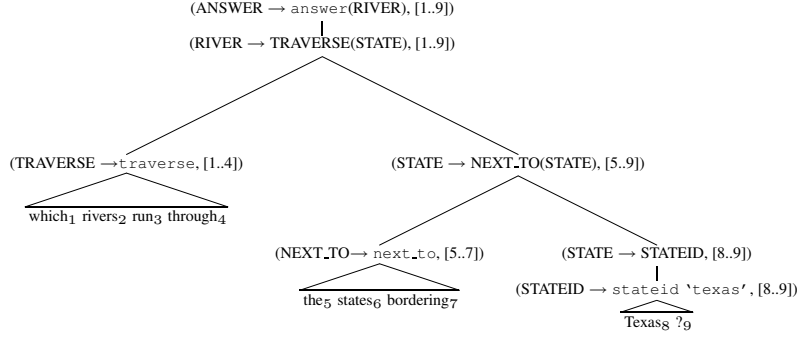


Figure 3: Semantic derivation of the NL sentence “Which rivers run through the states bordering Texas?” which gives MR as `answer(traverse(next_to(stateid(texas))))`.

node may not be in the same order as are the substrings of the sentence they should cover in a semantic derivation. For example, if the sentence was “Through the states that border Texas which rivers run?”, which has the same MR as the sentence in figure 3, then the order of the children of the node “RIVER → TRAVERSE(STATE)” would need to be reversed. To accommodate this, a semantic derivation tree is allowed to contain MR parse tree nodes in which the children have been permuted.

Note that given a semantic derivation of an NL sentence, it is trivial to obtain the corresponding MR simply as the string generated by the parse. Since children nodes may be permuted, this step also needs to permute them back to the way they should be according to the MRL productions. If a semantic derivation gives the correct MR of the NL sentence, then we call it a *correct semantic derivation* otherwise it is an *incorrect semantic derivation*.

2.2 Most Probable Semantic Derivation

Let $P_\pi(u)$ denote the probability that a production π of the MRL grammar covers the NL substring u . In other words, the NL substring u expresses the semantic concept of a production π with probability $P_\pi(u)$. In the next subsection we will describe how KRISP obtains these probabilities using string-kernel based SVM classifiers. Assuming these probabilities are independent of each other, the probability of a semantic derivation D of a sentence s is then:

$$P(D) = \prod_{(\pi, [i..j]) \in D} P_\pi(s[i..j])$$

The task of the semantic parser is to find the most probable derivation of a sentence s . This task can be recursively performed using the notion of a *partial derivation* $E_{n,s[i..j]}$, which stands

for a subtree of a semantic derivation tree with n as the left-hand-side (LHS) non-terminal of the root production and which covers s from index i to j . For example, the subtree rooted at the node “(STATE → NEXT_TO(STATE), [5..9])” in the derivation shown in figure 3 is a partial derivation which would be denoted as $E_{STATE,s[5..9]}$. Note that the derivation D of sentence s is then simply $E_{start,s[1..|s|]}$, where *start* is the start symbol of the MRL’s context free grammar, G .

Our procedure to find the most probable partial derivation $E_{n,s[i..j]}^*$ considers all possible subtrees whose root production has n as its LHS non-terminal and which cover s from index i to j . Mathematically, the most probable partial derivation $E_{n,s[i..j]}^*$ is recursively defined as:

$$E_{n,s[i..j]}^* = \text{makeTree}(\arg \max_{\substack{\pi = n \rightarrow n_1..n_t \in G, \\ (p_1, \dots, p_t) \in \text{partition}(s[i..j], t)}} (P_\pi(s[i..j]) \prod_{k=1..t} P(E_{n_k, p_k}^*)))$$

where $\text{partition}(s[i..j], t)$ is a function which returns the set of all partitions of $s[i..j]$ with t elements including their permutations. A partition of a substring $s[i..j]$ with t elements is a t -tuple containing t non-overlapping substrings of $s[i..j]$ which give $s[i..j]$ when concatenated. For example, (“the states bordering”, “Texas ?”) is a partition of the substring “the states bordering Texas ?” with 2 elements. The procedure $\text{makeTree}(\pi, (p_1, \dots, p_t))$ constructs a partial derivation tree by making π as its root production and making the most probable partial derivation trees found through the recursion as children subtrees which cover the substrings according to the partition $(p_1, \dots, p_t)$.

The most probable partial derivation $E_{n,s[i..j]}^*$ is found using the above equation by trying all productions $\pi = n \rightarrow n_1..n_t$ in G which have

n as the LHS, and all *partitions* with t elements of the substring $s[i..j]$ (n_1 to n_t are right-hand-side (RHS) non-terminals of π , terminals do not play any role in this process and are not shown for simplicity). The most probable partial derivation $E_{STATE, s[5..9]}^*$ for the sentence shown in figure 3 will be found by trying all the productions in the grammar with STATE as the LHS, for example, one of them being “STATE $\rightarrow$ NEXT_TO STATE”. Then for this sample production, all partitions, (p_1, p_2) , of the substring $s[5..9]$ with two elements will be considered, and the most probable derivations E_{NEXT_TO, p_1}^* and E_{STATE, p_2}^* will be found recursively. The recursion reaches base cases when the productions which have n on the LHS do not have any non-terminal on the RHS or when the substring $s[i..j]$ becomes smaller than the length t .

According to the equation, a production $\pi \in G$ and a partition $(p_1, \dots, p_t) \in \text{partition}(s[i..j], t)$ will be selected in constructing the most probable partial derivation. These will be the ones which maximize the product of the probability of π covering the substring $s[i..j]$ with the product of probabilities of all the recursively found most probable partial derivations consistent with the partition $(p_1, \dots, p_t)$.

A naive implementation of the above recursion is computationally expensive, but by suitably extending the well known Earley’s context-free parsing algorithm (Earley, 1970), it can be implemented efficiently. The above task has some resemblance to probabilistic context-free grammar (PCFG) parsing for which efficient algorithms are available (Stolcke, 1995), but we note that our task of finding the most probable semantic derivation differs from PCFG parsing in two important ways. First, the probability of a production is not independent of the sentence but depends on which substring of the sentence it covers, and second, the leaves of the tree are not individual terminals of the grammar but are substrings of words of the NL sentence. The extensions needed for Earley’s algorithm are straightforward and are described in detail in (Kate, 2005) but due to space limitation we do not describe them here. Our extended Earley’s algorithm does a beam search and attempts to find the ω (a parameter) most probable semantic derivations of an NL sentence s using the probabilities $P_\pi(s[i..j])$. To make this search faster, it uses a threshold, θ , to prune low probability derivation trees.

3 KRISP’s Training Algorithm

In this section, we describe how KRISP learns the classifiers which give the probabilities $P_\pi(u)$ needed for semantic parsing as described in the previous section. Given the training corpus of NL sentences paired with their MRs $\{(s_i, m_i) | i = 1..N\}$, KRISP first parses the MRs using the MRL grammar, G . We represent the parse of MR, m_i , by $\text{parse}(m_i)$.

Figure 4 shows pseudo-code for KRISP’s training algorithm. KRISP learns a semantic parser iteratively, each iteration improving upon the parser learned in the previous iteration. In each iteration, for every production π of G , KRISP collects positive and negative example sets. In the first iteration, the set $\mathcal{P}(\pi)$ of positive examples for production π contains all sentences, s_i , such that $\text{parse}(m_i)$ uses the production π . The set of negative examples, $\mathcal{N}(\pi)$, for production π includes all of the remaining training sentences. Using these positive and negative examples, an SVM classifier², C_π , is trained for each production π using a normalized string subsequence kernel. Following the framework of Lodhi et al. (2002), we define a kernel between two strings as the number of common subsequences they share. One difference, however, is that their strings are over characters while our strings are over words. The more the two strings share, the greater the similarity score will be.

Normally, SVM classifiers only predict the class of the test example but one can obtain class probability estimates by mapping the distance of the example from the SVM’s separating hyperplane to the range $[0,1]$ using a learned sigmoid function (Platt, 1999). The classifier C_π then gives us the probabilities $P_\pi(u)$. We represent the set of these classifiers by $\mathcal{C} = \{C_\pi | \pi \in G\}$.

Next, using these classifiers, the extended Earley’s algorithm, which we call EXTENDED_EARLEY in the pseudo-code, is invoked to obtain the ω best semantic derivations for each of the training sentences. The procedure *getMR* returns the MR for a semantic derivation. At this point, for many training sentences, the resulting most-probable semantic derivation may not give the correct MR. Hence, next, the system collects more refined positive and negative examples to improve the result in the next iteration. It

²We use the LIBSVM package available at: <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

```

function TRAIN_KRISP(training corpus  $\{(s_i, m_i) | i = 1..N\}$ , MRL grammar  $G$ )
for each  $\pi \in G$  // collect positive and negative examples for the first iteration
  for  $i = 1$  to  $N$  do
    if  $\pi$  is used in  $parse(m_i)$  then
      include  $s_i$  in  $\mathcal{P}(\pi)$ 
    else include  $s_i$  in  $\mathcal{N}(\pi)$ 
  for iteration = 1 to  $MAX\_ITR$  do
    for each  $\pi \in G$  do
       $C_\pi = trainSVM(\mathcal{P}(\pi), \mathcal{N}(\pi))$  // SVM training
    for each  $\pi \in G$   $\mathcal{P}(\pi) = \Phi$  // empty the positive examples, accumulate negatives though
    for  $i = 1$  to  $N$  do
       $D = EXTENDED\_EARLEY(s_i, G, P)$  // obtain best derivations
      if  $\nexists d \in D$  such that  $parse(m_i) = getMR(d)$  then
         $D = D \cup EXTENDED\_EARLEY\_CORRECT(s_i, G, P, m_i)$  // if no correct derivation then force to find one
       $d^* = \arg \max_{d \in D \& getMR(d) = parse(m_i)} P(d)$ 
      COLLECT_POSITIVES( $d^*$ ) // collect positives from maximum probability correct derivation
      for each  $d \in D$  do
        if  $P(d) > P(d^*)$  and  $getMR(d) \neq parse(m_i)$  then
          // collect negatives from incorrect derivation with larger probability than the correct one
          COLLECT_NEGATIVES( $d, d^*$ )
    return classifiers  $\mathcal{C} = \{C_\pi | \pi \in G\}$ 

```

Figure 4: KRISP’s training algorithm

is also possible that for some sentences, none of the obtained ω derivations give the correct MR. But as will be described shortly, the most probable derivation which gives the correct MR is needed to collect positive and negative examples for the next iteration. Hence in these cases, a version of the extended Earley’s algorithm, `EXTENDED_EARLEY_CORRECT`, is invoked which also takes the correct MR as an argument and returns the best ω derivations it finds, all of which give the correct MR. This is easily done by making sure all subtrees derived in the process are present in the parse of the correct MR.

From these derivations, positive and negative examples are collected for the next iteration. Positive examples are collected from the most probable derivation which gives the correct MR, figure 3 showed an example of a derivation which gives the correct MR. At each node in such a derivation, the substring covered is taken as a positive example for its production. Negative examples are collected from those derivations whose probability is higher than the most probable correct derivation but which do not give the correct MR. Figure 5 shows an example of an incorrect derivation. Here the function “`next_to`” is missing from the MR it produces. The following procedure is used to collect negative examples from incorrect derivations. The incorrect derivation and the most probable correct derivation are traversed simultaneously starting from the root using breadth-first traversal. The first nodes where their productions differ is detected, and all of the words covered by these nodes (in both derivations) are marked. In the correct and incorrect derivations shown in figures 3 and 5 respec-

tively, the first nodes where the productions differ are “(STATE $\rightarrow$ NEXT_TO(STATE), [5..9])” and “(STATE $\rightarrow$ STATEID, [8..9])”. Hence, the union of words covered by them: 5 to 9 (“*the states bordering Texas?*”), will be marked. For each of these marked words, the procedure considers all of the productions which cover it in the two derivations. The nodes of the productions which cover a marked word in the incorrect derivation but not in the correct derivation are used to collect negative examples. In the example, the node “(TRAVERSE $\rightarrow$ traverse, [1..7])” will be used to collect a negative example (i.e. the words 1 to 7 “*which rivers run through the states bordering*” will be a negative example for the production TRAVERSE $\rightarrow$ traverse) because the production covers the marked words “*the*”, “*states*” and “*bordering*” in the incorrect derivation but not in the correct derivation. With this as a negative example, hopefully in the next iteration, the probability of this derivation will decrease significantly and drop below the probability of the correct derivation.

In each iteration, the positive examples from the previous iteration are first removed so that new positive examples which lead to better correct derivations can take their place. However, negative examples are accumulated across iterations for better accuracy because negative examples from each iteration only lead to incorrect derivations and it is always good to include them. To further increase the number of negative examples, every positive example for a production is also included as a negative example for all the other productions having the same LHS. After a specified number of `MAX_ITR` iterations,

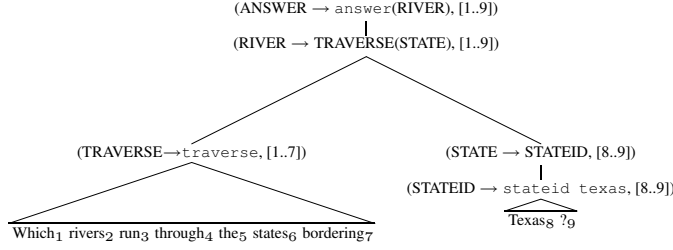


Figure 5: An incorrect semantic derivation of the NL sentence “Which rivers run through the states bordering Texas?” which gives the incorrect MR `answer(traverse(stateid(texas)))`.

the trained classifiers from the last iteration are returned. Testing involves using these classifiers to generate the most probable derivation of a test sentence as described in the subsection 2.2, and returning its MR.

The MRL grammar may contain productions corresponding to constants of the domain, for e.g., state names like “STATEID → ‘texas’”, or river names like “RIVERID → ‘colorado’” etc. Our system allows the user to specify such productions as *constant productions* giving the NL substrings, called *constant substrings*, which directly relate to them. For example, the user may give “Texas” as the constant substring for the production “STATEID → ‘texas’”. Then KRISP does not learn classifiers for these constant productions and instead decides if they cover a substring of the sentence or not by matching it with the provided constant substrings.

4 Experiments

4.1 Methodology

KRISP was evaluated on CLANG and GEOQUERY domains as described in section 2. The CLANG corpus was built by randomly selecting 300 pieces of coaching advice from the log files of the 2003 RoboCup Coach Competition. These formal advice instructions were manually translated into English (Kate et al., 2005). The GEOQUERY corpus contains 880 English queries collected from undergraduates and from real users of a web-based interface (Tang and Mooney, 2001). These were manually translated into their MRs. The average length of an NL sentence in the CLANG corpus is 22.52 words while in the GEOQUERY corpus it is 7.48 words, which indicates that CLANG is the harder corpus. The average length of the MRs is 13.42 tokens in the CLANG corpus while it is 6.46 tokens in the GEOQUERY corpus.

KRISP was evaluated using standard 10-fold cross validation. For every test sentence, only the

best MR corresponding to the most probable semantic derivation is considered for evaluation, and its probability is taken as the system’s confidence in that MR. Since KRISP uses a threshold, θ , to prune low probability derivation trees, it sometimes may fail to return any MR for a test sentence. We computed the number of test sentences for which KRISP produced MRs, and the number of these MRs that were correct. For CLANG, an output MR is considered correct if and only if it exactly matches the correct MR. For GEOQUERY, 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. 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).

In our experiments, the threshold θ was fixed to 0.05 and the beam size ω was 20. These parameters were found through pilot experiments. The maximum number of iterations (MAX_ITR) required was only 3, beyond this we found that the system only overfits the training corpus.

We compared our system’s performance with the following existing systems: the string and tree versions of SILT (Kate et al., 2005), a system that learns transformation rules relating NL phrases to MRL expressions; WASP (Wong and Mooney, 2006), a system that learns transformation rules using statistical machine translation techniques; SCISSOR (Ge and Mooney, 2005), a system that learns an integrated syntactic-semantic parser; and CHILL (Tang and Mooney, 2001) an ILP-based semantic parser. We also compared with the CCG-based semantic parser by Zettlemoyer et al. (2005), but their results are available only for the GEO880 corpus and their experimental set-up is also different from ours. Like KRISP, WASP and SCISSOR also give confidences to the MRs they generate which are used to plot precision-recall curves by measuring precisions and recalls at vari-

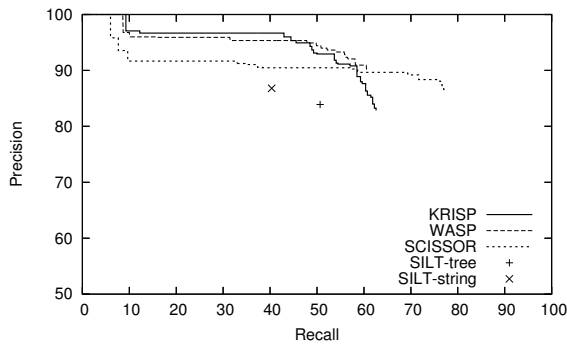


Figure 6: Results on the CLANG corpus.

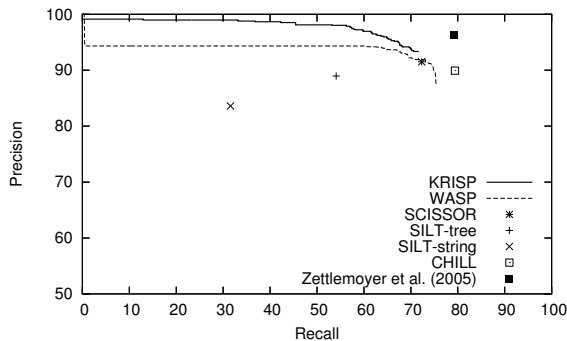


Figure 7: Results on the GEOQUERY corpus.

ous confidence levels. The results of the other systems are shown as points on the precision-recall graph.

4.2 Results

Figure 6 shows the results on the CLANG corpus. KRISP performs better than either version of SILT and performs comparable to WASP. Although SCISSOR gives less precision at lower recall values, it gives much higher maximum recall. However, we note that SCISSOR requires more supervision for the training corpus in the form of semantically annotated syntactic parse trees for the training sentences. CHILL could not be run beyond 160 training examples because its Prolog implementation runs out of memory. For 160 training examples it gave 49.2% precision with 12.67% recall.

Figure 7 shows the results on the GEOQUERY corpus. KRISP achieves higher precisions than WASP on this corpus. Overall, the results show that KRISP performs better than deterministic rule-based semantic parsers like CHILL and SILT and performs comparable to other statistical semantic parsers like WASP and SCISSOR.

4.3 Experiments with Other Natural Languages

We have translations of a subset of the GEOQUERY corpus with 250 examples (GEO250 corpus) in

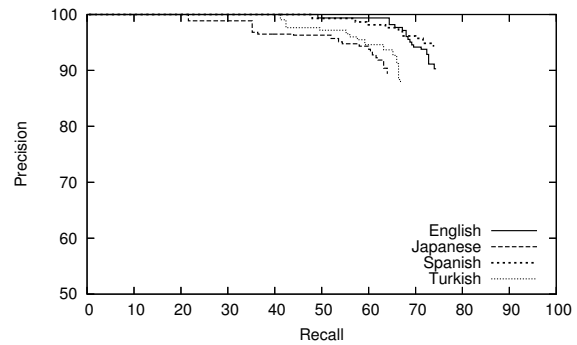


Figure 8: Results of KRISP on the GEO250 corpus for different natural languages.

three other natural languages: Spanish, Turkish and Japanese. Since KRISP's learning algorithm does not use any natural language specific knowledge, it is directly applicable to other natural languages. Figure 8 shows that KRISP performs competently on other languages as well.

4.4 Experiments with Noisy NL Sentences

Any real world application in which semantic parsers would be used to interpret natural language of a user is likely to face noise in the input. If the user is interacting through spontaneous speech and the input to the semantic parser is coming from the output of a speech recognition system then there are many ways in which noise could creep in the NL sentences: interjections (like um's and ah's), environment noise (like door slams, phone rings etc.), out-of-domain words, grammatically ill-formed utterances etc. (Zue and Glass, 2000). As opposed to the other systems, KRISP's string-kernel-based semantic parsing does not use hard-matching rules and should be thus more flexible and robust to noise. We tested this hypothesis by running experiments on data which was artificially corrupted with simulated speech recognition errors.

The interjections, environment noise etc. are likely to be recognized as real words by a speech recognizer. To simulate this, after every word in a sentence, with some probability P_{add} , an extra word is added which is chosen with probability proportional to its word frequency found in the British National Corpus (BNC), a good representative sample of English. A speech recognizer may sometimes completely fail to detect words, so with a probability of P_{drop} a word is sometimes dropped. A speech recognizer could also introduce noise by confusing a word with a high frequency phonetically close word. We sim-

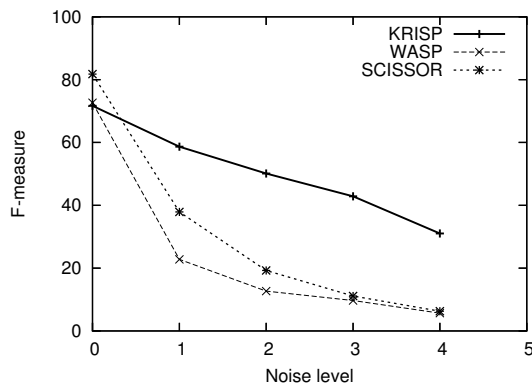


Figure 9: Results on the CLANG corpus with increasing amounts of noise in the test sentences.

ulate this type of noise by substituting a word in the corpus by another word, w , with probability $p^{ed(w)} * P(w)$, where p is a parameter, $ed(w)$ is w 's edit distance (Levenshtein, 1966) from the original word and $P(w)$ is w 's probability proportional to its word frequency. The edit distance which calculates closeness between words is character-based rather than based on phonetics, but this should not make a significant difference in the experimental results.

Figure 9 shows the results on the CLANG corpus with increasing amounts of noise, from level 0 to level 4. The noise level 0 corresponds to no noise. The noise parameters, P_{add} and P_{drop} , were varied uniformly from being 0 at level 0 and 0.1 at level 4, and the parameter p was varied uniformly from being 0 at level 0 and 0.01 at level 4. We are showing the best F-measure (harmonic mean of precision and recall) for each system at different noise levels. As can be seen, KRISP's performance degrades gracefully in the presence of noise while other systems' performance degrade much faster, thus verifying our hypothesis. In this experiment, only the test sentences were corrupted, we get qualitatively similar results when both training and test sentences are corrupted. The results are also similar on the GEOQUERY corpus.

5 Conclusions

We presented a new kernel-based approach to learn semantic parsers. SVM classifiers based on string subsequence kernels are trained for each of the productions in the meaning representation language. These classifiers are then used to compositionally build complete meaning representations of natural language sentences. We evaluated our system on two real-world corpora. The results showed that our system compares favorably

to other existing systems and is particularly robust to noise.

Acknowledgments

This research was supported by Defense Advanced Research Projects Agency under grant HR0011-04-1-0007.

References

- Mao Chen et al. 2003. Users manual: RoboCup soccer server manual for soccer server version 7.07 and later. Available at <http://sourceforge.net/projects/sserver/>.
- Nello Cristianini and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.
- Jay Earley. 1970. An efficient context-free parsing algorithm. *Communications of the Association for Computing Machinery*, 6(8):451–455.
- R. Ge and R. J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proc. of 9th Conf. on Computational Natural Language Learning (CoNLL-2005)*, pages 9–16, Ann Arbor, MI, July.
- R. J. Kate, Y. W. Wong, and R. J. Mooney. 2005. Learning to transform natural to formal languages. In *Proc. of 20th Natl. Conf. on Artificial Intelligence (AAAI-2005)*, pages 1062–1068, Pittsburgh, PA, July.
- Rohit J. Kate. 2005. A kernel-based approach to learning semantic parsers. Technical Report UT-AI-05-326, Artificial Intelligence Lab, University of Texas at Austin, Austin, TX, November.
- V. I. Levenshtein. 1966. Binary codes capable of correcting insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, February.
- Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444.
- Scott Miller, David Stallard, Robert Bobrow, and Richard Schwartz. 1996. A fully statistical approach to natural language interfaces. In *Proc. of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-96)*, pages 55–61, Santa Cruz, CA.
- John C. Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Alexander J. Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 185–208. MIT Press.
- Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates. 2004. Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In *Proc. of 20th Intl. Conf. on Computational Linguistics (COLING-04)*, Geneva, Switzerland, August.
- Patti J. Price. 1990. Evaluation of spoken language systems: The ATIS domain. In *Proc. of 3rd DARPA Speech and Natural Language Workshop*, pages 91–95, June.
- Andreas Stolcke. 1995. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2):165–201.
- L. R. Tang and R. J. Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proc. of the 12th European Conf. on Machine Learning*, pages 466–477, Freiburg, Germany.
- Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proc. of Human Language Technology Conf. / North American Association for Computational Linguistics Annual Meeting (HLT/NAACL-2006)*, New York City, NY. To appear.
- John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proc. of 13th Natl. Conf. on Artificial Intelligence (AAAI-96)*, pages 1050–1055, Portland, OR, August.
- Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorical grammars. In *Proc. of 21th Conf. on Uncertainty in Artificial Intelligence (UAI-2005)*, Edinburgh, Scotland, July.
- Victor W. Zue and James R. Glass. 2000. Conversational interfaces: Advances and challenges. In *Proc. of the IEEE*, volume 88(8), pages 1166–1180.

A Bootstrapping Approach to Unsupervised Detection of Cue Phrase Variants

Rashid M. Abdalla and Simone Teufel

Computer Laboratory, University of Cambridge
15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
rma33@cam.ac.uk, sht25@cam.ac.uk

Abstract

We investigate the unsupervised detection of semi-fixed cue phrases such as “*This paper proposes a novel approach...*”¹ from unseen text, on the basis of only a handful of seed cue phrases with the desired semantics. The problem, in contrast to bootstrapping approaches for Question Answering and Information Extraction, is that it is hard to find a constraining context for occurrences of semi-fixed cue phrases. Our method uses components of the cue phrase itself, rather than external context, to bootstrap. It successfully excludes phrases which are different from the target semantics, but which look superficially similar. The method achieves 88% accuracy, outperforming standard bootstrapping approaches.

1 Introduction

Cue phrases such as “*This paper proposes a novel approach to...*”, “*no method for... exists*” or even “*you will hear from my lawyer*” are semi-fixed in that they constitute a formulaic pattern with a clear semantics, but with syntactic and lexical variations which are hard to predict and thus hard to detect in unseen text (e.g. “*a new algorithm for... is suggested in the current paper*” or “*I envisage legal action*”). In scientific discourse, such *meta-discourse* (Myers, 1992; Hyland, 1998) abounds and plays an important role in marking the discourse structure of the texts.

Finding these variants can be useful for many text understanding tasks because semi-fixed cue phrases act as linguistic markers indicating the importance and/or the rhetorical role of some adjacent text. For the summarisation of scientific

papers, cue phrases such as “*Our paper deals with...*” are commonly used as indicators of extraction-worthiness of sentences (Kupiec et al., 1995). Re-generative (rather than extractive) summarisation methods may want to go further than that and directly use the knowledge that a certain sentence contains the particular research aim of a paper, or a claimed gap in the literature. Similarly, in the task of automatic routing of customer emails and automatic answering of some of these, the detection of threats of legal action could be useful.

However, systems that use cue phrases usually rely on manually compiled lists, the acquisition of which is time-consuming and error-prone and results in cue phrases which are genre-specific. Methods for finding cue phrases automatically include Hovy and Lin (1998) (using the ratio of word frequency counts in summaries and their corresponding texts), Teufel (1998) (using the most frequent n-grams), and Paice (1981) (using a pattern matching grammar and a lexicon of manually collected equivalence classes). The main issue with string-based pattern matching techniques is that they cannot capture syntactic generalisations such as active/passive constructions, different tenses and modification by adverbial, adjectival or prepositional phrases, appositions and other parenthetical material.

For instance, we may be looking for sentences expressing the goal or main contribution of a paper; Fig. 1 shows candidates of such sentences. Cases a)–e), which do indeed describe the authors’ goal, display a wide range of syntactic variation.

- | |
|---|
| a) In this paper, we introduce a method for similarity-based estimation of... |
| b) We introduce and justify a method ... |
| c) A method (described in section 1) is introduced |
| d) The method introduced here is a variation... |
| e) We wanted to introduce a method ... |
| f) We do not introduce a method ... |
| g) We introduce and adopt the method given in [1]... |
| h) Previously we introduced a similar method ... |
| i) They introduce a similar method ... |

Figure 1: Goal statements and syntactic variation – correct matches (a–e) and incorrect matches (f–i)

¹In contrast to standard work in discourse linguistics, which mostly considers sentence connectives and adverbials as cue phrases, our definition includes longer phrases, sometimes even entire sentences.

Cases f)–i) in contrast are false matches: they do not express the authors’ goals, although they are superficially similar to the correct contexts. While string-based approaches (Paice, 1981; Teufel, 1998) are too restrictive to cover the wide variation within the correct contexts, bag-of-words approaches such as Agichtein and Gravano’s (2000) are too permissive and would miss many of the distinctions between correct and incorrect contexts.

Lisacek et al. (2005) address the task of identifying “paradigm shift” sentences in the biomedical literature, i.e. statements of thwarted expectation. This task is somewhat similar to ours in its definition by rhetorical context. Their method goes beyond string-based matching: In order for a sentence to qualify, the right set of concepts must be present in a sentence, with any syntactic relationship holding between them. Each concept set is encoded as a fixed, manually compiled lists of strings. Their method covers only one particular context (the paradigm shift one), whereas we are looking for a method where many types of cue phrases can be acquired. Whereas it relies on manually assembled lists, we advocate data-driven acquisition of new contexts. This is generally preferable to manual definition, as language use is changing, inventive and hard to predict and as many of the relevant concepts in a domain may be infrequent (cf. the formulation “*be cursed*”, which was used in our corpus as a way of describing a method’s problems). It also allows the acquisition of cue phrases in new domains, where the exact prevalent meta-discourse might not be known.

Riloff’s (1993) method for learning information extraction (IE) patterns uses a syntactic parse and correspondences between the text and filled MUC-style templates to learn context in terms of lexico-semantic patterns. However, it too requires substantial hand-crafted knowledge: 1500 filled templates as training material, and a lexicon of semantic features for roughly 3000 nouns for constraint checking. Unsupervised methods for similar tasks include Agichtein and Gravano’s (2000) work, which shows that clusters of vector-space-based patterns can be successfully employed to detect specific IE relationships (companies and their headquarters), and Ravichandran and Hovy’s (2002) algorithm for finding patterns for a Question Answering (QA) task. Based on training material in the shape of pairs of question and answer terms – e.g., (e.g. {*Mozart*, 1756}), they learn the

- | |
|--|
| a) In this paper, we introduce a method for similarity-based estimation of...
b) Here, we present a similarity-based approach for estimation of...
c) In this paper, we propose an algorithm which is...
d) We will here define a technique for similarity-based... |
|--|

Figure 2: Context around cue phrases (lexical variants)

semantics holding between these terms (“birth year”) via frequent string patterns occurring in the context, such as “*A was born in B*”, by considering n-grams of all repeated substrings. What is common to these three works is that bootstrapping relies on constraints between the context *external* to the extracted material and the extracted material itself, and that the target extraction material is defined by real-world relations.

Our task differs in that the cue phrases we extract are based on general rhetorical relations holding in all scientific discourse. Our approach for finding semantically similar variants in an unsupervised fashion relies on bootstrapping of seeds from *within the cue phrase*. The assumption is that every semi-fixed cue phrase contains at least two main concepts whose syntax and semantics mutually constrain each other (e.g. verb and direct object in phrases such as “(we) present an approach for”). The expanded cue phrases are recognised in various syntactic contexts using a parser². General semantic constraints valid for groups of semantically similar cue phrases are then applied to model, e.g., the fact that it must be the authors who present the method, not somebody else.

We demonstrate that such an approach is more appropriate for our task than IE/QA bootstrapping mechanisms based on cue phrase-external context. Part of the reason for why normal bootstrapping does not work for our phrases is the difficulty of finding negatives contexts, essential in bootstrapping to evaluate the quality of the patterns automatically. IE and QA approaches, due to uniqueness assumptions of the real-world relations that these methods search for, have an automatic definition of *negative* contexts by hard constraints (i.e., all contexts involving Mozart and any other year are by definition of the wrong semantics; so are all contexts involving Microsoft and a city other than Redmond). As our task is not grounded in real-world relations but in rhetorical ones, constraints found in the context tend to be

²Thus, our task shows some parallels to work in paraphrasing (Barzilay and Lee, 2002) and syntactic variant generation (Jacquemin et al., 1997), but the methods are very different.

soft rather than hard (cf. Fig 2): while it is possible that strings such as “we” and “in this paper” occur more often in the context of a given cue phrase, they also occur in many other places in the paper where the cue phrase is not present. Thus, it is hard to define clear negative contexts for our task.

The novelty of our work is thus the new pattern extraction task (finding variants of semi-fixed cue phrases), a task for which it is hard to directly use the context the patterns appear in, and an iterative unsupervised bootstrapping algorithm for lexical variants, using phrase-internal seeds and ranking similar candidates based on relation strength between the seeds.

While our method is applicable to general cue phrases, we demonstrate it here with transitive verb–direct object pairs, namely a) cue phrases introducing a new methodology (and thus the main research goal of the scientific article; e.g. “In this paper, we propose a novel algorithm...” – we call those *goal-type* cue phrases; and b) cue phrases indicating continuation of previous other research (e.g. “Therefore, we adopt the approach presented in [1]...” – *continuation-type* cue phrases).

2 Lexical Bootstrapping Algorithm

The task of this module is to find lexical variants of the components of the seed cue phrases. Given the seed phrases “we introduce a method” and “we propose a model”, the algorithm starts by finding all direct objects of “introduce” in a given corpus and, using an appropriate similarity measure, ranks them according to their distributional similarity to the nouns “method” and “model”. Subsequently, the noun “method” is used to find transitive verbs and rank them according to their similarity to “introduce” and “propose”. In both cases, the ranking step retains variants that preserve the semantics of the cue phrase (e.g. “develop” and “approach”) and filters irrelevant terms that change the phrase semantics (e.g. “need” and “example”).

Stopping at this point would limit us to those terms that co-occur with the seed words in the training corpus. Therefore additional iterations using automatically generated verbs and nouns are applied in order to recover more and more variants. The full algorithm is given in Fig. 3.

The algorithm requires corpus data for the steps Hypothesize (producing a list of potential candidates) and Rank (testing them for similarity). We

Input: Tuples $\{A_1, A_2, \dots, A_m\}$ and $\{B_1, B_2, \dots, B_n\}$.
Initialisation: Set the concept-A reference set to $\{A_1, A_2, \dots, A_m\}$ and the concept-B reference set to $\{B_1, B_2, \dots, B_n\}$. Set the concept-A active element to A_1 and the concept-B active element to B_1 .

Recursion:

1. Concept B retrieval:

(i) **Hypothesize:** Find terms in the corpus which are in the desired relationship with the concept-A active element (e.g. direct objects of a verb active element). This results in the concept-B candidate set.

(ii) **Rank:** Rank the concept-B candidate set using a suitable ranking methodology that may make use of the concept-B reference set. In this process, each member of the candidate set is assigned a score.

(iii) **Accumulate:** Add the top s items of the concept-B candidate set to the concept-B accumulator list (based on empirical results, s is the rank of the candidate set during the initial iteration and 50 for the remaining iterations). If an item is already on the accumulator list, add its ranking score to the existing item’s score.

2. Concept A retrieval: as above, with concepts A and B swapped.

3. Updating active elements:

(i) Set the concept-B active element to the highest ranked instance in the concept-B accumulator list which has not been used as an active element before.

(ii) Set the concept-A active element to the highest ranked instance in the concept-A accumulator list which has not been used as an active element before.

Repeat steps 1-3 for k iterations

Output: top M words of concept-A (verb) accumulator list and top N words of concept-B (noun) accumulator list

Reference set: a set of seed words which define the collective semantics of the concept we are looking for in this iteration

Active element: the instance of the concept used in the current iteration for retrieving instances of the other concept. If we are finding lexical variants of Concept A by exploiting relationships between Concepts A and B, then the active element is from Concept B.

Candidate set: the set of candidate terms for one concept (eg. Concept A) obtained using an active element from the other concept (eg. Concept B). The more semantically similar a term in the candidate set is to the members of the reference set, the higher its ranking should be. This set contains verbs if the active element is a noun and vice versa.

Accumulator list: a sorted list that accumulates the ranked members of the candidate set.

Figure 3: Lexical variant bootstrapping algorithm

estimate frequencies for the Rank step from the written portion of the British National Corpus (BNC, Burnard (1995)), 90 Million words. For the Hypothesize step, we experiment with two data sets: First, the scientific subsection of the BNC (24 Million words), which we parse using RASP (Briscoe and Carroll, 2002); we then examine the grammatical relations (GRs) for transitive verb constructions, both in active and passive voice. This method guarantees that we find almost all transitive verb constructions cleanly; Carroll et al. (1999) report an accuracy of .85 for

DOs, Active: "AGENT_STRING AUX active-verb-element DETERMINER * POSTMOD"
DOs, Passive: "DETERMINER * AUX active-verb-element element"
TVs, Active: "AGENT_STRING AUX * DETERMINER active-noun- element POSTMOD"
TVs, Passive: "DET active-noun-element AUX * POSTMOD"

Figure 4: Query patterns for retrieving direct objects (DOs) and transitive verbs (TVs) in the Hypothesize step.

newspaper articles for this relation. Second, in order to obtain larger coverage and more current data we also experiment with Google Scholar³, an automatic web-based indexer of scientific literature (mainly peer-reviewed papers, technical reports, books, pre-prints and abstracts). Google Scholar snippets are often incomplete fragments which cannot be parsed. For practical reasons, we decided against processing the entire documents, and obtain an approximation to direct objects and transitive verbs with regular expressions over the result snippets in both active and passive voice (cf. Fig. 4), designed to be high-precision⁴. The amount of data available from BNC and Google Scholar is not directly comparable: harvesting Google Scholar snippets for both active and passive constructions gives around 2000 sentences per seed (Google Scholar returns up to 1000 results per query), while the number of BNC sentences containing seed words in active and passive form varies from 11 (“*formalism*”) to 5467 (“*develop*”) with an average of 1361 sentences for the experimental seed pairs.

Ranking

Having obtained our candidate sets (either from the scientific subsection of the BNC or from Google Scholar), the members are ranked using BNC frequencies. We investigate two ranking methodologies: frequency-based and context-based. Frequency-based ranking simply ranks each member of the candidate set by how many times it is retrieved together with the current active element. Context-based ranking uses a similarity measure for computing the scores, giving a higher score to those words that share sufficiently similar contexts with the members of the reference set. We consider similarity measures in a vector space defined either by a fixed window, by the sentence window, or by syntactic relationships. The score assigned to each word in the candidate set is the sum of its semantic similarity values computed with respect to each member in the reference set.

³<http://scholar.google.com>

⁴The capitalised words in these patterns are replaced by actual words (e.g. AGENT_STRING: We/I, DETERMINER: a/ an/our), and the extracted words (indicated by “*”) are lemmatised.

Syntactic contexts, as opposed to window-based contexts, constrain the context of a word to only those words that are grammatically related to it. We use verb-object relations in both active and passive voice constructions as did Pereira et al. (1993) and Lee (1999), among others. We use the cosine similarity measure for window-based contexts and the following commonly used similarity measures for the syntactic vector space: Hindle’s (1990) measure, the weighted Lin measure (Wu and Zhou, 2003), the α -Skew divergence measure (Lee, 1999), the Jensen-Shannon (JS) divergence measure (Lin, 1991), Jaccard’s coefficient (van Rijsbergen, 1979) and the Confusion probability (Essen and Steinbiss, 1992). The Jensen-Shannon measure $JS(x_1, x_2) = \sum_{y \in Y} \sum_{x \in \{x_1, x_2\}} \left(P(y|x) \log \left(\frac{P(y|x)}{\frac{1}{2}(P(y|x_1) + P(y|x_2))} \right) \right)$ subsequently performed best for our task. We compare the different ranking methodologies and data sets with respect to a manually-defined gold standard list of 20 goal-type verbs and 20 nouns. This list was manually assembled from Teufel (1999); WordNet synonyms and other plausible verbs and nouns found via Web searches on scientific articles were added. We ensured by searches on the ACL anthology that there is good evidence that the gold-standard words indeed occur in the right contexts, i.e. in goal statement sentences. As we want to find similarity metrics and data sources which result in accumulator lists with many of these gold members at high ranks, we need a measure that rewards exactly those lists. We use non-interpolated Mean Average Precision (MAP), a standard measure for evaluating ranked information retrieval runs, which combines precision and recall and ranges from 0 to 1⁵.

We use 8 pairs of 2-tuples as input (e.g. [*introduce*, *study*] & [*approach*, *method*]), randomly selected from the gold standard list. MAP was cal-

⁵ $MAP = \frac{1}{N} \sum_{j=1}^N AP_j = \frac{1}{N} \sum_{j=1}^N \frac{1}{M} \sum_{i=1}^M P(g_i)$ where $P(g_i) = \frac{n_{ij}}{r_{ij}}$ if g_i is retrieved and 0 otherwise, N is the number of seed combinations, M is the size of the golden list, g_i is the i^{th} member of the golden list and r_{ij} is its rank in the retrieved list of combination j while n_{ij} is the number of golden members found up to and including rank r_{ij} .

Ranking scheme	BNC	Google Scholar
Frequency-based	0.123	0.446
Sentence-window	0.200	0.344
Fixedsize-window	0.184	0.342
Hindle	0.293	0.416
Weighted Lin	0.358	0.509
α -Skew	0.361	0.486
Jensen-Shannon	0.404	0.550
Jaccard's coef.	0.301	0.436
Confusion prob.	0.171	0.293

Figure 5: MAPs after the first iteration

culated over the verbs and nouns retrieved using our algorithm and averaged. Fig. 5 summarises the MAP scores for the first iteration, where Google Scholar significantly outperformed the BNC. The best result for this iteration (MAP=.550) was achieved by combining Google Scholar and the Jensen-Shannon measure. The algorithm stops to iterate when no more improvement can be obtained, in this case after 4 iterations, resulting in a final MAP of .619.

Although α -Skew outperforms the simpler measures in ranking nouns, its performance on verbs is worse than the performance of Weighted Lin. While Lee (1999) argues that α -Skew's asymmetry can be advantageous for nouns, this probably does not hold for verbs: verb hierarchies have much shallower structure than noun hierarchies with most verbs concentrated on one level (Miller et al., 1990). This would explain why JS, which is symmetric compared to the α -Skew metric, performed better in our experiments.

In the evaluation presented here we therefore use Google Scholar data and the JS measure. An additional improvement (MAP=.630) is achieved when we incorporate a filter based on the following hypothesis: goal-type verbs should be more likely to have their direct objects preceded by indefinite articles rather than definite articles or possessive determiners (because a new method is introduced) whereas continuation-type verbs should prefer definite articles with their direct objects (as an existing method is involved).

3 Syntactic variants and semantic filters

The syntactic variant extractor takes as its input the raw text and the lists of verbs and nouns generated by the lexical bootstrapper. After RASP-parsing the input text, all instances of the input verbs are located and, based on the grammatical relations output by RASP⁶, a set of relevant en-

⁶The grammatical relations used are nsubj, dobj, iobj, aux, argmod, detmod, ncmmod and mod.

The agent of the verb (e.g., "We adopt. adopted by <i>the author</i> "), the agent's determiner and related adjectives.
The direct object of the verb, the object's determiner and adjectives, in addition to any post-modifiers (e.g., "...apply a method <i>proposed by [1]</i> ...", "...follow an approach of <i>[1]</i> ...")
Auxiliaries of the verb (e.g., "In a similar manner, we <i>may propose</i> a ...")
Adverbial modification of the verb (e.g., "We have <i>previously</i> presented a ...")
Prepositional phrases related to the verb (e.g., " <i>In this paper</i> we present. . .", "... adopted <i>from their work</i> ")

Figure 6: Grammatical relations considered

titles and modifiers for each verb is constructed, grouped into five categories (cf. Fig. 6).

Next, semantic filters are applied to each of the potential candidates (represented by the extracted entities and modifiers), and a fitness score is calculated. These constraints encode semantic principles that will apply to all cue phrases of that rhetorical category. Examples for constraints are: if work is referred to as being done in previous own work, it is probably not a goal statement; the work in a goal statement must be presented *here* or *in the current paper* (the concept of 'here-ness'); and the agents of a goal statement have to be the authors, not other people. While these filters are manually defined, they are modular, encode general principles, and can be combined to express a wide range of rhetorical contexts. We verified that around 20 semantic constraints are enough to cover a large sets of different cue phrases (the 1700 cue phrases from Teufel (1999)), though not all of these are implemented yet.

A nice side-effect of our approach is the simple characterisation of a cue phrase (by a syntactic relationship, some seed words for each concept, and some general, reusable semantic constraints). This characterisation is more informative and specific than string-based approaches, yet it has the potential for generalisation (useful if the cue phrases are ever manually assessed and put into a lexicon).

Fig. 7 shows successful extraction examples from our corpus⁷, illustrating the difficulty of the task: the system correctly identified sentences with syntactically complex goal-type and continuation-type cue phrases, and correctly rejected deceptive variants⁸.

⁷Numbers after examples give CmpLg archive numbers, followed by sentence numbers according to our preprocessing.

⁸The seeds in this example were [*analyse, present*] & [*architecture, method*] (for goal) and [*improve, adopt*] & [*model, method*] (for continuation).

Correctly found:	
Goal-type:	<i>What we aim in this paper is to propose a paradigm that enables partial/local generation through de-compositions and reorganizations of tentative local structures.</i> (9411021, S-5)
Continuation-type:	<i>In this paper we have discussed how the lexicographical concept of lexical functions, introduced by Melcuk to describe collocations, can be used as an interlingual device in the machine translation of such structures.</i> (9410009, S-126)
Correctly rejected:	
Goal-type:	<i>Perhaps the method proposed by Pereira et al. (1993) is the most relevant in our context.</i> (9605014, S-76)
Continuation-type:	<i>Neither Kamp nor Kehler extend their copying/ substitution mechanism to anything besides pronouns, as we have done.</i> (9502014, S-174)

Figure 7: Sentences correctly processed by our system

4 Gold standard evaluation

We evaluated the quality of the extracted phrases in two ways: by comparing our system output to gold standard annotation, and by human judgement of the quality of the returned sentences. In both cases bootstrapping was done using the seed tuples *[analyse, present]* & *[architecture, method]*. For the gold standard-evaluation, we ran our system on a test set of 121 scientific articles drawn from the CmpLg corpus (Teufel, 1999) – entirely different texts from the ones the system was trained on. Documents were manually annotated by the second author for (possibly more than one) goal-type sentence; annotation of that type has been previously shown to be reliable at $K=0.71$ (Teufel, 1999). Our evaluation recorded how often the system’s highest-ranked candidate was indeed a goal-type sentence; as this is a precision-critical task, we do not measure recall here.

We compared our system against our reimplementation of Ravichandran and Hovy’s (2002) paraphrase learning. The seed words were of the form {goal-verb, goal-noun}, and we submitted each of the 4 combinations of the seed pair to Google Scholar. From the top 1000 documents for each query, we harvested 3965 sentences containing both the goal-verb and the goal-noun. By considering all possible substrings, an extensive list of candidate patterns was assembled. Patterns with single occurrences were discarded, leaving a list of 5580 patterns (examples in Fig. 8). In order to rank the patterns by precision, the goal-verbs were submitted as queries and the top 1000 documents were downloaded for each. From these,

we <verb> a <noun> for
of a new <noun> to <verb> the
In this section , we <verb> the <noun> of
the <noun> <verb> in this paper
is to <verb> the <noun> after

Figure 8: Examples of patterns extracted using Ravichandran and Hovy’s (2002) method

Method	Correct sentences
Our system with bootstrapping	88 (73%)
Ravichandran and Hovy (2002)	58 (48%)
Our system, no bootstrapping, WordNet	50 (41%)
Our system, no bootstrapping, seeds only	37 (30%)

Figure 9: Gold standard evaluation: results

the precision of each pattern was calculated by dividing the number of strings matching the pattern instantiated with both the goal-verb and all WordNet synonyms of the goal-noun, by the number of strings matching the patterns instantiated with the goal-verb only. An important point here is that while the tight semantic coupling between the question and answer terms in the original method accurately identifies all the positive and negative examples, we can only approximate this by using a sensible synonym set for the seed goal-nouns. For each document in the test set, the sentence containing the pattern with the highest precision (if any) was extracted as the goal sentence.

We also compared our system to two baselines. We replaced the lists obtained from the lexical bootstrapping module with a) just the seed pair and b) the seed pair and all the WordNet synonyms of the components of the seed pair⁹.

The results of these experiments are given in Fig. 9. All differences are statistically significant with the χ^2 test at $p=0.01$ (except those between Ravichandran/Hovy and our non-bootstrapping/WordNet system). Our bootstrapping system outperforms the Ravichandran and Hovy algorithm by 34%. This is not surprising, because this algorithm was not designed to perform well in tasks where there is no clear negative context. The results also show that bootstrapping outperforms a general thesaurus such as WordNet.

Out of the 33 articles where our system’s favourite was not an annotated goal-type sentence, only 15 are due to bootstrapping errors (i.e., to an incorrect ranking of the lexical variants), corre-

⁹Bootstrapping should in principle do better than a thesaurus, as some of our correctly identified variants are not true synonyms (e.g., *theory* vs. *method*), and as noise through overgeneration of unrelated senses might occur unless automatic word sense disambiguation is performed.

System chose:	but should have chosen:
derive set	compare model
illustrate algorithm	present formalisation
discuss measures	present variations describe modifications propose measures
accommodate material	describe approach
examine material	present study

Figure 10: Wrong bootstrapping decisions

	Ceiling	System	Baseline
Exp. A	3.91	3.08	1.58
Exp.B	4.33	3.67	2.50

Figure 11: Extrinsic evaluation: judges' scores

sponding to a 88% accuracy of the bootstrapping module. Examples from those 15 error cases are given in Fig. 10. The other errors were due to the cue phrase not being a transitive verb–direct object pattern (e.g. *we show that, our goal is* and *we focus on*), so the system could not have found anything (11 cases, or an 80% accuracy), ungrammatical English or syntactic construction too complex, resulting in a lack of RASP detection of the crucial grammatical relation (2) and failure of the semantic filter to catch non-goal contexts (5).

5 Human evaluation

We next perform two human experiments to indirectly evaluate the quality of the automatically generated cue phrase variants. Given an abstract of an article and a sentence extracted from the article, judges are asked to assign a score ranging from 1 (low) to 5 (high) depending on how well the sentence expresses the goal of that article (Exp. A), or the continuation of previous work (Exp. B).

Each experiment involves 24 articles drawn randomly from a subset of 80 articles in the CmpLg corpus that contain manual annotation for goal-type and continuation-type sentences. The experiments use three external judges (graduate students in computational linguistics), and a Latin Square experimental design with three conditions: Baseline (see below), System-generated and Ceiling (extracted from the gold standard annotation used in Teufel (1999)). Judges were not told how the sentences were generated, and no judge saw an item in more than one condition.

The baseline for Experiment A was a random selection of sentences with the highest $TF*IDF$ scores, because goal-type sentences typically contain many content-words. The baseline for experiment B (continuation-type) were randomly selected sentences containing citations, because they

often co-occur with statements of continuation. In both cases, the length of the baseline sentence was controlled for by the average lengths of the gold standard and the system-extracted sentences in the document.

Fig. 11 shows that judges gave an average score of 3.08 to system-extracted sentences in Exp. A, compared with a baseline of 1.58 and a ceiling of 3.91¹⁰; in Exp. B, the system scored 3.67, with a higher baseline of 2.50 and a ceiling of 4.33. According to the Wilcoxon signed-ranks test at $\alpha = .01$, the system is indistinguishable from the gold standard, but significantly different from the baseline, in both experiments. Although this study is on a small scale, it indicates that humans judged sentences obtained with our method as almost equally characteristic of their rhetorical function as human-chosen sentences, and much better than non-trivial baselines.

6 Conclusion

In this paper we have investigated the automatic acquisition of semi-fixed cue phrases as a bootstrapping task which requires very little manual input for each cue phrase and yet generalises to a wide range of syntactic and lexical variants in running text. Our system takes a few seeds of the type of cue phrase as input, and bootstraps lexical variants from a large corpus. It filters out many semantically invalid contexts, and finds cue phrases in various syntactic variants. The system achieved 80% precision of goal-type phrases of the targeted syntactic shape (88% if only the bootstrapping module is evaluated), and good quality ratings from human judges. We found Google Scholar to perform better than BNC as source for finding hypotheses for lexical variants, which may be due to the larger amount of data available to Google Scholar. This seems to outweigh the disadvantage of only being able to use POS patterns with Google Scholar, as opposed to robust parsing with the BNC.

In the experiments reported, we bootstrap only from one type of cue phrase (transitive verbs and direct objects). This type covers a large proportion of the cue phrases needed practically, but our algorithm should in principle work for any kind of semi-fixed cue phrase, as long as they have two core concepts and a syntactic and semantic

¹⁰This score seems somewhat low, considering that these were the best sentences available as goal descriptions, according to the gold standard.

CUE PHRASE: “(previous) methods fail ” (Subj–Verb)
VARIANTS SEED 1: methodology, approach, technique...
VARIANTS SEED 2: be cursed, be incapable of, be restricted to, be troubled, degrade, fall prey to, ...
CUE PHRASE: “ advantage over previous methods ” (NP–PP postmod + adj–noun premod.)
VARIANTS SEED 1: benefit, breakthrough, edge, improvement, innovation, success, triumph...
VARIANTS SEED 2: available, better-known, cited, classic, common, conventional, current, customary, established, existing, extant, ...

Figure 12: Cues with other syntactic relationships

relation between them. Examples for such other types of phrases are given in Fig. 12; the second cue phrase involves a complex syntactic relationship between the two seeds (or possibly it could be considered as a cue phrase with three seeds). We will next investigate if the positive results presented here can be maintained for other syntactic contexts and for cue phrases with more than two seeds.

The syntactic variant extractor could be enhanced in various ways, eg. by resolving anaphora in cue phrases. A more sophisticated model of syntactically weighted vector space (Pado and Lapata, 2003) may help improve the lexical acquisition phase. Another line for future work is bootstrapping meaning across cue phrases within the same rhetorical class, e.g. to learn that *we propose a method for X* and *we aim to do X* are equivalent. As some papers will contain both variants of the cue phrase, with very similar material (X) in the vicinity, they could be used as starting point for experiments to validate cue phrase equivalence.

7 Acknowledgements

This work was funded by the EPSRC projects CITRAZ (GR/S27832/01, “Rhetorical Citation Maps and Domain-independent Argumentative Zoning”) and SCIBORG (EP/C010035/1, “Extracting the Science from Scientific Publications”).

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries*.

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

Ted Briscoe and John Carroll. 2002. Robust accurate statistical annotation of general text. In *Proc. of LREC*.

Lou Burnard, 1995. *Users Reference Guide, British National Corpus Version 1.0*. Oxford University, UK.

John Carroll, Guido Minnen, and Ted Briscoe. 1999. Corpus annotation for parser evaluation. In *Proceedings of Linguistically Interpreted Corpora (LINC-99), EACL-workshop*.

Ute Essen and Volker Steinbiss. 1992. Co-occurrence smoothing for stochastic language modelling. In *Proc. of ICASSP*.

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

Edvard Hovy and Chin-Yew Lin. 1998. Automated text summarization and the Summarist system. In *Proc. of the TIPSTER Text Program*.

Ken Hyland. 1998. Persuasion and context: The pragmatics of academic metadiscourse. *Journal of Pragmatics*, 30(4):437–455.

Christian Jacquemin, Judith Klavans, and Evelyn Tzoukermann. 1997. Expansion of multi-word terms for indexing and retrieval using morphology and syntax. In *Proc. of the ACL*.

Julian Kupiec, Jan O. Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Proc. of SIGIR-95*.

Lillian Lee. 1999. Measures of distributional similarity. In *Proc. of the ACL*.

Jianhua Lin. 1991. Divergence measures based on the Shannon entropy. *IEEE transactions on Information Theory*, 37(1):145–151.

Frederique Lisacek, Christine Chichester, Aaron Kaplan, and Sandor Agnes. 2005. Discovering paradigm shift patterns in biomedical abstracts: Application to neurodegenerative diseases. In *Proc. of the SMMB*.

George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Five papers on WordNet. Technical report, Cognitive Science Laboratory, Princeton University.

Greg Myers. 1992. In this paper we report...—speech acts and scientific facts. *Journal of Pragmatics*, 17(4):295–313.

Sebastian Pado and Mirella Lapata. 2003. Constructing semantic space models from parsed corpora. In *Proc. of ACL*.

Chris D. Paice. 1981. The automatic generation of literary abstracts: an approach based on the identification of self-indicating phrases. In Robert Norman Oddy, Stephen E. Robertson, Cornelis Joost van Rijsbergen, and P. W. Williams, editors, *Information Retrieval Research*, Butterworth, London, UK.

Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of English words. In *Proc. of the ACL*.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proc. of the ACL*.

Ellen Riloff. 1993. Automatically constructing a dictionary for information extraction tasks. In *Proc. of AAAI-93*.

Simone Teufel. 1998. Meta-discourse markers and problem-structuring in scientific articles. In *Proceedings of the ACL-98 Workshop on Discourse Structure and Discourse Markers*.

Simone Teufel. 1999. *Argumentative Zoning: Information Extraction from Scientific Text*. Ph.D. thesis, School of Cognitive Science, University of Edinburgh, UK.

Cornelis Joost van Rijsbergen. 1979. *Information Retrieval*. Butterworth, London, UK, 2nd edition.

Hua Wu and Ming Zhou. 2003. Synonymous collocation extraction using translation information. In *Proc. of the ACL*.

Semantic Role Labeling via FrameNet, VerbNet and PropBank

Ana-Maria Giuglea and Alessandro Moschitti

Department of Computer Science

University of Rome "Tor Vergata"

Rome, Italy

agiuglea@gmail.com

moschitti@info.uniroma2.it

Abstract

This article describes a robust semantic parser that uses a broad knowledge base created by interconnecting three major resources: FrameNet, VerbNet and PropBank. The FrameNet corpus contains the examples annotated with semantic roles whereas the VerbNet lexicon provides the knowledge about the syntactic behavior of the verbs. We connect VerbNet and FrameNet by mapping the FrameNet frames to the VerbNet Intersective Levin classes. The PropBank corpus, which is tightly connected to the VerbNet lexicon, is used to increase the verb coverage and also to test the effectiveness of our approach. The results indicate that our model is an interesting step towards the design of more robust semantic parsers.

1 Introduction

During the last years a noticeable effort has been devoted to the design of lexical resources that can provide the training ground for automatic semantic role labelers. Unfortunately, most of the systems developed until now are confined to the scope of the resource used for training. A very recent example in this sense was provided by the CONLL 2005 shared task (Carreras and Màrquez, 2005) on PropBank (PB) (Kingsbury and Palmer, 2002) role labeling. The systems that participated in the task were trained on the Wall Street Journal corpus (WSJ) and tested on portions of WSJ and Brown corpora. While the best F-measure recorded on WSJ was 80%, on the Brown corpus, the F-measure dropped below 70%. The most significant causes for this performance decay

were highly ambiguous and unseen predicates (i.e. predicates that do not have training examples).

The same problem was again highlighted by the results obtained with and without the frame information in the Senseval-3 competition (Litkowski, 2004) of FrameNet (Johnson et al., 2003) role labeling task. When such information is not used by the systems, the performance decreases by 10 percent points. This is quite intuitive as the semantics of many roles strongly depends on the focused frame. Thus, we cannot expect a good performance on new domains in which this information is not available.

A solution to this problem is the automatic frame detection. Unfortunately, our preliminary experiments showed that given a FrameNet (FN) predicate-argument structure, the task of identifying the associated frame can be performed with very good results when the verb predicates have enough training examples, but becomes very challenging otherwise. The predicates belonging to new application domains (i.e. not yet included in FN) are especially problematic since there is no training data available.

Therefore, we should rely on a semantic context alternative to the frame (Giuglea and Moschitti, 2004). Such context should have a wide coverage and should be easily derivable from FN data. A very good candidate seems to be the Intersective Levin class (ILC) (Dang et al., 1998) that can be found as well in other predicate resources like PB and VerbNet (VN) (Kipper et al., 2000).

In this paper we have investigated the above claim by designing a semi-automatic algorithm that assigns ILCs to FN verb predicates and by carrying out several semantic role labeling (SRL) experiments in which we replace the frame with the ILC information. We used support vector ma-

chines (Vapnik, 1995) with (a) polynomial kernels to learn the semantic role classification and (b) Tree Kernels (Moschitti, 2004) for learning both frame and ILC classification. Tree kernels were applied to the syntactic trees that encode the subcategorization structures of verbs. This means that, although FN contains three types of predicates (nouns, adjectives and verbs), we only concentrated on the verb predicates and their roles.

The results show that: (1) ILC can be derived with high accuracy for both FN and Probank and (2) ILC can replace the frame feature with almost no loss in the accuracy of the SRL systems. At the same time, ILC provides better predicate coverage as it can also be learned from other corpora (e.g. PB).

In the remainder of this paper, Section 2 summarizes previous work done on FN automatic role detection. It also explains in more detail why models based exclusively on this corpus are not suitable for free-text parsing. Section 3 focuses on VN and PB and how they can enhance the robustness of our semantic parser. Section 4 describes the mapping between frames and ILCs whereas Section 5 presents the experiments that support our thesis. Finally, Section 6 summarizes the conclusions.

2 Automatic Semantic Role Labeling

One of the goals of the FN project is to design a linguistic ontology that can be used for the automatic processing of semantic information. The associated hierarchy contains an extensive semantic analysis of verbs, nouns, adjectives and situations in which they are used, called *frames*. The basic assumption on which the frames are built is that each word evokes a particular situation with specific participants (Fillmore, 1968). The word that evokes a particular frame is called *target word* or *predicate* and can be an adjective, noun or verb. The participant entities are defined using semantic roles and they are called *frame elements*.

Several models have been developed for the automatic detection of the frame elements based on the FN corpus (Gildea and Jurafsky, 2002; Thompson et al., 2003; Litkowski, 2004). While the algorithms used vary, almost all the previous studies divide the task into: 1) the identification of the verb arguments to be labeled and 2) the tagging of each argument with a role. Also, most of the models agree on the core features as be-

ing: *Predicate, Headword, Phrase Type, Governing Category, Position, Voice* and *Path*. These are the initial features adopted by Gildea and Jurafsky (2002) (henceforth G&J) for both frame element identification and role classification.

One difference among previous machine-learning models is whether they used the frame information or not. The impact of the frame feature over unseen predicates and words is particularly interesting for us. The results obtained by G&J provide some interesting insights in this direction. In one of their experiments, they used the frame to generalize from predicates seen in the training data to unseen predicates, which belonged to the same frame. The overall performance increased showing that when no training data is available for a target word we can use data from the same frame.

Other studies suggest that the frame is crucial when trying to eliminate the major sources of errors. In their error analysis, (Thompson et al., 2003) pinpoints that the verb arguments with headwords that are rare in a particular frame but not rare over the whole corpus are especially hard to classify. For these cases the frame is very important because it provides the context information needed to distinguish between different word senses.

Overall, the experiments presented in G&J's study correlated with the results obtained in the Senseval-3 competition show that the frame feature increases the performance and decreases the amount of annotated examples needed in training (i.e. frame usage improves the generalization ability of the learning algorithm). On the other hand, the results obtained without the frame information are very poor.

These results show that having broader frame coverage is very important for robust semantic parsing. Unfortunately, the 321 frames that contain at least one verb predicate cover only a small fraction of the English verb lexicon and of the possible domains. Also from these 321 frames only 100 were considered to have enough training data and were used in Senseval-3 (see (Litkowski, 2004) for more details).

Our approach for solving such problems involves the usage of a frame-like feature, namely the Intersective Levin class (ILC). We show that the ILC can replace the frame with almost no loss in performance. At the same time, ILC provides better coverage as it can be learned also from other

corpora (e.g. PB).

The next section provides the theoretical support for the unified usage of FN, VN and PB, explaining why and how it is possible to link them.

3 Linking FrameNet to VerbNet and PropBank

In general, predicates belonging to the same FN frame have a coherent syntactic behavior that is also different from predicates pertaining to other frames (G&J). This finding is consistent with theories of linking that claim that the syntactic behavior of a verb can be predicted from its semantics (Levin, 1993). This insight justifies the attempt to use ILCs instead of the frame feature when classifying FN semantic roles (Giuglea and Moschitti, 2004).

The main advantage of using Levin classes comes from the fact that other resources like PB and the VN lexicon contain this kind of information. Thus, we can train an ILC classifier also on the PB corpus, considerably increasing the verb knowledge base at our disposal. Another advantage derives from the syntactic criteria that were applied in defining the Levin's clusters. As shown later in this article, the syntactic nature of these classes makes them easier to classify than frames when using only syntactic and lexical features.

More precisely, Levin's clusters are formed according to diathesis alternation criteria which are variations in the way verbal arguments are grammatically expressed when a specific semantic phenomenon arises. For example, two different types of diathesis alternations are the following:

(a) Middle Alternation

[*Subject, Agent* The butcher] cuts [*Direct Object, Patient* **the meat**].
[*Subject, Patient* **The meat**] cuts easily.

(b) Causative/inchoative Alternation

[*Subject, Agent* Janet] broke [*Direct Object, Patient* **the cup**].
[*Subject, Patient* **The cup**] broke.

In both cases, what is alternating is the grammatical function that the Patient role takes when changing from the transitive use of the verb to the intransitive one. The semantic phenomenon accompanying these types of alternations is the change of focus from the entity performing the action to the theme of the event.

Levin documented 79 alternations which constitute the building blocks for the verb classes. Although alternations are chosen as the primary means for identifying the classes, additional properties related to subcategorization, morphology and extended meanings of verbs are taken into account as well. Thus, from a syntactic point of view, the verbs in one Levin class have a regular behavior, different from the verbs pertaining to other classes. Also, the classes are semantically coherent and all verbs belonging to one class share the same participant roles.

This constraint of having the same semantic roles is further ensured inside the VN lexicon which is constructed based on a more refined version of the Levin's classification, called Intersective Levin classes (ILCs) (Dang et al., 1998). The lexicon provides a regular association between the syntactic and semantic properties of each of the described classes. It also provides information about the syntactic frames (alternations) in which the verbs participate and the set of possible semantic roles.

One corpus associated with the VN lexicon is PB. The annotation scheme of PB ensures that the verbs belonging to the same Levin class share similarly labeled arguments. Inside one ILC, to one argument corresponds one semantic role numbered sequentially from ARG0 to ARG5. The adjunct roles are labeled ARGM.

Levin classes were constructed based on regularities exhibited at grammatical level and the resulting clusters were shown to be semantically coherent. As opposed, the FN frames were built on semantic bases, by putting together verbs, nouns and adjectives that evoke the same situations. Although different in conception, the FN verb clusters and VN verb clusters have common properties¹:

1. Different syntactic properties between distinct verb clusters (as proven by the experiments in G&J)
2. A shared set of possible semantic roles for all verbs pertaining to the same cluster.

Having these insights, we have assigned a correspondent VN class not to each verb predicate but rather to each frame. In doing this we have applied the simplifying assumption that a frame has a

¹See section 4.4 for more details

unique corresponding Levin class. Thus, we have created a one-to-many mapping between the ILCs and the frames. In order to create a pair (FN frame, VN class), our mapping algorithm checks both the syntactic and semantic consistency by comparing the role frequency distributions on different syntactic positions for the two candidates. The algorithm is described in detail in the next section.

4 Mapping FrameNet frames to VerbNet classes

The mapping algorithm consists of three steps: (a) we link the frames and ILCs that have the largest number of verbs in common and we create a set of pairs (FN frame, VN class) (see Table 1); (b) we refine the pairs obtained in the previous step based on diathesis alternation criteria, i.e. the verbs pertaining to the FN frame have to undergo the same diathesis alternation that characterize the corresponding VN class (see Table 2) and (c) we manually check the resulting mapping.

4.1 The mapping algorithm

Given a frame, F , we choose as candidate for the mapping the ILC, C , that has the largest number of verbs in common with it (see Table 1, line (I)). If the number is greater or equal than three we form a pair $\langle F, C \rangle$ that will be tested in the second step of the algorithm. Only the frames that have more than 3 verb lexical units are candidates for this step (frames with less than 3 members cannot pass condition (II)). This excludes a number of 60 frames that will be subsequently manually mapped.

In order to assign a VN class to a frame, we have to verify that the verbs belonging to the FN frame participate in the same diathesis alternation criteria used to define the VN class. Thus, the pairs $\langle F, C \rangle$ formed in step 1 of the mapping algorithm have to undergo a validation step that verifies the similarity between the enclosed FN frame and VN class. This validation process has several sub-steps:

First, we make use of the property (2) of the Levin classes and FN frames presented in the previous section. According to this property, all verbs pertaining to one frame or ILC have the same participant roles. Thus, a first test of compatibility between a frame and a Levin class is that they share the same participant roles. As FN is annotated with frame-specific semantic roles, we manually mapped these roles into the VN set of the-

```

INPUT
VN = {C | C is a VerbNet class}
VN Class C = {v | v is a verb of C}
FN = {F | F is a FrameNet frame}
FN frame F = {v | v is a verb of F}
OUTPUT
Pairs = {⟨F, C⟩ | F ∈ FN, C ∈ VN : F maps to C}
COMPUTE PAIRS:
Let Pairs = ∅
for each F ∈ FN
  (I) compute C* = arg max_{C ∈ VN} |F ∩ C|
  (II) if |F ∩ C*| ≥ 3 then Pairs = Pairs ∪ ⟨F, C*⟩

```

Table 1: Linking FrameNet frames and VerbNet classes.

```

TR = {θi : θi is the i - th theta role of VerbNet }

for each ⟨F, C⟩ ∈ Pairs
   $\vec{A}^F = \langle o_1, \dots, o_n \rangle, o_i = \# \langle \theta_i, F, pos = adjacent \rangle$ 
   $\vec{D}^F = \langle o_1, \dots, o_n \rangle, o_i = \# \langle \theta_i, F, pos = distant \rangle$ 
   $\vec{A}^C = \langle o_1, \dots, o_n \rangle, o_i = \# \langle \theta_i, C, pos = adjacent \rangle$ 
   $\vec{D}^C = \langle o_1, \dots, o_n \rangle, o_i = \# \langle \theta_i, C, pos = distant \rangle$ 

   $Score_{F,C} = \frac{2}{3} \times \frac{\vec{A}^F \cdot \vec{A}^C}{\|\vec{A}^F\| \times \|\vec{A}^C\|} + \frac{1}{3} \times \frac{\vec{D}^F \cdot \vec{D}^C}{\|\vec{D}^F\| \times \|\vec{D}^C\|}$ 

```

Table 2: Mapping algorithm - refining step.

matic roles. Given a frame, we assigned thematic roles to all frame elements that are associated with verbal predicates. For example the *Speaker*, *Addressee*, *Message* and *Topic* roles from the *Telling* frame were respectively mapped into the *Agent*, *Recipient*, *Theme* and *Topic* theta roles.

Second, we build a frequency distribution of VN thematic roles on different syntactic positions. Based on our observation and previous studies (Merlo and Stevenson, 2001), we assume that each ILC has a distinct frequency distribution of roles on different grammatical slots. As we do not have matching grammatical functions in FN and VN, we approximate that *subjects* and *direct objects* are more likely to appear on positions adjacent to the predicate, while *indirect objects* appear on more distant positions. The same intuition is successfully used by G&J to design the *Position* feature.

For each thematic role θ_i we acquired from VN and FN data the frequencies with which θ_i appears on an adjacent A or distant D positions in a given frame or VN class (i.e. $\# \langle \theta_i, class, position \rangle$). Therefore, for each frame and class, we obtain two vectors with thematic role frequencies corresponding respectively to the adjacent and distant positions (see Table 2). We compute a score for each

Score	No. of Frames	Not mapped	Correct	Overall Correct
[0,0.5]	118	48.3%	82.5%	89.6%
(0.5,0.75]	69	0	84%	
(0.75,1]	72	0	100%	

Table 3: Results of the mapping algorithm.

pair $\langle F, C \rangle$ using the normalized scalar product.

The core arguments, which tend to occupy adjacent positions, show a minor syntactic variability and are more reliable than adjunct roles. To account for this in the overall score, we multiply the adjacent and the distant scores by 2/3 and 1/3, respectively. This limits the impact of adjunct roles like Temporal and Location.

The above frequency vectors are computed for FN directly from the corpus of predicate-argument structure examples associated with each frame. The examples associated with the VN lexicon are extracted from the PB corpus. In order to do this we apply a preprocessing step in which each label Arg0..5 is replaced with its corresponding thematic role given the ILC of the predicate. We assign the same roles to the adjuncts all over PB as they are general for all verb classes. The only exception is ARG-M-DIR that can correspond to Source, Goal or Path. We assign different roles to this adjunct based on the prepositions. We ignore some adjuncts like ARG-M-ADV or ARG-M-DIS because they cannot bear a thematic role.

4.2 Mapping Results

We found that only 133 VN classes have correspondents among FN frames. Moreover, from the frames mapped with an automatic score smaller than 0.5 almost a half did not match any of the existing VN classes². A summary of the results is depicted in Table 3. The first column contains the automatic score provided by the mapping algorithm when comparing frames with ILCs. The second column contains the number of frames for each score interval. The third column contains the percentage of frames that did not have a corresponding VN class and finally the fourth and fifth columns contain the accuracy of the mapping algorithm for each interval score and for the whole task, respectively.

We mention that there are 3,672 distinct verb senses in PB and 2,351 distinct verb senses in

²The automatic mapping is improved by manually assigning the FN frames of the pairs that receive a score lower than 0.5.

FN. Only 501 verb senses are in common between the two corpora which means 13.64% of PB and 21.31% of FN. Thus, by training an ILC classifier on both PB and FN we extend the number of available verb senses to 5,522.

4.3 Discussion

In the literature, other studies compared the Levin classes with the FN frames, e.g. (Baker and Ruppenhofer, 2002; Giuglea and Moschitti, 2004; Shi and Mihalcea, 2005). Their findings suggest that although the two set of clusters are roughly equivalent there are also several types of mismatches:

1. Levin classes that are narrower than the corresponding frames,
2. Levin classes that are broader than the corresponding frames and
3. Overlapping groups.

For our task, point 2 does not pose a problem. Points 1 and 3 however suggest that there are cases in which to one FN frame corresponds more than one Levin class. By investigating such cases, we noted that the mapping algorithm consistently assigns scores below 75% to cases that match problem 1 (two Levin classes inside one frame) and below 50% to cases that match problem 3 (more than two Levin classes inside one frame). Thus, to increase the accuracy of our results, a first step should be to assign independently an ILC to each of the verbs pertaining to frames with score lower than 0.75%.

Nevertheless the current results are encouraging as they show that the algorithm is achieving its purpose by successfully detecting syntactic incoherences that can be subsequently corrected manually. Also, in the next section we will show that our current mapping achieves very good results, giving evidence for the effectiveness of the Levin class feature.

5 Experiments

In the previous sections we have presented the algorithm for annotating the verb predicates of FrameNet (FN) with Intersective Levin classes (ILCs). In order to show the effectiveness of this annotation and of the ILCs in general we have performed several experiments.

First, we trained (1) an ILC multiclassifier from FN, (2) an ILC multiclassifier from PB and (3) a

	Run 51.3.2	Cooking 45.3	Characterize 29.2	Other_cos 45.4	Say 37.7	Correspond 36.1	Multiclassifier
PB #Train Instances	262	6	2,945	2,207	9,707	259	52,172
PB #Test Instances	5	5	134	149	608	20	2,742
PB Results	75	33.33	96.3	97.24	100	88.89	92.96
FN #Train Instances	5,381	138	765	721	1,860	557	46,734
FN #Test Instances	1,343	35	40	184	1,343	111	11,650
FN Results	96.36	72.73	95.73	92.43	94.43	78.23	92.63

Table 4: F1s of some individual ILC classifiers and the overall multiclassifier accuracy (180 classes on PB and 133 on FN).

	Body_part	Crime	Degree	Agent	Multiclassifier
FN #Train Instances	1,511	39	765	6,441	102,724
FN #Test Instances	356	5	187	1,643	25,615
LF+Gold Frame	90.91	88.89	70.51	93.87	90.8
LF+Gold ILC	90.80	88.89	71.52	92.01	88.23
LF+Automatic Frame	84.87	88.89	70.10	87.73	85.64
LF+Automatic ILC	85.08	88.89	69.62	87.74	84.45
LF	79.76	75.00	64.17	80.82	80.99

Table 5: F1s of some individual FN role classifiers and the overall multiclassifier accuracy (454 roles).

frame multiclassifier from FN. We compared the results obtained when trying to classify the VN class with the results obtained when classifying frame. We show that ILCs are easier to detect than FN frames.

Our second set of experiments regards the automatic labeling of FN semantic roles on FN corpus when using as features: gold frame, gold ILC, automatically detected frame and automatically detected ILC. We show that in all situations in which the VN class feature is used, the accuracy loss, compared to the usage of the frame feature, is negligible. This suggests that the ILC can successfully replace the frame feature for the task of semantic role labeling.

Another set of experiments regards the generalization property of the ILC. We show the impact of this feature when very few training data is available and its evolution when adding more and more training examples. We again perform the experiments for: gold frame, gold ILC, automatically detected frame and automatically detected ILC.

Finally, we simulate the difficulty of free text by annotating PB with FN semantic roles. We used PB because it covers a different set of verbal predicates and also because it is very different from FN at the level of vocabulary and sometimes even syntax. These characteristics make PB a difficult testbed for the semantic role models trained on FN.

In the following section we present the results

obtained for each of the experiments mentioned above.

5.1 Experimental setup

The corpora available for the experiments were PB and FN. PB contains about 54,900 predicates and gold parse trees. We used sections from 02 to 22 (52,172 predicates) to train the ILC classifiers and Section 23 (2,742 predicates) for testing purposes. The number of ILCs is 180 in PB and 133 on FN, i.e. the classes that we were able to map.

For the experiments on FN corpus, we extracted 58,384 sentences from the 319 frames that contain at least one verb annotation. There are 128,339 argument instances of 454 semantic roles. In our evaluation we use only verbal predicates. Moreover, as there is no fixed split between training and testing, we randomly selected 20% of sentences for testing and 80% for training. The sentences were processed using Charniak’s parser (Charniak, 2000) to generate parse trees automatically.

The classification models were implemented by means of the SVM-light-TK software available at <http://ai-nlp.info.uniroma2.it/moschitti> which encodes tree kernels in the SVM-light software (Joachims, 1999). We used the default parameters. The classification performance was evaluated using the F_1 measure for the individual role and ILC classifiers and the accuracy for the multiclassifiers.

5.2 Automatic VerbNet class vs. automatic FrameNet frame detection

In these experiments, we classify ILCs on PB and frames on FN. For the training stage we use SVMs with Tree Kernels.

The main idea of tree kernels is the modeling of a $K_T(T_1, T_2)$ function which computes the number of common substructures between two trees T_1 and T_2 . Thus, we can train SVMs with structures drawn directly from the syntactic parse tree of the sentence. The kernel that we employed in our experiments is based on the SCF structure devised in (Moschitti, 2004). We slightly modified SCF by adding the headwords of the arguments, useful for representing the selectional preferences (more details are given in (Giuglea and Moschitti, 2006)).

For frame detection on FN, we trained our classifier on 46,734 training instances and tested on 11,650 testing instances, obtaining an accuracy of 91.11%. For ILC detection the results are depicted in Table 4. The first six columns report the F_1 measure of some verb class classifiers whereas the last column shows the global multiclassifier accuracy. We note that ILC detection is more accurate than the frame detection on both FN and PB. Additionally, the ILC results on PB are similar with those obtained for the ILCs on FN. This suggests that the training corpus does not have a major influence. Also, the SCF-based tree kernel seems to be robust in what concerns the quality of the parse trees. The performance decay is very small on FN that uses automatic parse trees with respect to PB that contains gold parse trees.

5.3 Automatic semantic role labeling on FrameNet

In the experiments involving semantic role labeling, we used SVMs with polynomial kernels. We adopted the standard features developed for semantic role detection by Gildea and Jurafsky (see Section 2). Also, we considered some of the features designed by (Pradhan et al., 2005): *First and Last Word/POS in Constituent*, *Subcategorization*, *Head Word of Prepositional Phrases* and the *Syntactic Frame* feature from (Xue and Palmer, 2004). For the rest of the paper, we will refer to these features as being literature features (LF). The results obtained when using the literature features alone or in conjunction with the gold frame feature, gold ILC, automatically detected frame feature and automatically detected ILC are depicted in Table 5.

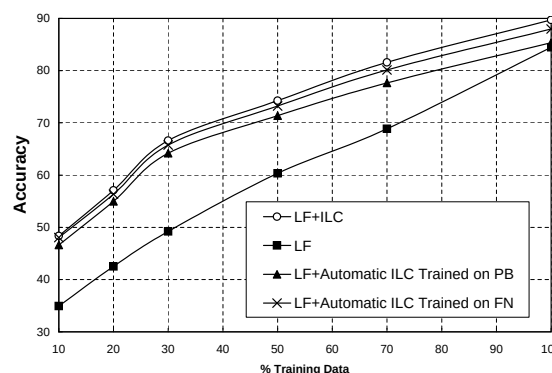


Figure 1: Semantic role learning curve.

The first four columns report the F_1 measure of some role classifiers whereas the last column shows the global multiclassifier accuracy. The first row contains the number of training and testing instances and each of the other rows contains the performance obtained for different feature combinations. The results are reported for the labeling task as the argument-boundary detection task is not affected by the frame-like features (G&J).

We note that automatic frame produces an accuracy very close to the one obtained with automatic ILC suggesting that this is a very good candidate for replacing the frame feature. Also, both automatic features are very effective and they decrease the error rate by 20%.

To test the impact of ILC on SRL with different amount of training data, we additionally draw the learning curves with respect to different features: LF, LF+ (gold) ILC, LF+automatic ILC trained on PB and LF+automatic ILC trained on FN. As can be noted, the automatic ILC information provided by the ILC classifiers (trained on FN or PB) performs almost as good as the gold ILC.

5.4 Annotating PB with FN semantic roles

To show that our approach can be suitable for semantic role free-text annotation, we have automatically classified PB sentences³ with the FN semantic-role classifiers. In order to measure the quality of the annotation, we randomly selected 100 sentences and manually verified them. We measured the performance obtained with and without the automatic ILC feature. The sentences contained 189 arguments from which 35 were incorrect when ILC was used compared to 72 incorrect in the absence of this feature, i.e. an accuracy of 81% with ILC versus 62% without it. This demonstrates the importance of the ILC feature

³The results reported are only for role classification.

outside the scope of FN where the frame feature is not available.

6 Conclusions

In this paper we have shown that the ILC feature can successfully replace the FN frame feature. By doing that we could interconnect FN to VN and PB obtaining better verb coverage and a more robust semantic parser. Our good results show that we have defined an effective framework which is a promising step toward the design of more robust semantic parsers.

In the future, we intend to measure the effectiveness of our system by testing FN SRL on a larger portion of PB or on other corpora containing a larger verb set.

References

- Collin Baker and Josef Ruppenhofer. 2002. Framenets frames vs. levin's verb classes. In *28th Annual Meeting of the Berkeley Linguistics Society*.
- Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of CoNLL-2005*.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NACL00*, Seattle, Washington.
- Hoa Trang Dang, Karin Kipper, Martha Palmer, and Joseph Rosenzweig. 1998. Investigating regular sense extensions based on intersective levin classes. In *Coling-ACL98*.
- Charles J. Fillmore. 1968. The case for case. In *Universals in Linguistic Theory*.
- Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistic*.
- Ana-Maria Giuglea and Alessandro Moschitti. 2004. Knowledge discovering using FrameNet, VerbNet and PropBank. In *Proceedings of Workshop on Ontology and Knowledge Discovering at ECML 2004*, Pisa, Italy.
- Ana-Maria Giuglea and Alessandro Moschitti. 2006. Shallow semantic parsing based on FrameNet, VerbNet and PropBank. In *Proceedings of the 17th European Conference on Artificial Intelligence*, Riva del Garda, Italy.
- 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*.
- Christopher Johnson, Miriam Petruck, Collin Baker, Michael Ellsworth, Josef Ruppenhofer, and Charles Fillmore. 2003. *Framenet: Theory and practice*. Berkeley, California.
- Paul Kingsbury and Martha Palmer. 2002. From Treebank to PropBank. In *LREC02*.
- Karin Kipper, Hoa Trang Dang, and Martha Palmer. 2000. Class-based construction of a verb lexicon. In *AAAI00*.
- Beth Levin. 1993. *English Verb Classes and Alternations A Preliminary Investigation*. Chicago: University of Chicago Press.
- Kenneth Litkowski. 2004. Senseval-3 task automatic labeling of semantic roles. In *Senseval-3*.
- Paola Merlo and Suzanne Stevenson. 2001. Automatic verb classification based on statistical distribution of argument structure. *CL Journal*.
- Alessandro Moschitti. 2004. A study on convolution kernels for shallow semantic parsing. In *ACL04*, Barcelona, Spain.
- Sameer Pradhan, Kadri Hacioglu, Valeri Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support vector learning for semantic argument classification. *Machine Learning Journal*.
- Lei Shi and Rada Mihalcea. 2005. Putting pieces together: Combining FrameNet, VerbNet and WordNet for robust semantic parsing. In *Proceedings of Cicling 2005, Mexico*.
- Cynthia A. Thompson, Roger Levy, and Christopher Manning. 2003. A generative model for semantic role labeling. In *14th European Conference on Machine Learning*.
- V. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer.
- Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP 2004*, Barcelona, Spain. Association for Computational Linguistics.

Multilingual Legal Terminology on the Jibiki Platform: The LexALP Project

Gilles Sérasset, Francis Brunet-Manquat

Université Joseph Fourier,
Laboratoire CLIPS-IMAG, BP 53
38041 Grenoble Cedex 9 - France,
Gilles.Serasset@imag.fr
Francis.Brunet-Manquat@imag.fr

Elena Chiocchetti

EURAC Research
Viale Druso 1
39100 Bozen/Bolzano - Italy
Elena.Chiocchetti@eurac.edu

Abstract

This paper presents the particular use of “Jibiki” (Papillon’s web server development platform) for the LexALP¹ project. LexALP’s goal is to harmonise the terminology on spatial planning and sustainable development used within the Alpine Convention², so that the member states are able to cooperate and communicate efficiently in the four official languages (French, German, Italian and Slovene). To this purpose, LexALP uses the Jibiki platform to build a term bank for the contrastive analysis of the specialised terminology used in six different national legal systems and four different languages. In this paper we present how a generic platform like Jibiki can cope with a new kind of dictionary.

1 Introduction

One of the most time-consuming hindrances to supranational law drafting and convention negotiation is the lack of understanding among negotiators and technical writers. This is not only due to the fact that different languages are involved, but mainly to the inherent differences in the legal systems. Countries that speak the same language (like France and part of Switzerland) may use the same word to represent different legal concepts³,

as defined in their respective legal traditions. The same concept may be referred to in different ways according to the legal system⁴. Also, terms that may superficially seem to be translations of each other can represent different legal notions⁵.

In order to concretely address these problems, several institutions representing translators, terminologists, legal experts and computational linguists joined in the LexALP project, co-funded by EU’s INTERREG IIIb Alpine Space programme. The objective of the project is to compare the specialised terminology of six different national legal systems (Austria, France, Germany, Italy, Switzerland and Slovenia) and three supranational systems (EU law, international law and the particular framework of the Alpine Convention) in the four official languages of the Alpine Convention, which is an international framework agreement signed by all countries of the Alpine arc and the EU. This contrastive analysis serves as a basis for the work of a group of experts (the Harmonising Group) who will determine translation equivalents in French, Italian, German and Slovene (one-to-one correspondence) in the fields of spatial planning and sustainable development for use within the Convention, thus optimising the understanding between the Alpine states at supranational level.

The tools that are to be developed for these objectives comprise a corpus bank and a term bank. The corpus bank is developed by adapting the bistro system (Streiter et al., 2006; Streiter et al., 2004). The term bank is based on the Jibiki plat-

¹Legal Language Harmonisation System for Environment and Spatial Planning within the Multilingual Alps

²<http://www.convenzionedellealpi.org>

³E.g.: In the German-speaking province of Bolzano Italy the *Landeshauptmann* is the president of the provincial council, with much more limited competence than the Austrian *Landeshauptmann*, who is head of one of the states (*Bundesland*) that are part of the Austrian federation.

⁴See for instance the European Union use of *chien drogue* while French legislation calls them *chien renifleur*.

⁵For example, in Italy an *elezione suppletiva* is commonly held whenever an elected deputy or senator either resigns or dies. In Germany in such cases the first non-elected candidate is called to parliament. *Ersatzwahlen* are a rare phenomenon, foreseen in some very specific cases.

form (Mangeot et al., 2003; Sérasset, 2004).

This paper details the way the Jibiki platform is used in order to cope with a new dictionary structure. The platform provides dictionary access and edition services without any new and specific development.

After a brief overview of the Jibiki platform, we describe the choices made by the LexALP team for the structure and organisation of their term bank. Then, we show how this structure is described using Jibiki metadata description languages. Finally, we give some details on the resulting LexALP Information System.

2 Jibiki, The Papillon Dictionary Development Platform

2.1 Overview

The Jibiki platform has been designed to support the collaborative development of multilingual dictionaries. This platform is used as the basis of the Papillon project web site⁶.

This platform offers several services to its users:

- access to many different dictionaries from a single easy to use query form,
- advance search for particular dictionary entries through an advanced search form,
- creation and edition of dictionary entries.

What makes the Jibiki platform quite unique is the fact that it provides these services *regardless of the dictionary structure*. In other words it may be used by any dictionary builder to give access *and* collaboratively edit *any* dictionary, provided that the resulting dictionary will be freely accessible online.

2.2 Jibiki Platform Architecture

The Jibiki platform is a framework used to set up a web server dedicated to the collaborative development of multilingual dictionaries. All services provided by the platform are organised as classical 3-tier architectures with a presentation layer (in charge of the interface with users), a business layer (which provides the services per se) and a data layer (in charge of the storage of persistent data).

In order to adapt the Jibiki platform to a new dictionary, the dictionary manager does not have

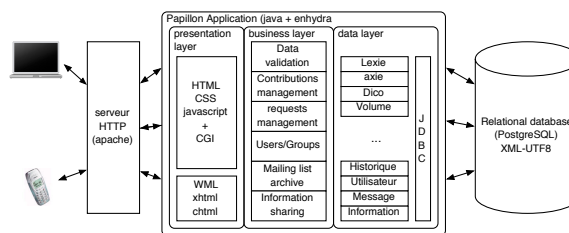


Figure 1: The Jibiki platform general architecture

to write specific java code nor specific dynamic web pages. The only necessary information used by the platform consists in:

- a description of the dictionary volumes and their relations,
- a mapping between the envisaged dictionary structure and a simple hypothetical dictionary structure (called CDM)⁷,
- the definition of the XML structure of each envisaged dictionary volume by way of XML schemas,
- the development of a specific edition interface as a standard xhtml form (that can be adapted from an automatically generated draft).

3 The LexALP Terminology Structure

3.1 Overview

The objective of the LexALP project is to compare the specialised terminology of six different national legal systems and three supranational systems in four different languages, and to harmonise it, thus optimising communication between the Alpine states at supranational level. To achieve this objective, the terminology of the Alpine Convention is described and compared to the equivalent terms used in national legislation. The resulting terminology entries feed a specific term bank that will support the harmonisation work.

As the project deals with legal terms, which refer to concepts that are proper of the considered national law or international convention, equivalence problems are the norm, given that concepts are not “stable” between the different national legislations. Standard terminology techniques for other fields can not be applied to the field of law, where the standardisation approach (Felber, 1987;

⁶<http://www.papillon-dictionary.org/>

⁷This mapping is sufficient for simple dictionary access

Felber, 1994) is not applicable. For this, we chose to use “acceptions” as they are defined in the Papillon dictionary (Sérasset, 1994) to represent the equivalence links between concepts of the different legal systems (Arntz, 1993).

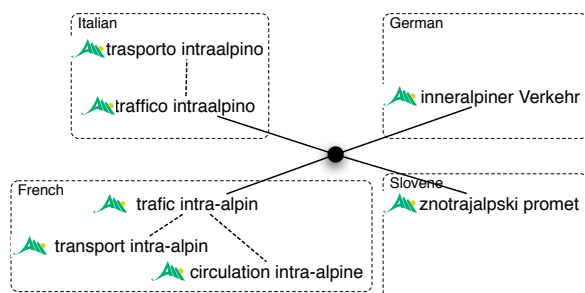


Figure 2: An Alpine Convention concept in four languages

The example given in figure 2 shows a concept defined in the Alpine Convention. This concept has the same definition in the four languages of the Alpine Convention but is expressed by different *denominations*. The Alpine Convention also uses the terms “circulation intra-alpine” or “transport intra-alpin” which are identified as synonyms by the terminologist.

This illustrates the first goal of the LexALP project. In different texts, the same concept may be realised by different terms in the same language. This may lead to inefficient communication. Hence, a single term has to be determined as part of a *harmonised quadruplet of translation equivalents*. The other denominations will be represented in the term bank as non-harmonised synonyms in order to direct drafting and translating within the Alpine Convention towards a more clear and consistent terminology use for interlingual and supranational communication.

In this example, the lexicographers and jurists did not identify any existing concept in the different national laws that could be considered close enough to the concept analysed. This is coherent with the minutes from the French National Assembly which clearly states that the term “trafic intra-alpin” (among others) should be clarified by a declaration to be added to the Alpine Convention.

Figure 3 shows an analogous quadrilingual example where the Alpine Convention concept may be related to a legal term defined in the French laws. In this example the French term is distinguished from the Alpine Convention terms, because these concepts belong to different legal sys-

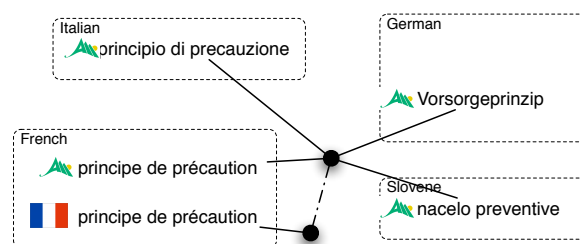


Figure 3: A quadrilingual term extracted from the Alpine Convention with reference to its equivalent at French national level

tems (and are not identically defined in them). Hence, the terminologists created distinct acceptions, one for each concept. These acceptions are related by a *translation link*.

This illustrates the second goal of the project, which is to help with the fine comprehension of the Alpine Convention and with the detailed knowledge necessary to evaluate the implementation and implementability of the convention in the different legal systems.

As a by-product of the project, one can see that there is an indirect relation between concepts from different national legal systems (by way of their respective relation to the concepts of the Alpine Convention). However, establishing these indirect relations is not one of the main objectives of the LexALP project and would require more direct contrastive analysis.

3.2 Macro- and Micro- Structures

The LexALP term bank consists in 5 volumes (for French, German, Italian, Slovene and English) containing all term descriptions (grammatical information, definition, contexts etc.). The translation links are established through a central acception volume. Figure 2 and 3 show examples of terms extracted from the Alpine Convention, synonymy links in the French and Italian volumes, as well as inter-lingual relations by way of acceptions.

All language volumes share the same microstructure. This structure is stored in XML.

Figure 4 shows the xml structure of the French term “trafic intra-alpin”, as defined in the Alpine Convention. The term entry is associated to a unique identifier used to establish relations between volume entries. Each term entry belongs to one (and only one) legal system. The example term belongs to the Alpine Convention legal

```

<entry id="fra.trafic_intra-alpin.1010743.e"
  lang="fra"
  legalSystem="AC"
  process_status="FINALISED"
  status="HARMONISED">
  <term>trafic intra-alpin</term>
  <grammar>n.m.</grammar>
  <domain>Transport</domain>
  <usage frequency="common"
    geographical-code="INT"
    technical="false"/>
  <relatedTerm isHarmonised="false"
    relationToTerm="Synonym"
    termref="">
    transport intra-alpin
  </relatedTerm>
  <relatedTerm isHarmonised="false"
    relationToTerm="Synonym"
    termref="">
    circulation intra-alpine
  </relatedTerm>
  <definition>
    [T]rafic constitué de trajets ayant leur
    point de départ et/ou d'arrivée à l'inté-
    rieur de l'espace alpin.
  </definition>
  <source url="">Prot. Transp., art. 2</source>
  <context url="http://www...">
    Des projets routiers à grand débit pour
    le trafic intra-alpin peuvent être réalisés,
    si [...].
  </context>
</entry>

```

Figure 4: XML form of the term “trafic intra-alpin”.

system⁸ (code AC). The set of known legal systems includes of course countries belonging to the Alpine Space (Austria, France, Germany, Italy, Slovenia and Switzerland⁹) but also international treaties or conventions. The entry also bears the information on its status (harmonised or rejected) and its process status (to be processed, provisionally processed or finalised).

The term itself and its part of speech is also given, with the general domain to which the term belongs, along with some usage notes. In these usage notes, the attribute `geographical-code` allows for discrimination between terms defined in national (or federal) laws and terms defined in regional laws as in some of the countries involved legislative power is distributed at different levels.

Then the term may be related to other terms. These relations may lead to simple strings of texts (as in the given example) or to autonomous term entries in the dictionary by the use of the `termref` attribute. The relation itself is specified in the `relationToTerm` attribute. The current schema allows for the representation of relations

⁸Strictly speaking, the Alpine Convention does not constitute a legal system per se.

⁹Also Liechtenstein and Monaco are parties to the Alpine Convention, however, their legal systems are not terminologically processed within LexALP.

between concepts (synonymy, hyponymy and hyperonymy), as well as relations between graphies (variant, abbreviation, acronym, etc.).

Then, a definition and a context may be given. Both should be extracted from legal texts, which must be identified in the source field.

An interlingual acception (or *axie*) is a place holder for relations. Each interlingual acception may be linked to several term entries in the language volumes through `termref` elements and to other interlingual acceptations through `axieref` elements, as illustrated in figure 5.

```

<axie id="axi..1011424.e">
  <termref
    idref="ita.traffico_intraalpino.1010654.e"
    lang="ita"/>
  <termref
    idref="fra.trafic_intra-alpin.1010743.e"
    lang="fra"/>
  <termref
    idref="deu.inneralpinen_Verkehr.1011065.e"
    lang="deu"/>
  <termref
    idref="slo.znotrajalpski_promet.1011132.e"
    lang="slo"/>
  <axieref idref="" />
  <misc></misc>
</axie>

```

Figure 5: XML form of the interlingual acception illustrated in figure 2.

4 LexALP Information System

4.1 Overview

Building such a term bank can only be envisaged as a collaborative work involving terminologists, translators and legal experts from all the involved countries. Hence, the LexALP consortium has set up a centralised information system that is used to gather all textual and terminological data.

This information system is organized in two main parts. The first one is dedicated to corpus management. It allows the users to upload legal texts that will serve to bootstrap the terminology work (by way of candidate term extraction) and to let terminologists find occurrences of the term they are working on, in order for them to provide definitions or contexts.

The second part is dedicated to terminology work per se. It has been developed with the Jibiki platform described in section 2. In this section, we show the LexALP Information System functionality, along with the metadata required to implement it with Jibiki.

4.2 Dictionary Browsing

The first main service consists in browsing the currently developed dictionary. It consists in two different query interfaces (see figures 6 and 7) and a unique result presentation interface (see figure 10).

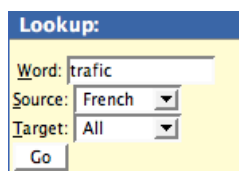


Figure 6: Simple search interface present on all pages of the LexALP Information System

```
<dictionary-metadata
[...]
d:category="multilingual"
d:fullname="LexALP multilingual Term Base"
d:name="LexALP"
d:owner="LexALP consortium"
d:type="pivot">
  <languages>
    <source-language d:lang="deu"/>
    <source-language d:lang="fra"/>
    <target-language d:lang="deu"/>
    <target-language d:lang="fra"/>
    [...]
  </languages>
  [...]
  <volumes>
    <volume-metadata-ref name="LexALP_fra"
      source-language="fra"
      xlink:href="LexALP_fra-metadata.xml"/>
    <volume-metadata-ref name="LexALP_deu"
      source-language="deu"
      xlink:href="LexALP_deu-metadata.xml"/>
    [...]
    <volume-metadata-ref name="LexALP_axi"
      source-language="axi"
      xlink:href="LexALP_axi-metadata.xml"/>
  </volumes>
  <xsl-style-sheet name="LexALP" default="true"
    xlink:href="LexALP-view.xsl"/>
  <xsl-style-sheet name="short-list"
    xlink:href="short-list-view.xsl"/>
</dictionary-metadata>
```

Figure 8: Excerpt of the dictionary descriptor

In the provided examples, the user of the system specifies an entry (a term), or part of it, and a language in which the search is to be done. The expected behaviour may only be achieved if :

- the system knows in which volume the search is to be performed,
- the system knows where, in the volume entry, the headword is to be found,
- the system is able to produce a presentation for the retrieved XML structures.

However, as the Jibiki platform is entirely independent of the underlying dictionary structure

```
<volume-metadata
[...]
dbname="lexalpfra"
dictname="LexALP"
name="LexALP_fra"
source-language="fra">
  <cdm-elements>
    <cdm-entry-id index="true"
      xpath="/volume/entry/@id"/>
    <cdm-headword d:lang="fra" index="true"
      xpath="/volume/entry/term/text()"/>
    <cdm-pos d:lang="fra" index="true"
      xpath="/volume/entry/grammar/text()"/>
    [...]
  </cdm-elements>
  <xmlschema-ref xlink:href="lexalp.xsd"/>
  <template-entry-ref
    xlink:href="lexalp_fra-template.xml"/>
  <template-interface-ref
    xlink:href="lexalp-interface.xhtml"/>
</volume-metadata>
```

Figure 9: Excerpt of a volume descriptor

(which makes it highly adaptable), the expected result may only be achieved if additional metadata is added to the system.

These pieces of information are to be found in the mandatory dictionary descriptor. It consists in a structure defined in the Dictionary Metadata Language (DML), as set of metadata structures and a specific XML namespace defined in (Mangeot, 2001).

Figure 8 gives an excerpt of this descriptor. The metadata first identify the dictionary by giving it a name and a type. In this example the dictionary is a *pivot* dictionary (DML also defines *monolingual* and *bilingual* dictionary types). The descriptor also defines the set of source and target languages. Finally, the dictionary is defined as a set of volumes, each volume being described in another file. As the LexALP dictionary is a pivot dictionary, there should be a volume for the artificial language *axi*, which is the pivot volume.

Figure 9 shows an excerpt of the description of the French volume of the LexALP dictionary. After specifying the name of the dictionary, the descriptor provides a set of *cdm-elements*. These elements are used to identify standard dictionary elements (that can be found in several dictionaries) in the specific dictionary structure. For instance, the descriptor tells the system that the headword of the dictionary (*cdm-headword*) is to be found by applying the specified *xpath*¹⁰ to the dictionary structure.

With this set of metadata, the system knows that:

¹⁰an *xpath* is a standard way to extract a sub-part of any XML structure

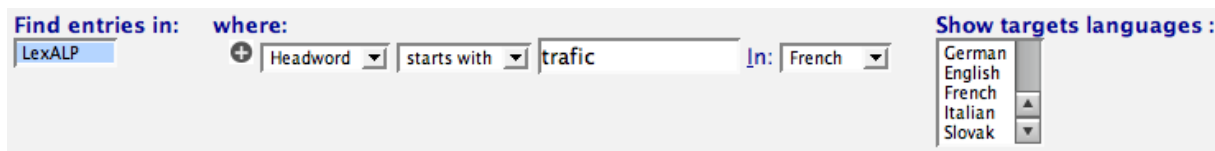


Figure 7: Advanced search interface

- requests on French should be directed to the LexALP_fra volume,
- the requested headword will be found in the text of the `term` element of the volume entry element,

Hence, the system can easily perform a request and retrieve the desired XML entries. The only remaining step is to produce a presentation for the user, based on the retrieved entries. This is achieved by way of a `xsl`¹¹ stylesheet. This stylesheet is specified either on the dictionary level (for common presentations) or on the volume level (for volume specific presentation).

In the given example, the dictionary administrator provided two presentations called LexALP (the default one, as shown in figure 10) and `short-list`, both of them defined in the dictionary descriptor.

This mechanism allows for the definition of presentation outputs in `xhtml` (for online browsing) or for presentation output in `pdf` (for dictionary export and print).

4.3 Dictionary Edition

The second main service provided by the Jibiki platform is to allow terminologists to collaboratively develop the envisaged dictionary. In this sense, Jibiki is quite unique as it federates, on the very same platform the construction and diffusion of a structured dictionary.

As before, Jibiki may be used to edit *any* dictionary. Hence, it needs some metadata information in order to work:

- the *complete* definition of the dictionary entry structures by way of an XML schema,
- a template describing an empty entry structure,

¹¹XSL is a standard way to transform an XML structure into another structure (XML or not).

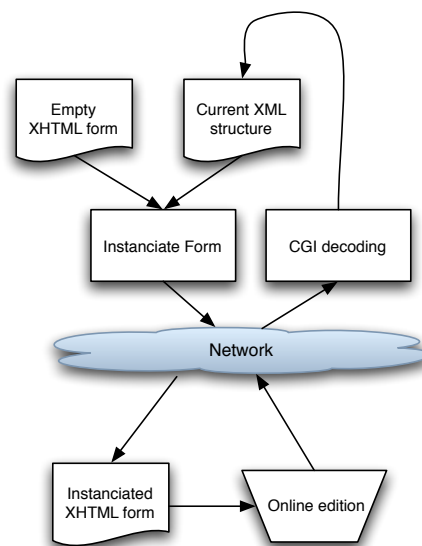


Figure 11: Basic flow chart of the editing service

- a `xhtml` form used to edit a dictionary entry structure (which can be adapted from an automatically generated one).

When this information is known, the Jibiki platform provides a specific web page to edit a dictionary entry structure. As shown in figure 11, the XML structure is projected into the given empty XHTML form. This form is served as a standard web page on the client browser. After manual editing, the resulting form is sent back to the Jibiki platform as CGI¹² data. The Jibiki platform decodes this data and modifies the edited XML structure accordingly. Then the process iterates as long as necessary. Figure 12 shows an example of such a dynamically created web page.

After each update, the resulting XML structure is stored in the dictionary database. However, it is not available to other users until it is marked as *finished* by the contributor (by clicking on the *save* button). If the contributor leaves the web page without saving the entry, he will be able to retrieve it and finish his contribution later.

¹²Common Gateway Interface

TRAFIC INTRA-ALPIN, n.m. [HARMONISED/FINALISED]  kwic

TRANSPORT INTRA-ALPIN [kwic] (Synonym)

CIRCULATION INTRA-ALPINE [kwic] (Synonym)

[T]rafic constitué de trajets ayant leur point de départ et/ou d'arrivée à l'intérieur de l'espace alpin.

Source: Prot. Transp., art. 2

Context: Des projets routiers à grand débit pour le trafic intra-alpin peuvent être réalisés, si [...] les besoins en matière de transports ne peuvent être satisfaits, ni par une meilleure utilisation des capacités routières et ferroviaires existantes, ni par l'extension ou la construction d'infrastructures ferroviaires ou fluvio-maritimes, ni par l'amélioration d'un transport combiné, ni par d'autres mesures relatives à l'organisation des transports [...]. [goto]

INNERALPINER VERKEHR, n.m. [HARMONISED/FINALISED]  kwic

Verkehr mit Ziel und Quelle im Alpenraum inklusive Verkehr mit Ziel oder Quelle im Alpenraum.

Source: Prot. Verk., Art. 2/Cieri

Context: Die Vertragsparteien verpflichten sich zu einer nachhaltigen Verkehrspolitik, die [...] den inneralpinen und alpenquerenden Verkehr durch Steigerung der Effektivität und Effizienz der Verkehrssysteme und durch Förderung umwelt- und ressourcenschonenderer Verkehrsträger unter wirtschaftlich tragbaren Kosten gewährleistet [...]. [goto]

TRAFFICO INTRAALPINO, n.m. [HARMONISED/FINALISED]  kwic

TRASPORTO INTRAALPINO [kwic] (Synonym)

Traffico o trasporto con origine e destinazione all'interno del territorio alpino incluso il traffico o trasporto con origine o destinazione nel territorio alpino.

Source: Prot. Traspt., art. 2/Cieri

Context: Le Parti contraenti si impegnano ad attuare una politica sostenibile dei trasporti tesa a [...] ridurre gli effetti negativi e i rischi derivanti dal traffico intraalpino e transalpino ad un livello che sia tollerabile per l'uomo, la fauna e la flora e il loro habitat [...]. [goto]

ZNOTRAJALPSKI PROMET, n.m. [HARMONISED/UNPROCESSED]  kwic

Promet s ciljem in izvorom v alpskem prostoru (notranji promet), vključno s prometom s ciljem ali izvorom v alpskem prostoru.

Source: Prot. Promet, čl. 2

Context: Cestni projekt višjega reda za znotrajalpski promet se lahko udejanji le takrat, če [...] potreb po prevoznih zmogljivostih ni moč zadovoljiti z boljšo izkoriščenostjo obstoječih cestnih in železniških zmogljivosti, z rekonstrukcijo ali novogradnjo železniških in plovniških infrastruktur ter z izboljšanjem kombiniranega prometa kot tudi z nadaljnjimi prometno-organizacijskimi ukrepi [...]. [goto]

Figure 10: Query result presentation interface

LexALP edition interface

legalSystem: /

geographical-code:

Usage: frequency: ☐ is a technical term?

☐ ☐ Domains:

☐

☐ ☐ Related terms:

☐ [harmonised ? ☒] {termref: }

☐ [harmonised ? ☐] {termref: }

Definition (created by:)

☐ ☐ Sources:

☐

☐ ☐ Contexts:

☐

☐ ☐ Notes:

☐

misc:

Figure 12: Edition interface of a LexALP French entry

At each step of the contribution (after each update) and at each step of dictionary editing (after each save), the previous state is saved and the contributor (or the dictionary administrator) is able to browse the history of changes and to revert the entry to a previous version.

5 Conclusion

In this article we give some details on the way the Jibiki platform allows the diffusion and the online editing of a dictionary, regardless of its structure (monolingual, bilingual (directed or not) or multilingual (multi-bilingual or pivot based)).

Initially developed to support the editing of the Papillon multilingual dictionary¹³, the Jibiki platform proved useful for the development of other very different dictionaries. It is currently used for the development of the GDEF (Grand Dictionnaire Estonien-Français) project¹⁴ an Estonian French bilingual dictionary. This article also shows the use of the platform for the development of a European term bank for legal terms on spatial planning and sustainable development in the LexALP project.

Adapting the Jibiki platform to a new dictionary requires the definition of several metadata information, taking the form of several XML files. While not trivial, this metadata definition does not require any competence in computer development. This adaptation may therefore also be done by experimented linguists. Moreover, when the dictionary microstructure needs to evolve, this evolution does not require any programming. Hence the Jibiki platform gives linguists great liberty in their decisions.

Another positive aspect of Jibiki is that it integrates diffusion and editing services on the same platform. This allows for a tighter collaboration between linguists and users and also allows for the involvement of motivated users to the editing process.

The Jibiki platform is freely available for use by any willing team of lexicographer/terminologists, provided that the resulting dictionary data will be freely available for online browsing.

In this article, we also presented the choices made by the LexALP consortium to structure a term bank used for the description and harmonisation of legal terms in the domain of spatial plan-

ning and sustainable development of the Alpine Space. In such a domain, classical techniques used in multilingual terminology cannot be used as the term cannot be defined by reference to a stable/shared semantic level (each country having its own set of non-equivalent legal concepts).

References

- Reiner Arntz. 1993. Terminological equivalence and translation. In H. Sonneveld and K. Loening, editors, *Terminology. Applications in Interdisciplinary Communication*, pages 5–19. Amsterdam et Philadelphia, John Benjamins Publishing Company.
- Helmut Felber, 1987. *Manuel de terminologie*. UNESCO, Paris.
- Helmut Felber. 1994. Terminology research: Its relation to the theory of science. *ALFA*, 8(7):163–172.
- Mathieu Mangeot, Gilles Sérasset, and Mathieu Lafourcade. 2003. Construction collaborative d’une base lexicale multilingue, le projet Papillon. *TAL*, 44(2):151–176.
- Mathieu Mangeot. 2001. *Environnements centralisés et distribués pour lexicographes et lexicologues en contexte multilingue*. Thèse de nouveau doctorat, spécialité informatique, Université Joseph Fourier Grenoble I, Septembre.
- Gilles Sérasset. 1994. Interlingual lexical organisation for multilingual lexical databases in nadia. In Makoto Nagao, editor, *COLING-94*, volume 1, pages 278–282, August.
- Gilles Sérasset. 2004. A generic collaborative platform for multilingual lexical database development. In Gilles Sérasset, editor, *COLING 2004 Multilingual Linguistic Resources*, pages 73–79, Geneva, Switzerland, August 28. COLING.
- Oliver Streiter, Leonhard Voltmer, Isabella Ties, and Natascia Ralli. 2004. BISTRO, the online platform for terminology management: structuring terminology without entry structures. In *The translation of domain specific languages and multilingual terminology*, number 3 in *Linguistica Antverpiensia New Series*. Hoger Instituut voor Vertalers en Tolken, Hogeschool Antwerpen.
- Oliver Streiter, Leonhard Voltmer, Isabella Ties, Natascia Ralli, and Verena Lyding. 2006. BISTRO: Data structure, term tools and interface. *Terminology Science and Research*, 16.

¹³<http://www.papillon-dictionary.org/>

¹⁴<http://estfra.ee/>

Leveraging Reusability: Cost-effective Lexical Acquisition for Large-scale Ontology Translation

G. Craig Murray

Bonnie J. Dorr

Jimmy Lin

Institute for Advanced Computer Studies

University of Maryland

{gcraigm, bdorr, jimmylin}@umd.edu

Jan Hajič

Pavel Pecina

Institute for Formal and Applied Linguistics

Charles University

{hajic, pecina}@ufal.mff.cuni.cz

Abstract

Thesauri and ontologies provide important value in facilitating access to digital archives by representing underlying principles of organization. Translation of such resources into multiple languages is an important component for providing multilingual access. However, the specificity of vocabulary terms in most ontologies precludes fully-automated machine translation using general-domain lexical resources. In this paper, we present an efficient process for leveraging human translations when constructing domain-specific lexical resources. We evaluate the effectiveness of this process by producing a probabilistic phrase dictionary and translating a thesaurus of 56,000 concepts used to catalogue a large archive of oral histories. Our experiments demonstrate a cost-effective technique for accurate machine translation of large ontologies.

1 Introduction

Multilingual access to digital collections is an important problem in today's increasingly interconnected world. Although technologies such as cross-language information retrieval and machine translation help humans access information they could not otherwise find or understand, they are often inadequate for highly specific domains.

Most digital collections of any significant size use a system of organization that facilitates easy access to collection contents. Generally, the organizing principles are captured in the form of a controlled vocabulary of keyword phrases (de-

scriptors) representing specific concepts. These descriptors are usually arranged in a hierarchic thesaurus or ontology, and are assigned to collection items as a means of providing access (either via searching for keyword phases, browsing the hierarchy, or a combination both). MeSH (Medical Subject Headings) serves as a good example of such an ontology; it is a hierarchically-arranged collection of controlled vocabulary terms manually assigned to medical abstracts in a number of databases. It provides multilingual access to the contents of these databases, but maintaining translations of such a complex structure is challenging (Nelson, et al, 2004).

For the most part, research in multilingual information access focuses on the content of digital repositories themselves, often neglecting significant knowledge that is explicitly encoded in the associated ontologies. However, information systems cannot utilize such ontologies by simply applying off-the-shelf machine translation. General-purpose translation resources provide insufficient coverage of the vocabulary contained within these domain-specific ontologies.

This paper tackles the question of how one might efficiently translate a large-scale ontology to facilitate multilingual information access. If we need humans to assist in the translation process, how can we maximize access while minimizing cost? Because human translation is associated with a certain cost, it is preferable not to incur costs of retranslation whenever components of translated text are reused. Moreover, when exhaustive human translation is not practical, the most "useful" components should be translated first. Identifying reusable elements and prioritizing their translation based on utility is essential to maximizing effectiveness and reducing cost.

We present a process of prioritized translation that balances the issues discussed above. Our work is situated in the context of the MALACH project, an NSF-funded effort to improve multilingual information access to large archives of spoken language (Gustman, et al., 2002). Our process leverages a small set of manually-acquired English-Czech translations to translate a large ontology of keyword phrases, thereby providing Czech speakers access to 116,000 hours of video testimonies in 32 languages. Starting from an initial out-of-vocabulary (OOV) rate of 85%, we show that a small set of prioritized translations can be elicited from human informants, aligned, decomposed and then recombined to cover 90% of the access value in a complex ontology. Moreover, we demonstrate that prioritization based on hierarchical position and frequency of use facilitates extremely efficient reuse of human input. Evaluations show that our technique is able to boost performance of a simple translation system by 65%.

2 The Problem

The USC Shoah Foundation Institute for Visual History and Education manages what is presently the world's largest archive of videotaped oral histories (USC, 2006). The archive contains 116,000 hours of video from the testimonies of over 52,000 survivors, liberators, rescuers and witnesses of the Holocaust. If viewed end to end, the collection amounts to 13 years of continuous video. The Shoah Foundation uses a hierarchically arranged thesaurus of 56,000 keyword phrases representing domain-specific concepts. These are assigned to time-points in the video testimonies as a means of indexing the video content. Although the testimonies in the collection represent 32 different languages, the thesaurus used to catalog them is currently available only in English. Our task was to translate this resource to facilitate multilingual access, with Czech as the first target language.

Our first pass at automating thesaurus translation revealed that only 15% of the words in the vocabulary could be found in an available aligned corpus (Čmejrek, et al., 2004). The rest of the vocabulary was not available from general resources. Lexical information for translating these terms had to be acquired from human input. Reliable access to digital archives requires accuracy. Highly accurate human translations incur a cost that is generally proportional to the number of words being translated. However, the keyword phrases in the Shoah Foundation's ar-

chive occur in a Zipfian distribution—a relatively small number of terms provide access to a large portion of the video content. Similarly, a great number of highly specific terms describe only a small fraction of content. Therefore, not every keyword phrase in the thesaurus carries the same value for access to the archive. The hierarchical arrangement of keyword phrases presents another issue: some concepts, while not of great value for access to segments of video, may be important for organizing other concepts and for browsing the hierarchy. These factors must be balanced in developing a cost-effective process that maximizes utility.

3 Our Solution

This paper presents a prioritized human-in-the-loop approach to translating large-scale ontologies that is fast, efficient, and cost effective. Using this approach, we collected 3,000 manual translations of keyword phrases and reused the translated terms to generate a lexicon for automated translation of the rest of the thesaurus. The process begins by prioritizing keyword phrases for manual translation in terms of their value in accessing the collection and the reusability of their component terms. Translations collected from one human informant are then checked and aligned to the original English terms by a second informant. From these alignments we induce a probabilistic English-Czech phrase dictionary.

To test the effectiveness of this process we implemented a simple translation system that utilizes the newly generated lexical resources. Section 4 reports on two evaluations of the translation output that quantify the effectiveness of our human-in-the-loop approach.

3.1 Maximizing Value and Reusability

To quantify their utility, we defined two values for each keyword phrase in the thesaurus: a *thesaurus value*, representing the importance of the keyword phrase for providing access to the collection, and a *translation value*, representing the usefulness of having the keyword phrase translated. These values are not identical, but the second is related to the first.

Thesaurus value: Keyword phrases in the Shoah Foundation's thesaurus are arranged into a poly-hierarchy in which child nodes may have multiple parents. Internal (non-leaf) nodes of the hierarchy are used to organize concepts and support concept browsing. Some internal nodes are also used to index video content. Leaf nodes are

very specific and are only used to index video content. Thus, the usefulness of any keyword phrase for providing access to the digital collection is directly related to the concept's position in the thesaurus hierarchy.

A fragment of the hierarchy is shown in Figure 1. The keyword phrase “Auschwitz II-Birkenau (Poland: Death Camp)”, which describes a Nazi death camp, is assigned to 17,555 video segments in the collection. It has broader (parent) terms and narrower (child) terms. Some of the broader and narrower terms are also assigned to segments, but not all. Notably, “German death camps” is not assigned to any video segments. However, “German death camps” has very important narrower terms including “Auschwitz II-Birkenau” and others.

From this example, we can see that an internal node is valuable in providing access to its children, even if the keyword phrase itself is not assigned to any segments. The value we assign to any term must reflect this fact. If we were to reduce cost by translating only the nodes assigned to video segments, we would neglect nodes that are crucial for browsing. However, if we value a node by the sum value of all its children, grandchildren, etc., the resulting calculation would bias the top of the hierarchy. Any prioritization based on this method would lead to translation of the top of the hierarchy first. Given limited resources, leaf nodes might never be translated. Support for searching and browsing calls for different approaches to prioritization.

To strike a balance between these factors, we calculate a *thesaurus value*, which represents the importance of each keyword phrase to the thesaurus as a whole. This value is computed as:

$$h_k = \text{count}(s_k) + \frac{\sum_{i \in \text{children}(k)} h_i}{|\text{children}(k)|}$$

For leaf nodes in our thesaurus, this value is simply the number of video segments to which the concept has been assigned. For parent nodes, the *thesaurus value* is the number of segments (if any) to which the node has been assigned, *plus* the average of the *thesaurus value* of any child nodes.

This recursive calculation yields a micro-averaged value that represents the reachability of segments via downward edge traversals from a given node in the hierarchy. That is, it gives a kind of weighted value for the number of segments described by a given keyword phrase or its narrower-term keyword phrases.



Figure 1. Sample keyword phrase with broader and narrower terms

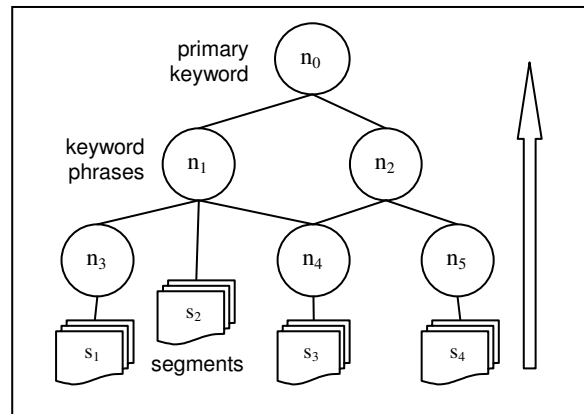


Figure 2. Bottom-up micro-averaging

For example, in Figure 2 each of the leaf nodes n_3 , n_4 , and n_5 have values based solely on the number of segments to which they are assigned. Node n_1 has value both as an access point to the segments at s_2 and as an access point to the keyword phrases at nodes n_3 and n_4 . Other internal nodes, such as n_2 have value only in providing access to other nodes/keyword phrases. Working from the bottom of the hierarchy up to the primary node (n_0) we can compute the *thesaurus value* for each node in the hierarchy. In our example, we start with nodes n_3 through n_5 , counting the number of the segments that have been assigned each keyword phrase. Then we move up to nodes n_1 and n_2 . At n_1 we count the number of segments s_2 to which n_1 was assigned and add that count to the average of the *thesaurus values* for n_3 , and n_4 . At n_2 we simply average the *thesaurus values* for n_4 and n_5 . The final values quantify how valuable the translation of any given keyword phrase would be in providing access to video segments.

Translation value: After obtaining the *thesaurus value* for each node, we can compute the *translation value* for each word in the vocabulary

as the sum of the thesaurus value for every keyword phrase that contains that word:

$$t_w = \sum_{k \in K_w} h_k \quad \text{where } K_w = \{x \mid \text{phrase } x \text{ contains } w\}$$

For example, the word “Auschwitz” occurs in 35 concepts. As a candidate for translation, it carries a large impact, both in terms of the number of keyword phrases that contains this word, and the potential value of those keyword phrases (once they are translated) in providing access to segments in the archive. The end result is a list of vocabulary words and the impact that correct translation of each word would have on the overall value of the translated thesaurus.

We elicited human translations of entire keyword phrases rather than individual vocabulary terms. Having humans translate individual words without their surrounding context would have been less efficient. Also, the value any keyword phrase holds for translation is only indirectly related to its own value as a point of access to the collection (i.e., its *thesaurus value*). Some keyword phrases contain words with high *translation value*, but the keyword phrase itself has low *thesaurus value*. Thus, the value gained by translating any given phrase is more accurately estimated by the total value of any untranslated words it contains. Therefore, we prioritized the order of keyword phrase translations based on the *translation value* of the untranslated words in each keyword phrase.

Our next step was to iterate through the thesaurus keyword phrases, prioritizing their translation based on the assumption that any words contained in a keyword phrase of higher priority would already have been translated. Starting from the assumption that the entire thesaurus is untranslated, we select the one keyword phrase that contains the most valuable un-translated words—we simply add up the *translation value* of all the untranslated words in each keyword phrase, and select the keyword phrase with the highest value. We add this keyword phrase to a prioritized list of items to be manually translated and we remove it from the list of untranslated phrases. We update our vocabulary list and, assuming translations of all the words in the prior keyword phrase to now be translated (neglecting issues such as morphology), we again select the keyword phrase that contains the most valuable untranslated words. We iterate the process until all vocabulary terms have been included at least one keyword phrases on the prioritized list. Ultimately we end up with an ordered list of the

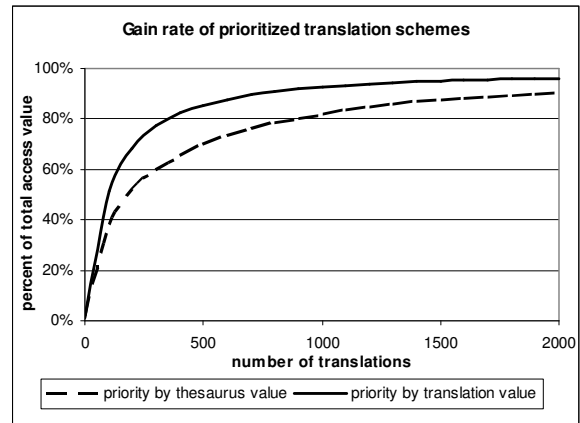


Figure 3. Gain rate of access value based on number of human translations

keyword phrases that should be translated to cover the entire vocabulary, with the most important words being covered first.

A few words about additional characteristics of this approach: note that it is greedy and biased toward longer keyword phrases. As a result, some words may be translated more than once because they appear in more than one keyword phrase with high *translation value*. This side effect is actually desirable. To build an accurate translation dictionary, it is helpful to have more than one translation of frequently occurring words, especially for morphologically rich languages such as Czech. Our technique makes the operational assumption that translations of a word gathered in one context can be reused in another context. Obviously this is not always true, but contexts of use are relatively stable in controlled vocabularies. Our evaluations address the acceptability of this operational assumption and demonstrate that the technique yields acceptable translations.

Following this process model, the most important elements of the thesaurus will be translated first, and the most important vocabulary terms will quickly become available for automated translation of keyword phrases with high *thesaurus value* that do not make it onto the prioritized list for manual translation (i.e., low *translation value*). The overall access value of the thesaurus rises very quickly after initial translations. With each subsequent human translation of keyword phrases on the prioritized list, we gain tremendous value in terms of providing non-English access to the collection of video testimonies. Figure 3 shows this rate of gain. It can be seen that prioritization based on *translation value* gives a much higher yield of total access than prioritization based on *thesaurus value*.

3.2 Alignment and Decomposition

Following the prioritization scheme above, we obtained professional translations for the top 3000 English keyword phrases. We tokenized these translations and presented them to another bilingual Czech speaker for verification and alignment. This second informant marked each Czech word in a translated keyword phrase with a link to the equivalent English word(s). Multiple links were used to convey the relationship between a single word in one language and a string of words in another. The output of the alignment process was then used to build a probabilistic dictionary of words and phrases.

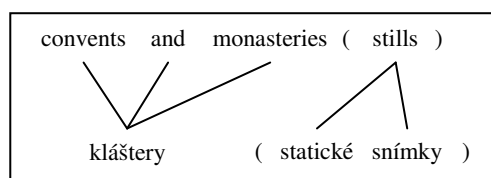


Figure 4. Sample alignment

Figure 4 shows an example of an aligned translation. The word “stills” is recorded as a translation for “statické snímky” and “kláštery” is recorded as a translation for “convents and monasteries.” We count the number of occurrences of each alignment in all of the translations and calculate probabilities for each Czech word or phrase given an English word or phrase. For example, in the top 3000 keyword phrases “stills” appears 29 times. It was aligned with “statické snímky” 28 times and only once with “statické záběry”, giving us a translation probability of $28/29=0.9655$ for “statické snímky”.

Human translation of the 3000 English keyword phrases into Czech took approximately 70 hours, and the alignments took 55 hours. The overall cost of human input (translation and alignment) was less than 1000 €. The projected cost of full translation for the entire thesaurus would have been close to 20000 € and would not have produced any reusable resources. Naturally, costs for building resources in this manner will vary, but in our case the cost savings is approximately twenty fold.

3.3 Machine Translation

To demonstrate the effectiveness of our approach, we show that a probabilistic dictionary, induced through the process we just described, facilitates high quality machine translation of the rest of the thesaurus. We evaluated translation quality using a relatively simple translation system. However, more sophisticated systems can draw equal benefit from the same lexical resources.

Our translation system implemented a greedy coverage algorithm with a simple back-off strategy. It first scans the English input to find the longest matching substring in our dictionary, and replaces it with the most likely Czech translation. Building on the example above, the system looks up “monasteries and convents stills” in the dictionary, finds no translation, and backs off to “monasteries and convents”, which is translated to “kláštery”. Had this phrase translation not been found, the system would have attempted to find a match for the individual tokens. Failing a match in our dictionary, the system then backs off to the Prague Czech-English Dependency Treebank dictionary, a much larger dictionary with broader scope. If no match is found in either dictionary for the full token, we stem the token and look for matches based on the stem. Finally, tokens whose translations can not be found are simply passed through untranslated.

A minimal set of heuristic rules was applied to reordering the Czech tokens but the output is primarily phrase by phrase/word by word translation. Our evaluation scores below will partially reflect the simplicity of our system. Our system is simple by design. Any improvement or degradation to the *input* of our system has direct influence on the *output*. Thus, measures of translation accuracy for our system can be directly interpreted as quality measures for the lexical resources used and the process by which they were developed.

4 Evaluation

We performed two different types of evaluation to validate our process. First, we compared our system output to human reference translations using Bleu (Papineni, et al., 2002), a widely-accepted objective metric for evaluation of machine translations. Second, we showed corrected and uncorrected machine translations to Czech speakers and collected subjective judgments of fluency and accuracy.

For evaluation purposes, we selected 418 keyword phrases to be used as target translations. These phrases were selected using a stratified sampling technique so that different levels of *thesaurus value* would be represented. There was no overlap between these keyword phrases and the 3000 prioritized keyword phrases used to build our lexicon. Prior to machine translation we obtained at least two independent human-generated reference translations for each of the 418 keyword phrases.

After collecting the first 2500 prioritized translations, we induced a probabilistic dictionary and generated machine translations of the 418 target keyword phrases. These were then corrected by native Czech speakers, who adjusted word order, word choice, and morphology. We use this set of human-corrected machine translations as a second reference for evaluation.

Measuring the difference between our uncorrected machine translations (MT) and the human-generated reference establishes how accurate our translations are compared to an independently established target. Measuring the difference between our MT and the human-corrected machine translations (corrected MT) establishes how acceptable our translations are. We also measured the difference between corrected MT and the human-generated translations. We take this to be an upper bound on realistic system performance.

The results from our objective evaluation are shown in Figure 5. Each set of bars in the graph shows performance after adding a different number of aligned translations into the lexicon (i.e., performance after adding 500, 1000, ..., 3000 aligned translations.) The zero condition is our baseline: translations generated using only the dictionary available in the Prague Czech-English Dependency Treebank. Three different reference sets are shown: human-generated, corrected MT, and a combination of the two.

There is a notable jump in Bleu score after the very first translations are added into our probabilistic dictionary. Without any elicitation and alignment we got a baseline score of 0.46 (against the human-generated reference translations). After the aligned terms from only 500 translations were added to our dictionary, our Bleu score rose to 0.66. After aligned terms from 3000 translations were added, we achieved 0.69. Using corrected MT as the reference our Bleu scores improve from 0.48 to 0.79. If human-generated and human-corrected references are both considered to be correct translations, the improvement goes from .49 to .80. Regardless of the reference set, there is a consistent performance improvement as more and more translations are added. We found the same trend using the TER metric on a smaller data set (Murray, et al., 2006). The fact that the Bleu scores continue to rise indicates that our approach is successful in quickly expanding the lexicon with accurate translations. It is important to point out that Bleu scores are not meaningful in an absolute sense; the scores here should be interpreted with respect to each other. The trend

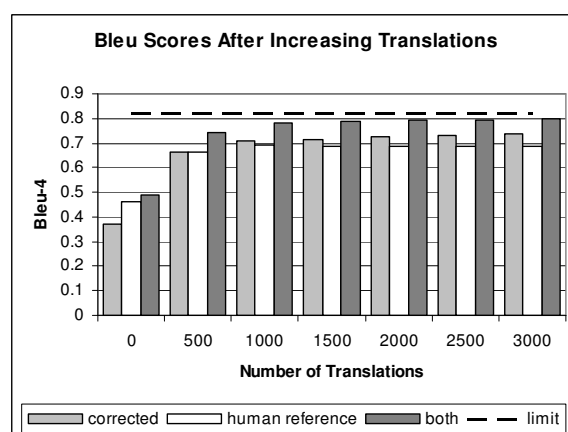


Figure 5. Objective evaluation results

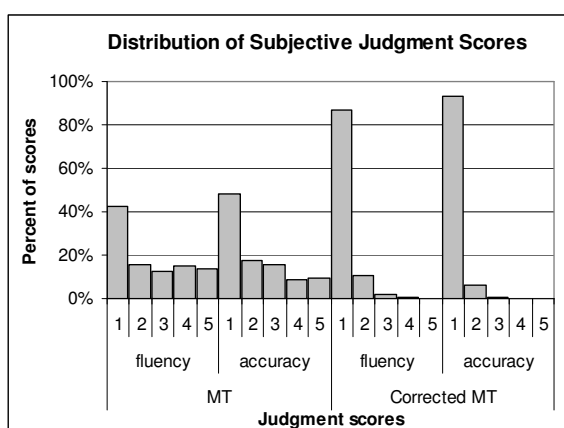


Figure 6. Subjective evaluation results

in scores strongly indicates that our prioritization scheme is effective for generating a high-quality translation lexicon at relatively low cost.

To determine an upper bound on machine performance, we compared our corrected MT output to the initial human-generated reference translations, which were collected prior to machine translation. Corrected MT achieved a Bleu score of 0.82 when compared to the human-generated reference translations. This upper bound is the "limit" indicated in Figure 5.

To determine the impact of external resources, we removed the Prague Czech-English Dependency Treebank dictionary as a back-off resource and retranslated keyword phrases using only the lexicons induced from our aligned translations. The results of this experiment showed only marginal degradation of the output. Even when as few as 500 aligned translations were used for our dictionary, we still achieved a Bleu score of 0.65 against the human reference translations. This means that even for languages where prior resources are not available our prioritization scheme successfully addresses the OOV problem.

In our subjective evaluation, we presented a random sample of our system output to seven

native Czech speakers and collected judgments of accuracy and fluency using a 5-point Likert scale (1=good, 3=neutral, 5=bad). An overview of the results is presented in Figure 6. Scores are shown for corrected and uncorrected MT. In all cases, the mode is 1 (i.e., good fluency and good accuracy). 59% of the machine translated phrases were rated 2 or better for fluency. 66% were rated 2 or better for accuracy. Only a small percentage of the translations had meanings that were far from the intended meaning. Disfluencies were primarily due to errors in morphology and word order.

5 Related Work

Several studies have taken a knowledge-acquisition approach to collecting multilingual word pairs. For example, Sadat et al. (2003) automatically extracted bilingual word pairs from comparable corpora. This approach is based on the simple assumption that if two words are mutual translations, then their most frequent collocates are likely to be mutual translations as well. However, the approach requires large comparable corpora, the collection of which presents non-trivial challenges. Others have made similar mutual-translation assumptions for lexical acquisition (Echizen-ya, et al., 2005; Kaji & Aizono, 1996; Rapp, 1999; Tanaka & Iwasaki, 1996). Most make use of either parallel corpora or a bilingual dictionary for the task of bilingual term extraction. Echizen-ya, et al. (2005) avoided using a bilingual dictionary, but required a parallel corpus to achieve their goal; whereas Fung (2000) and others have relied on pre-existing bilingual dictionaries. In either case, large bilingual resources of some kind are required. In addition, these approaches focused on the extraction of single-word pairs, not phrasal units.

Many recent approaches to dictionary and thesaurus translation are geared toward providing domain-specific thesauri to specialists in a particular field, e.g., medical terminology (Déjean, et al., 2005) and agricultural terminology (Chun & Wenlin, 2002). Researchers on these projects are faced with either finding human translators who are specialized enough to manage the domain-particular translations—or applying automatic techniques to large-scale parallel corpora where data sparsity poses a problem for low-frequency terms. Data sparsity is also an issue for more general state-of-the-art bilingual alignment approaches (Brown, et al., 2000; Och & Ney, 2003; Wantanabe & Sumita, 2003).

6 Conclusion

The task of translating large ontologies can be recast as a problem of implementing fast and efficient processes for acquiring task-specific lexical resources. We developed a method for prioritizing keyword phrases from an English thesaurus of concepts and elicited Czech translations for a subset of the keyword phrases. From these, we decomposed phrase elements for reuse in an English-Czech probabilistic dictionary. We then applied the dictionary in machine translation of the rest of the thesaurus.

Our results show an overall improvement in machine translation quality after collecting only a few hundred human translations. Translation quality continued to rise as more and more human translations were added. The test data used in our evaluations are small relative to the overall task. However, we fully expect these results to hold across larger samples and for more sophisticated translation systems.

We leveraged the reusability of translated words to translate a thesaurus of 56,000 keyword phrases using information gathered from only 3000 manual translations. Our probabilistic dictionary was acquired at a fraction of the cost of manually translating the entire thesaurus. By prioritizing human translations based on the *translation value* of the words and the *thesaurus value* of the keyword phrases in which they appear, we optimized the rate of return on investment. This allowed us to choose a trade-off point between cost and utility. For this project we chose to stop human translation at a point where less than 0.01% of the value of the thesaurus would be gained from each additional human translation. This choice produced a high-quality lexicon with significant positive impact on machine translation systems. For other applications, a different trade-off point will be appropriate, depending on the initial OOV rate and the importance of detailed coverage.

The value of our work lies in the process model we developed for cost-effective elicitation of lexical resources. The metrics we established for assessing the impact of each translation item are key to our approach. We use these to optimize the value gained from each human translation. In our case the items were keyword phrases arranged in a hierarchical thesaurus that describes an ontology of concepts. The operational value of these keyword phrases was determined by the access they provide to video segments in a large archive of oral histories. However, our technique is not limited to this application.

We have shown that careful prioritization of elicited human translations facilitates cost-effective thesaurus translation with minimal human input. Our use of a prioritization scheme addresses the most important deficiencies in the vocabulary first. We induced a framework where the utility of lexical resources gained from each additional human translation becomes smaller and smaller. Under such a framework, choosing the number of human translation to elicit becomes merely a function of the financial resources available for the task.

Acknowledgments

Our thanks to Doug Oard for his contribution to this work. Thanks also to our Czech informants: Robert Fischmann, Eliska Kozakova, Alena Prunerova and Martin Smok; and to Soumya Bhat for her programming efforts.

This work was supported in part by NSF IIS Award 0122466 and NSF CISE RI Award EIA0130422. Additional support also came from grants of the MSMT CR #1P05ME786 and #MSM0021620838, and the Grant Agency of the CR #GA405/06/0589.

References

- Brown, P. F., Della-Pietra, V. J., Della-Pietra, S. A., & Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2), 263-311.
- Chun, C., & Wenlin, L. (2002). The translation of agricultural multilingual thesaurus. In *Proceedings of the Third Asian Conference for Information Technology in Agriculture*. Beijing, China: Chinese Academy of Agricultural Sciences (CAAS) and Asian Federation for Information Technology in Agriculture (AFITA).
- Čmejrek, M., Cuřín, J., Havelka, J., Hajič, J., & Kubon, V. (2004). Prague Czech-English dependency treebank: Syntactically annotated resources for machine translation. In *4th International Conference on Language Resources and Evaluation* Lisbon, Portugal.
- Déjean, H., Gaussier, E., Renders, J.-M., & Sadat, F. (2005). Automatic processing of multilingual medical terminology: Applications to thesaurus enrichment and cross-language information retrieval. *Artificial Intelligence in Medicine*, 33(2), 111-124.
- Echizen-ya, H., Araki, K., & Momouchi, Y. (2005). Automatic acquisition of bilingual rules for extraction of bilingual word pairs from parallel corpora. In *Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition* (pp. 87-96).
- Fung, P. (2000). A statistical view of bilingual lexicon extraction: From parallel corpora to non-parallel corpora. In Jean Veronis (ed.), *Parallel Text Processing*. Dordrecht: Kluwer Academic Publishers.
- Gustman, Soergel, Oard, Byrne, Picheny, Ramabhadran, & Greenberg. (2002). Supporting access to large digital oral history archives. In *Proceedings of the Joint Conference on Digital Libraries*. Portland, Oregon. (pp. 18-27).
- Kaji, H., & Aizono, T. (1996). Extracting word correspondences from bilingual corpora based on word co-occurrence information. In *Proceedings of COLING '96* (pp. 23-28).
- Murray, G. C., Dorr, B., Lin, J., Hajič, J., & Pecina, P. (2006). Leveraging recurrent phrase structure in large-scale ontology translation. In *Proceedings of the 11th Annual Conference of the European Association for Machine Translation*. Oslo, Norway.
- Nelson, S. J., Schopen, M., Savage, A. G., Schulman, J.-L., & Arluk, N. (2004). The MeSH translation maintenance system: Structure, interface design, and implementation. In *Proceedings of the 11th World Congress on Medical Informatics*. (pp. 67-69). Amsterdam: IOS Press.
- Och, F. J., & Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1), 19-51.
- Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics* (pp. 331-318).
- Rapp, R. (1999). Automatic identification of word translations from unrelated English and German corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*. (pp. 519-526).
- Sadat, F., Yoshikawa, M., & Uemura, S. (2003). Enhancing cross-language information retrieval by an automatic acquisition of bilingual terminology from comparable corpora. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 397-398).
- Tanaka, K., & Iwasaki, H. (1996). Extraction of lexical translations from non-aligned corpora. In *Proceedings of COLING '96*. (pp. 580-585).
- USC. (2006) USC Shoah Foundation Institute for Visual History and Education, [online] <http://www.usc.edu/schools/college/vhi>
- Watanabe, T., & Sumita, E. (2003). Example-based decoding for statistical machine translation. In *Proceedings of MT Summit IX* (pp. 410-417).

Accurate Collocation Extraction Using a Multilingual Parser

Violeta Seretan

Language Technology Laboratory
University of Geneva
2, rue de Candolle, 1211 Geneva
Violeta.Seretan@latl.unige.ch

Eric Wehrli

Language Technology Laboratory
University of Geneva
2, rue de Candolle, 1211 Geneva
Eric.Wehrli@latl.unige.ch

Abstract

This paper focuses on the use of advanced techniques of text analysis as support for collocation extraction. A hybrid system is presented that combines statistical methods and multilingual parsing for detecting accurate collocational information from English, French, Spanish and Italian corpora. The advantage of relying on full parsing over using a traditional window method (which ignores the syntactic information) is first theoretically motivated, then empirically validated by a comparative evaluation experiment.

1 Introduction

Recent computational linguistics research fully acknowledged the stringent need for a systematic and appropriate treatment of phraseological units in natural language processing applications (Sag et al., 2002). Syntagmatic relations between words — also called *multi-word expressions*, or “idiosyncratic interpretations that cross word boundaries” (Sag et al., 2002, 2) — constitute an important part of the lexicon of a language: according to Jackendoff (1997), they are at least as numerous as the single words, while according to Mel’čuk (1998) they outnumber single words ten to one.

Phraseological units include a wide range of phenomena, among which we mention compound nouns (*dead end*), phrasal verbs (*ask out*), idioms (*lend somebody a hand*), and collocations (*fierce battle*, *daunting task*, *schedule a meeting*). They pose important problems for NLP applications, both text analysis and text production perspectives being concerned.

In particular, collocations¹ are highly problematic, for at least two reasons: first, because their linguistic status and properties are unclear (as pointed out by McKeown and Radev (2000), their definition is rather vague, and the distinction from other types of expressions is not clearly drawn); second, because they are prevalent in language. Mel’čuk (1998, 24) claims that “collocations make up the lions share of the phraseme inventory”, and a recent study referred in (Pearce, 2001) showed that each sentence is likely to contain at least one collocation.

Collocational information is not only useful, but also indispensable in many applications. In machine translation, for instance, it is considered “the key to producing more acceptable output” (Orliac and Dillinger, 2003, 292).

This article presents a system that extracts accurate collocational information from corpora by using a syntactic parser that supports several languages. After describing the underlying methodology (section 2), we report several extraction results for English, French, Spanish and Italian (section 3). Then we present in sections 4 and 5 a comparative evaluation experiment proving that a hybrid approach leads to more accurate results than a classical approach in which syntactic information is not taken into account.

2 Hybrid Collocation Extraction

We consider that syntactic analysis of source corpora is an inescapable precondition for collocation extraction, and that the syntactic structure of source text has to be taken into account in order to ensure the quality and interpretability of results.

¹To put it simply, collocations are non-idiomatical, but restricted, conventional lexical combinations.

As a matter of fact, some of the existing collocation extraction systems already employ (but only to a limited extent) linguistic tools in order to support the collocation identification in text corpora. For instance, lemmatizers are often used for recognizing all the inflected forms of a lexical item, and POS taggers are used for ruling out certain categories of words, e.g., in (Justeson and Katz, 1995).

Syntactic analysis has long since been recognized as a prerequisite for collocation extraction (for instance, by Smadja²), but the traditional systems simply ignored it because of the lack, at that time, of efficient and robust parsers required for processing large corpora. Oddly enough, this situation is nowadays perpetuated, in spite of the dramatic advances in parsing technology. Only a few exceptions exist, e.g., (Lin, 1998; Krenn and Evert, 2001).

One possible reason for this might be the way that collocations are generally understood, as a purely statistical phenomenon. Some of the best-known definitions are the following: “Collocations of a given word are statements of the habitual and customary places of that word” (Firth, 1957, 181); “arbitrary and recurrent word combination” (Benson, 1990); or “sequences of lexical items that habitually co-occur” (Cruse, 1986, 40). Most of the authors make no claims with respect to the grammatical status of the collocation, although this can indirectly be inferred from the examples they provide.

On the contrary, other definitions state explicitly that a collocation is an expression of language: “co-occurrence of two or more lexical items as realizations of structural elements within a given syntactic pattern” (Cowie, 1978); “a sequence of two or more consecutive words, that has characteristics of a syntactic and semantic unit” (Choueka, 1988). Our approach is committed to these latter definitions, hence the importance we lend to using appropriate extraction methodologies, based on syntactic analysis.

The hybrid method we developed relies on the parser Fips (Wehrli, 2004), that implements the Government and Binding formalism and supports several languages (besides the ones mentioned in

the abstract, a few others are also partly dealt with). We will not present details about the parser here; what is relevant for this paper is the type of syntactic structures it uses. Each constituent is represented by a simplified X-bar structure (without intermediate level), in which to the lexical head is attached a list of left constituents (its specifiers) and right constituents (its complements), and each of these are in turn represented by the same type of structure, recursively.

Generally speaking, a collocation extraction can be seen as a two-stage process:

- I. in stage one, collocation candidates are identified from the text corpora, based on criteria which are specific to each system;
- II. in stage two, the candidates are scored and ranked using specific *association measures* (a review can be found in (Manning and Schütze, 1999; Evert, 2004; Pecina, 2005)).

According to this description, in our approach the parser is used in the first stage of extraction, for identifying the collocation candidates. A pair of lexical items is selected as a candidate only if there is a syntactic relation holding between the two items (one being the head of the current parse structure, and the other the lexical head of its specifier/complement). Therefore, the criterion we employ for candidate selection is the syntactic proximity, as opposed to the linear proximity used by traditional, window-based methods.

As the parsing goes on, the syntactic word pairs are extracted from the parse structures created, from each head-specifier or head-complement relation. The pairs obtained are then partitioned according to their syntactic configuration (e.g., noun + adjectival or nominal specifier, noun + argument, noun + adjective in predications, verb + adverbial specifier, verb + argument (subject, object), verb + adjunct, etc). Finally, the log-likelihood ratios test (henceforth LLR) (Dunning, 1993) is applied on each set of pairs. We call this method hybrid, since it combines syntactic and statistical information (about word and co-occurrence frequency).

The following examples — which, like all the examples in this paper, are actual extraction results — demonstrate the potential of our system to detect collocation candidates, even if subject to complex syntactic transformations.

²“Ideally, in order to identify lexical relations in a corpus one would need to first parse it to verify that the words are used in a single phrase structure. However, in practice, free-style texts contain a great deal of nonstandard features over which automatic parsers would fail. This fact is being seriously challenged by current research (...), and might not be true in the near future” (Smadja, 1993, 151).

- 1.a) *raise question*: The *question* of political leadership has been *raised* several times by previous speakers.
- 1.b) *play role*: What *role* can Canada's immigration program *play* in helping developing nations... ?
- 1.c) *make mistake*: We could look back and probably see a lot of *mistakes* that all parties including Canada perhaps may have *made*.

3 Multilingual Extraction Results

In this section, we present several extraction results obtained with the system presented in section 2. The experiments were performed on data in the four languages, and involved the following corpora: for English and French, a subpart of the Hansard Corpus of proceedings from the Canadian Parliament; for Italian, documents from the Swiss Parliament; and for Spanish, a news corpus distributed by the Linguistic Data Consortium.

Some statistics on these corpora, some processing details and quantitative results are provided in Table 1. The first row lists the corpora size (in tokens); the next three rows show some parsing statistics³, and the last rows display the number of collocation candidates extracted and of candidates for which the LLR score could be computed⁴.

Statistics	English	French	Spanish	Italian
tokens	3509704	1649914	1023249	287804
sentences	197401	70342	67502	12008
compl. parse	139498	50458	13245	4511
avg. length	17.78	23.46	15.16	23.97
pairs	725025	370932	162802	58258
(extracted)	276670	147293	56717	37914
pairs	633345	308410	128679	47771
(scored)	251046	131384	49495	30586

Table 1: Extraction statistics

In Table 2 we list the top collocations (of length two) extracted for each language. We do not specifically discuss here multilingual issues in collocation extraction; these are dealt with in a separate paper (Seretan and Wehrli, 2006).

³The low rate of completely parsed sentences for Spanish and Italian are due to the relatively reduced coverage of the parsers of these two languages (under development). However, even if a sentence is not assigned a complete parse tree, some syntactic pairs can still be collected from the partial parses.

⁴The log-likelihood ratios score is undefined for those pairs having a cell of the contingency table equal to 0.

Language	Key1	Key2	LLR score
English	federal	government	7229.69
	reform	party	6530.69
	house	common	6006.84
	minister	finance	5829.05
	acting	speaker	5551.09
	red	book	5292.63
	create	job	4131.55
	right	Hon	4117.52
	official	opposition	3640.00
	deputy	speaker	3549.09
French	premier	ministre	4317.57
	bloc	québécois	3946.08
	discours	trône	3894.04
	vérificateur	général	3796.68
	parti	réformiste	3615.04
	gouvernement	fédéral	3461.88
	missile	croisière	3147.42
	Chambre	commune	3083.02
	livre	rouge	2536.94
	secrétaire	parlementaire	2524.68
Spanish	banco	central	4210.48
	millón	dólar	3312.68
	millón	peso	2335.00
	libre	comercio	2169.02
	nuevo	peso	1322.06
	tasa	interés	1179.62
	deuda	externo	1119.91
	cámara	representante	1015.07
	asamblea	ordinario	992.85
	papel	comercial	963.95
Italian	consiglio	federale	3513.19
	scrivere	consiglio	594.54
	unione	europeo	479.73
	servizio	pubblico	452.92
	milione	franco	447.63
	formazione	continuo	388.80
	iniziativa	popolare	383.68
	testo	interpellanza	377.46
	punto	vista	373.24
	scrivere	risposta	348.77

Table 2: Top ten collocations extracted for each language

The collocation pairs obtained were further processed with a procedure of long collocations extraction described elsewhere (Seretan et al., 2003). Some examples of collocations of length 3, 4 and 5 obtained are: *minister of Canadian heritage, house proceed to statement by, secretary to leader of government in house of common* (En), *question adresser à ministre, programme de aide à rénovation résidentielle, agent employer force susceptible causer* (Fr), *bolsa de comercio local, peso en cuota de fondo de inversión, permitir uso de papel de deuda externo* (Sp), *consiglio federale disporre, creazione di nuovo posto di lavoro, costituire fattore penalizzante per regione* (It)⁵.

⁵Note that the output of the procedure contains lemmas rather than inflected forms.

4 Comparative Evaluation Hypotheses

4.1 Does Parsing Really Help?

Extracting collocations from raw text, without pre-processing the source corpora, offers some clear advantages over linguistically-informed methods such as ours, which is based on the syntactic analysis: *speed* (in contrast, parsing large corpora of texts is expected to be much more time consuming), *robustness* (symbolic parsers are often not robust enough for processing large quantities of data), *portability* (no need to a priori define syntactic configurations for collocations candidates).

On the other hand, these basic systems suffer from the combinatorial explosion if the candidate pairs are chosen from a large search space. To cope with this problem, a candidate pair is usually chosen so that both words are inside a context ('collocational') window of a small length. A 5-word window is the norm, while longer windows prove impractical (Dias, 2003).

It has been argued that a window size of 5 is actually sufficient for capturing most of the collocational relations from texts in English. But there is no evidence sustaining that the same holds for other languages, like German or the Romance ones that exhibit freer word order. Therefore, as window-based systems miss the 'long-distance' pairs, their recall is presumably lower than that of parse-based systems. However, the parser could also miss relevant pairs due to inherent analysis errors.

As for precision, the window systems are susceptible to return more noise, produced by the grammatically unrelated pairs inside the collocational window. By dividing the number of grammatical pairs by the total number of candidates considered, we obtain the overall precision with respect to grammaticality; this result is expected to be considerably worse in the case of basic method than for the parse-based methods, just by virtue of the parsing task. As for the overall precision with respect to collocability, we expect the proportional figures to be preserved. This is because the parser-based methods return less, but better pairs (i.e., only the pairs identified as grammatical), and because collocations are a subset of the grammatical pairs.

Summing up, the evaluation hypothesis that can be stated here is the following: parse-based methods outperform basic methods thanks to a drastic reduction of noise. While unquestionable under

the assumption of perfect parsing, this hypothesis has to be empirically validated in an actual setting.

4.2 Is More Data Better Than Better Data?

The hypothesis above refers to the overall precision and recall, that is, relative to the entire list of selected candidates. One might argue that these numbers are less relevant for practice than they are from a theoretical (evaluation) perspective, and that the exact composition of the list of candidates identified is unimportant if only the top results (i.e., those pairs situated above a threshold) are looked at by a lexicographer or an application.

Considering a threshold for the *n*-best candidates works very much in the favor of basic methods. As the amount of data increases, there is a reduction of the noise among the best-scored pairs, which tend to be more grammatical because the likelihood of encountering many similar noisy pairs is lower. However, as the following example shows, noisy pairs may still appear in top, if they occur often in a longer collocation:

- 2.a) les essais du missile de croisière
- 2.b) essai - croisière

The pair *essai - croisière* is marked by the basic systems as a collocation because of the recurrent association of the two words in text as part or the longer collocation *essai du missile de croisière*. It is an grammatically unrelated pair, while the correct pairs reflecting the right syntactic attachment are *essai missile* and *missile (de) croisière*.

We mentioned that parsing helps detecting the 'long-distance' pairs that are outside the limits of the collocational window. Retrieving all such complex instances (including all the extraposition cases) certainly augment the recall of extraction systems, but this goal might seem unjustified, because the risk of not having a collocation represented at all diminishes as more and more data is processed. One might think that systematically missing long-distance pairs might be very simply compensated by supplying the system with more data, and thus that larger data is a valid alternative to performing complex processing.

While we agree that the inclusion of more data compensates for the 'difficult' cases, we do consider this truly helpful in deriving collocational information, for the following reasons: (1) more data means more noise for the basic methods; (2) some collocations might systematically appear in

a complex grammatical environment (such as passive constructions or with additional material inserted between the two items); (3) more importantly, the complex cases not taken into account alter the frequency profile of the pairs concerned.

These observations entitle us to believe that, even when more data is added, the n -best precision might remain lower for the basic methods with respect to the parse-based ones.

4.3 How Real the Counts Are?

Syntactic analysis (including shallower levels of linguistic analysis traditionally used in collocation extraction, such as lemmatization, POS tagging, or chunking) has two main functions.

On the one hand, it guides the extraction system in the candidate selection process, in order to better pinpoint the pairs that might form collocations and to exclude the ones considered as inappropriate (e.g., the pairs combining function words, such as a preposition followed by a determiner).

On the other, parsing supports the association measures that will be applied on the selected candidates, by providing more exact frequency information on words — the inflected forms count as instances of the same lexical item — and on their co-occurrence frequency — certain pairs might count as instance of the same pair, others do not.

In the following example, the pair *loi modifier* is an instance of a subject-verb collocation in 3.a), and of a verb-object collocation type in 3.b). Basic methods are unable to distinguish between the two types, and therefore count them as equivalent.

3.a) *Loi modifiant* la Loi sur la responsabilité civile

3.b) la *loi* devrait être *modifiée*

Parsing helps to create a more realistic frequency profile for the candidate pairs, not only because of the grammaticality constraint it applies on the pairs (wrong pairs are excluded), but also because it can detect the long-distance pairs that are outside the collocational window.

Given that the association measures rely heavily on the frequency information, the erroneous counts have a direct influence on the ranking of candidates and, consequently, on the top candidates returned. We believe that in order to achieve a good performance, extraction systems should be as close as possible to the real frequency counts

and, of course, to the real syntactic interpretation provided in the source texts⁶.

Since parser-based methods rely on more accurate frequency information for words and their co-occurrence than window methods, it follows that the n -best list obtained with the first methods will probably show an increase in quality over the second.

To conclude this section, we enumerate the hypotheses that have been formulated so far: (1) Parse methods provide a noise-freer list of collocation candidates, in comparison with the window methods; (2) Local precision (of best-scored results) with respect to grammaticality is higher for parse methods, since in basic methods some noise still persists, even if more data is included; (3) Local precision with respect to collocability is higher for parse methods, because they use a more realistic image of word co-occurrence frequency.

5 Comparative Evaluation

We compare our hybrid method (based on syntactic processing of texts) against the window method classically used in collocation extraction, from the point of view of their precision with respect to grammaticality and collocability.

5.1 The Method

The n -best extraction results, for a given n (in our experiment, n varies from 50 to 500 at intervals of 50) are checked in each case for grammatical well-formedness and for lexicalization. By lexicalization we mean the quality of a pair to constitute (part of) a multi-word expression — be it compound, collocation, idiom or another type of syntagmatic lexical combination. We avoid giving collocability judgments since the classification of multi-word expressions cannot be made precisely and with objective criteria (McKeown and Radev, 2000). We rather distinguish between lexicalizable and trivial combinations (completely regular productions, such as *big house*, *buy bread*, that do not deserve a place in the lexicon). As in (Choueka, 1988) and (Evert, 2004), we consider that a dominant feature of collocations is that they are unpredictable for speakers and therefore have to be stored into a lexicon.

⁶To exemplify this point: the pair *développement humain* (which has been detected as a collocation by the basic method) looks like a valid expression, but the source text consistently offers a different interpretation: *développement des ressources humaines*.

Each collocation from the n -best list at the different levels considered is therefore annotated with one of the three flags: 1. ungrammatical; 2. trivial combination; 3. multi-word expression (MWE).

On the one side, we evaluate the results of our hybrid, parse-based method; on the other, we simulate a window method, by performing the following steps: POS-tag the source texts; filter the lexical items and retain only the open-class POS; consider all their combinations within a collocational window of length 5; and, finally, apply the log-likelihood ratios test on the pairs of each configuration type.

In accordance with (Evert and Kermes, 2003), we consider that the comparative evaluation of collocation extraction systems should not be done at the end of the extraction process, but separately for each stage: after the candidate selection stage, for evaluating the quality (in terms of grammaticality) of candidates proposed; and after the application of collocability measures, for evaluating the measures applied. In each of these cases, different evaluation methodologies and resources are required. In our case, since we used the same measure for the second stage (the log-likelihood ratios test), we could still compare the final output of basic and parse-based methods, as given by the combination of the first stage with the same collocability measure.

Again, similarly to Krenn and Evert (2001), we believe that the homogeneity of data is important for the collocability measures. We therefore applied the LLR test on our data after first partitioning it into separate sets, according to the syntactical relation holding in each candidate pair. As the data used in the basic method contains no syntactic information, the partitioning was done based on POS-combination type.

5.2 The Data

The evaluation experiment was performed on the whole French corpus used in the extraction experiment (section 2), that is, a subpart of the Hansard corpus of Canadian Parliament proceedings. It contains 112 text files totalling 8.43 MB, with an average of 628.1 sentences/file and 23.46 tokens/sentence (as detected by the parser). The total number of tokens is 1,649,914.

On the one hand, the texts were parsed and 370,932 candidate pairs were extracted using the

hybrid method we presented. Among the pairs extracted, 11.86% (44,002 pairs) were multi-word expressions identified at parse-time, since present in the parser's lexicon. The log-likelihood ratios test was applied on the rest of pairs. A score could be associated to 308,410 of these pairs (corresponding to 131,384 types); for the others, the score was undefined.

On the other hand, the texts were POS-tagged using the same parser as in the first case. If in the first case the candidate pairs were extracted during the parsing, in the second they were generated after the open-class filtering. From 673,789 POS-filtered tokens, a number of 1,024,888 combinations (560,073 types) were created using the 5-length window criterion, while taking care not to cross a punctuation mark. A score could be associated to 1,018,773 token pairs (554,202 types), which means that the candidate list is considerably larger than in the first case. The processing time was more than twice longer than in the first case, because of the large amount of data to handle.

5.3 Results

The 500 best-scored collocations retrieved with the two methods were manually checked by three human judges and annotated, as explained in 5.1, as either ungrammatical, trivial or MWE. The agreement statistics on the annotations for each method are shown in Table 3.

Method	Agr.	1,2,3	1,2	1,3	2,3
parse	observed	285	365	362	340
	k-score	55.4%	62.6%	69%	64%
window	observed	226	339	327	269
	k-score	43.1%	63.8%	61.1%	48%

Table 3: Inter-annotator agreement

For reporting n -best precision results, we used as reference set the annotated pairs on which at least two of the three annotators agreed. That is, from the 500 initial pairs retrieved with each method, 497 pairs were retained in the first case (parse method), and 483 pairs in the second (window method).

Table 4 shows the comparative evaluation results for precision at different levels in the list of best-scored pairs, both with respect to grammaticality and to collocability (or, more exactly, the potential of a pair to constitute a MWE). The numbers show that a drastic reduction of noise is achieved by parsing the texts. The error rate with

<i>n</i>	Precision (gram.)		Precision (MWE)	
	<i>window</i>	<i>parse</i>	<i>window</i>	<i>parse</i>
50	94.0	96.0	80.0	72.0
100	91.0	98.0	75.0	74.0
150	87.3	98.7	72.7	73.3
200	85.5	98.5	70.5	74.0
250	82.8	98.8	67.6	69.6
300	82.3	98.7	65.0	69.3
350	80.3	98.9	63.7	67.4
400	80.0	99.0	62.5	67.0
450	79.6	99.1	61.1	66.0
500	78.3	99.0	60.1	66.0

Table 4: Comparative evaluation results

respect to grammaticality is, on average, 15.9% for the window method; with parsing, it drops to 1.5% (i.e., 10.6 times smaller).

This result confirms our hypothesis regarding the local precision which was stated in section 4.2. Despite the inherent parsing errors, the noise reduction is substantial. It is also worth noting that we compared our method against a rather high baseline, as we made a series of choices susceptible to alleviate the candidates identification with the window-based method: we filtered out function words, we used a parser for POS-tagging (that eliminated POS-ambiguity), and we filtered out cross-punctuation pairs.

As for the MWE precision, the window method performs better for the first 100 pairs⁷; on the remaining part, the parsing-based method is on average 3.7% better. The precision curve for the window method shows a more rapid degradation than it does for the other. Therefore we can conclude that parsing is especially advantageous if one investigates more than the first hundred results (as it seems reasonable for large extraction experiments).

In spite of the rough classification we used in annotation, we believe that the comparison performed is nonetheless meaningful since results should be first checked for grammaticality and 'triviality' before defining more difficult tasks such as collocability.

6 Conclusion

In this paper, we provided both theoretical and empirical arguments in the favor of performing syntactic analysis of texts prior to the extraction of collocations with statistical methods.

⁷A closer look at the data revealed that this might be explained by some inconsistencies between annotations.

Part of the extraction work that, like ours, relies on parsing was cited in section 2. Most often, it concerns chunking rather than complete parsing; specific syntactic configurations (such as adjective-noun, preposition-noun-verb); and languages other than the ones we deal with (usually, English and German). Parsing has been also used after extraction (Smadja, 1993) for filtering out invalid results. We believe that this is not enough and that parsing is required prior to the application of statistical tests, for computing a realistic frequency profile for the pairs tested.

As for evaluation, unlike most of the existing work, we are not concerned here with comparing the performance of association measures (cf. (Evert, 2004; Pecina, 2005) for comprehensive references), but with a contrastive evaluation of syntactic-based and standard extraction methods, combined with the same statistical computation.

Our study finally clear the doubts on the usefulness of parsing for collocation extraction. Previous work that quantified the influence of parsing on the quality of results suggested the performance for tagged and parsed texts is similar (Evert and Kermes, 2003). This result applies to a quite rigid syntactic pattern, namely adjective-noun in German. But a preceding study on noun-verb pairs (Breidt, 1993) came to the conclusion that good precision can only be achieved for German with parsing. Its author had to simulate parsing because of the lack, at the time, of parsing tools for German. Our report, that concerns an actual system and a large data set, validates Breidt's finding for a new language (French).

Our experimental results confirm the hypotheses put forth in section 4, and show that parsing (even if imperfect) benefits to extraction, notably by a drastic reduction of the noise in the top of the significance list. In future work, we consider investigating other levels of the significance list, extending the evaluation to other languages, comparing against shallow-parsing methods instead of the window method, and performing recall-based evaluation as well.

Acknowledgements

We would like to thank Jorge Antonio Leoni de Leon, Mar Ndiaye, Vincenzo Pallotta and Yves Scherrer for participating to the annotation task. We are also grateful to Gabrielle Musillo and to the anonymous reviewers of an earlier version of

this paper for useful comments and suggestions.

References

- Morton Benson. 1990. Collocations and general-purpose dictionaries. *International Journal of Lexicography*, 3(1):23–35.
- Elisabeth Breidt. 1993. Extraction of V-N-collocations from text corpora: A feasibility study for German. In *Proceedings of the Workshop on Very Large Corpora: Academic and Industrial Perspectives*, Columbus, U.S.A.
- Yaacov Choueka. 1988. Looking for needles in a haystack, or locating interesting collocational expressions in large textual databases expressions in large textual databases. In *Proceedings of the International Conference on User-Oriented Content-Based Text and Image Handling*, pages 609–623, Cambridge, MA.
- Anthony P. Cowie. 1978. The place of illustrative material and collocations in the design of a learner's dictionary. In P. Stevens, editor, *In Honour of A.S. Hornby*, pages 127–139. Oxford: Oxford University Press.
- D. Alan Cruse. 1986. *Lexical Semantics*. Cambridge University Press, Cambridge.
- Gaël Dias. 2003. Multiword unit hybrid extraction. In *Proceedings of the ACL Workshop on Multiword Expressions*, pages 41–48, Sapporo, Japan.
- Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.
- Stefan Evert and Hannah Kermes. 2003. Experiments on candidate data for collocation extraction. In *Companion Volume to the Proceedings of the 10th Conference of The European Chapter of the Association for Computational Linguistics*, pages 83–86, Budapest, Hungary.
- Stefan Evert. 2004. *The Statistics of Word Cooccurrences: Word Pairs and Collocations Word Pairs and Collocations*. Ph.D. thesis, University of Stuttgart.
- John Rupert Firth. 1957. *Papers in Linguistics 1934-1951*. Oxford Univ. Press, Oxford.
- Ray Jackendoff. 1997. *The Architecture of the Language Faculty*. MIT Press, Cambridge, MA.
- John S. Justeson and Slava M. Katz. 1995. Technical terminology: Some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1:9–27.
- Brigitte Krenn and Stefan Evert. 2001. Can we do better than frequency? A case study on extracting PP-verb collocations. In *Proceedings of the ACL Workshop on Collocations*, pages 39–46, Toulouse, France.
- Dekang Lin. 1998. Extracting collocations from text corpora. In *First Workshop on Computational Terminology*, pages 57–63, Montreal.
- Christopher Manning and Heinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Mass.
- Kathleen R. McKeown and Dragomir R. Radev. 2000. Collocations. In Robert Dale, Hermann Moisl, and Harold Somers, editors, *A Handbook of Natural Language Processing*, pages 507–523. Marcel Dekker, New York, U.S.A.
- Igor Mel'čuk. 1998. Collocations and lexical functions. In Anthony P. Cowie, editor, *Phraseology. Theory, Analysis, and Applications*, pages 23–53. Clarendon Press, Oxford.
- Brigitte Orliac and Mike Dillinger. 2003. Collocation extraction for machine translation. In *Proceedings of Machine Translation Summit IX*, pages 292–298, New Orleans, Louisiana, U.S.A.
- Darren Pearce. 2001. Synonymy in collocation extraction. In *WordNet and Other Lexical Resources: Applications, Extensions and Customizations (NAACL 2001 Workshop)*, pages 41–46, Carnegie Mellon University, Pittsburgh.
- Pavel Pecina. 2005. An extensive empirical study of collocation extraction methods. In *Proceedings of the ACL Student Research Workshop*, pages 13–18, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2002)*, pages 1–15, Mexico City.
- Violeta Seretan and Eric Wehrli. 2006. Multilingual collocation extraction: Issues and solutions solutions. In *Proceedings of COLING/ACL Workshop on Multilingual Language Resources and Interoperability*, Sydney, Australia, July. To appear.
- Violeta Seretan, Luka Nerima, and Eric Wehrli. 2003. Extraction of multi-word collocations using syntactic bigram composition. In *Proceedings of the Fourth International Conference on Recent Advances in NLP (RANLP-2003)*, pages 424–431, Borovets, Bulgaria.
- Frank Smadja. 1993. Retrieving collocations from text: Xtract. *Computational Linguistics*, 19(1):143–177.
- Eric Wehrli. 2004. Un modèle multilingue d'analyse syntaxique. In A. Auchlin et al., editor, *Structures et discours - Mélanges offerts à Eddy Roulet*, pages 311–329. Éditions Nota bene, Québec.

Scalable Inference and Training of Context-Rich Syntactic Translation Models

Michel Galley*, Jonathan Graehl†, Kevin Knight†‡, Daniel Marcu†‡,
Steve DeNeefe†, Wei Wang‡ and Ignacio Thayer†

*Columbia University †University of Southern California ‡Language Weaver, Inc.
Dept. of Computer Science Information Sciences Institute 4640 Admiralty Way
New York, NY 10027 Marina del Rey, CA 90292 Marina del Rey, CA 90292

galley@cs.columbia.edu, {graehl,knight,marcu,sdeneefe}@isi.edu,
wwang@languageweaver.com, thayer@google.com

Abstract

Statistical MT has made great progress in the last few years, but current translation models are weak on re-ordering and target language fluency. Syntactic approaches seek to remedy these problems. In this paper, we take the framework for acquiring multi-level syntactic translation rules of (Galley et al., 2004) from aligned tree-string pairs, and present two main extensions of their approach: first, instead of merely computing a single derivation that minimally explains a sentence pair, we construct a large number of derivations that include contextually richer rules, and account for multiple interpretations of unaligned words. Second, we propose probability estimates and a training procedure for weighting these rules. We contrast different approaches on real examples, show that our estimates based on multiple derivations favor phrasal re-orderings that are linguistically better motivated, and establish that our larger rules provide a 3.63 BLEU point increase over minimal rules.

1 Introduction

While syntactic approaches seek to remedy word-ordering problems common to statistical machine translation (SMT) systems, many of the earlier models—particularly child re-ordering models—fail to account for human translation behavior. Galley et al. (2004) alleviate this modeling problem and present a method for acquiring millions of syntactic transfer rules from bilingual corpora, which we review below. Here, we make the following new contributions: (1) we show how to acquire larger rules that crucially condition on more syntactic context, and show how to compute multiple derivations for each training example, capturing both large and small rules, as well as multiple interpretations for unaligned words; (2) we develop probability models for these multi-level transfer rules, and give estimation methods for assigning probabilities to very large rule sets. We contrast our work with (Galley et al., 2004), highlight some severe limitations of probability estimates computed from single derivations, and

demonstrate that it is critical to account for many derivations for each sentence pair. We also use real examples to show that our probability models estimated from a large number of derivations favor phrasal re-orderings that are linguistically well motivated. An empirical evaluation against a state-of-the-art SMT system similar to (Och and Ney, 2004) indicates positive prospects. Finally, we show that our contextually richer rules provide a 3.63 BLEU point increase over those of (Galley et al., 2004).

2 Inferring syntactic transformations

We assume we are given a source-language (e.g., French) sentence f , a target-language (e.g., English) parse tree π , whose yield e is a translation of f , and a word alignment a between f and e . Our aim is to gain insight into the process of transforming π into f and to discover grammatically-grounded translation rules. For this, we need a formalism that is expressive enough to deal with cases of syntactic divergence between source and target languages (Fox, 2002): for any given (π, f, a) triple, it is useful to produce a derivation that minimally explains the transformation between π and f , while remaining consistent with a . Galley et al. (2004) present one such formalism (henceforth “GHKM”).

2.1 Tree-to-string alignments

It is appealing to model the transformation of π into f using tree-to-string (**xRs**) transducers, since their theory has been worked out in an extensive literature and is well understood (see, e.g., (Graehl and Knight, 2004)). Formally, transformational rules r_i presented in (Galley et al., 2004) are equivalent to 1-state **xRs** transducers mapping a given pattern (subtree to match in π) to a right hand side string. We will refer to them as $lhs(r_i)$ and $rhs(r_i)$, respectively. For example, some **xRs**

rules may describe the transformation of *does not* into *ne ... pas* in French. A particular instance may look like this:

$$\text{VP}(\text{AUX}(\text{does}), \text{RB}(\text{not}), x_0:\text{VB}) \rightarrow \text{ne}, x_0, \text{pas}$$

$lhs(r_i)$ can be any arbitrary syntax tree fragment. Its leaves are either lexicalized (e.g. *does*) or variables (x_0, x_1 , etc). $rhs(r_i)$ is represented as a sequence of target-language words and variables.

Now we give a brief overview of how such transformational rules are acquired automatically in GHKM.¹ In Figure 1, the $(\pi, \mathbf{f}, \mathbf{a})$ triple is represented as a directed graph G (edges going downward), with no distinction between edges of π and alignments. Each node of the graph is labeled with its **span** and **complement span** (the latter in *italic>* in the figure). The span of a node n is defined by the indices of the first and last word in $\mathbf{f}$ that are reachable from n . The complement span of n is the union of the spans of all nodes n' in G that are neither descendants nor ancestors of n . Nodes of G whose spans and complement spans are non-overlapping form the **frontier set** $F \in G$.

What is particularly interesting about the frontier set? For any frontier of graph G containing a given node $n \in F$, spans on that frontier define an ordering between n and each other frontier node n' . For example, the span of VP[4-5] either precedes or follows, but never overlaps the span of any node n' on any graph frontier. This property does not hold for nodes outside of F . For instance, PP[4-5] and VBG[4] are two nodes of the same graph frontier, but they cannot be ordered because of their overlapping spans.

The purpose of **xRs** rules in this framework is to order constituents along sensible frontiers in G , and all frontiers containing undefined orderings, as between PP[4-5] and VBG[4], must be disregarded during rule extraction. To ensure that **xRs** rules are prevented from attempting to re-order any such pair of constituents, these rules are designed in such a way that variables in their *lhs* can only match nodes of the frontier set. Rules that satisfy this property are said to be **induced** by G .² For example, rule (d) in Table 1 is valid according to GHKM, since the spans corresponding to

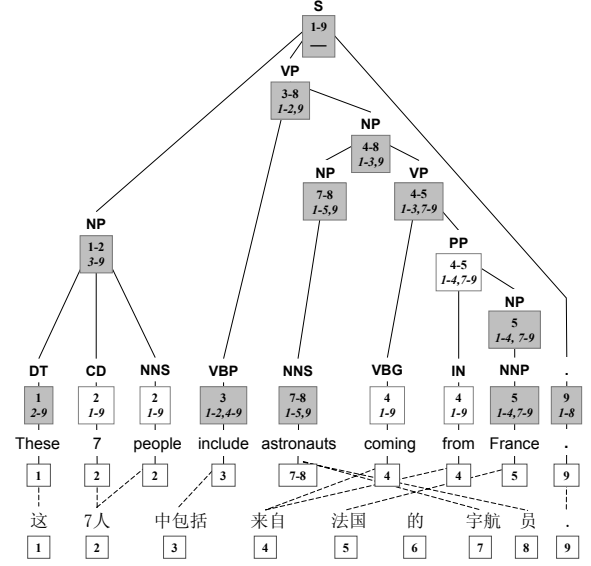


Figure 1: **Spans** and **complement-spans** determine what rules are extracted. Constituents in gray are members of the **frontier set**; a minimal rule is extracted from each of them.

(a)	$S(x_0:\text{NP}, x_1:\text{VP}, x_2:.) \rightarrow x_0, x_1, x_2$
(b)	$\text{NP}(x_0:\text{DT}, \text{CD}(7), \text{NNS}(\text{people})) \rightarrow x_0, 7\text{人}$
(c)	$\text{DT}(\text{these}) \rightarrow \text{这}$
(d)	$\text{VP}(x_0:\text{VBP}, x_1:\text{NP}) \rightarrow x_0, x_1$
(e)	$\text{VBP}(\text{include}) \rightarrow \text{中包括}$
(f)	$\text{NP}(x_0:\text{NP}, x_1:\text{VP}) \rightarrow x_1, \text{的}, x_0$
(g)	$\text{NP}(x_0:\text{NNS}) \rightarrow x_0$
(h)	$\text{NNS}(\text{astronauts}) \rightarrow \text{宇航, 员}$
(i)	$\text{VP}(\text{VBG}(\text{coming}), \text{PP}(\text{IN}(\text{from}), x_0:\text{NP})) \rightarrow \text{来自}, x_0$
(j)	$\text{NP}(x_0:\text{NNP}) \rightarrow x_0$
(k)	$\text{NNP}(\text{France}) \rightarrow \text{法国}$
(l)	$.(.) \rightarrow .$

Table 1: A minimal derivation corresponding to Figure 1.

its *rhs* constituents (VBP[3] and NP[4-8]) do not overlap. Conversely, $\text{NP}(x_0:\text{DT}, x_1:\text{CD}, x_2:\text{NNS})$ is not the *lhs* of any rule extractible from G , since its frontier constituents CD[2] and NNS[2] have overlapping spans.³ Finally, the GHKM procedure produces a single derivation from G , which is shown in Table 1.

The concern in GHKM was to extract minimal rules, whereas ours is to extract rules of any arbitrary size. **Minimal** rules defined over G are those that cannot be decomposed into simpler rules induced by the same graph G , e.g., all rules in Table 1. We call minimal a derivation that only contains minimal rules. Conversely, a **composed** rule results from the composition of two or more minimal rules, e.g., rule (b) and (c) compose into:

$$\text{NP}(\text{DT}(\text{these}), \text{CD}(7), \text{NNS}(\text{people})) \rightarrow \text{这}, 7\text{人}$$

³It is generally reasonable to also require that the root n of $lhs(r_i)$ be part of F , because no rule induced by G can compose with r_i at n , due to the restrictions imposed on the extraction procedure, and r_i wouldn't be part of any valid derivation.

¹Note that we use a slightly different terminology.

²Specifically, an **xRs** rule r_i is extracted from G by taking a subtree $\gamma \in \pi$ as $lhs(r_i)$, appending a variable to each leaf node of γ that is internal to π , adding those variables to $rhs(r_i)$, ordering them in accordance to $\mathbf{a}$, and if necessary inserting any word of $\mathbf{f}$ to ensure that $rhs(r_i)$ is a sequence of contiguous spans (e.g., [4-5][6][7-8] for rule (f) in Table 1).

2.3 Algorithm

The linear-time algorithm presented in GHKM is only a particular case of the more general one we describe here, which is used to extract all rules, minimal and composed, induced by G . Similarly to the GHKM algorithm, ours performs a top-down traversal of G , but differs in the operations it performs at each node $n \in F$: we must explore all subtrees rooted at n , find all consistent ways of attaching unaligned words of $\mathbf{f}$, and build valid derivations in accordance to these attachments.

We use a table **or-dforest** $[x, y, c]$ to store OR-nodes, in which each OR-node can be uniquely defined by a syntactic category c and a span $[x, y]$ (which may cover unaligned words of $\mathbf{f}$). This table is used to prevent the same partial derivation to be followed multiple times (the in-degrees of OR-nodes generally become large with composed rules). Furthermore, to avoid over-generating unaligned words, the root and variables in each rule are represented with their spans. For example, in Figure 2(b), the second and third child of the top-most OR-node respectively span across [4-5][6-8] and [4-6][7-8] (after constituent reordering). In the former case, 的 will eventually be realized in an NP, and in the latter case, in a VP.

The preprocessing step consists of assigning spans and complement spans to nodes of G , in the first case by a bottom-up exploration of the graph, and in the latter by a top-down traversal. To assign complement spans, we assign the complement span of any node n to each of its children, and for each of them, add the span of the child to the complement span of all other children. In another traversal of G , we determine the minimal rule extractible from each node in F .

We explore all tree fragments rooted at n by maintaining an open and a closed queue of rules extracted from n (q_o and q_c). At each step, we pick the smallest rule in q_o , and for each of its variable nodes, try to discover new rules (‘successor rules’) by means of composition with minimal rules, until a given threshold on rule size or maximum number of rules in q_c is reached. There may be more than one successor per rule, since we must account for all possible spans than can be assigned to non-lexical leaves of a rule. Once a threshold is reached, or if the open queue is empty, we connect a new OR-node to all rules that have just been extracted from n , and add it to **or-dforest**. Finally, we proceed recursively, and extract new rules from

each node at the frontier of the minimal rule rooted at n . Once all nodes of F have been processed, the **or-dforest** table contains a representation encoding only valid derivations.

3 Probability models

The overall goal of our translation system is to transform a given source-language sentence $\mathbf{f}$ into an appropriate translation $\mathbf{e}$ in the set $\mathbf{E}$ of all possible target-language sentences. In a noisy-channel approach to SMT, we use Bayes’ theorem and choose the English sentence $\hat{\mathbf{e}} \in \mathbf{E}$ that maximizes:⁴

$$\hat{\mathbf{e}} = \arg \max_{\mathbf{e} \in \mathbf{E}} \left\{ Pr(\mathbf{e}) \cdot Pr(\mathbf{f}|\mathbf{e}) \right\} \quad (1)$$

$Pr(\mathbf{e})$ is our language model, and $Pr(\mathbf{f}|\mathbf{e})$ our translation model. In a grammatical approach to MT, we hypothesize that syntactic information can help produce good translation, and thus introduce dependencies on target-language syntax trees. The function to optimize becomes:

$$\hat{\mathbf{e}} = \arg \max_{\mathbf{e} \in \mathbf{E}} \left\{ Pr(\mathbf{e}) \cdot \sum_{\pi \in \tau(\mathbf{e})} Pr(\mathbf{f}|\pi) \cdot Pr(\pi|\mathbf{e}) \right\} \quad (2)$$

$\tau(\mathbf{e})$ is the set of all English trees that yield the given sentence $\mathbf{e}$. Estimating $Pr(\pi|\mathbf{e})$ is a problem equivalent to syntactic parsing and thus is not discussed here. Estimating $Pr(\mathbf{f}|\pi)$ is the task of syntax-based translation models (SBTM).

Given a rule set R , our SBTM makes the common assumption that left-most compositions of **xRs** rules $\theta_i = r_1 \circ \dots \circ r_n$ are independent from one another in a given derivation $\theta_i \in \Theta$, where Θ is the set of all derivations constructible from $G = (\pi, \mathbf{f}, \mathbf{a})$ using rules of R . Assuming that Λ is the set of all subtree decompositions of π corresponding to derivations in Θ , we define the estimate:

$$Pr(\mathbf{f}|\pi) = \frac{1}{|\Lambda|} \sum_{\theta_i \in \Theta} \prod_{r_j \in \theta_i} p(rhs(r_j)|lhs(r_j)) \quad (3)$$

under the assumption:

$$\sum_{r_j \in R: lhs(r_j)=lhs(r_i)} p(rhs(r_j)|lhs(r_j)) = 1 \quad (4)$$

It is important to notice that the probability distribution defined in Equation 3 requires a normalization factor ($|\Lambda|$) in order to be tight, i.e., sum to 1 over all strings $\mathbf{f}_i \in \mathbf{F}$ that can be derived

⁴We denote general probability distributions with $Pr(\cdot)$ and use $p(\cdot)$ for probabilities assigned by our models.

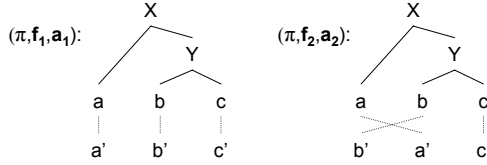


Figure 3: Example corpus.

from π . A simple example suffices to demonstrate it is not tight without normalization. Figure 3 contains a sample corpus from which four rules can be extracted:

- $r_1: X(a, Y(b, c)) \rightarrow a', b', c'$
- $r_2: X(a, Y(b, c)) \rightarrow b', a', c'$
- $r_3: X(a, x_0:Y) \rightarrow a', x_0$
- $r_4: Y(b, c) \rightarrow b', c'$

From Equation 4, the probabilities of r_3 and r_4 must be 1, and those of r_1 and r_2 must sum to 1. Thus, the total probability mass, which is distributed across two possible output strings $a'b'c'$ and $b'a'c'$, is: $p(a'b'c'|\pi) + p(b'a'c'|\pi) = p_1 + p_3 \cdot p_4 + p_2 = 2$, where $p_i = p(rhs(r_i)|lhs(r_i))$.

It is relatively easy to prove that the probabilities of all derivations that correspond to a given decomposition $\lambda_i \in \Lambda$ sum to 1 (the proof is omitted due to constraints on space). From this property we can immediately conclude that the model described by Equation 3 is tight.⁵

We examine two estimates $p(rhs(r)|lhs(r))$. The first one is the relative frequency estimator conditioning on left hand sides:

$$p(rhs(r)|lhs(r)) = \frac{f(r)}{\sum_{r': lhs(r')=lhs(r)} f(r')} \quad (5)$$

$f(r)$ represents the number of times rule r occurred in the derivations of the training corpus.

One of the major negative consequences of extracting only minimal rules from a corpus is that an estimator such as Equation 5 can become extremely biased. This again can be observed from Figure 3. In the minimal-rule extraction of GHKM, only three rules are extracted from the example corpus, i.e. rules r_2 , r_3 , and r_4 . Let's assume now that the triple (π, f_1, a_1) is represented 99 times, and (π, f_2, a_2) only once. Given a tree π , the model trained on that corpus can generate the two strings $a'b'c'$ and $b'a'c'$ only through two derivations, $r_3 \circ r_4$ and r_2 , respectively. Since all rules in that example have probability 1, and

⁵If each tree fragment in π is the lhs of some rule in R , then we have $|\Lambda| = 2^n$, where n is the number of nodes of the frontier set $F \in G$ (each node is a binary choice point).

given that the normalization factor $|\Lambda|$ is 2, both probabilities $p(a'b'c'|\pi)$ and $p(b'a'c'|\pi)$ are 0.5. On the other hand, if all rules are extracted and incorporated into our relative-frequency probability model, r_1 seriously counterbalances r_2 and the probability of $a'b'c'$ becomes: $\frac{1}{2} \cdot (\frac{99}{100} + 1) = .995$ (since it differs from .99, the estimator remains biased, but to a much lesser extent).

An alternative to the conditional model of Equation 3 is to use a joint model conditioning on the root node instead of the entire left hand side:

$$p(r|root(r)) = \frac{f(r)}{\sum_{r': root(r')=root(r)} f(r')} \quad (6)$$

This can be particularly useful if no parser or syntax-based language model is available, and we need to rely on the translation model to penalize ill-formed parse trees. Section 6 will describe an empirical evaluation based on this estimate.

4 EM training

In our previous discussion of parameter estimation, we did not explore the possibility that one derivation in a forest may be much more plausible than the others. If we knew which derivation in each forest was the “true” derivation, then we could straightforwardly collect rule counts off those derivations. On the other hand, if we had good rule probabilities, we could compute the most likely (Viterbi) derivations for each training example. This is a situation in which we can employ EM training, starting with uniform rule probabilities. For each training example, we would like to: (1) score each derivation θ_i as a product of the probabilities of the rules it contains, (2) compute a conditional probability p_i for each derivation θ_i (conditioned on the observed training pair) by normalizing those scores to add to 1, and (3) collect weighted counts for each rule in each θ_i , where the weight is p_i . We can then normalize the counts to get refined probabilities, and iterate; the corpus likelihood is guaranteed to improve with each iteration. While it is infeasible to enumerate the millions of derivations in each forest, Graehl and Knight (2004) demonstrate an efficient algorithm. They also analyze how to train arbitrary tree transducers into two steps. The first step is to build a derivation forest for each training example, where the forest contains those derivations licensed by the (already supplied) transducer's rules. The second step employs EM on those derivation forests, running in time proportional to the size of the

Best minimal-rule derivation (C_m)			$p(r)$
(a)	$S(x_0:NP-C\ x_1:VP\ x_2:.) \rightarrow x_0\ x_1\ x_2$		.845
(b)	$NP-C(x_0:NPB) \rightarrow x_0$		.82
(c)	$NPB(DT(the)\ x_0:NNS) \rightarrow x_0$		.507
(d)	$NNS(gunmen) \rightarrow \text{枪手}$		.559
(e)	$VP(VBD(were)\ x_0:VP-C) \rightarrow x_0$		.434
(f)	$VP-C(x_0:VBN\ x_1:PP) \rightarrow x_1\ x_0$		.374
(g)	$PP(x_0:IN\ x_1:NP-C) \rightarrow x_0\ x_1$		.64
(h)	$IN(by) \rightarrow \text{被}$		.0067
(i)	$NP-C(x_0:NPB) \rightarrow x_0$		.82
(j)	$NPB(DT(the)\ x_0:NN) \rightarrow x_0$		.586
(k)	$NN(police) \rightarrow \text{警方}$		.0429
(l)	$VBN(killed) \rightarrow \text{击毙}$		.0072
(m)	$(.) \rightarrow .$		.981
Best composed-rule derivation (C_4)			$p(r)$
(o)	$S(NP-C(NPB(DT(the)\ NNS(gunmen)))\ x_0:VP\ (.)) \rightarrow \text{枪手}\ x_0\ .$		1
(p)	$VP(VBD(were)\ VP-C(x_0:VBN\ PP(IN(by)\ x_1:NP-C))) \rightarrow \text{被}\ x_1\ x_0$		0.00724
(q)	$NP-C(NPB(DT(the)\ NN(police))) \rightarrow \text{警方}$		0.173
(r)	$VBN(killed) \rightarrow \text{击毙}$		0.00719

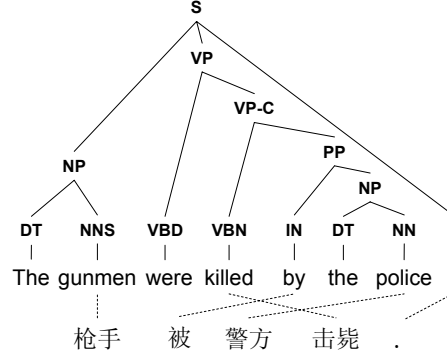


Figure 4: Two most probable derivations for the graph on the right: the top table restricted to minimal rules; the bottom one, much more probable, using a large set of composed rules. Note: the derivations are constrained on the (π, f, a) triple, and thus include some non-literal translations with relatively low probabilities (e.g. *killed*, which is more commonly translated as 死亡).

rule set	nb. of rules	nb. of nodes	deriv-time	EM-time
C_m	4M	192M	2 h.	4 h.
C_3	142M	1255M	52 h.	34 h.
C_4	254M	2274M	134 h.	60 h.

Table 2: Rules and derivation nodes for a 54M-word, 1.95M sentence pair English-Chinese corpus, and time to build derivations (on 10 cluster nodes) and run 50 EM iterations.

forests. We only need to borrow the second step for our present purposes, as we construct our own derivation forests when we acquire our rule set.

A major challenge is to scale up this EM training to large data sets. We have been able to run EM for 50 iterations on our Chinese-English 54-million word corpus. The derivation forests for this corpus contain 2.2 billion nodes; the largest forest contains 1.1 million nodes. The outcome is to assign probabilities to over 254 million rules. Our EM runs with either lhs normalization or lhs -root normalization. In the former case, each lhs has an average of three corresponding rhs 's that compete with each other for probability mass.

5 Model coverage

We now present some examples illustrating the benefit of composed rules. We trained three $p(rhs(r_i)|lhs(r_i))$ models on a 54 million-word English-Chinese parallel corpus (Table 2): the first one (C_m) with only minimal rules, and the two others (C_3 and C_4) additionally considering composed rules with no more than three, respectively four, internal nodes in $lhs(r_i)$. We evaluated these models on a section of the NIST 2002 evaluation corpus, for which we built derivation forests and

$lhs: S(x_0:NP-C\ VP(x_1:VBD\ x_2:NP-C)\ x_3:.)$		
corpus	rhs_i	$p(rhs_i lhs)$
Chinese (minimal)	$x_1\ x_0\ x_2\ x_3$	.3681
	$x_0\ x_1\ \text{第}\ x_3\ x_2$	.0357
	$x_2\ ,\ x_0\ x_1\ x_3$	.0287
	$x_0\ x_1\ \text{第}\ x_3\ x_2\ .$	.0267
Chinese (composed)	$x_0\ x_1\ x_2\ x_3$	.9047
	$x_0\ x_1\ ,\ x_2\ x_3$	.016
	$x_0\ ,\ x_1\ x_2\ x_3$	.0083
	$x_0\ x_1\ \neg\ x_2\ x_3$	.0072
Arabic (composed)	$x_1\ x_0\ x_2\ x_3$	.5874
	$x_0\ x_1\ x_2\ x_3$	.4027
	$x_1\ x_2\ x_0\ x_3$	.0077
	$x_1\ x_0\ x_2\ "x_3$	.0001

Table 3: Our model transforms English subject-verb-object (SVO) structures into Chinese SVO and into Arabic VSO. With only minimal rules, Chinese VSO is wrongly preferred.

extracted the most probable one (Viterbi) for each sentence pair (based on an automatic alignment produced by GIZA). We noticed in general that Viterbi derivations according to C_4 make extensive usage of composed rules, as it is the case in the example in Figure 4. It shows the best derivation according to C_m and C_4 on the unseen (π, f, a) triple displayed on the right. The second derivation ($\log p = -11.6$) is much more probable than the minimal one ($\log p = -17.7$). In the case of C_m , we can see that many small rules must be applied to explain the transformation, and at each step, the decision regarding the re-ordering of constituents is made with little syntactic context. For example, from the perspective of a decoder, the word *by* is immediately transformed into a preposition (IN), but it is in general useful to know which particular function word is present in the sentence to motivate good re-orderings in the up-

lhs_1 : NP-C(x_0 :NPB PP(IN(of) x_1 :NP-C))				(NP- <i>of</i> -NP)			
lhs_2 : PP(IN(of) NP-C(x_0 :NPB PP(IN(of) NP-C(x_1 :NPB x_2 :VP))))				(of-NP-of-NP-VP)			
lhs_3 : VP(VBD(<i>said</i>) SBAR-C(IN(<i>that</i>) x_0 :S-C))				(said-that-S)			
lhs_4 : SBAR(WHADVP(WRB(<i>when</i>)) S-C(x_0 :NP-C VP(VBP(<i>are</i>) x_1 :VP-C)))				(when-NP-are-VP)			
rhs_{1i}	$p(rhs_{1i} lhs_1)$	rhs_{2i}	$p(rhs_{2i} lhs_2)$	rhs_{3i}	$p(rhs_{3i} lhs_3)$	rhs_{4i}	$p(rhs_{4i} lhs_4)$
$x_1 x_0$	.54	x_2 的 x_1 的 x_0	.6754	说, x_0	.6062	在 $x_1 x_0$ 时	.6618
$x_0 x_1$	.2351	在 x_2 的 x_1 的 x_0	.035	说 x_0	.1073	当 $x_1 x_0$ 时	.0724
x_1 的 x_0	.0334	x_2 的 x_1 的 x_0 ,	.0263	表示, x_0	.0591	在 $x_1 x_0$ 时,	.0579
$x_1 x_0$ 的	.026	x_2 的 x_1 的 x_0 有	.0116	他说, x_0	.0234	, 在 $x_1 x_0$ 时	.0289

Table 4: Translation probabilities promote linguistically motivated constituent re-orderings (for lhs_1 and lhs_2), and enable non-constituent (lhs_3) and non-contiguous (lhs_4) phrasal translations.

per levels of the tree. A rule like (e) is particularly unfortunate, since it allows the word *were* to be added without any other evidence that the VP should be in passive voice. On the other hand, the composed-rule derivation of C_4 incorporates more linguistic evidence in its rules, and re-orderings are motivated by more syntactic context. Rule (p) is particularly appropriate to create a passive VP construct, since it expects a Chinese passive marker (被), an NP-C, and a verb in its rhs , and creates the *were ... by* construction at once in the left hand side.

5.1 Syntactic translation tables

We evaluate the promise of our SBTM by analyzing instances of translation tables (t-table). Table 3 shows how a particular form of SVO construction is transformed into Chinese, which is also an SVO language. While the t-table for Chinese composed rules clearly gives good estimates for the “correct” $x_0 x_1$ ordering ($p = .9$), i.e. subject before verb, the t-table for minimal rules unreasonably gives preference to verb-subject ordering ($x_1 x_0$, $p = .37$), because the most probable transformation ($x_0 x_1$) does not correspond to a minimal rule. We obtain different results with Arabic, an VSO language, and our model effectively learns to move the subject after the verb ($p = .59$).

lhs_1 in Table 4 shows that our model is able to learn large-scale constituent re-orderings, such as re-ordering NPs in a NP-*of*-NP construction, and put the modifier first as it is more commonly the case in Chinese ($p = .54$). If more syntactic context is available as in lhs_2 , our model provides much sharper estimates, and appropriately reverses the order of three constituents with high probability ($p = .68$), inserting modifiers first (possessive markers 的 are needed here for better syntactic disambiguation).

A limitation of earlier syntax-based systems is their poor handling of non-constituent phrases. Table 4 shows that our model can learn rules for

such phrases, e.g., *said that* (lhs_3). While the *that* has no direct translation, our model effectively learns to separate 说 (said) from the relative clause with a comma, which is common in Chinese.

Another promising prospect of our model seems to lie in its ability to handle non-contiguous phrases, a feature that state of the art systems such as (Och and Ney, 2004) do not incorporate. The *when*-NP-*are*-VP construction of lhs_4 presents such a case. Our model identifies that *are* needs to be deleted, that *when* translates into the phrase 在 ... 时, and that the NP needs to be moved after the VP in Chinese ($p = .66$).

6 Empirical evaluation

The task of our decoder is to find the most likely English tree π that maximizes all models involved in Equation 2. Since **xRs** rules can be converted to context-free productions by increasing the number of non-terminals, we implemented our decoder as a standard CKY parser with beam search. Its rule binarization is described in (Zhang et al., 2006).

We compare our syntax-based system against an implementation of the alignment template (AlTemp) approach to MT (Och and Ney, 2004), which is widely considered to represent the state of the art in the field. We registered both systems in the NIST 2005 evaluation; results are presented in Table 5. With a difference of 6.4 BLEU points for both language pairs, we consider the results of our syntax-based system particularly promising, since these are the highest scores to date that we know of using linguistic syntactic transformations. Also, on the one hand, our AlTemp system represents quite mature technology, and incorporates highly tuned model parameters. On the other hand, our syntax decoder is still work in progress: only one model was used during search, i.e., the EM-trained root-normalized SBTM, and as yet no language model is incorporated in the search (whereas the search in the AlTemp system uses two phrase-based translation models and

	Syntactic	AlTemp
Arabic-to-English	40.2	46.6
Chinese-to-English	24.3	30.7

Table 5: BLEU-4 scores for the 2005 NIST test set.

	C_m	C_3	C_4
Chinese-to-English	24.47	27.42	28.1

Table 6: BLEU-4 scores for the 2002 NIST test set, with rules of increasing sizes.

12 other feature functions). Furthermore, our decoder doesn’t incorporate any syntax-based language model, and admittedly our ability to penalize ill-formed parse trees is still limited.

Finally, we evaluated our system on the NIST-02 test set with the three different rule sets (see Table 6). The performance with our largest rule set represents a 3.63 BLEU point increase (14.8% relative) compared to using only minimal rules, which indicates positive prospects for using even larger rules. While our rule inference algorithm scales to higher thresholds, one important area of future work will be the improvement of our decoder, conjointly with analyses of the impact in terms of BLEU of contextually richer rules.

7 Related work

Similarly to (Poutsma, 2000; Wu, 1997; Yamada and Knight, 2001; Chiang, 2005), the rules discussed in this paper are equivalent to productions of synchronous tree substitution grammars. We believe that our tree-to-string model has several advantages over tree-to-tree transformations such as the ones acquired by Poutsma (2000). While tree-to-tree grammars are richer formalisms that provide the potential benefit of rules that are linguistically better motivated, modeling the syntax of both languages comes as an extra cost, and it is admittedly more helpful to focus our syntactic modeling effort on the target language (e.g., English) in cases where it has syntactic resources (parsers and treebanks) that are considerably more available than for the source language. Furthermore, we think there is, overall, less benefit in modeling the syntax of the source language, since the input sentence is fixed during decoding and is generally already grammatical.

With the notable exception of Poutsma, most related works rely on models that are restricted to synchronous context-free grammars (SCFG). While the state-of-the-art hierarchical SMT system (Chiang, 2005) performs well despite stringent constraints imposed on its context-free gram-

mar, we believe its main advantage lies in its ability to extract hierarchical rules across phrasal boundaries. Context-free grammars (such as Penn Treebank and Chiang’s grammars) make independence assumptions that are arguably often unreasonable, but as our work suggests, relaxations of these assumptions by using contextually richer rules results in translations of increasing quality. We believe it will be beneficial to account for this finding in future work in syntax-based SMT and in efforts to improve upon (Chiang, 2005).

8 Conclusions

In this paper, we developed probability models for the multi-level transfer rules presented in (Galley et al., 2004), showed how to acquire larger rules that crucially condition on more syntactic context, and how to pack multiple derivations, including interpretations of unaligned words, into derivation forests. We presented some theoretical arguments for not limiting extraction to minimal rules, validated them on concrete examples, and presented experiments showing that contextually richer rules provide a 3.63 BLEU point increase over the minimal rules of (Galley et al., 2004).

Acknowledgments

We would like to thank anonymous reviewers for their helpful comments and suggestions. This work was partially supported under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022.

References

- D. Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL*.
- H. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proc. of EMNLP*, pages 304–311.
- M. Galley, M. Hopkins, K. Knight, and D. Marcu. 2004. What’s in a translation rule? In *Proc. of HLT/NAACL-04*.
- J. Graehl and K. Knight. 2004. Training tree transducers. In *Proc. of HLT/NAACL-04*, pages 105–112.
- F. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.
- A. Poutsma. 2000. Data-oriented translation. In *Proc. of COLING*, pages 635–641.
- D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.
- K. Yamada and K. Knight. 2001. A syntax-based statistical translation model. In *Proc. of ACL*, pages 523–530.
- H. Zhang, L. Huang, D. Gildea, and K. Knight. 2006. Synchronous binarization for machine translation. In *Proc. of HLT/NAACL*.

Modelling lexical redundancy for machine translation

David Talbot and Miles Osborne

School of Informatics, University of Edinburgh
2 Buccleuch Place, Edinburgh, EH8 9LW, UK

d.r.talbot@sms.ed.ac.uk, miles@inf.ed.ac.uk

Abstract

Certain distinctions made in the lexicon of one language may be *redundant* when translating into another language. We quantify redundancy among source types by the similarity of their distributions over target types. We propose a language-independent framework for minimising *lexical redundancy* that can be optimised directly from parallel text. Optimisation of the source lexicon for a given target language is viewed as model selection over a set of cluster-based translation models.

Redundant distinctions between types may exhibit monolingual regularities, for example, inflexion patterns. We define a prior over model structure using a Markov random field and learn features over sets of monolingual types that are predictive of bilingual redundancy. The prior makes model selection more robust without the need for language-specific assumptions regarding redundancy. Using these models in a phrase-based SMT system, we show significant improvements in translation quality for certain language pairs.

1 Introduction

Data-driven machine translation (MT) relies on models that can be efficiently estimated from parallel text. Token-level independence assumptions based on word-alignments can be used to decompose parallel corpora into manageable units for parameter estimation. However, if training data is scarce or language pairs encode significantly different information in the lexicon, such as Czech and English, additional independence assumptions may assist the model estimation process.

Standard statistical translation models use separate parameters for each pair of source and target

types. In these models, distinctions in either lexicon that are redundant to the translation process will result in unwarranted model complexity and make parameter estimation from limited parallel data more difficult. A natural way to eliminate such *lexical redundancy* is to group types into homogeneous clusters that do not differ significantly in their distributions over types in the other language. Cluster-based translation models capture the corresponding independence assumptions.

Previous work on bilingual clustering has focused on coarse partitions of the lexicon that resemble automatically induced part-of-speech classes. These were used to model generic word-alignment patterns such as noun-adjective re-ordering between English and French (Och, 1998). In contrast, we induce fine-grained partitions of the lexicon, conceptually closer to automatic lemmatisation, optimised specifically to assign translation probabilities. Unlike lemmatisation or stemming, our method specifically quantifies lexical redundancy in a bilingual setting and does not make language-specific assumptions.

We tackle the problem of redundancy in the translation lexicon via Bayesian model selection over a set of cluster-based translation models. We search for the model, defined by a clustering of the source lexicon, that maximises the *marginal likelihood* of target tokens in parallel data. In this optimisation, source types are combined into clusters if their distributions over target types are too similar to warrant distinct parameters.

Redundant distinctions between types may exhibit regularities within a language, for instance, inflexion patterns. These can be used to guide model selection. Here we show that the inclusion of a model ‘prior’ over the lexicon structure leads to more robust translation models. Although *a priori* we do not know which monolingual features characterise redundancy for a given language pair, by defining a model over the prior monolingual

space of source types and cluster assignments, we can introduce an inductive bias that allows clustering decisions in different parts of the lexicon to influence one another via *monolingual* features. We use an EM-type algorithm to learn weights for a Markov random field parameterisation of this prior over lexicon structure.

We obtain significant improvements in translation quality as measured by BLEU, incorporating these optimised model within a phrase-based SMT system for three different language pairs. The MRF prior improves the results and picks up features that appear to agree with linguistic intuitions of redundancy for the language pairs considered.

2 Lexical redundancy between languages

In statistical MT, the source and target lexicons are usually defined as the sets of distinct types observed in the parallel training corpus for each language. Such models may not be optimal for certain language pairs and training regimes.

A word-level statistical translation model approximates the probability $Pr(E|F)$ that a source type indexed by F will be translated as a target type indexed by E . Standard models, e.g. Brown et al. (1993), consist of discrete probability distributions with separate parameters for each unique pairing of a source and target types; no attempt is made to leverage structure within the event spaces $\mathcal{E}$ and $\mathcal{F}$ during parameter estimation. This results in a large number of parameters that must be estimated from limited amounts of parallel corpora.

We refer to distinctions made between lexical types in one language that do not result in different distributions over types in the other language as *lexically redundant* for the language pair. Since the role of the translation model is to determine a distribution over target types given a source type, when the corresponding target distributions do not vary significantly over a set of source types, the model gains nothing by maintaining a distinct set of parameters for each member of this set.

Lexical redundancy may arise when languages differ in the specificity with which they refer to the same concepts. For instance, colours of the spectrum may be partitioned differently (e.g. *blue* in English v.s. *sinii* and *goluboi* in Russian). It will also arise when languages explicitly encode different information in the lexicon. For example, translating from French to English, a standard model would treat the following pairs of source and tar-

get types as distinct events with entirely unrelated parameters: $(vert, green)$, $(verte, green)$, $(verts, green)$ and $(vertes, green)$. Here the French types differ only in their final suffixes due to adjectival agreement. Since there is no equivalent mechanism in English, these distinctions are redundant with respect to this target language.

Distinctions that are redundant in the source lexicon when translating into one language may, however, be significant when translating into another. For instance, the French adjectival number agreement (the addition of an *s*) may be significant when translating to Russian which also marks adjectives for number (the inflexion to *-ye*).

We can remove redundancy from the translation model by conflating redundant types, e.g. $\overline{vert} \doteq \{vert, verte, verts, vertes\}$, and averaging bilingual statistics associated with these events.

3 Eliminating redundancy in the model

Redundancy in the translation model can be viewed as unwarranted model complexity. A cluster-based translation model defined via a hard-clustering of the lexicon can reduce this complexity by introducing additional independence assumptions: given the source cluster label, c_j , the target type, e_i , is assumed to be independent of the exact source type, f_j , observed, i.e., $p(e_i|f_j) \approx p(e_i|c_j)$. Optimising the model for lexical redundancy can be viewed as model selection over a set of such cluster-based translation models.

We formulate model search as a *maximum a posteriori* optimisation: the data-dependent term, $p(D|C)$, quantifies evidence provided for a model, C , by bilingual training data, D , while the prior, $p(C)$, can assert a preference for a particular model structure (clustering of the source lexicon) on the basis of monolingual features. Both terms have parameters that are estimated from data. Formally, we search for C^* ,

$$\begin{aligned} C^* &= \arg \max_C p(C|D) \\ &= \arg \max_C p(C)p(D|C). \end{aligned} \quad (1)$$

Evaluating the data-dependent term, $p(D|C)$, for different partitions of the source lexicon, we can compare how well different models predict the target tokens aligned in a parallel corpus. This term will prefer models that group together source types with similar distributions over target types. By using the *marginal likelihood* (integrating out the parameters of the translation model) to calculate

$p(D|C)$, we can account explicitly for the complexity of the translation model and compare models with different *numbers* of clusters as well as different assignments of types to clusters.

In addition to an implicit uniform prior over cluster labels as in k -means clustering (e.g. Chou (1991)), we also consider a Markov random field (MRF) parameterisation of the $p(C)$ term to capture monolingual regularities in the lexicon. The MRF induces dependencies between clustering decisions in different parts of the lexicon via a monolingual feature space biasing the search towards models that exhibit monolingual regularities. Rather than assuming *a priori* knowledge of redundant distinctions in the source language, we use an EM algorithm to update parameters for features defined over sets of source types on the basis of existing cluster assignments. While initially the model search will be guided only by information from the bilingual statistics in $p(D|C)$, monolingual regularities in the lexicon, such as inflexion patterns, may gradually be propagated through the model as $p(C)$ becomes informative. Our experiments suggest that the MRF prior enables more robust model selection.

As stated, the model selection procedure accounts for redundancy in the source lexicon using the target distributions. The target lexicon can be optimised analogously. Clustering target types allows the implementation of independence assumptions asserting that the exact specification of a target type is independent of the source type given knowledge of the target cluster label. For example, when translating an English adjective into French it may be more efficient to use the translation model to specify only that the translation lies within a certain set of French adjectives, corresponding to a single lemma, and have the language model select the exact form. Our experiments suggest that it can be useful to account for redundancy in both languages in this way; this can be incorporated simply within our optimisation procedure.

In Section 3.1 we describe the bilingual marginal likelihood, $p(D|C)$, clustering procedure; in Section 3.2 we introduce the MRF parameterisation of the prior, $p(C)$, over model structure; and in Section 3.3, we describe algorithmic approximations.

3.1 Bilingual model selection

Assume we are optimising the source lexicon (the target lexicon is optimised analogously). A clus-

tering of the lexicon is a unique mapping $C_F : \mathcal{F} \rightarrow \mathcal{C}_F$ defined for all $f \in \mathcal{F}$ where, in addition to all source types observed in the parallel training corpus, $\mathcal{F}$ may include items seen in other monolingual corpora (and, in the case of the source lexicon only, the development and test data). The standard SMT lexicon can be viewed as a clustering with each type observed in the parallel training corpus assigned to a distinct cluster and all other types assigned to a single ‘unknown word’ cluster.

We optimise a conditional model of target tokens from word-aligned parallel corpora, $D = \{D_{c_0}, \dots, D_{c_N}\}$, where D_{c_i} represents the set of target words that were aligned to the set of source types in cluster c_i . We assume that each target token in the corpus is generated conditionally *i.i.d.* given the cluster label of the source type to which it is aligned. Sufficient statistics for this model consist of co-occurrence counts of source and target types summed across each source cluster,

$$\#_{c_f}(e) \doteq \sum_{f' \in c_f} \#(e, f'). \quad (2)$$

Maximising the likelihood of the data under this model would require us to specify the number of clusters (the size of the lexicon) in advance. Instead we place a Dirichlet prior parameterised by α^1 over the translation model parameters of each cluster, $\mu_{c_f, e}$, defining the conditional distributions over target types. Given a clustering, the Dirichlet prior, and independent parameters, the distribution over data and parameters factorises,

$$\begin{aligned} p(D, \mu | C_F, \alpha) &= \prod_{c_f \in \mathcal{C}_F} p(D_{c_f}, \mu_{c_f} | c_f, \alpha) \\ &\propto \prod_{c_f \in \mathcal{C}_F} \prod_{e \in \mathcal{E}} \mu_{c_f, e}^{\alpha - 1 + \#_{c_f}(e)} \end{aligned}$$

We optimise cluster assignments with respect to the *marginal likelihood* which averages the likelihood of the set of counts assigned to a cluster, D_{c_f} , under the current model over the prior,

$$p(D_{c_f} | \alpha, c_f) = \int p(\mu_{c_f} | \alpha) p(D_{c_f} | \mu_{c_f}, c_f) d\mu_{c_f}.$$

This can be evaluated analytically for a Dirichlet prior with multinomial parameters.

Assuming a (fixed) uniform prior over model structure, $p(C)$, model selection involves iteratively re-assigning source types to clusters such as to maximise the marginal likelihood. Re-assignments may alter the total number of clusters

¹Distinct from the prior over model structure, $p(C)$.

at any point. Updates can be calculated locally, for instance, given the sets of target tokens D_{c_i} and D_{c_j} aligned to source types currently in clusters c_i and c_j , the change in log marginal likelihood if clusters c_i and c_j are merged into cluster $\bar{c}$ is,

$$\Delta_{c_i, c_j \rightarrow \bar{c}} = \log \frac{p(D_{\bar{c}} | \alpha, \bar{c})}{p(D_{c_i} | \alpha, c_i) p(D_{c_j} | \alpha, c_j)}, \quad (3)$$

which is a *Bayes factor* in favour of the hypothesis that D_{c_i} and D_{c_j} were sampled from the same distribution (Wolpert, 1995). Unlike its equivalent in maximum likelihood clustering, Eq.(3) may assume positive values favouring a smaller number of clusters when the data does not support a more complex hypothesis. The more complex model, with c_i and c_j modelled separately, is penalised for being able to model a wider range of data sets.

The hyperparameter, α , is tied across clusters and taken to be proportional to the marginal (the ‘background’) distribution over target types in the corpus. Under this prior, source types aligned to the same target types, will be clustered together more readily if these target types are less frequent in the corpus as a whole.

3.2 Markov random field model prior

As described above we consider a Markov random field (MRF) parameterisation of the prior over model structure, $p(C)$. This defines a distribution over cluster assignments of the source lexicon as a whole based solely on monolingual characteristics of the lexical types and the relations between their respective cluster assignments.

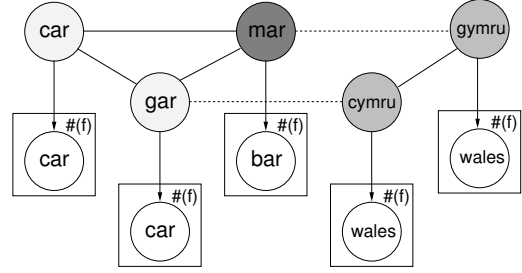
Viewed as graph, each variable in the MRF is modelled as conditionally independent of all other variables given the values of its neighbours (the Markov property; (Geman and Geman, 1984)). Each variable in the MRF prior corresponds to a lexical source type and its cluster assignment. Fig. 1 shows a section of the complete model including the MRF prior for a Welsh source lexicon; shading denotes cluster assignments and English target tokens are shown as directed nodes.² From the Markov property it follows that this prior decomposes over neighbourhoods,

$$p_{\text{MRF}}(C) \propto e^{\beta \sum_{f \in \mathcal{F}} \sum_{f' \in \mathcal{N}_f} \sum_i \lambda_i \psi_i(f, f', c_f, c_{f'})}$$

Here $\mathcal{N}_f$ is the set of neighbours of source type f ; i indexes a set of functions $\psi_i(\cdot)$ that pick out features of a clique; each function has a parameter λ_i

²The *plates* represent repeated sampling; each Welsh source type may be aligned to multiple English tokens.

Figure 1: Model with Markov random field prior



that we learn from the data; these are tied across the graph. β is a free parameter used to control the overall contribution of the prior in Eq. (1). Here features are defined over pairs of types but higher-order interactions can also be modelled. We only consider ‘positive’ prior knowledge that is indicative of redundancy among source types. Hence all features are non-zero only when their arguments are assigned to the same cluster.

Features can be defined over any aspects of the lexicon; in our experiments we use binary features over constrained string edits between types. The following feature would be 1, for instance, if the Welsh types *cymru* and *gymru* (see Fig. 1), were assigned to the same cluster.³

$$\psi_1(f_i = (c \sim) \wedge f_j = (g \sim) \wedge c_i = c_j)$$

Setting the parameters of the MRF prior over this feature space by hand would require *a priori* knowledge of redundancies for the language pair. In the absence of such knowledge, we use an iterative EM algorithm to update the parameters on the basis of the previous solution to the bilingual clustering procedure. EM parameter estimation forces the cluster assignments of the MRF prior to agree with those obtained on the basis of bilingual data using monolingual features alone. Since features are tied across the MRF, patterns that characterise redundant relations between types will be re-enforced across the model. For instance (see Fig. 1), if *cymru* and *gymru* are clustered together, the parameter for feature ψ_1 , shown above, may increase. This induces a prior preference for *car* and *gar* to form a cluster on subsequent iterations. A similar feature defined for *mar* and *gar* in the *a priori* string edit feature space, on the other hand, may remain uninformative if not observed frequently on pairs of types assigned to the same clusters. In this way, the model learns to

³Here $\sim$ matches a common substring of both arguments.

generalise language-specific redundancy patterns from a large *a priori* feature space. Changes in the prior due to re-assignments can be calculated locally and combined with the marginal likelihood.

3.3 Algorithmic approximations

The model selection procedure is an EM algorithm. Each source type is initially assigned to its own cluster and the MRF parameters, λ_i , are initialised to zero. A greedy E-step iteratively re-assigns each source type to the cluster that maximises Eq. (1); cluster statistics are updated after any re-assignment. To reduce computation, we only consider re-assignments that would cause at least one (non-zero) feature in the MRF to fire, or to clusters containing types sharing target word-alignments with the current type; types may also be re-assigned to a cluster of their own at any iteration. When clustering both languages simultaneously, we average ‘target’ statistics over the number of events in each ‘target’ cluster in Eq. (2).

We re-estimate the MRF parameters after each pass through the vocabulary. These are updated according to MLE using a *pseudolikelihood approximation* (Besag, 1986). Since MRF parameters can only be non-zero for features observed on types clustered together during an E-step, we use lazy instantiation to work with a large implicit feature set defined by a constrained string edit.

The algorithm has two free parameters: α determining the strength of the Dirichlet prior used in the marginal likelihood, $p(D|C)$, and β which determines the contribution of $p_{\text{MRF}}(C)$ to Eq. (1).

4 Experiments

Phrase-based SMT systems have been shown to outperform word-based approaches (Koehn et al., 2003). We evaluate the effects of lexicon model selection on translation quality by considering two applications within a phrase-based SMT system.

4.1 Applications to phrase-based SMT

A phrase-based translation model can be estimated in two stages: first a parallel corpus is aligned at the word-level and then phrase pairs are extracted (Koehn et al., 2003). Aligning tokens in parallel sentences using the IBM Models (Brown et al., 1993), (Och and Ney, 2003) may require less information than full-blown translation since the task is constrained by the source and target tokens present in each sentence pair. In the phrase-level translation table, however, the model must assign

Source	Tokens	Types	Singletons	Test	OOV
Czech	468K	54K	29K	6K	469
French	5682K	53K	19K	16K	112
Welsh	4578K	46K	18K	15K	64

Table 1: Parallel corpora used in the experiments.

probabilities to a potentially unconstrained set of target phrases. We anticipate the optimal model sizes to be different for these two tasks.

We can incorporate an optimised lexicon at the word-alignment stage by mapping tokens in the training corpus to their cluster labels. The mapping will not change the number of tokens in a sentence, hence the word-alignments can be associated with the original corpus (see Exp. 1).

To extrapolate a mapping over phrases from our type-level models we can map each type within a phrase to its corresponding cluster label. This, however, results in a large number of distinct phrases being collapsed down to a single ‘clustered phrase’. Using these directly may spread probability mass too widely. Instead we use them to smooth the phrase translation model (see Exp. 2). Here we consider a simple interpolation scheme; they could also be used within a backoff model (Yang and Kirchhoff, 2006).

4.2 Experimental set-up

The system we use is described in (Koehn, 2004). The phrase-based translation model includes phrase-level and lexical weightings in both directions. We use the decoder’s default behaviour for unknown words copying them verbatim to the output. Smoothed trigram language models are estimated on training sections of the parallel corpus.

We used the parallel sections of the Prague Treebank (Cmejrek et al., 2004), French and English sections of the Europarl corpus (Koehn, 2005) and parallel text from the Welsh Assembly⁴ (see Table1). The source languages, Czech, French and Welsh, were chosen on the basis that they may exhibit different degrees of redundancy with respect to English and that they differ morphologically. Only the Czech corpus has explicit morphological annotation.

4.3 Models

All models used in the experiments are defined as mappings of the source and target vocabularies. The target vocabulary includes all distinct types

⁴This Welsh-English parallel text is in the public domain. Contact the first author for details.

seen in the training corpus; the source vocabulary also includes types seen only in development and test data. Free parameters were set to maximize our evaluation metric, BLEU, on development data. The results are reported on the test sets (see Table 1). The baseline mappings used were:

- *standard*: the identity mapping;
- *max-pref*: a prefix of no more than n letters;
- *min-freq*: a prefix with a frequency of at least n in the parallel training corpus.
- *lemmatize*: morphological lemmas (Czech)

standard corresponds to the standard SMT lexicon. *max-pref* and *min-freq* are both simple stemming algorithms that can be applied to raw text. These mappings result in models defined over fewer distinct events that will have higher frequencies; *min-freq* optimises the latter directly. We optimise over (possibly different) values of n for source and target languages. The *lemmatize* mapping which maps types to their lemmas was only applicable to the Czech corpus.

The optimised lexicon models define mappings directly via their clusterings of the vocabulary. We consider the following four models:

- *src*: clustered source lexicon;
- *src+mrf*: as *src* with MRF prior;
- *src+trg*: clustered source and target lexicons;
- *src+trg+mrf*: as *src+trg* with MRF priors.

In each case we optimise over α (a single value for both languages) and, when using the MRF prior, over β (a single value for both languages).

4.4 Experiments

The two sets of experiments evaluate the baseline models and optimised lexicon models during word-alignment and phrase-level translation model estimation respectively.

- Exp. 1: map the parallel corpus, perform word-alignment; estimate the phrase translation model using the original corpus.
- Exp. 2: smooth the phrase translation model,

$$p(\mathbf{e}|\mathbf{f}) = \frac{\#(\mathbf{e}, \mathbf{f}) + \gamma\#(\mathbf{c}_e, \mathbf{c}_f)}{\#(\mathbf{f}) + \gamma\#(\mathbf{c}_f)}$$

Here $\mathbf{e}$, $\mathbf{f}$ and $\mathbf{c}_e$, $\mathbf{c}_f$ are phrases mapped under the *standard* model and the model being tested respectively; γ is set once for all

experiments on development data. Word-alignments were generated using the optimal *max-pref* mapping for each training set.

5 Results

Table 2 shows the changes in BLEU when we incorporate the lexicon mappings during the word-alignment process. The standard SMT lexicon model is not optimal, as measured by BLEU, for any of the languages or training set sizes considered. Increases over this baseline, however, diminish with more training data. For both Czech and Welsh, the explicit model selection procedure that we have proposed results in better translations than all of the baseline models when the MRF prior is used; again these increases diminish with larger training sets. We note that the stemming baseline models appear to be more effective for Czech than for Welsh. The impact of the MRF prior is also greater for smaller training sets.

Table 3 shows the results of using these models to smooth the phrase translation table.⁵ With the exception of Czech, the improvements are smaller than for Exp 1. For all source languages and models we found that it was optimal to leave the target lexicon unmapped when smoothing the phrase translation model.

Using *lemmatize* for word-alignment on the Czech corpus gave BLEU scores of 32.71 and 37.21 for the 10K and 21K training sets respectively; used to smooth the phrase translation model it gave scores of 33.96 and 37.18.

5.1 Discussion

Model selection had the largest impact for smaller data sets suggesting that the complexity of the standard model is most excessive in sparse data conditions. The larger improvements seen for Czech and Welsh suggest that these languages encode more redundant information in the lexicon with respect to English. Potential sources could be grammatical case markings (Czech) and mutation patterns (Welsh). The impact of the MRF prior for smaller data sets suggests it overcomes sparsity in the bilingual statistics during model selection.

The location of redundancies, in the form of case markings, at the *ends* of words in Czech as assumed by the stemming algorithms may explain why these performed better on this language than

⁵The *standard* model in Exp. 2 is equivalent to the optimised *max-pref* in Exp. 1.

Table 2: BLEU scores with optimised lexicon applied during word-alignment (Exp. 1)

Model	Czech-English		French-English				Welsh-English			
	10K sent.	21K	10K	25K	100K	250K	10K	25K	100K	250K
standard	32.31	36.17	20.76	23.17	26.61	27.63	35.45	39.92	45.02	46.47
max-pref	34.18	37.34	21.63	23.94	26.45	28.25	35.88	41.03	44.82	46.11
min-freq	33.95	36.98	21.22	23.77	26.74	27.98	36.23	40.65	45.38	46.35
src	33.95	37.27	21.43	24.42	26.99	27.82	36.98	40.98	45.81	46.45
src+mrf	33.97	37.89	21.63	24.38	26.74	28.39	37.36	41.13	46.50	46.56
src+trg	34.24	38.28	22.05	24.02	26.53	27.80	36.83	41.31	45.22	46.51
src+trg+mrf	34.70	38.44	22.33	23.95	26.69	27.75	37.56	42.19	45.18	46.48

Table 3: BLEU scores with optimised lexicon used to smooth phrase-based translation model (Exp. 2)

Model	Czech-English		French-English				Welsh-English			
	10K sent.	21K	10K	25K	100K	250K	10K	25K	100K	250K
(standard) ⁵	34.18	37.34	21.63	23.94	26.45	28.25	35.88	41.03	44.82	46.11
max-pref	35.63	38.81	22.49	24.10	26.99	28.26	37.31	40.09	45.57	46.41
min-freq	34.65	37.75	21.14	23.41	26.29	27.47	36.40	40.84	45.75	46.45
src	34.38	37.98	21.28	24.17	26.88	28.35	36.94	39.99	45.75	46.65
src+mrf	36.24	39.70	22.02	24.10	26.82	28.09	37.81	41.04	46.16	46.51

Table 4: System output (Welsh 25K; Exp. 2)

Src	ehangu o ffilm i deledu.
Ref	an expansion from film into television.
standard	expansion of footage to <i>deledu</i> .
max-pref	expansion of <i>ffilm</i> to television.
src+mrf	expansion of film to television.
Src	yw gwarchod cymru fel gwlad brydferth
Ref	safeguarding wales as a picturesque country
standard	protection of wales as a country <i>brydferth</i>
max-pref	protection of wales as a country <i>brydferth</i>
src+mrf	protecting wales as a beautiful country
Src	cynhyrchu canlyniadau llai na pherfffaith
Ref	produces results that are less than perfect
standard	produce results less than <i>pherfffaith</i>
max-pref	produce results less than <i>pherfffaith</i>
src+mrf	generates less than perfect results
Src	y dynodiad o graidd y broblem
Ref	the identification of the nub of the problem
standard	the <i>dynodiad</i> of the heart of the problem
max-pref	the <i>dynodiad</i> of the heart of the problem
src+mrf	the identified crux of the problem

on Welsh. The highest scoring features in the MRF (see Table 5) show that Welsh redundancies, on the other hand, are primarily between *initial* characters. Inspection of system output confirms that OOV types could be mapped to known Welsh words with the MRF prior but not via stemming (see Table 4). For each language pair the MRF learned features that capture intuitively redundant patterns: adjectival endings for French, case markings for Czech, and mutation patterns for Welsh.

The greater improvements in Exp. 1 were mirrored by higher compression rates for these lexicons (see Table. 6) supporting the conjecture that word-alignment requires less information than full-blown translation. The results of the *lemma-*

Table 5: Features learned by MRF prior

Czech	French	Welsh
($\sim$, $\sim m$)	($\sim$, $\sim s$)	($c \sim$, $g \sim$)
($\sim$, $\sim u$)	($\sim$, $\sim e$)	($d \sim$, $dd \sim$)
($\sim$, $\sim a$)	($\sim$, $\sim es$)	($d \sim$, $t \sim$)
($\sim$, $\sim ch$)	($\sim e$, $\sim es$)	($b \sim$, $p \sim$)
($\sim$, $\sim ho$)	($\sim e$, $\sim er$)	($c \sim$, $ch \sim$)
($\sim a$, $\sim u$)	($\sim e$, $\sim ent$)	($b \sim$, $f \sim$)

Note: Features defined over pairs of source types assigned to the same cluster; here $\sim$ matches a common substring.

Table 6: Optimal lexicon size (ratio of raw vocab.)

	Czech	French	Welsh
Word-alignment	0.26	0.22	0.24
TM smoothing	0.28	0.38	0.51

tize model on Czech show the model selection procedure improving on a simple supervised baseline.

6 Related Work

Previous work on automatic bilingual word clustering has been motivated somewhat differently and not made use of cluster-based models to assign translation probabilities directly (Wang et al., 1996), (Och, 1998). There is, however, a large body of work using morphological analysis to define cluster-based translation models similar to ours but in a supervised manner (Zens and Ney, 2004), (Niessen and Ney, 2004). These approaches have used morphological annotation (e.g. lemmas and part of speech tags) to provide explicit supervision. They have also involved manually specifying which morphological distinc-

tions are redundant (Goldwater and McClosky, 2005). In contrast, we attempt to learn both equivalence classes and redundant relations automatically. Our experiments with orthographic features suggest that some morphological redundancies can be acquired in an unsupervised fashion.

The marginal likelihood hard-clustering algorithm that we propose here for translation model selection can be viewed as a Bayesian k -means algorithm and is an application of Bayesian model selection techniques, e.g., (Wolpert, 1995). The Markov random field prior over model structure extends the fixed uniform prior over clusters implicit in k -means clustering and is common in computer vision (Geman and Geman, 1984). Recently Basu et al. (2004) used an MRF to embody hard constraints within semi-supervised clustering. In contrast, we use an iterative EM algorithm to *learn* soft constraints within the ‘prior’ monolingual space based on the results of clustering with bilingual statistics.

7 Conclusions and Future Work

We proposed a framework for modelling lexical redundancy in machine translation and tackled optimisation of the lexicon via Bayesian model selection over a set of cluster-based translation models. We showed improvements in translation quality incorporating these models within a phrase-based SMT system. Additional gains resulted from the inclusion of an MRF prior over model structure. We demonstrated that this prior could be used to learn weights for monolingual features that characterise bilingual redundancy. Preliminary experiments defining MRF features over morphological annotation suggest this model can also identify redundant distinctions categorised linguistically (for instance, that *morphological case* is redundant on Czech nouns and adjectives with respect to English, while *number* is redundant only on adjectives). In future work we will investigate the use of linguistic resources to define feature sets for the MRF prior. Lexical redundancy would ideally be addressed in the context of phrases, however, computation and statistical estimation may then be significantly more challenging.

Acknowledgements

The authors would like to thank Philipp Koehn for providing training scripts used in this work; and Steve Renals, Mirella Lapata and members of the Edinburgh SMT Group for valuable comments. This work was supported by an MRC Priority Area Studentship to the School of Informatics, University of Edinburgh.

References

- Sugato Basu, Mikhail Bilenko, and Raymond J. Mooney. 2004. A probabilistic framework for semi-supervised clustering. In *Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004)*.
- Julian Besag. 1986. The statistical analysis of dirty pictures. *Journal of the Royal Society Series B*, 48(2):259–302.
- Peter Brown, Stephen Della Pietra, Vincent Della Pietra, and Robert Mercer. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.
- Philip A. Chou. 1991. Optimal partitioning for classification and regression trees. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(4).
- M. Cmejrek, J. Curin, J. Havelka, J. Hajic, and V. Kubon. 2004. Prague Czech-English dependency treebank: Syntactically annotated resources for machine translation. In *4th International Conference on Language Resources and Evaluation, Lisbon, Portugal*
- S. Geman and D. Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6:721–741.
- Sharon Goldwater and David McClosky. 2005. Improving statistical MT through morphological analysis. In *Proc. of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the HLT/NAACL 2003*.
- Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the AMTA 2004*.
- Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit 2005*.
- S. Niessen and H. Ney. 2004. Statistical machine translation with scarce resources using morpho-syntactic information. *Computational Linguistics*, 30(2):181–204.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- F.-J. Och. 1998. An efficient method for determining bilingual word classes. In *Proc. of the European Chapter of the Association for Computational Linguistics 1998*.
- Ye-Yi Wang, John Lafferty, and Alex Waibel. 1996. Word clustering with parallel spoken language corpora. In *Proc. of 4th International Conference on Spoken Language Processing, ICSLP 96, Philadelphia, PA*.
- D.H. Wolpert. 1995. Determining whether two data sets are from the same distribution. In *15th international workshop on Maximum Entropy and Bayesian Methods*.
- Mei Yang and Katrin Kirchhoff. 2006. Phrase-based back-off models for machine translation of highly inflected languages. In *Proc. of the European Chapter of the Association for Computational Linguistics 2006*.
- R. Zens and H. Ney. 2004. Improvements in phrase-based statistical machine translation. In *Proc. of the Human Language Technology Conference (HLT-NAACL 2004)*.

Empirical Lower Bounds on the Complexity of Translational Equivalence *

Benjamin Wellington

Computer Science Dept.
New York University
New York, NY 10003

{lastname}@cs.nyu.edu

Sonja Waxmonsky

Computer Science Dept.
University of Chicago[†]
Chicago, IL, 60637

wax@cs.uchicago.edu

I. Dan Melamed

Computer Science Dept.
New York University
New York, NY, 10003

{lastname}@cs.nyu.edu

Abstract

This paper describes a study of the patterns of translational equivalence exhibited by a variety of bitexts. The study found that the complexity of these patterns in every bitext was higher than suggested in the literature. These findings shed new light on why “syntactic” constraints have not helped to improve statistical translation models, including finite-state phrase-based models, tree-to-string models, and tree-to-tree models. The paper also presents evidence that inversion transduction grammars cannot generate some translational equivalence relations, even in relatively simple real bitexts in syntactically similar languages with rigid word order. Instructions for replicating our experiments are at <http://nlp.cs.nyu.edu/GenPar/ACL06>

1 Introduction

Translational equivalence is a mathematical relation that holds between linguistic expressions with the same meaning. The most common explicit representations of this relation are word alignments between sentences that are translations of each other. The complexity of a given word alignment can be measured by the difficulty of decomposing it into its atomic units under certain constraints detailed in Section 2. This paper describes a study of the distribution of alignment complexity in a variety of bitexts. The study considered word alignments both in isolation and in combination with independently generated parse trees for one or both sentences in each pair. Thus, the study

is relevant to finite-state phrase-based models that use no parse trees (Koehn et al., 2003), tree-to-string models that rely on one parse tree (Yamada and Knight, 2001), and tree-to-tree models that rely on two parse trees (Groves et al., 2004, e.g.).

The word alignments that are the least complex on our measure coincide with those that can be generated by an inversion transduction grammar (ITG). Following Wu (1997), the prevailing opinion in the research community has been that more complex patterns of word alignment in real bitexts are mostly attributable to alignment errors. However, the experiments in Section 3 show that more complex patterns occur surprisingly often even in highly reliable alignments in relatively simple bitexts. As discussed in Section 4, these findings shed new light on why “syntactic” constraints have not yet helped to improve the accuracy of statistical machine translation.

Our study used two kinds of data, each controlling a different confounding variable. First, we wanted to study alignments that contained as few errors as possible. So unlike some other studies (Zens and Ney, 2003; Zhang et al., 2006), we used manually annotated alignments instead of automatically generated ones. The results of our experiments on these data will remain relevant regardless of improvements in technology for automatic word alignment.

Second, we wanted to measure how much of the complexity is not attributable to systematic translation divergences, both in the languages as a whole (SVO vs. SOV), and in specific constructions (English *not* vs. French *ne...pas*). To eliminate this source of complexity of translational equivalence, we used English/English bitexts. We are not aware of any previous studies of word alignments in monolingual bitexts.

Even manually annotated word alignments vary in their reliability. For example, annotators sometimes link many words in one sentence to many

* Thanks to David Chiang, Liang Huang, the anonymous reviewers, and members of the NYU Proteus Project for helpful feedback. This research was supported by NSF grant #'s 0238406 and 0415933.

[†] SW made most of her contribution while at NYU.

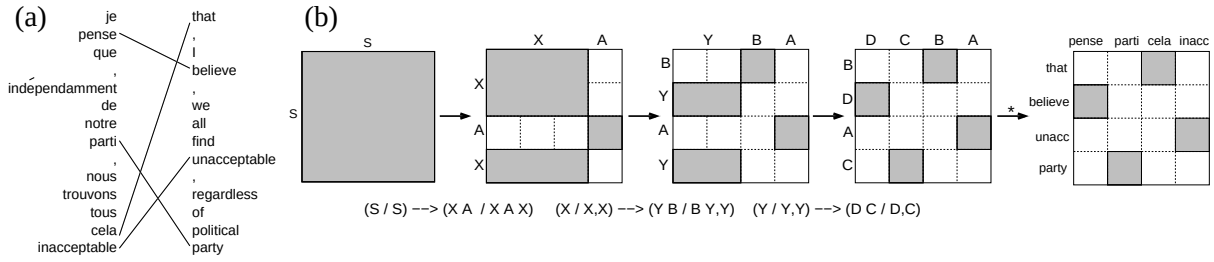


Figure 1: (a) Part of a word alignment. (b) Derivation of this word alignment using only binary and nullary productions requires one gap per nonterminal, indicated by commas in the production rules.

words in the other, instead of making the effort to tease apart more fine-grained distinctions. A study of such word alignments might say more about the annotation process than about the translational equivalence relation in the data. The inevitable noise in the data motivated us to focus on lower bounds, complementary to Fox (2002), who wrote that her results “should be looked on as more of an upper bound.” (p. 307) As explained in Section 3, we modified all unreliable alignments so that they cannot increase the complexity measure. Thus, we arrived at complexity measurements that were underestimates, but reliably so. It is almost certain that the true complexity of translational equivalence is higher than what we report.

2 A Measure of Alignment Complexity

Any translation model can memorize a training sentence pair as a unit. For example, given a sentence pair like *(he left slowly / slowly he left)* with the correct word alignment, a phrase-based translation model can add a single 3-word biphrase to its phrase table. However, this biphrase would not help the model predict translations of the individual words in it. That’s why phrase-based models typically decompose such training examples into their sub-biphases and remember them too. Decomposing the translational equivalence relations in the training data into smaller units of knowledge can improve a model’s ability to generalize (Zhang et al., 2006). In the limit, to maximize the chances of covering arbitrary new data, a model should decompose the training data into the smallest possible units, and learn from them.¹ For phrase-based models, this stipulation implies phrases of length one. If the model is a synchronous rewriting system, then it should be able to generate every training sentence pair as the yield of a *binary-*

¹Many popular models learn from larger units at the same time, but the size of the smallest learnable unit is what’s important for our purposes.

branching synchronous derivation tree, where every word-to-word link is generated by a different derivation step. For example, a model that uses production rules could generate the previous example using the synchronous productions $(S, S) \rightarrow (X Y / Y X)$; $(X, X) \rightarrow (U V / U V)$; $(Y, Y) \rightarrow (\text{slowly}, \text{slowly})$; $(U, U) \rightarrow (\text{he}, \text{he})$; and $(V, V) \rightarrow (\text{left}, \text{left})$.

A problem arises when this kind of decomposition is attempted for the alignment in Figure 1(a). If each link is represented by its own nonterminal, and production rules must be binary-branching, then some of the nonterminals involved in generating this alignment need discontinuities, or **gaps**. Figure 1(b) illustrates how to generate the sentence pair and its word alignment in this manner. The nonterminals X and Y have one discontinuity each.

More generally, for any positive integer k , it is possible to construct a word alignment that cannot be generated using binary production rules whose nonterminals all have fewer than k gaps (Satta and Peserico, 2005). Our study measured the complexity of a word alignment as the minimum number of gaps needed to generate it under the following constraints:

1. Each step of the derivation generates no more than two different nonterminals.
2. Each word-to-word link is generated from a separate nonterminal.²

Our measure of alignment complexity is analogous to what Melamed et al. (2004) call “fan-out.”³ The least complex alignments on this measure — those that can be generated with zero gaps — are precisely those that can be generated by an

²If we imagine that each word is generated from a separate nonterminal as in GCNF (Melamed et al., 2004), then constraint 2 becomes a special case of constraint 1.

³For grammars that generate bitexts, fan-out is equal to the maximum number of allowed gaps plus two.

bitext	# SPs	min	median	max	95% C.I.
Chinese/English	491	4	24	52	.02
Romanian/English	200	2	19	76	.03
Hindi/English	90	1	10	40	.04
Spanish/English	199	4	23	49	.03
French/English	447	2	15	29	.01
Eng/Eng MTEval	5253	2	26	92	.01
Eng/Eng fiction	6263	2	15	97	.01

Table 1: Number of sentence pairs and minimum/median/maximum sentence lengths in each bitext. All failure rates reported later have a 95% confidence interval that is no wider than the value shown for each bitext.

ITG. For the rest of the paper, we restrict our attention to binary derivations, except where explicitly noted otherwise.

To measure the number of gaps needed to generate a given word alignment, we used a bottom-up hierarchical alignment algorithm to infer a binary synchronous parse tree that was consistent with the alignment, using as few gaps as possible. A hierarchical alignment algorithm is a type of synchronous parser where, instead of constraining inferences by the production rules of a grammar, the constraints come from word alignments and possibly other sources (Wu, 1997; Melamed and Wang, 2005). A bottom-up hierarchical aligner begins with word-to-word links as constituents, where some of the links might be to nothing (“NULL”). It then repeatedly composes constituents with other constituents to make larger ones, trying to find a constituent that covers the entire input.

One of the important design choices in this kind of study is how to treat multiple links attached to the same word token. Word aligners, both human and automatic, are often inconsistent about whether they intend such sets of links to be disjunctive or conjunctive. In accordance with its focus on lower bounds, the present study treated them as disjunctive, to give the hierarchical alignment algorithm more opportunities to use fewer gaps. This design decision is one of the main differences between our study and that of Fox (2002), who treated links to the same word conjunctively.

By treating many-to-one links disjunctively, our measure of complexity ignored a large class of discontinuities. Many types of discontinuous constituents exist in text independently of any translation. Simard et al. (2005) give examples such as English verb-particle constructions, and the French negation *ne...pas*. The disparate elements of such constituents would usually be aligned to the same word in a translation. However, when

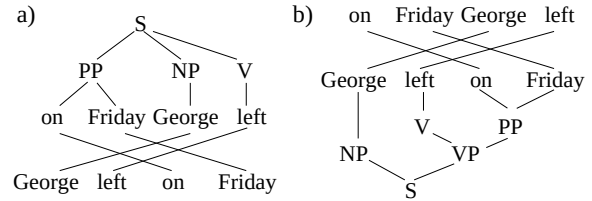


Figure 2: a) With a parse tree constraining the top sentence, a hierarchical alignment is possible without gaps. b) With a parse tree constraining the bottom sentence, no such alignment exists.

our hierarchical aligner saw two words linked to one word, it ignored one of the two links. Our lower bounds would be higher if they accounted for this kind of discontinuity.

3 Experiments

3.1 Data

We used two monolingual bitexts and five bilingual bitexts. The Romanian/English and Hindi/English data came from Martin et al. (2005). For Chinese/English and Spanish/English, we used the data from Ayan et al. (2005). The French/English data were those used by Mihalcea and Pedersen (2003). The monolingual bitext labeled “MTEval” in the tables consists of multiple independent translations from Chinese to English (LDC, 2002). The other monolingual bitext, labeled “fiction,” consists of two independent translations from French to English of Jules Verne’s novel *20,000 Leagues Under the Sea*, sentence-aligned by Barzilay and McKeown (2001).

From the monolingual bitexts, we removed all sentence pairs where either sentence was longer than 100 words. Table 1 gives descriptive statistics for the remaining data. The table also shows the upper bound of the 95% confidence intervals for the coverage rates reported later. The results of experiments on different bitexts are not directly comparable, due to the varying genres and sentence lengths.

3.2 Constraining Parse Trees

One of the main independent variables in our experiments was the number of monolingual parse trees used to constrain the hierarchical alignments. To induce models of translational equivalence, some researchers have tried to use such trees to constrain bilingual constituents: The span of every node in the constraining parse tree must coincide with the relevant monolingual span of some

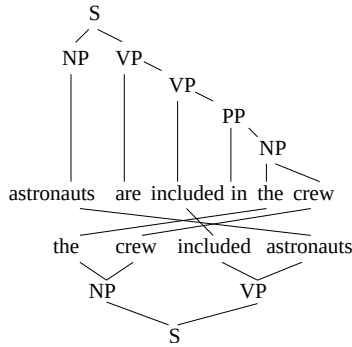


Figure 3: A word alignment that cannot be generated without gaps in a manner consistent with both parse trees.

node in the bilingual derivation tree. These additional constraints can thwart attempts at hierarchical alignment that might have succeeded otherwise. Figure 2a shows a word alignment and a parse tree that can be hierarchically aligned without gaps. *George* and *left* can be composed in both sentences into a constituent without crossing any phrase boundaries in the tree, as can *on* and *Friday*. These two constituents can then be composed to cover the entire sentence pair. On the other hand, if a constraining tree is applied to the other sentence as shown in Figure 2b, then the word alignment and tree constraint conflict. The projection of the VP is discontinuous in the top sentence, so the links that it covers cannot be composed into a constituent without gaps. On the other hand, if a gap is allowed, then the VP can compose as *on Friday ... left* in the top sentence, where the ellipsis represents a gap. This VP can then compose with the NP complete a synchronous parse tree. Some authors have applied constraining parse trees to both sides of the bitext. The example in Figure 3 can be hierarchically aligned using either one of the two constraining trees, but gaps are necessary to align it with both trees.

3.3 Methods

We parsed the English side of each bilingual bitext and both sides of each English/English bitext using an off-the-shelf syntactic parser (Bikel, 2004), which was trained on sections 02-21 of the Penn English Treebank (Marcus et al., 1993).

Our bilingual bitexts came with manually annotated word alignments. For the monolingual bitexts, we used an automatic word aligner based on a cognate heuristic and a list of 282 function words compiled by hand. The aligner linked two words to each other only if neither of them was on

the function word list and their longest common subsequence ratio (Melamed, 1995) was at least 0.75. Words that were not linked to another word in this manner were linked to NULL. For the purposes of this study, a word aligned to NULL is a non-constraint, because it can always be composed without a gap with some constituent that is adjacent to it on just one side of the bitext. The number of automatically induced non-NULL links was lower than what would be drawn by hand.

We modified the word alignments in all bitexts to minimize the chances that alignment errors would lead to an over-estimate of alignment complexity. All of the modifications involved adding links to NULL. Due to our disjunctive treatment of conflicting links, the addition of a link to NULL can decrease but cannot increase the complexity of an alignment. For example, if we added the links (*cela*, NULL) and (NULL, *that*) to the alignment in Figure 1, the hierarchical alignment algorithm could use them instead of the link between *cela* and *that*. It could thus generate the modified alignment without using a gap. We added NULL links in two situations. First, if a subset of the links in an alignment formed a many-to-many mapping but did not form a bipartite clique (i.e. every word on one side linked to every word on the other side), then we added links from each of these words to NULL. Second, if n words on one side of the bitext aligned to m words on the other side with $m > n$ then we added NULL links for each of the words on the side with m words.

After modifying the alignments and obtaining monolingual parse trees, we measured the alignment complexity of each bitext using a hierarchical alignment algorithm, as described in Section 2. Separate measurements were taken with zero, one, and two constraining parse trees. The synchronous parser in the GenPar toolkit⁴ can be configured for all of these cases (Burbank et al., 2005).

Unlike Fox (2002) and Galley et al. (2004), we measured failure rates per corpus rather than per sentence pair or per node in a constraining tree. This design was motivated by the observation that if a translation model cannot correctly model a certain word alignment, then it is liable to make incorrect inferences about arbitrary parts of that alignment, not just the particular word links involved in a complex pattern. The failure rates we report represent lower bounds on the fraction of training data

⁴<http://nlp.cs.nyu.edu/GenPar>

# of gaps allowed →	0/0	0/1 or 1/0
Chinese/English	26 = 5%	0 = 0%
Romanian/English	1 = 0%	0 = 0%
Hindi/English	2 = 2%	0 = 0%
Spanish/English	3 = 2%	0 = 0%
French/English	3 = 1%	0 = 0%

Table 2: Failure rates for hierarchical alignment of bilingual bitexts under word alignment constraints only.

# of gaps allowed on non-English side →	0	1	2
Chinese/English	298 = 61%	28 = 6%	0 = 0%
Romanian/English	82 = 41%	6 = 3%	1 = 0%
Hindi/English	33 = 37%	1 = 1%	0 = 0%
Spanish/English	75 = 38%	4 = 2%	0 = 0%
French/English	67 = 15%	2 = 0%	0 = 0%

Table 3: Failure rates for hierarchical alignment of bilingual bitexts under the constraints of a word alignment and a monolingual parse tree on the English side.

that is susceptible to misinterpretation by overconstrained translation models.

3.4 Summary Results

Table 2 shows the lower bound on alignment failure rates with and without gaps for five languages paired with English. This table represents the case where the only constraints are from word alignments. Wu (1997) has “been unable to find real examples” of cases where hierarchical alignment would fail under these conditions, at least in “fixed-word-order languages that are lightly inflected, such as English and Chinese.” (p. 385). In contrast, we found examples in all bitexts that could not be hierarchically aligned without gaps, including at least 5% of the Chinese/English sentence pairs. Allowing constituents with a single gap on one side of the bitext decreased the observed failure rate to zero for all five bitexts.

Table 3 shows what happened when we used monolingual parse trees to restrict the compositions on the English side. The failure rates were above 35% for four of the five language pairs, and 61% for Chinese/English! Again, the failure rate fell dramatically when one gap was allowed on the unconstrained (non-English) side of the bitext. Allowing two gaps on the non-English side led to almost complete coverage of these word alignments.

Table 3 does not specify the number of gaps allowed on the English side, because varying this parameter never changed the outcome. The only way that a gap on that side could increase coverage is if there was a node in the constraining parse tree that

# of gaps →	0/0	0/1	0/2
0 CTs	171 = 3%	0 = 0%	0 = 0%
1 CTs	1792 = 34%	143 = 3%	7 = 0%
2 CTs	3227 = 61%	3227 = 61%	3227 = 61%

Table 4: Failure rates for hierarchical alignment of the MTEval bitext, over varying numbers of gaps and constraining trees (CTs).

# of gaps →	0/0	0/1	0/2
0 CTs	23 = 0%	0 = 0%	0 = 0%
1 CTs	655 = 10%	22 = 0%	1 = 0%
2 CTs	1559 = 25%	1559 = 25%	1559 = 25%

Table 5: Failure rates for hierarchical alignment of the fiction bitext, over varying numbers of gaps and constraining trees (CTs).

had at least four children whose translations were in one of the complex permutations. The absence of such cases in the data implies that the failure rates under the constraints of one parse tree would be identical even if we allowed production rules of rank higher than two.

Table 4 shows the alignment failure rates for the MTEval bitext. With word alignment constraints only, 3% of the sentence pairs could not be hierarchically aligned without gaps. Allowing a single gap on one side decreased this failure rate to zero. With a parse tree constraining constituents on one side of the bitext and with no gaps, alignment failure rates rose from 3% to 34%, but allowing a single gap on the side of the bitext that was not constrained by a parse tree brought the failure rate back down to 3%. With two constraining trees the failure rate was 61%, and allowing gaps did not lower it, for the same reasons that allowing gaps on the tree-constrained side made no difference in Table 3.

The trends in the fiction bitext (Table 5) were similar to those in the MTEval bitext, but the coverage was always higher, for two reasons. First, the median sentence size was lower in the fiction bitext. Second, the MTEval translators were instructed to translate as literally as possible, but the fiction translators paraphrased to make the fiction more interesting. This freedom in word choice reduced the frequency of cognates and thus imposed fewer constraints on the hierarchical alignment, which resulted in looser estimates of the lower bounds. We would expect the opposite effect with hand-aligned data (Galley et al., 2004).

To study how sentence length correlates with the complexity of translational equivalence, we took subsets of each bitext while varying the max-

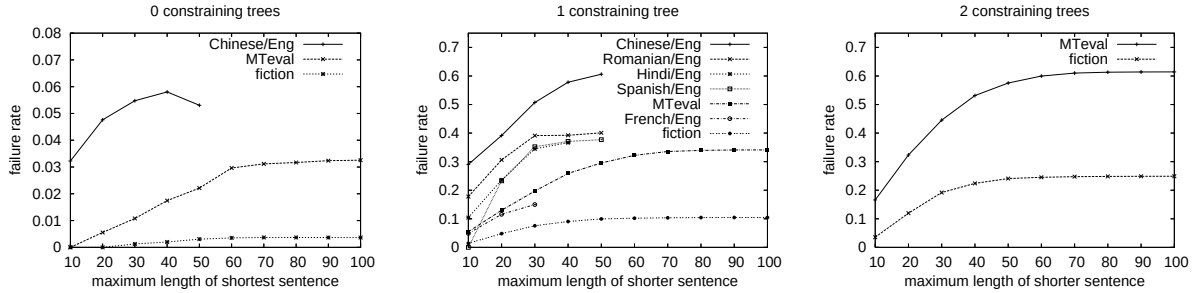


Figure 4: Failure rates for hierarchical alignment without gaps vs. maximum length of shorter sentence.

category →	1	2	3
valid reordering	12	10	5
parser error	n/a	16	25
same word used differently	15	4	0
erroneous cognates	3	0	0
total sample size	30	30	30
initial failure rate (%)	3.25	31.9	38.4
% false negatives	60±7	66±7	84±3
adjusted failure rate (%)	1.3±.22	11±2.2	6±1.1

Table 6: Detailed analysis of hierarchical alignment failures in MTEval bitext.

imum length of the shorter sentence in each pair.⁵ Figure 4 plots the resulting alignment failure rates with and without constraining parse trees. The lines in these graphs are not comparable to each other because of the variety of genres involved.

3.5 Detailed Failure Analysis

We examined by hand 30 random sentence pairs from the MTEval bitext in each of three different categories: (1) the set of sentence pairs that could not be hierarchically aligned without gaps, even without constraining parse trees; (2) the set of sentence pairs that could not be hierarchically aligned without gaps with one constraining parse tree, but that did not fall into category 1; and (3) the set of sentence pairs that could not be hierarchically aligned without gaps with two constraining parse trees, but that did not fall into category 1 or 2. Table 6 shows the results of this analysis.

In category 1, 60% of the word alignments that could not be hierarchically aligned without gaps were caused by word alignment errors. E.g.:

- 1a GlaxoSmithKline’s second-best selling **drug** may have to face competition.
- 1b **Drug** maker GlaxoSmithKline may have to face competition on its second best selling product.

The word *drug* appears in both sentences, but for different purposes, so *drug* and *drug* should not

⁵The length of the shorter sentence is the upper bound on the number of non-NULL word alignments.

have been linked.⁶ Three errors were caused by words like *targeted* and *started*, which our word alignment algorithm deemed cognates. 12 of the hierarchical alignment failures in this category were true failures. For example:

- 2a **Cheney** denied **yesterday** that the **mission** of his trip was to organize an assault on Iraq, while in **Manama**.
- 2b **Yesterday** in **Manama**, **Cheney** denied that the **mission** of his trip was to organize an assault on Iraq.

The alignment pattern of the words in bold is the familiar (3,1,4,2) permutation, as in Figure 1. Most of the 12 true failures were due to movement of prepositional phrases. The freedom of movement for such modifiers would be greater in bitexts that involve languages with less rigid word order than English.

Of the 30 sentence pairs in category 2, 16 could not be hierarchically aligned due to parser errors and 4 due to faulty word alignments. 10 were due to valid word reordering. In the following example, a co-referring pronoun causes the word alignment to fail with a constraining tree on the second sentence:

- 3a But **Chretien** appears to have changed his stance after meeting with Bush in Washington last Thursday.
- 3b But after **Chretien** talked to Bush last Thursday in Washington, **he** seemed to change his original stance.

25 of the 30 sentence pairs in category 3 failed to align due to parser error. 5 examples failed because of valid word reordering. 1 of the 5 reorderings was due to a difference between active voice and passive voice, as in Figure 3.

The last row of Table 6 takes the various reasons for alignment failure into account. It estimates what the failure rates would be if the monolingual parses and word alignments were perfect, with 95% confidence intervals. These revised rates emphasize the importance of reliable word alignments for this kind of study.

⁶This sort of error is likely to happen with other word alignment algorithms too, because words and their common translations are likely to be linked even if they’re not translationally equivalent in the given sentence.

4 Discussion

Figure 1 came from a real bilingual bitext, and Example 2 in Section 3.5 came from a real monolingual bitext.⁷ Neither of these examples can be hierarchically aligned correctly without gaps, even without constraining parse trees. The received wisdom in the literature led us to expect no such examples in bilingual bitexts, let alone in monolingual bitexts. See <http://nlp.cs.nyu.edu/GenPar/ACL06> for more examples. The English/English lower bounds are very loose, because the automatic word aligner would not link words that were not cognates. Alignment failure rates on a hand aligned bitext would be higher. We conclude that the ITG formalism cannot account for the “natural” complexity of translational equivalence, even when translation divergences are factored out.

Perhaps our most surprising results were those involving one constraining parse tree. These results explain why constraints from independently generated monolingual parse trees have not improved statistical translation models. For example, Koehn et al. (2003) reported that “requiring constituents to be syntactically motivated does not lead to better constituent pairs, but only fewer constituent pairs, with loss of a good amount of valuable knowledge.” This statement is consistent with our findings. However, most of the knowledge loss could be prevented by allowing a gap. With a parse tree constraining constituents on the English side, the coverage failure rate was 61% for the Chinese/English bitext (top row of Table 3), but allowing a gap decreased it to 6%. Zhang and Gildea (2004) found that their alignment method, which did not use external syntactic constraints, outperformed the model of Yamada and Knight (2001). However, Yamada and Knight’s model could explain only the data that would pass the no-gap test in our experiments with one constraining tree (first column of Table 3). Zhang and Gildea’s conclusions might have been different if Yamada and Knight’s model were allowed to use discontinuous constituents. The second row of Table 4 suggests that when constraining parse trees are used without gaps, at least 34% of training sentence pairs are likely to introduce noise into the model, even if systematic syntactic differences between languages are factored out. We should not

⁷The examples were shortened for the sake of space and clarity.

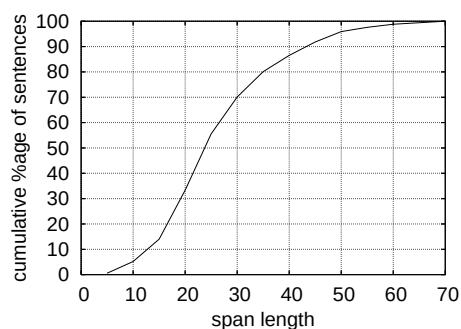


Figure 5: Lengths of spans covering words in (3,1,4,2) permutations.

be surprised when such constraints do more harm than good.

To increase the chances that a translation model can explain complex word alignments, some authors have proposed various ways of extending a model’s domain of locality. For example, Callison-Burch et al. (2005) have advocated for longer phrases in finite-state phrase-based translation models. We computed the phrase length that would be necessary to cover the words involved in each (3,1,4,2) permutation in the MTEval bitext. Figure 5 shows the cumulative percentage of these cases that would be covered by phrases up to a certain length. Only 9 of the 171 cases (5.2%) could be covered by phrases of length 10 or less. Analogous techniques for tree-structured translation models involve either allowing each nonterminal to generate both terminals and other nonterminals (Groves et al., 2004; Chiang, 2005), or, given a constraining parse tree, to “flatten” it (Fox, 2002; Zens and Ney, 2003; Galley et al., 2004). Both of these approaches can increase coverage of the training data, but, as explained in Section 2, they risk losing generalization ability.

Our study suggests that there might be some benefits to an alternative approach using discontinuous constituents, as proposed, e.g., by Melamed et al. (2004) and Simard et al. (2005). The large differences in failure rates between the first and second columns of Table 3 are largely independent of the tightness of our lower bounds. Synchronous parsing with discontinuities is computationally expensive in the worst case, but recently invented data structures make it feasible for typical inputs, as long as the number of gaps allowed per constituent is fixed at a small maximum (Waxmonsky and Melamed, 2006). More research is needed to investigate the trade-off between these costs and benefits.

5 Conclusions

This paper presented evidence of phenomena that can lead to complex patterns of translational equivalence in bitexts of any language pair. There were surprisingly many examples of such patterns that could not be analyzed using binary-branching structures without discontinuities. Regardless of the languages involved, the translational equivalence relations in most real bitexts of non-trivial size cannot be generated by an inversion transduction grammar. The low coverage rates without gaps under the constraints of independently generated monolingual parse trees might be the main reason why “syntactic” constraints have not yet increased the accuracy of SMT systems. Allowing a single gap in bilingual phrases or other types of constituent can improve coverage dramatically.

References

- Necip Ayan, Bonnie J. Dorr, and Christof Monz. 2005. Alignment link projection using transformation-based learning. In *EMNLP*.
- Regina Barzilay and Kathleen McKeown. 2001. Extracting paraphrases from a parallel corpus. In *ACL*.
- Andrea Burbank, Marine Carpuat, Stephen Clark, Markus Dreyer and Pamela Fox, Declan Groves, Keith Hall, Mary Hearne, I. Dan Melamed, Yihai Shen, Andy Way, Ben Wellington, and Dekai Wu. 2005. Final Report on Statistical Machine Translation by Parsing. JHU CLSP. <http://www.clsp.jhu.edu/ws2005/groups/statistical/report.html>
- Dan Bikel. 2004. A distributional analysis of a lexicalized statistical parsing model. In *EMNLP*.
- Chris Callison-Burch, Colin Bannard, and Josh Scroeder. 2005. Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In *ACL*.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *ACL*.
- Bonnie Dorr. 1994. Machine translation divergences: A formal description and proposed solution. *Computational Linguistics* 20(4):597–633.
- Heidi Fox. 2002. Phrasal cohesion and statistical machine translation. In *EMNLP*.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What’s in a translation rule? In *HLT-NAACL*.
- Declan Groves, Mary Hearne, and Andy Way. 2004. Robust sub-sentential alignment of phrase-structure trees. In *COLING*.
- Philipp Koehn, Franz Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL*.
- Mitchell Marcus, Beatrice Santorini, and Mary-Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- Joel Martin, Rada Mihalcea, and Ted Pedersen. 2005. Word alignments for languages with scarce resources. In *ACL Workshop on Building and Using Parallel Texts*.
- I. Dan Melamed. 1995. Automatic evaluation and uniform filter cascades for inducing *N*-best translation lexicons. In *ACL Workshop on Very Large Corpora*.
- I. Dan Melamed, Giorgio Satta, and Benjamin Wellington. 2004. Generalized multitext grammars. In *ACL*.
- I. Dan Melamed and Wei Wang. 2005. Generalized Parsers for Machine Translation. NYU Proteus Project Technical Report 05-001 <http://nlp.cs.nyu.edu/pubs/>.
- Rada Mihalcea and Ted Pedersen. 2003. An evaluation exercise for word alignment. In *HLT-NAACL Workshop on Building and Using Parallel Texts*.
- LDC. 2002. NIST MT evaluation data, Linguistic Data Consortium catalogue # LDC2002E53. <http://projects.ldc.upenn.edu/TIDES/mt2003.html>.
- Giorgio Satta and Enoch Peserico. 2005. Some computational complexity results for synchronous context-free grammars. In *EMNLP*.
- Michel Simard, Nicola Cancedda, Bruno Cavestro, Marc Dymetman, Eric Guassier, Cyril Goutte, and Kenji Yamada. 2005. Translating with non-contiguous phrases. In *EMNLP*.
- Sonjia Waxmonsky and I. Dan Melamed. 2006. A dynamic data structure for parsing with discontinuous constituents. NYU Proteus Project Technical Report 06-001 <http://nlp.cs.nyu.edu/pubs/>.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *ACL*.
- Richard Zens and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *ACL*.
- Hao Zhang and Daniel Gildea. 2004. Syntax-based alignment: Supervised or unsupervised? In *COLING*.
- Hao Zhang, Liang Huang, Daniel Gildea, and Kevin Knight. 2006. Synchronous binarization for machine translation. In *HLT-NAACL*.

A Hierarchical Bayesian Language Model based on Pitman-Yor Processes

Yee Whye Teh

School of Computing,
National University of Singapore,
3 Science Drive 2, Singapore 117543.
tehyw@comp.nus.edu.sg

Abstract

We propose a new hierarchical Bayesian n -gram model of natural languages. Our model makes use of a generalization of the commonly used Dirichlet distributions called Pitman-Yor processes which produce power-law distributions more closely resembling those in natural languages. We show that an approximation to the hierarchical Pitman-Yor language model recovers the exact formulation of interpolated Kneser-Ney, one of the best smoothing methods for n -gram language models. Experiments verify that our model gives cross entropy results superior to interpolated Kneser-Ney and comparable to modified Kneser-Ney.

1 Introduction

Probabilistic language models are used extensively in a variety of linguistic applications, including speech recognition, handwriting recognition, optical character recognition, and machine translation. Most language models fall into the class of n -gram models, which approximate the distribution over sentences using the conditional distribution of each word given a context consisting of only the previous $n - 1$ words,

$$P(\text{sentence}) \approx \prod_{i=1}^T P(\text{word}_i | \text{word}_{i-n+1}^{i-1}) \quad (1)$$

with $n = 3$ (trigram models) being typical. Even for such a modest value of n the number of parameters is still tremendous due to the large vocabulary size. As a result direct maximum-likelihood parameter fitting severely overfits to the training

data, and smoothing methods are indispensable for proper training of n -gram models.

A large number of smoothing methods have been proposed in the literature (see (Chen and Goodman, 1998; Goodman, 2001; Rosenfeld, 2000) for good overviews). Most methods take a rather ad hoc approach, where n -gram probabilities for various values of n are combined together, using either interpolation or back-off schemes. Though some of these methods are intuitively appealing, the main justification has always been empirical—better perplexities or error rates on test data. Though arguably this should be the only real justification, it only answers the question of *whether* a method performs better, not *how* nor *why* it performs better. This is unavoidable given that most of these methods are not based on internally coherent Bayesian probabilistic models, which have explicitly declared prior assumptions and whose merits can be argued in terms of how closely these fit in with the known properties of natural languages. Bayesian probabilistic models also have additional advantages—it is relatively straightforward to improve these models by incorporating additional knowledge sources and to include them in larger models in a principled manner. Unfortunately the performance of previously proposed Bayesian language models had been dismal compared to other smoothing methods (Nadas, 1984; MacKay and Peto, 1994).

In this paper, we propose a novel language model based on a hierarchical Bayesian model (Gelman et al., 1995) where each hidden variable is distributed according to a Pitman-Yor process, a nonparametric generalization of the Dirichlet distribution that is widely studied in the statistics and probability theory communities (Pitman and Yor, 1997; Ishwaran and James, 2001; Pitman, 2002).

Our model is a direct generalization of the hierarchical Dirichlet language model of (MacKay and Peto, 1994). Inference in our model is however not as straightforward and we propose an efficient Markov chain Monte Carlo sampling scheme.

Pitman-Yor processes produce power-law distributions that more closely resemble those seen in natural languages, and it has been argued that as a result they are more suited to applications in natural language processing (Goldwater et al., 2006). We show experimentally that our hierarchical Pitman-Yor language model does indeed produce results superior to interpolated Kneser-Ney and comparable to modified Kneser-Ney, two of the currently best performing smoothing methods (Chen and Goodman, 1998). In fact we show a stronger result—that interpolated Kneser-Ney can be interpreted as a particular approximate inference scheme in the hierarchical Pitman-Yor language model. Our interpretation is more useful than past interpretations involving marginal constraints (Kneser and Ney, 1995; Chen and Goodman, 1998) or maximum-entropy models (Goodman, 2004) as it can recover the exact formulation of interpolated Kneser-Ney, and actually produces superior results. (Goldwater et al., 2006) has independently noted the correspondence between the hierarchical Pitman-Yor language model and interpolated Kneser-Ney, and conjectured improved performance in the hierarchical Pitman-Yor language model, which we verify here.

Thus the contributions of this paper are threefold: in proposing a language model with excellent performance and the accompanying advantages of Bayesian probabilistic models, in proposing a novel and efficient inference scheme for the model, and in establishing the direct correspondence between interpolated Kneser-Ney and the Bayesian approach.

We describe the Pitman-Yor process in Section 2, and propose the hierarchical Pitman-Yor language model in Section 3. In Sections 4 and 5 we give a high level description of our sampling based inference scheme, leaving the details to a technical report (Teh, 2006). We also show how interpolated Kneser-Ney can be interpreted as approximate inference in the model. We show experimental comparisons to interpolated and modified Kneser-Ney, and the hierarchical Dirichlet language model in Section 6 and conclude in Section 7.

2 Pitman-Yor Process

Pitman-Yor processes are examples of nonparametric Bayesian models. Here we give a quick description of the Pitman-Yor process in the context of a unigram language model; good tutorials on such models are provided in (Ghahramani, 2005; Jordan, 2005). Let W be a fixed and finite vocabulary of V words. For each word $w \in W$ let $G(w)$ be the (to be estimated) probability of w , and let $G = [G(w)]_{w \in W}$ be the vector of word probabilities. We place a Pitman-Yor process prior on G :

$$G \sim \text{PY}(d, \theta, G_0) \quad (2)$$

where the three parameters are: a discount parameter $0 \leq d < 1$, a strength parameter $\theta > -d$ and a mean vector $G_0 = [G_0(w)]_{w \in W}$. $G_0(w)$ is the a priori probability of word w : before observing any data, we believe word w should occur with probability $G_0(w)$. In practice this is usually set uniformly $G_0(w) = 1/V$ for all $w \in W$. Both θ and d can be understood as controlling the amount of variability around G_0 in different ways. When $d = 0$ the Pitman-Yor process reduces to a Dirichlet distribution with parameters θG_0 .

There is in general no known analytic form for the density of $\text{PY}(d, \theta, G_0)$ when the vocabulary is finite. However this need not deter us as we will instead work with the distribution over sequences of words induced by the Pitman-Yor process, which has a nice tractable form and is sufficient for our purpose of language modelling. To be precise, notice that we can treat both G and G_0 as distributions over W , where word $w \in W$ has probability $G(w)$ (respectively $G_0(w)$). Let $x_1, x_2, \dots$ be a sequence of words drawn independently and identically (i.i.d.) from G . We shall describe the Pitman-Yor process in terms of a generative procedure that produces $x_1, x_2, \dots$ iteratively with G marginalized out. This can be achieved by relating $x_1, x_2, \dots$ to a separate sequence of i.i.d. draws $y_1, y_2, \dots$ from the mean distribution G_0 as follows. The first word x_1 is assigned the value of the first draw y_1 from G_0 . Let t be the current number of draws from G_0 (currently $t = 1$), c_k be the number of words assigned the value of draw y_k (currently $c_1 = 1$), and $c. = \sum_{k=1}^t c_k$ be the current number of draws from G . For each subsequent word $x_{c.+1}$, we either assign it the value of a previous draw y_k with probability $\frac{c_k - d}{\theta + c.}$ (increment c_k ; set $x_{c.+1} \leftarrow y_k$), or we assign it the value of a new draw from G_0

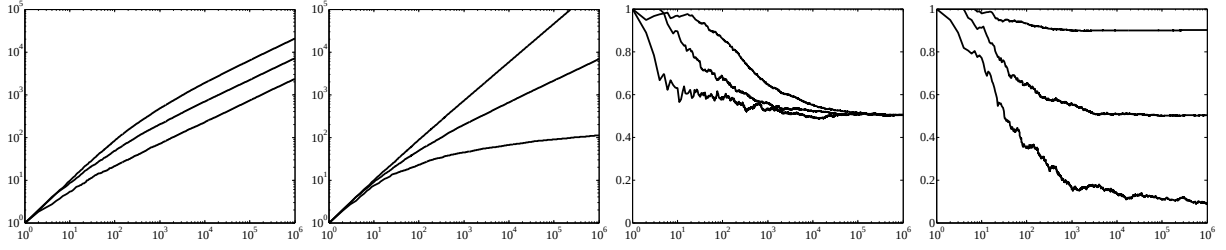


Figure 1: First panel: number of unique words as a function of the number of words drawn on a log-log scale, with $d = .5$ and $\theta = 1$ (bottom), 10 (middle) and 100 (top). Second panel: same, with $\theta = 10$ and $d = 0$ (bottom), $.5$ (middle) and $.9$ (top). Third panel: proportion of words appearing only once, as a function of the number of words drawn, with $d = .5$ and $\theta = 1$ (bottom), 10 (middle), 100 (top). Last panel: same, with $\theta = 10$ and $d = 0$ (bottom), $.5$ (middle) and $.9$ (top).

with probability $\frac{\theta+dt}{\theta+c}$ (increment t ; set $c_t = 1$; draw $y_t \sim G_0$; set $x_{c.+1} \leftarrow y_t$).

The above generative procedure produces a sequence of words drawn i.i.d. from G , with G marginalized out. It is informative to study the Pitman-Yor process in terms of the behaviour it induces on this sequence of words. Firstly, notice the rich-gets-richer clustering property: the more words have been assigned to a draw from G_0 , the more likely subsequent words will be assigned to the draw. Secondly, the more we draw from G_0 , the more likely a new word will be assigned to a new draw from G_0 . These two effects together produce a power-law distribution where many unique words are observed, most of them rarely. In particular, for a vocabulary of unbounded size and for $d > 0$, the number of unique words scales as $O(\theta T^d)$ where T is the total number of words. For $d = 0$, we have a Dirichlet distribution and the number of unique words grows more slowly as $O(\theta \log T)$.

Figure 1 demonstrates the power-law behaviour of the Pitman-Yor process and how this depends on d and θ . In the first two panels we show the average number of unique words among 10 sequences of T words drawn from G , as a function of T , for various values of θ and d . We see that θ controls the overall number of unique words, while d controls the asymptotic growth of the number of unique words. In the last two panels, we show the proportion of words appearing only once among the unique words; this gives an indication of the proportion of words that occur rarely. We see that the asymptotic behaviour depends on d but not on θ , with larger d 's producing more rare words.

This procedure for generating words drawn

from G is often referred to as the Chinese restaurant process (Pitman, 2002). The metaphor is as follows. Consider a sequence of customers (corresponding to the words drawn from G) visiting a Chinese restaurant with an unbounded number of tables (corresponding to the draws from G_0), each of which can accommodate an unbounded number of customers. The first customer sits at the first table, and each subsequent customer either joins an already occupied table (assign the word to the corresponding draw from G_0), or sits at a new table (assign the word to a new draw from G_0).

3 Hierarchical Pitman-Yor Language Models

We describe an n -gram language model based on a hierarchical extension of the Pitman-Yor process. An n -gram language model defines probabilities over the current word given various contexts consisting of up to $n - 1$ words. Given a context $\mathbf{u}$, let $G_{\mathbf{u}}(w)$ be the probability of the current word taking on value w . We use a Pitman-Yor process as the prior for $G_{\mathbf{u}}[G_{\mathbf{u}}(w)]_{w \in W}$, in particular,

$$G_{\mathbf{u}} \sim \text{PY}(d_{|\mathbf{u}|}, \theta_{|\mathbf{u}|}, G_{\pi(\mathbf{u})}) \quad (3)$$

where $\pi(\mathbf{u})$ is the suffix of $\mathbf{u}$ consisting of all but the earliest word. The strength and discount parameters are functions of the length $|\mathbf{u}|$ of the context, while the mean vector is $G_{\pi(\mathbf{u})}$, the vector of probabilities of the current word given all but the earliest word in the context. Since we do not know $G_{\pi(\mathbf{u})}$ either, We recursively place a prior over $G_{\pi(\mathbf{u})}$ using (3), but now with parameters $\theta_{|\pi(\mathbf{u})|}$, $d_{|\pi(\mathbf{u})|}$ and mean vector $G_{\pi(\pi(\mathbf{u}))}$ instead. This is repeated until we get to $G_{\emptyset}$, the vector of probabilities over the current word given the

empty context $\emptyset$. Finally we place a prior on $G_\emptyset$:

$$G_\emptyset \sim \text{PY}(d_0, \theta_0, G_0) \quad (4)$$

where G_0 is the global mean vector, given a uniform value of $G_0(w) = 1/V$ for all $w \in W$. Finally, we place a uniform prior on the discount parameters and a $\text{Gamma}(1, 1)$ prior on the strength parameters. The total number of parameters in the model is $2n$.

The structure of the prior is that of a suffix tree of depth n , where each node corresponds to a context consisting of up to $n-1$ words, and each child corresponds to adding a different word to the beginning of the context. This choice of the prior structure expresses our belief that words appearing earlier in a context have (a priori) the least importance in modelling the probability of the current word, which is why they are dropped first at successively higher levels of the model.

4 Hierarchical Chinese Restaurant Processes

We describe a generative procedure analogous to the Chinese restaurant process of Section 2 for drawing words from the hierarchical Pitman-Yor language model with all $G_{\mathbf{u}}$'s marginalized out. This gives us an alternative representation of the hierarchical Pitman-Yor language model that is amenable to efficient inference using Markov chain Monte Carlo sampling and easy computation of the predictive probabilities for test words. The correspondence between interpolated Kneser-Ney and the hierarchical Pitman-Yor language model is also apparent in this representation.

Again we may treat each $G_{\mathbf{u}}$ as a distribution over the current word. The basic observation is that since $G_{\mathbf{u}}$ is Pitman-Yor process distributed, we can draw words from it using the Chinese restaurant process given in Section 2. Further, the only operation we need of its parent distribution $G_{\pi(\mathbf{u})}$ is to draw words from it too. Since $G_{\pi(\mathbf{u})}$ is itself distributed according to a Pitman-Yor process, we can use another Chinese restaurant process to draw words from that. This is recursively applied until we need draws from the global mean distribution G_0 , which is easy since it is just uniform. We refer to this as the hierarchical Chinese restaurant process.

Let us introduce some notations. For each context $\mathbf{u}$ we have a sequence of words $x_{\mathbf{u}1}, x_{\mathbf{u}2}, \dots$ drawn i.i.d. from $G_{\mathbf{u}}$ and another sequence of

words $y_{\mathbf{u}1}, y_{\mathbf{u}2}, \dots$ drawn i.i.d. from the parent distribution $G_{\pi(\mathbf{u})}$. We use l to index draws from $G_{\mathbf{u}}$ and k to index the draws from $G_{\pi(\mathbf{u})}$. Define $t_{\mathbf{u}wk} = 1$ if $y_{\mathbf{u}k}$ takes on value w , and $t_{\mathbf{u}wk} = 0$ otherwise. Each word $x_{\mathbf{u}l}$ is assigned to one of the draws $y_{\mathbf{u}k}$ from $G_{\pi(\mathbf{u})}$. If $y_{\mathbf{u}k}$ takes on value w define $c_{\mathbf{u}wk}$ as the number of words $x_{\mathbf{u}l}$ drawn from $G_{\mathbf{u}}$ assigned to $y_{\mathbf{u}k}$, otherwise let $c_{\mathbf{u}wk} = 0$. Finally we denote marginal counts by dots. For example, $c_{\mathbf{u} \cdot k}$ is the number of $x_{\mathbf{u}l}$'s assigned the value of $y_{\mathbf{u}k}$, $c_{\mathbf{u}w \cdot}$ is the number of $x_{\mathbf{u}l}$'s with value w , and $t_{\mathbf{u} \cdot}$ is the current number of draws $y_{\mathbf{u}k}$ from $G_{\pi(\mathbf{u})}$. Notice that we have the following relationships among the $c_{\mathbf{u}w \cdot}$'s and $t_{\mathbf{u}w \cdot}$:

$$\begin{cases} t_{\mathbf{u}w \cdot} = 0 & \text{if } c_{\mathbf{u}w \cdot} = 0; \\ 1 \leq t_{\mathbf{u}w \cdot} \leq c_{\mathbf{u}w \cdot} & \text{if } c_{\mathbf{u}w \cdot} > 0; \end{cases} \quad (5)$$

$$c_{\mathbf{u}w \cdot} = \sum_{\mathbf{u}': \pi(\mathbf{u}') = \mathbf{u}} t_{\mathbf{u}'w \cdot} \quad (6)$$

Pseudo-code for drawing words using the hierarchical Chinese restaurant process is given as a recursive function $\text{DrawWord}(\mathbf{u})$, while pseudo-code for computing the probability that the next word drawn from $G_{\mathbf{u}}$ will be w is given in $\text{WordProb}(\mathbf{u}, w)$. The counts are initialized at all $c_{\mathbf{u}wk} = t_{\mathbf{u}wk} = 0$.

Function $\text{DrawWord}(\mathbf{u})$:

Returns a new word drawn from $G_{\mathbf{u}}$.

If $\mathbf{u} = 0$, return $w \in W$ with probability $G_0(w)$.

Else with probabilities proportional to:

$c_{\mathbf{u}wk} - d_{|\mathbf{u}|} t_{\mathbf{u}wk}$: assign the new word to $y_{\mathbf{u}k}$.
Increment $c_{\mathbf{u}wk}$; return w .

$\theta_{|\mathbf{u}|} + d_{|\mathbf{u}|} t_{\mathbf{u} \cdot}$: assign the new word to a new draw $y_{\mathbf{u}k^{\text{new}}}$ from $G_{\pi(\mathbf{u})}$.

Let $w \leftarrow \text{DrawWord}(\pi(\mathbf{u}))$;

set $t_{\mathbf{u}wk^{\text{new}}} = c_{\mathbf{u}wk^{\text{new}}} = 1$; return w .

Function $\text{WordProb}(\mathbf{u}, w)$:

Returns the probability that the next word after context $\mathbf{u}$ will be w .

If $\mathbf{u} = 0$, return $G_0(w)$. Else return

$$\frac{c_{\mathbf{u}w \cdot} - d_{|\mathbf{u}|} t_{\mathbf{u}w \cdot}}{\theta_{|\mathbf{u}|} + c_{\mathbf{u} \cdot}} + \frac{\theta_{|\mathbf{u}|} + d_{|\mathbf{u}|} t_{\mathbf{u} \cdot}}{\theta_{|\mathbf{u}|} + c_{\mathbf{u} \cdot}} \text{WordProb}(\pi(\mathbf{u}), w).$$

Notice the self-reinforcing property of the hierarchical Pitman-Yor language model: the more a word w has been drawn in context $\mathbf{u}$, the more likely will we draw w again in context $\mathbf{u}$. In fact word w will be reinforced for other contexts that share a common suffix with $\mathbf{u}$, with the probability of drawing w increasing as the length of the

common suffix increases. This is because w will be more likely under the context of the common suffix as well.

The hierarchical Chinese restaurant process is equivalent to the hierarchical Pitman-Yor language model insofar as the distribution induced on words drawn from them are exactly equal. However, the probability vectors G_u 's have been marginalized out in the procedure, replaced instead by the assignments of words x_{ul} to draws y_{uk} from the parent distribution, i.e. the seating arrangement of customers around tables in the Chinese restaurant process corresponding to G_u . In the next section we derive tractable inference schemes for the hierarchical Pitman-Yor language model based on these seating arrangements.

5 Inference Schemes

In this section we give a high level description of a Markov chain Monte Carlo sampling based inference scheme for the hierarchical Pitman-Yor language model. Further details can be obtained at (Teh, 2006). We also relate interpolated Kneser-Ney to the hierarchical Pitman-Yor language model.

Our training data $\mathcal{D}$ consists of the number of occurrences c_{uw} of each word w after each context $\mathbf{u}$ of length exactly $n - 1$. This corresponds to observing word w drawn c_{uw} times from G_u . Given the training data $\mathcal{D}$, we are interested in the posterior distribution over the latent vectors $\mathcal{G} = \{G_v : \text{all contexts } \mathbf{v}\}$ and parameters $\Theta = \{\theta_m, d_m : 0 \leq m \leq n - 1\}$:

$$p(\mathcal{G}, \Theta | \mathcal{D}) = p(\mathcal{G}, \Theta, \mathcal{D}) / p(\mathcal{D}) \quad (7)$$

As mentioned previously, the hierarchical Chinese restaurant process marginalizes out each G_u , replacing it with the seating arrangement in the corresponding restaurant, which we shall denote by S_u . Let $\mathcal{S} = \{S_v : \text{all contexts } \mathbf{v}\}$. We are thus interested in the equivalent posterior over seating arrangements instead:

$$p(\mathcal{S}, \Theta | \mathcal{D}) = p(\mathcal{S}, \Theta, \mathcal{D}) / p(\mathcal{D}) \quad (8)$$

The most important quantities we need for language modelling are the predictive probabilities: what is the probability of a test word w after a context $\mathbf{u}$? This is given by

$$p(w | \mathbf{u}, \mathcal{D}) = \int p(w | \mathbf{u}, \mathcal{S}, \Theta) p(\mathcal{S}, \Theta | \mathcal{D}) d(\mathcal{S}, \Theta) \quad (9)$$

where the first probability on the right is the predictive probability under a particular setting of seating arrangements $\mathcal{S}$ and parameters Θ , and the overall predictive probability is obtained by averaging this with respect to the posterior over $\mathcal{S}$ and Θ (second probability on right). We approximate the integral with samples $\{\mathcal{S}^{(i)}, \Theta^{(i)}\}_{i=1}^I$ drawn from $p(\mathcal{S}, \Theta | \mathcal{D})$:

$$p(w | \mathbf{u}, \mathcal{D}) \approx \sum_{i=1}^I p(w | \mathbf{u}, \mathcal{S}^{(i)}, \Theta^{(i)}) \quad (10)$$

while $p(w | \mathbf{u}, \mathcal{S}, \Theta)$ is given by the function $\text{WordProb}(\mathbf{u}, w)$:

$$p(w | 0, \mathcal{S}, \Theta) = 1/V \quad (11)$$

$$p(w | \mathbf{u}, \mathcal{S}, \Theta) = \frac{c_{\mathbf{u}w} - d_{|\mathbf{u}|} t_{\mathbf{u}w}}{\theta_{|\mathbf{u}|} + c_{\mathbf{u}..}} + \frac{\theta_{|\mathbf{u}|} + d_{|\mathbf{u}|} t_{\mathbf{u}..}}{\theta_{|\mathbf{u}|} + c_{\mathbf{u}..}} p(w | \pi(\mathbf{u}), \mathcal{S}, \Theta) \quad (12)$$

where the counts are obtained from the seating arrangement $\mathcal{S}_u$ in the Chinese restaurant process corresponding to G_u .

We use Gibbs sampling to obtain the posterior samples $\{\mathcal{S}, \Theta\}$ (Neal, 1993). Gibbs sampling keeps track of the current state of each variable of interest in the model, and iteratively resamples the state of each variable given the current states of all other variables. It can be shown that the states of variables will converge to the required samples from the posterior distribution after a sufficient number of iterations. Specifically for the hierarchical Pitman-Yor language model, the variables consist of, for each $\mathbf{u}$ and each word x_{ul} drawn from G_u , the index k_{ul} of the draw from $G_{\pi(\mathbf{u})}$ assigned x_{ul} . In the Chinese restaurant metaphor, this is the index of the table which the l th customer sat at in the restaurant corresponding to G_u . If x_{ul} has value w , it can only be assigned to draws from $G_{\pi(\mathbf{u})}$ that has value w as well. This can either be a preexisting draw with value w , or it can be a new draw taking on value w . The relevant probabilities are given in the functions $\text{DrawWord}(\mathbf{u})$ and $\text{WordProb}(\mathbf{u}, w)$, where we treat x_{ul} as the last word drawn from G_u . This gives:

$$p(k_{ul} = k | \mathcal{S}^{-ul}, \Theta) \propto \frac{\max(0, c_{\mathbf{u}x_{ul}k} - d)}{\theta + c_{\mathbf{u}..}^{-ul}} \quad (13)$$

$$p(k_{ul} = k^{\text{new}} \text{ with } y_{\mathbf{u}k^{\text{new}}} = x_{ul} | \mathcal{S}^{-ul}, \Theta) \propto \frac{\theta + d t_{\mathbf{u}..}^{-ul}}{\theta + c_{\mathbf{u}..}^{-ul}} p(x_{ul} | \pi(\mathbf{u}), \mathcal{S}^{-ul}, \Theta) \quad (14)$$

where the superscript $-u$ means the corresponding set of variables or counts with x_u excluded. The parameters Θ are sampled using an auxiliary variable sampler as detailed in (Teh, 2006). The overall sampling scheme for an n -gram hierarchical Pitman-Yor language model takes $O(nT)$ time and requires $O(M)$ space per iteration, where T is the number of words in the training set, and M is the number of unique n -grams. During test time, the computational cost is $O(nI)$, since the predictive probabilities (12) require $O(n)$ time to calculate for each of I samples.

The hierarchical Pitman-Yor language model produces discounts that grow gradually as a function of n -gram counts. Notice that although each Pitman-Yor process G_u only has one discount *parameter*, the predictive probabilities (12) produce different discount *values* since t_{uw} can take on different values for different words w . In fact t_{uw} will on average be larger if c_{uw} is larger; averaged over the posterior, the actual amount of discount will grow slowly as the count c_{uw} grows. This is shown in Figure 2 (left), where we see that the growth of discounts is sublinear.

The correspondence to interpolated Kneser-Ney is now straightforward. If we restrict t_{uw} to be at most 1, that is,

$$t_{uw} = \min(1, c_{uw}) \quad (15)$$

$$c_{uw} = \sum_{u': \pi(u')=u} t_{u'w} \quad (16)$$

we will get the same discount value so long as $c_{uw} > 0$, i.e. absolute discounting. Further supposing that the strength parameters are all $\theta_{|u|} = 0$, the predictive probabilities (12) now directly reduces to the predictive probabilities given by interpolated Kneser-Ney. Thus we can interpret interpolated Kneser-Ney as the approximate inference scheme (15,16) in the hierarchical Pitman-Yor language model.

Modified Kneser-Ney uses the same values for the counts as in (15,16), but uses a different valued discount for each value of c_{uw} up to a maximum of $c^{(\max)}$. Since the discounts in a hierarchical Pitman-Yor language model are limited to between 0 and 1, we see that modified Kneser-Ney is not an approximation of the hierarchical Pitman-Yor language model.

6 Experimental Results

We performed experiments on the hierarchical Pitman-Yor language model on a 16 million word corpus derived from APNews. This is the same dataset as in (Bengio et al., 2003). The training, validation and test sets consist of about 14 million, 1 million and 1 million words respectively, while the vocabulary size is 17964. For trigrams with $n = 3$, we varied the training set size between approximately 2 million and 14 million words by six equal increments, while we also experimented with $n = 2$ and 4 on the full 14 million word training set. We compared the hierarchical Pitman-Yor language model trained using the proposed Gibbs sampler (HPYLM) against interpolated Kneser-Ney (IKN), modified Kneser-Ney (MKN) with maximum discount cut-off $c^{(\max)} = 3$ as recommended in (Chen and Goodman, 1998), and the hierarchical Dirichlet language model (HDLM).

For the various variants of Kneser-Ney, we first determined the parameters by conjugate gradient descent in the cross-entropy on the validation set. At the optimal values, we folded the validation set into the training set to obtain the final n -gram probability estimates. This procedure is as recommended in (Chen and Goodman, 1998), and takes approximately 10 minutes on the full training set with $n = 3$ on a 1.4 Ghz PIII. For HPYLM we inferred the posterior distribution over the latent variables and parameters given both the training and validation sets using the proposed Gibbs sampler. Since the posterior is well-behaved and the sampler converges quickly, we only used 125 iterations for burn-in, and 175 iterations to collect posterior samples. On the full training set with $n = 3$ this took about 1.5 hours.

Perplexities on the test set are given in Table 1. As expected, HDLM gives the worst performance, while HPYLM performs better than IKN. Perhaps surprisingly HPYLM performs slightly worse than MKN. We believe this is because HPYLM is not a perfect model for languages and as a result posterior estimates of the parameters are not optimized for predictive performance. On the other hand parameters in the Kneser-Ney variants are optimized using cross-validation, so are given optimal values for prediction. To validate this conjecture, we also experimented with HPYCV, a hierarchical Pitman-Yor language model where the parameters are obtained by fitting them in a slight generalization of IKN where the strength param-

T	n	IKN	MKN	HPYLM	HPYCV	HDLM
2e6	3	148.8	144.1	145.7	144.3	191.2
4e6	3	137.1	132.7	134.3	132.7	172.7
6e6	3	130.6	126.7	127.9	126.4	162.3
8e6	3	125.9	122.3	123.2	121.9	154.7
10e6	3	122.0	118.6	119.4	118.2	148.7
12e6	3	119.0	115.8	116.5	115.4	144.0
14e6	3	116.7	113.6	114.3	113.2	140.5
14e6	2	169.9	169.2	169.6	169.3	180.6
14e6	4	106.1	102.4	103.8	101.9	136.6

Table 1: Perplexities of various methods and for various sizes of training set T and length of n -grams.

eters $\theta_{|u|}$'s are allowed to be positive and optimized over along with the discount parameters using cross-validation. Seating arrangements are Gibbs sampled as in Section 5 with the parameter values fixed. We find that HPYCV performs better than MKN (except marginally worse on small problems), and has best performance overall. Note that the parameter values in HPYCV are still not the optimal ones since they are obtained by cross-validation using IKN, an approximation to a hierarchical Pitman-Yor language model. Unfortunately cross-validation using a hierarchical Pitman-Yor language model inferred using Gibbs sampling is currently too costly to be practical.

In Figure 2 (right) we broke down the contributions to the cross-entropies in terms of how many times each word appears in the test set. We see that most of the differences between the methods appear as differences among rare words, with the contribution of more common words being negligible. HPYLM performs worse than MKN on words that occurred only once (on average) and better on other words, while HPYCV is reversed and performs better than MKN on words that occurred only once or twice and worse on other words.

7 Discussion

We have described using a hierarchical Pitman-Yor process as a language model and shown that it gives performance superior to state-of-the-art methods. In addition, we have shown that the state-of-the-art method of interpolated Kneser-Ney can be interpreted as approximate inference in the hierarchical Pitman-Yor language model.

In the future we plan to study in more detail

the differences between our model and the variants of Kneser-Ney, to consider other approximate inference schemes, and to test the model on larger data sets and on speech recognition. The hierarchical Pitman-Yor language model is a fully Bayesian model, thus we can also reap other benefits of the paradigm, including having a coherent probabilistic model, ease of improvements by building in prior knowledge, and ease in using as part of more complex models; we plan to look into these possible improvements and extensions.

The hierarchical Dirichlet language model of (MacKay and Peto, 1994) was an inspiration for our work. Though (MacKay and Peto, 1994) had the right intuition to look at smoothing techniques as the outcome of hierarchical Bayesian models, the use of the Dirichlet distribution as a prior was shown to lead to non-competitive cross-entropy results. Our model is a nontrivial but direct generalization of the hierarchical Dirichlet language model that gives state-of-the-art performance. We have shown that with a suitable choice of priors (namely the Pitman-Yor process), Bayesian methods can be competitive with the best smoothing techniques.

The hierarchical Pitman-Yor process is a natural generalization of the recently proposed hierarchical Dirichlet process (Teh et al., 2006). The hierarchical Dirichlet process was proposed to solve a different problem—that of clustering, and it is interesting to note that such a direct generalization leads us to a good language model. Both the hierarchical Dirichlet process and the hierarchical Pitman-Yor process are examples of Bayesian nonparametric processes. These have recently received much attention in the statistics and machine learning communities because they can relax previously strong assumptions on the parametric forms of Bayesian models yet retain computational efficiency, and because of the elegant way in which they handle the issues of model selection and structure learning in graphical models.

Acknowledgement

I wish to thank the Lee Kuan Yew Endowment Fund for funding, Joshua Goodman for answering many questions regarding interpolated Kneser-Ney and smoothing techniques, John Blitzer and Yoshua Bengio for help with datasets, Anoop Sarkar for interesting discussion, and Hal Daume III, Min Yen Kan and the anonymous reviewers for

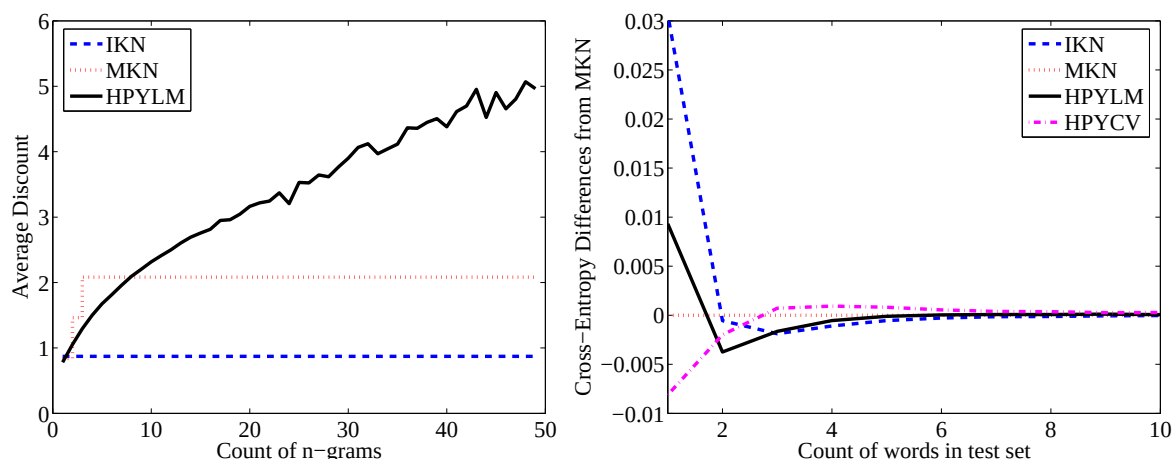


Figure 2: Left: Average discounts as a function of n -gram counts in IKN (bottom line), MKN (middle step function), and HPYLM (top curve). Right: Break down of cross-entropy on test set as a function of the number of occurrences of test words. Plotted is the sum over test words which occurred c times of cross-entropies of IKN, MKN, HPYLM and HPYCV, where c is as given on the x -axis and MKN is used as a baseline. Lower is better. Both panels are for the full training set and $n = 3$.

helpful comments.

References

- Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.
- S.F. Chen and J.T. Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Computer Science Group, Harvard University.
- A. Gelman, J. Carlin, H. Stern, and D. Rubin. 1995. *Bayesian data analysis*. Chapman & Hall, London.
- Z. Ghahramani. 2005. Nonparametric Bayesian methods. Tutorial presentation at the UAI Conference.
- S. Goldwater, T.L. Griffiths, and M. Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems*, volume 18.
- J.T. Goodman. 2001. A bit of progress in language modeling. Technical Report MSR-TR-2001-72, Microsoft Research.
- J.T. Goodman. 2004. Exponential priors for maximum entropy models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- H. Ishwaran and L.F. James. 2001. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173.
- M.I. Jordan. 2005. Dirichlet processes, Chinese restaurant processes and all that. Tutorial presentation at the NIPS Conference.
- R. Kneser and H. Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1.
- D.J.C. MacKay and L.C.B. Peto. 1994. A hierarchical Dirichlet language model. *Natural Language Engineering*.
- A. Nadas. 1984. Estimation of probabilities in the language model of the IBM speech recognition system. *IEEE Transaction on Acoustics, Speech and Signal Processing*, 32(4):859–861.
- R.M. Neal. 1993. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto.
- J. Pitman and M. Yor. 1997. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25:855–900.
- J. Pitman. 2002. Combinatorial stochastic processes. Technical Report 621, Department of Statistics, University of California at Berkeley. Lecture notes for St. Flour Summer School.
- R. Rosenfeld. 2000. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8).
- Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei. 2006. Hierarchical Dirichlet processes. *To appear in Journal of the American Statistical Association*.
- Y. W. Teh. 2006. A Bayesian interpretation of interpolated Kneser-Ney. Technical Report TRA2/06, School of Computing, National University of Singapore.

A Phonetic-Based Approach to Chinese Chat Text Normalization

Yunqing Xia, Kam-Fai Wong

Department of S.E.E.M.
The Chinese University of Hong Kong
Shatin, Hong Kong

{yqxia, kfwong}@se.cuhk.edu.hk

Wenjie Li

Department of Computing
The Hong Kong Polytechnic University
Kowloon, Hong Kong
cswjli@comp.polyu.edu.hk

Abstract

Chatting is a popular communication media on the Internet via ICQ, chat rooms, etc. Chat language is different from natural language due to its anomalous and dynamic natures, which renders conventional NLP tools inapplicable. The dynamic problem is enormously troublesome because it makes static chat language corpus outdated quickly in representing contemporary chat language. To address the dynamic problem, we propose the phonetic mapping models to present mappings between chat terms and standard words via phonetic transcription, i.e. Chinese Pinyin in our case. Different from character mappings, the phonetic mappings can be constructed from available standard Chinese corpus. To perform the task of dynamic chat language term normalization, we extend the source channel model by incorporating the phonetic mapping models. Experimental results show that this method is effective and stable in normalizing dynamic chat language terms.

1 Introduction

Internet facilitates online chatting by providing ICQ, chat rooms, BBS, email, blogs, etc. Chat language becomes ubiquitous due to the rapid proliferation of Internet applications. Chat language text appears frequently in chat logs of online education (Heard-White, 2004), customer relationship management (Gianforte, 2003), etc. On the other hand, web-based chat rooms and BBS systems are often abused by solicitors of terrorism, pornography and crime (McCullagh, 2004). Thus there is a social urgency to understand online chat language text.

Chat language is anomalous and dynamic. Many words in chat text are anomalous to natural language. Chat text comprises of ill-edited terms and anomalous writing styles. We refer chat terms to the anomalous words in chat text. The dynamic nature reflects that chat language changes more frequently than natural languages. For example, many popular chat terms used in last year have been discarded and replaced by new ones in this year. Details on these two features are provided in Section 2.

The anomalous nature of Chinese chat language is investigated in (Xia et al., 2005). Pattern matching and SVM are proposed to recognize the ambiguous chat terms. Experiments show that F-1 measure of recognition reaches 87.1% with the biggest training set. However, it is also disclosed that quality of both methods drops significantly when training set is older. The dynamic nature is investigated in (Xia et al., 2006a), in which an error-driven approach is proposed to detect chat terms in dynamic Chinese chat terms by combining standard Chinese corpora and NIL corpus (Xia et al., 2006b). Language texts in standard Chinese corpora are used as negative samples and chat text pieces in the NIL corpus as positive ones. The approach calculates confidence and entropy values for the input text. Then threshold values estimated from the training data are applied to identify chat terms. Performance equivalent to the methods in existence is achieved consistently. However, the issue of normalization is addressed in their work. Dictionary based chat term normalization is not a good solution because the dictionary cannot cover new chat terms appearing in the dynamic chat language.

In the early stage of this work, a method based on source channel model is implemented for chat term normalization. The problem we encounter is addressed as follows. To deal with the anomalous nature, a chat language corpus is constructed with chat text collected from the Internet. How-

ever, the dynamic nature renders the static corpus outdated quickly in representing contemporary chat language. The dilemma is that timely chat language corpus is nearly impossible to obtain. The sparse data problem and dynamic problem become crucial in chat term normalization. We believe that some information beyond character should be discovered to help addressing these two problems.

Observation on chat language text reveals that most Chinese chat terms are created via phonetic transcription, i.e. Chinese Pinyin in our case. A more exciting finding is that the phonetic mappings between standard Chinese words and chat terms remain stable in dynamic chat language. We are thus enlightened to make use of the phonetic mapping models, instead of character mapping models, to design a normalization algorithm to translate chat terms to their standard counterparts. Different from the character mapping models constructed from chat language corpus, the phonetic mapping models are learned from a standard language corpus because they attempt to model mappings probabilities between any two Chinese characters in terms of phonetic transcription. Now the sparse data problem can thus be appropriately addressed. To normalize the dynamic chat language text, we extend the source channel model by incorporating phonetic mapping models. We believe that the dynamic problem can be resolved effectively and robustly because the phonetic mapping models are stable.

The remaining sections of this paper are organized as follows. In Section 2, features of chat language are analyzed with evidences. In Section 3, we present methodology and problems of the source channel model approach to chat term normalization. In Section 4, we present definition, justification, formalization and parameter estimation for the phonetic mapping model. In Section 5, we present the extended source channel model that incorporates the phonetic mapping models. Experiments and results are presented in Section 6 as well as discussions and error analysis. We conclude this paper in Section 7.

2 Feature Analysis and Evidences

Observation on NIL corpus discloses the anomalous and dynamic features of chat language.

2.1 Anomalous

Chat language is explicitly anomalous in two aspects. Firstly, some chat terms are anomalous entries to standard dictionaries. For example, “介

里(here, *jie4 li3*)” is not a standard word in any contemporary Chinese dictionary while it is often used to replace “这里(here, *zhe4 li3*)” in chat language. Secondly, some chat terms can be found in standard dictionaries while their meanings in chat language are anomalous to the dictionaries. For example, “偶(even, *ou3*)” is often used to replace “我(me, *wo2*)” in chat text. But the entry that “偶” occupies in standard dictionary is used to describe even numbers. The latter case is constantly found in chat text, which makes chat text understanding fairly ambiguous because it is difficult to find out whether these terms are used as standard words or chat terms.

2.2 Dynamic

Chat text is deemed dynamic due to the fact that a large proportion of chat terms used in last year may become obsolete in this year. On the other hand, ample new chat terms are born. This feature is not as explicit as the anomalous nature. But it is as crucial. Observation on chat text in NIL corpus reveals that chat term set changes along with time very quickly.

An empirical study is conducted on five chat text collections extracted from YESKY BBS system (bbs.yesky.com) within different time periods, i.e. Jan. 2004, July 2004, Jan. 2005, July 2005 and Jan. 2006. Chat terms in each collection are picked out by hand together with their frequencies so that five chat term sets are obtained. The top 500 chat terms with biggest frequencies in each set are selected to calculate re-occurring rates of the earlier chat term sets on the later ones.

Set	Jul-04	Jan-05	Jul-05	Jan-06	Avg.
Jan-04	0.882	0.823	0.769	0.706	0.795
Jul-04	-	0.885	0.805	0.749	0.813
Jan-05	-	-	0.891	0.816	0.854
Jul-05	-	-	-	0.875	0.875

Table 1. Chat term re-occurring rates. The rows represent the earlier chat term sets and the columns the later ones.

The surprising finding in Table 1 is that 29.4% of chat terms are replaced with new ones within two years and about 18.5% within one year. The changing speed is much faster than that in standard language. This thus proves that chat text is dynamic indeed. The dynamic nature renders the static corpus outdated quickly. It poses a challenging issue on chat language processing.

3 Source Channel Model and Problems

The source channel model is implemented as baseline method in this work for chat term normalization. We brief its methodology and problems as follows.

3.1 The Model

The source channel model (SCM) is a successful statistical approach in speech recognition and machine translation (Brown, 1990). SCM is deemed applicable to chat term normalization due to similar task nature. In our case, SCM aims to find the character string $C = \{c_i\}_{i=1,2,\dots,n}$ that the given input chat text $T = \{t_i\}_{i=1,2,\dots,n}$ is most probably translated to, i.e. $t_i \rightarrow c_i$, as follows.

$$\hat{C} = \arg \max_C p(C|T) = \arg \max_C \frac{p(T|C)p(C)}{p(T)} \quad (1)$$

Since $p(T)$ is a constant for C , so $\hat{C}$ should also maximize $p(T|C)p(C)$. Now $p(C|T)$ is decomposed into two components, i.e. chat term translation observation model $p(T|C)$ and language model $p(C)$. The two models can be both estimated with maximum likelihood method using the trigram model in NIL corpus.

3.2 Problems

Two problems are notable in applying SCM in chat term normalization. First, data sparseness problem is serious because timely chat language corpus is expensive thus small due to dynamic nature of chat language. NIL corpus contains only 12,112 pieces of chat text created in eight months, which is far from sufficient to train the chat term translation model. Second, training effectiveness is poor due to the dynamic nature. Trained on static chat text pieces, the SCM approach would perform poorly in processing chat text in the future. Robustness on dynamic chat text thus becomes a challenging issue in our research.

Updating the corpus with recent chat text constantly is obviously not a good solution to the above problems. We need to find some information beyond character to help addressing the sparse data problem and dynamic problem. Fortunately, observation on chat terms provides us convincing evidence that the underlying phonetic mappings exist between most chat terms and their standard counterparts. The phonetic mappings are found promising in resolving the two problems.

4 Phonetic Mapping Model

4.1 Definition of Phonetic Mapping

Phonetic mapping is the bridge that connects two Chinese characters via phonetic transcription, i.e. Chinese Pinyin in our case. For example, “介” $\xrightarrow{(zhe, jie, 0.56)}$ “这” is the phonetic mapping connecting “这(this, zhe4)” and “介(interrupt, jie4)”, in which “zhe” and “jie” are Chinese Pinyin for “这” and “介” respectively. 0.56 is phonetic similarity between the two Chinese characters. Technically, the phonetic mappings can be constructed between any two Chinese characters within any Chinese corpus. In chat language, any Chinese character can be used in chat terms, and phonetic mappings are applied to connect chat terms to their standard counterparts. Different from the dynamic character mappings, the phonetic mappings can be produced with standard Chinese corpus before hand. They are thus stable over time.

4.2 Justifications on Phonetic Assumption

To make use of phonetic mappings in normalization of chat language terms, an assumption must be made that chat terms are mainly formed via phonetic mappings. To justify the assumption, two questions must be answered. First, how many percent of chat terms are created via phonetic mappings? Second, why are the phonetic mapping models more stable than character mapping models in chat language?

Mapping type	Count	Percentage
Chinese word/phrase	9370	83.3%
English capital	2119	7.9%
Arabic number	1021	8.0%
Other	1034	0.8%

Table 2. Chat term distribution in terms of mapping type.

To answer the first question, we look into chat term distribution in terms of mapping type in Table 2. It is revealed that 99.2 percent of chat terms in NIL corpus fall into the first four phonetic mapping types that make use of phonetic mappings. In other words, 99.2 percent of chat terms can be represented by phonetic mappings. 0.8% chat terms come from the OTHER type, emoticons for instance. The first question is undoubtedly answered with the above statistics.

To answer the second question, an observation is conducted again on the five chat term sets described in Section 2.2. We create phonetic map-

pings manually for the 500 chat terms in each set. Then five phonetic mapping sets are obtained. They are in turn compared against the standard phonetic mapping set constructed with Chinese Gigaword. Percentage of phonetic mappings in each set covered by the standard set is presented in Table 3.

Set	Jan-04	Jul-04	Jan-05	Jul-05	Jan-06
percentage	98.7	99.3	98.9	99.3	99.1

Table 3. Percentages of phonetic mappings in each set covered by standard set.

By comparing Table 1 and Table 3, we find that phonetic mappings remain more stable than character mappings in chat language text. This finding is convincing to justify our intention to design effective and robust chat language normalization method by introducing phonetic mappings to the source channel model. Note that about 1% loss in these percentages comes from chat terms that are not formed via phonetic mappings, emoticons for example.

4.3 Formalism

The phonetic mapping model is a five-tuple, i.e.

$$\langle T, C, pt(T), pt(C), Pr_{pm}(T|C) \rangle,$$

which comprises of chat term character T , standard counterpart character C , phonetic transcription of T and C , i.e. $pt(T)$ and $pt(C)$, and the mapping probability $Pr_{pm}(T|C)$ that T is mapped to C via the phonetic mapping $T \xrightarrow{(pt(T), pt(C), Pr_{pm}(T|C))} C$ (hereafter briefed by $T \xrightarrow{M} C$).

As they manage mappings between any two Chinese characters, the phonetic mapping models should be constructed with a standard language corpus. This results in two advantages. One, sparse data problem can be addressed appropriately because standard language corpus is used. Two, the phonetic mapping models are as stable as standard language. In chat term normalization, when the phonetic mapping models are used to represent mappings between chat term characters and standard counterpart characters, the dynamic problem can be addressed in a robust manner.

Differently, the character mapping model used in the SCM (see Section 3.1) connects two Chinese characters directly. It is a three-tuple, i.e.

$$\langle T, C, Pr_{cm}(T|C) \rangle,$$

which comprises of chat term character T , standard counterpart character C and the mapping probability $Pr_{cm}(T|C)$ that T is mapped to C via this character mapping. As they must be constructed from chat language training samples, the character mapping models suffer from data sparseness problem and dynamic problem.

4.4 Parameter Estimation

Two questions should be answered in parameter estimation. First, how are the phonetic mapping space constructed? Second, how are the phonetic mapping probabilities estimated?

To construct the phonetic mapping models, we first extract all Chinese characters from standard Chinese corpus and use them to form candidate character mapping models. Then we generate phonetic transcription for the Chinese characters and calculate phonetic probability for each candidate character mapping model. We exclude those character mapping models holding zero probability. Finally, the character mapping models are converted to phonetic mapping models with phonetic transcription and phonetic probability incorporated.

The phonetic probability is calculated by combining phonetic similarity and character frequencies in standard language as follows.

$$Pr_{ob_{pm}}(A, \bar{A}) = \frac{(fr_{slc}(\bar{A}) \times ps(A, \bar{A}))}{\sum_i (fr_{slc}(A_i) \times ps(A, A_i))} \quad (2)$$

In Equation (2) $\{A_i\}$ is the character set in which each element A_i is similar to character A in terms of phonetic transcription. $fr_{slc}(c)$ is a function returning frequency of given character c in standard language corpus and $ps(c_1, c_2)$ phonetic similarity between character c_1 and c_2 .

Phonetic similarity between two Chinese characters is calculated based on Chinese Pinyin as follows.

$$\begin{aligned} ps(A, \bar{A}) &= Sim(py(A), py(\bar{A})) \\ &= Sim(initial(py(A)), initial(py(\bar{A}))) \\ &\quad \times Sim(final(py(A)), final(py(\bar{A}))) \end{aligned} \quad (3)$$

In Equation (3) $py(c)$ is a function that returns Chinese Pinyin of given character c , and $initial(x)$ and $final(x)$ return initial (*shengmu*) and final (*yunmu*) of given Chinese Pinyin x respectively. For example, Chinese Pinyin for the Chinese character “这” is “zhe”, in which “zh” is initial and “e” is final. When initial or final is

empty for some Chinese characters, we only calculate similarity of the existing parts.

An algorithm for calculating similarity of initial pairs and final pairs is proposed in (Li et al., 2003) based on letter matching. Problem of this algorithm is that it always assigns zero similarity to those pairs containing no common letter. For example, initial similarity between “ch” and “q” is set to zero with this algorithm. But in fact, pronunciations of the two initials are very close to each other in Chinese speech. So non-zero similarity values should be assigned to these special pairs before hand (e.g., similarity between “ch” and “q” is set to 0.8). The similarity values are agreed by some native Chinese speakers. Thus Li et al.’s algorithm is extended to output a pre-defined similarity value before letter matching is executed in the original algorithm. For example, Pinyin similarity between “chi” and “qi” is calculated as follows.

$$\text{Sim}(\text{chi}, \text{qi}) = \text{Sim}(\text{ch}, \text{q}) \times \text{Sim}(\text{i}, \text{i}) = 0.8 \times 1 = 0.8$$

5 Extended Source Channel Model

We extend the source channel model by inserting phonetic mapping models $M = \{m_i\}_{i=1,2,\dots,n}$ into equation (1), in which chat term character t_i is mapped to standard character c_i via m_i , i.e. $t_i \xrightarrow{m_i} c_i$. The extended source channel model (XSCM) is mathematically addressed as follows.

$$\begin{aligned} \hat{C} &= \arg \max_{C,M} p(C|M,T) \\ &= \arg \max_{C,M} \frac{p(T|M,C)p(M|C)p(C)}{p(T)} \end{aligned} \quad (4)$$

Since $p(T)$ is a constant, $\hat{C}$ and $\hat{M}$ should also maximize $p(T|M,C)p(M|C)p(C)$. Now three components are involved in XSCM, i.e. chat term normalization observation model $p(T|M,C)$, phonetic mapping model $p(M|C)$ and language model $p(C)$.

Chat Term Normalization Observation Model. We assume that mappings between chat terms and their standard Chinese counterparts are independent of each other. Thus chat term normalization probability can be calculated as follows.

$$p(T|M,C) = \prod_i p(t_i|m_i,c_i) \quad (5)$$

The $p(t_i|m_i,c_i)$ ’s are estimated using maximum likelihood estimation method with Chinese character trigram model in NIL corpus.

Phonetic Mapping Model. We assume that the phonetic mapping models depend merely on the current observation. Thus the phonetic mapping probability is calculated as follows.

$$p(M|C) = \prod_i p(m_i|c_i) \quad (6)$$

in which $p(m_i|c_i)$ ’s are estimated with equation (2) and (3) using a standard Chinese corpus.

Language Model. The language model $p(C)$ ’s can be estimated using maximum likelihood estimation method with Chinese character trigram model on NIL corpus.

In our implementation, Katz Backoff smoothing technique (Katz, 1987) is used to handle the sparse data problem, and Viterbi algorithm is employed to find the optimal solution in XSCM.

6 Evaluation

6.1 Data Description

Training Sets

Two types of training data are used in our experiments. We use news from Xinhua News Agency in LDC Chinese Gigaword v.2 (CNGIGA) (Graf et al., 2005) as standard Chinese corpus to construct phonetic mapping models because of its excellent coverage of standard Simplified Chinese. We use NIL corpus (Xia et al., 2006b) as chat language corpus. To evaluate our methods on size-varying training data, six chat language corpora are created based on NIL corpus. We select 6056 sentences from NIL corpus randomly to make the first chat language corpus, i.e. C#1. In every next corpus, we add extra 1,211 random sentences. So 7,267 sentences are contained in C#2, 8,478 in C#3, 9,689 in C#4, 10,200 in C#5, and 12,113 in C#6.

Test Sets

Test sets are used to prove that chat language is dynamic and XSCM is effective and robust in normalizing dynamic chat language terms. Six time-varying test sets, i.e. T#1 ~ T#6, are created in our experiments. They contain chat language sentences posted from August 2005 to Jan 2006. We randomly extract 1,000 chat language sentences posted in each month. So timestamp of the six test sets are in temporal order, in which timestamp of T#1 is the earliest and that of T#6 the newest.

The normalized sentences are created by hand and used as standard normalization answers.

6.2 Evaluation Criteria

We evaluate two tasks in our experiments, i.e. recognition and normalization. In recognition, we use precision (p), recall (r) and f-1 measure (f) defined as follows.

$$p = \frac{x}{x+y} \quad r = \frac{x}{x+z} \quad f = \frac{2 \times p \times r}{p+r} \quad (7)$$

where x denotes the number of true positives, y the false positives and z the true negatives.

For normalization, we use accuracy (a), which is commonly accepted by machine translation researchers as a standard evaluation criterion. Every output of the normalization methods is compared to the standard answer so that normalization accuracy on each test set is produced.

6.3 Experiment I: SCM vs. XSCM Using Size-varying Chat Language Corpora

In this experiment we investigate on quality of XSCM and SCM using same size-varying training data. We intend to prove that chat language is dynamic and phonetic mapping models used in XSCM are helpful in addressing the dynamic problem. As no standard Chinese corpus is used in this experiment, we use standard Chinese text in chat language corpora to construct phonetic mapping models in XSCM. This violates the basic assumption that the phonetic mapping models should be constructed with standard Chinese corpus. So results in this experiment should be used only for comparison purpose. It would be unfair to make any conclusion on general performance of XSCM method based on results in this experiments.

We train the two methods with each of the six chat language corpora, i.e. C#1 ~ C#6 and test them on six time-varying test sets, i.e. T#1 ~ T#6. F-1 measure values produced by SCM and XSCM in this experiment are present in Table 3.

Three tendencies should be pointed out according to Table 3. The first tendency is that f-1 measure in both methods drops on time-varying test sets (see Figure 1) using same training chat language corpora. For example, both SCM and XSCM perform best on the earliest test set T#1 and worst on newest T#4. We find that the quality drop is caused by the dynamic nature of chat language. It is thus revealed that chat language is indeed dynamic. We also find that quality of XSCM drops less than that of SCM. This proves that phonetic mapping models used in XSCM are helpful in addressing the dynamic problem. However, quality of XSCM in this experiment

still drops by 0.05 on the six time-varying test sets. This is because chat language text corpus is used as standard language corpus to model the phonetic mappings. Phonetic mapping models constructed with chat language corpus are far from sufficient. We will investigate in Experiment-II to prove that stable phonetic mapping models can be constructed with real standard language corpus, i.e. CNGIGA.

Test Set		T#1	T#2	T#3	T#4	T#5	T#6
S C M	C#1	0.829	0.805	0.762	0.701	0.739	0.705
	C#2	0.831	0.807	0.767	0.711	0.745	0.715
	C#3	0.834	0.811	0.774	0.722	0.751	0.722
	C#4	0.835	0.814	0.779	0.729	0.753	0.729
	C#5	0.838	0.816	0.784	0.737	0.761	0.737
	C#6	0.839	0.819	0.789	0.743	0.765	0.743
X S C M	C#1	0.849	0.840	0.820	0.790	0.805	0.790
	C#2	0.850	0.841	0.824	0.798	0.809	0.796
	C#3	0.850	0.843	0.824	0.797	0.815	0.800
	C#4	0.851	0.844	0.829	0.805	0.819	0.805
	C#5	0.852	0.846	0.833	0.811	0.823	0.811
	C#6	0.854	0.849	0.837	0.816	0.827	0.816

Table 3. F-1 measure by SCM and XSCM on six test sets with six chat language corpora.

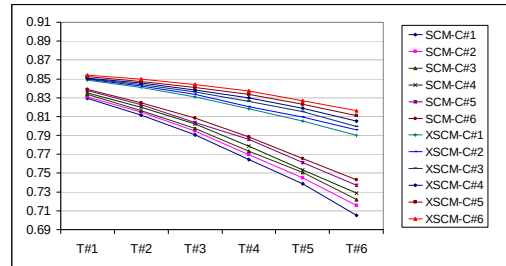


Figure 1. Tendency on f-1 measure in SCM and XSCM on six test sets with six chat language corpora.

The second tendency is f-1 measure of both methods on same test sets drops when trained with size-varying chat language corpora. For example, both SCM and XSCM perform best on the largest training chat language corpus C#6 and worst on the smallest corpus C#1. This tendency reveals that both methods favor bigger training chat language corpus. So extending the chat language corpus should be one choice to improve quality of chat language term normalization.

The last tendency is found on quality gap between SCM and XSCM. We calculate f-1 measure gaps between two methods using same training sets on same test sets (see Figure 2). Then the tendency is made clear. Quality gap between SCM and XSCM becomes bigger when test set

becomes newer. On the oldest test set T#1, the gap is smallest, while on the newest test set T#6, the gap reaches biggest value, i.e. around 0.09. This tendency reveals excellent capability of XSCM in addressing dynamic problem using the phonetic mapping models.

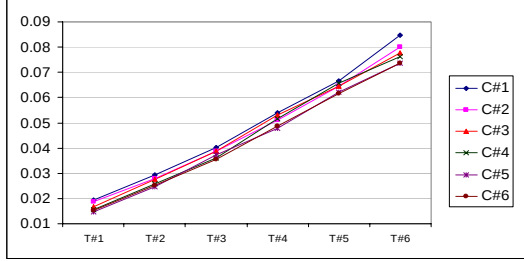


Figure 2. Tendency on f-1 measure gap in SCM and XSCM on six test sets with six chat language corpora.

6.4 Experiment II: SCM vs. XSCM Using Size-varying Chat Language Corpora and CNGIGA

In this experiment we investigate on quality of SCM and XSCM when a real standard Chinese language corpus is incorporated. We want to prove that the dynamic problem can be addressed effectively and robustly when CNGIGA is used as standard Chinese corpus.

We train the two methods on CNGIGA and each of the six chat language corpora, i.e. C#1 ~ C#6. We then test the two methods on six time-varying test sets, i.e. T#1 ~ T#6. F-1 measure values produced by SCM and XSCM in this experiment are present in Table 4.

Test Set		T#1	T#2	T#3	T#4	T#5	T#6
S C M	C#1	0.849	0.840	0.820	0.790	0.735	0.703
	C#2	0.850	0.841	0.824	0.798	0.743	0.714
	C#3	0.850	0.843	0.824	0.797	0.747	0.720
	C#4	0.851	0.844	0.829	0.805	0.748	0.727
	C#5	0.852	0.846	0.833	0.811	0.758	0.734
	C#6	0.854	0.849	0.837	0.816	0.763	0.740
X S C M	C#1	0.880	0.878	0.883	0.878	0.881	0.878
	C#2	0.883	0.883	0.888	0.882	0.884	0.880
	C#3	0.885	0.885	0.890	0.884	0.887	0.883
	C#4	0.890	0.888	0.893	0.888	0.893	0.887
	C#5	0.893	0.892	0.897	0.892	0.897	0.892
	C#6	0.898	0.896	0.900	0.897	0.901	0.896

Table 4. F-1 measure by SCM and XSCM on six test sets with six chat language corpora and CNGIGA.

Three observations are conducted on our results. First, according to Table 4, f-1 measure of

SCM with same training chat language corpora drops on time-varying test sets, but XSCM produces much better f-1 measure consistently using CNGIGA and same training chat language corpora (see Figure 3). This proves that phonetic mapping models are helpful in XSCM method. The phonetic mapping models contribute in two aspects. On the one hand, they improve quality of chat term normalization on individual test sets. On the other hand, satisfactory robustness is achieved consistently.

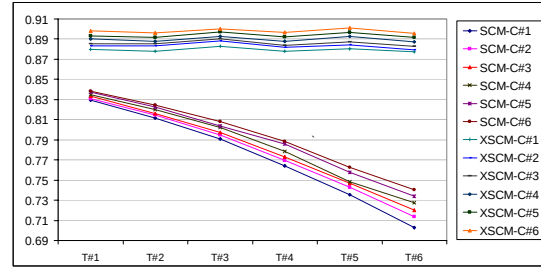


Figure 3. Tendency on f-1 measure in SCM and XSCM on six test sets with six chat language corpora and CNGIGA.

The second observation is conducted on phonetic mapping models constructed with CNGIGA. We find that 4,056,766 phonetic mapping models are constructed in this experiment, while only 1,303,227 models are constructed with NIL corpus in Experiment I. This reveals that coverage of standard Chinese corpus is crucial to phonetic mapping modeling. We then compare two character lists constructed with two corpora. The 100 characters most frequently used in NIL corpus are rather different from those extracted from CNGIGA. We can conclude that phonetic mapping models should be constructed with a sound corpus that can represent standard language.

The last observation is conducted on f-1 measure achieved by same methods on same test sets using size-varying training chat language corpora. Both methods produce best f-1 measure with biggest training chat language corpus C#6 on same test sets. This again proves that bigger training chat language corpus could be helpful to improve quality of chat language term normalization. One question might be asked whether quality of XSCM converges on size of the training chat language corpus. This question remains open due to limited chat language corpus available to us.

6.5 Error Analysis

Typical errors in our experiments belong mainly to the following two types.

Err.1 Ambiguous chat terms

Example-1: 我还是 8 米

In this example, XSCM finds no chat term while the correct normalization answer is “我还是不明 (I still don't understand)”. Error illustrated in Example-1 occurs when chat terms “8(eight, *ba1*)” and “米(meter, *mi3*)” appear in a chat sentence together. In chat language, “米” in some cases is used to replace “明(understand, *ming2*)”, while in other cases, it is used to represent a unit for length, i.e. meter. When number “8” appears before “米”, it is difficult to tell whether they are chat terms within sentential context. In our experiments, 93 similar errors occurred. We believe this type of errors can be addressed within discoursal context.

Err.2 Chat terms created in manners other than phonetic mapping

Example-2: 忧虑 ing

In this example, XSCM does not recognize “ing” while the correct answer is “(正在)忧虑 (I'm worrying)”. This is because chat terms created in manners other than phonetic mapping are excluded by the phonetic assumption in XSCM method. Around 1% chat terms fall out of phonetic mapping types. Besides chat terms holding same form as showed in Example-2, we find that emoticon is another major exception type. Fortunately, dictionary-based method is powerful enough to handle the exceptions. So, in a real system, the exceptions are handled by an extra component.

7 Conclusions

To address the sparse data problem and dynamic problem in Chinese chat text normalization, the phonetic mapping models are proposed in this paper to represent mappings between chat terms and standard words. Different from character mappings, the phonetic mappings are constructed from available standard Chinese corpus. We extend the source channel model by incorporating the phonetic mapping models. Three conclusions can be made according to our experiments. Firstly, XSCM outperforms SCM with same training data. Secondly, XSCM produces higher performance consistently on time-varying test sets. Thirdly, both SCM and XSCM perform best with biggest training chat language corpus.

Some questions remain open to us regarding optimal size of training chat language corpus in XSCM. Does the optimal size exist? Then what

is it? These questions will be addressed in our future work. Moreover, bigger context will be considered in chat term normalization, discourse for instance.

Acknowledgement

Research described in this paper is partially supported by the Chinese University of Hong Kong under the Direct Grant Scheme project (2050330) and Strategic Grant Scheme project (4410001).

References

- Brown, P. F., J. Cocke, S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer and P. S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, v.16 n.2, p.79-85.
- Gianforte, G.. 2003. From Call Center to Contact Center: How to Successfully Blend Phone, Email, Web and Chat to Deliver Great Service and Slash Costs. RightNow Technologies.
- Graf, D., K. Chen, J.Kong and K. Maeda. 2005. Chinese Gigaword Second Edition. LDC Catalog Number LDC2005T14.
- Heard-White, M., Gunter Saunders and Anita Pincas. 2004. Report into the use of CHAT in education. Final report for project of Effective use of CHAT in Online Learning, Institute of Education, University of London.
- James, F.. 2000. Modified Kneser-Ney Smoothing of n-gram Models. RIACS Technical Report 00.07.
- Katz, S. M.. 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.
- Li, H., W. He and B. Yuan. 2003. An Kind of Chinese Text Strings' Similarity and its Application in Speech Recognition. *Journal of Chinese Information Processing*, 2003 Vol.17 No.1 P.60-64.
- McCullagh, D.. 2004. Security officials to spy on chat rooms. News provided by CNET Networks. November 24, 2004.
- Xia, Y., K.-F. Wong and W. Gao. 2005. NIL is not Nothing: Recognition of Chinese Network Informal Language Expressions. 4th SIGHAN Workshop at IJCNLP'05, pp.95-102.
- Xia, Y. and K.-F. Wong. 2006a. Anomaly Detecting within Dynamic Chinese Chat Text. *EACL'06 NEW TEXT workshop*, pp.48-55.
- Xia, Y., K.-F. Wong and W. Li. 2006b. Constructing A Chinese Chat Text Corpus with A Two-Stage Incremental Annotation Approach. *LREC'06*.

Discriminative Pruning of Language Models for Chinese Word Segmentation

Jianfeng Li Haifeng Wang Dengjun Ren Guohua Li

Toshiba (China) Research and Development Center

5/F., Tower W2, Oriental Plaza, No.1, East Chang An Ave., Dong Cheng District

Beijing, 100738, China

{lijianfeng, wanghaifeng, rendengjun,
liguohua}@rdc.toshiba.com.cn

Abstract

This paper presents a discriminative pruning method of n -gram language model for Chinese word segmentation. To reduce the size of the language model that is used in a Chinese word segmentation system, importance of each bigram is computed in terms of discriminative pruning criterion that is related to the performance loss caused by pruning the bigram. Then we propose a step-by-step growing algorithm to build the language model of desired size. Experimental results show that the discriminative pruning method leads to a much smaller model compared with the model pruned using the state-of-the-art method. At the same Chinese word segmentation F-measure, the number of bigrams in the model can be reduced by up to 90%. Correlation between language model perplexity and word segmentation performance is also discussed.

1 Introduction

Chinese word segmentation is the initial stage of many Chinese language processing tasks, and has received a lot of attention in the literature (Sproat et al., 1996; Sun and Tsou, 2001; Zhang et al., 2003; Peng et al., 2004). In Gao et al. (2003), an approach based on source-channel model for Chinese word segmentation was proposed. Gao et al. (2005) further developed it to a linear mixture model. In these statistical models, language models are essential for word segmentation disambiguation. However, an uncom-

pressed language model is usually too large for practical use since all realistic applications have memory constraints. Therefore, language model pruning techniques are used to produce smaller models. Pruning a language model is to eliminate a number of parameters explicitly stored in it, according to some pruning criteria. The goal of research for language model pruning is to find criteria or methods, using which the model size could be reduced effectively, while the performance loss is kept as small as possible.

A few criteria have been presented for language model pruning, including count cut-off (Jelinek, 1990), weighted difference factor (Seymore and Rosenfeld, 1996), Kullback-Leibler distance (Stolcke, 1998), rank and entropy (Gao and Zhang, 2002). These criteria are general for language model pruning, and are not optimized according to the performance of language model in specific tasks.

In recent years, discriminative training has been introduced to natural language processing applications such as parsing (Collins, 2000), machine translation (Och and Ney, 2002) and language model building (Kuo et al., 2002; Roark et al., 2004). To the best of our knowledge, it has not been applied to language model pruning.

In this paper, we propose a discriminative pruning method of n -gram language model for Chinese word segmentation. It differentiates from the previous pruning approaches in two respects. First, the pruning criterion is based on performance variation of word segmentation. Second, the model of desired size is achieved by adding valuable bigrams to a base model, instead of by pruning bigrams from an unpruned model.

We define a misclassification function that approximately represents the likelihood that a sentence will be incorrectly segmented. The

variation value of the misclassification function caused by adding a parameter to the base model is used as the criterion for model pruning. We also suggest a step-by-step growing algorithm that can generate models of any reasonably desired size. We take the pruning method based on Kullback-Leibler distance as the baseline. Experimental results show that our method outperforms the baseline significantly with small model size. With the F-Measure of 96.33%, number of bigrams decreases by up to 90%. In addition, by combining the discriminative pruning method with the baseline method, we obtain models that achieve better performance for any model size. Correlation between language model perplexity and system performance is also discussed.

The remainder of the paper is organized as follows. Section 2 briefly discusses the related work on language model pruning. Section 3 proposes our discriminative pruning method for Chinese word segmentation. Section 4 describes the experimental settings and results. Result analysis and discussions are also presented in this section. We draw the conclusions in section 5.

2 Related Work

A simple way to reduce the size of an n -gram language model is to exclude those n -grams occurring infrequently in training corpus. It is named as count cut-off method (Jelinek, 1990). Because counts are always integers, the size of the model can only be reduced to discrete values.

Gao and Lee (2000) proposed a distribution-based pruning. Instead of pruning n -grams that are infrequent in training data, they prune n -grams that are likely to be infrequent in a new document. Experimental results show that it is better than traditional count cut-off method.

Seymore and Rosenfeld (1996) proposed a method to measure the difference of the models before and after pruning each n -gram, and the difference is computed as:

$$-N(h_j, w_i) \times [\log P'(w_i | h_j) - \log P(w_i | h_j)] \quad (1)$$

Where $P(w_i | h_j)$ denotes the conditional probabilities assigned by the original model, and $P'(w_i | h_j)$ denotes the probabilities in the pruned model. $N(h_j, w_i)$ is the discounted frequency of n -gram event $h_j w_i$. Seymore and Rosenfeld (1996) showed that this method is more effective than the traditional cut-off method.

Stolcke (1998) presented a more sound criterion for computing the difference of models before and after pruning each n -gram, which is

called relative entropy or Kullback-Leibler distance. It is computed as:

$$-\sum_{w_i, h_j} P(w_i, h_j) [\log P'(w_i | h_j) - \log P(w_i | h_j)] \quad (2)$$

The sum is over all words w_i and histories h_j . This criterion removes some of the approximations employed in Seymore and Rosenfeld (1996). In addition, Stolcke (1998) presented a method for efficient computation of the Kullback-Leibler distance of each n -gram.

In Gao and Zhang (2002), three measures are studied for the purpose of language model pruning. They are probability, rank, and entropy. Among them, probability is very similar to that proposed by Seymore and Rosenfeld (1996). Gao and Zhang (2002) also presented a method of combining two criteria, and showed the combination of rank and entropy achieved the smallest models.

3 Discriminative Pruning for Chinese Word Segmentation

3.1 Problem Definition

In this paper, discussions are restricted to bigram language model $P(w_y | w_x)$. In a bigram model, three kinds of parameters are involved: bigram probability $P_m(w_y | w_x)$ for seen bigram $w_x w_y$ in training corpus, unigram probability $P_m(w)$ and backoff coefficient $\alpha_m(w)$ for any word w . For any w_x and w_y in the vocabulary, bigram probability $P(w_y | w_x)$ is computed as:

$$P(w_y | w_x) = \begin{cases} P_m(w_y | w_x) & \text{if } c(w_x, w_y) > 0 \\ \alpha_m(w_x) \times P_m(w_y) & \text{if } c(w_x, w_y) = 0 \end{cases} \quad (3)$$

As equation (3) shows, the probability of an unseen bigram is computed by the product of the unigram probability and the corresponding backoff coefficient. If we remove a seen bigram from the model, we can still yield a bigram probability for it, by regarding it as an unseen bigram. Thus, we can reduce the number of bigram probabilities explicitly stored in the model. By doing this, model size decreases. This is the foundation for bigram model pruning.

The research issue is to find an effective criterion to compute "importance" of each bigram. Here, "importance" indicates the performance loss caused by pruning the bigram. Generally, given a target model size, the method for language model pruning is described in Figure 1.

In fact, deciding which bigrams should be excluded from the model is equivalent to deciding

which bigrams should be included in the model. Hence, we suggest a growing algorithm through which a model of desired size can also be achieved. It is illustrated in Figure 2. Here, two terms are introduced. *Full-bigram model* is the unpruned model containing all seen bigrams in training corpus. And *base model* is currently the unigram model.

For the discriminative pruning method suggested in this paper, growing algorithm instead of pruning algorithm is applied to generate the model of desired size. In addition, "importance" of each bigram indicates the performance improvement caused by adding a bigram into the base model.

1. Given the desired model size, compute the number of bigrams that should be pruned. The number is denoted as m ;
2. Compute "importance" of each bigram;
3. Sort all bigrams in the language model, according to their "importance";
4. Remove m most "unimportant" bigrams from the model;
5. Re-compute backoff coefficients in the model.

Figure 1. Language Model Pruning Algorithm

1. Given the desired model size, compute the number of bigrams that should be added into the base model. The number is denoted as n ;
2. Compute "importance" of each bigram included in the full-bigram model but excluded from the base model;
3. Sort the bigrams according to their "importance";
4. Add n most "important" bigrams into the base model;
5. Re-compute backoff coefficients in the base model.

Figure 2. Growing Algorithm for Language Model Pruning

3.2 Discriminative Pruning Criterion

Given a Chinese character string S , a word segmentation system chooses a sequence of words W^* as the segmentation result, satisfying:

$$W^* = \arg \max_W (\log P(W) + \log P(S|W)) \quad (4)$$

The sum of the two logarithm probabilities in equation (4) is called *discriminant function*:

$$g(S, W; \Lambda, \Gamma) = \log P(W) + \log P(S|W) \quad (5)$$

Where Γ denotes a language model that is used to compute $P(W)$, and Λ denotes a generative model that is used to compute $P(S|W)$. In language model pruning, Λ is an invariable.

The discriminative pruning criterion is inspired by the comparison of segmented sentences using full-bigram model Γ_F and using base model Γ_B . Given a sentence S , full-bigram model chooses W_F^* as the segmentation result, and base model chooses W_B^* as the segmentation result, satisfying:

$$W_F^* = \arg \max_W g(S, W; \Lambda, \Gamma_F) \quad (6)$$

$$W_B^* = \arg \max_W g(S, W; \Lambda, \Gamma_B) \quad (7)$$

Here, given a language model Γ , we define a *misclassification function* representing the difference between discriminant functions of W_F^* and W_B^* :

$$d(S; \Lambda, \Gamma) = g(S, W_F^*; \Lambda, \Gamma) - g(S, W_B^*; \Lambda, \Gamma) \quad (8)$$

The misclassification function reflects which one of W_F^* and W_B^* is inclined to be chosen as the segmentation result. If $W_F^* \neq W_B^*$, we may extract some hints from the comparison of them, and select a few valuable bigrams. By adding these bigrams to base model, we should make the model choose the correct answer between W_F^* and W_B^* . If $W_F^* = W_B^*$, no hints can be extracted.

Let W_0 be the known correct word sequence. Under the precondition $W_F^* \neq W_B^*$, we describe our method in the following three cases.

Case 1: $W_F^* = W_0$ and $W_B^* \neq W_0$

Here, full-bigram model chooses the correct answer, while base model does not. Based on equation (6), (7) and (8), we know that $d(S; \Lambda, \Gamma_B) > 0$ and $d(S; \Lambda, \Gamma_F) < 0$. It implies that adding bigrams into base model may lead the misclassification function from positive to negative. Which bigram should be added depends on the variation of misclassification function caused by adding it. If adding a bigram makes the misclassification function become smaller, it should be added with higher priority.

We add each bigram individually to Γ_B , and then compute the variation of the misclassification function. Let Γ' denotes the model after add-

ing bigram $w_x w_y$ into Γ_B . According to equation (5) and (8), we can write the misclassification function using Γ_B and Γ' separately:

$$d(S; \Lambda, \Gamma_B) = \log P_B(W_B^*) + \log P_\Lambda(S | W_B^*) - \log P_B(W_F^*) - \log P_\Lambda(S | W_F^*) \quad (9)$$

$$d(S; \Lambda, \Gamma') = \log P'(W_B^*) + \log P_\Lambda(S | W_B^*) - \log P'(W_F^*) - \log P_\Lambda(S | W_F^*) \quad (10)$$

Where $P_B(\cdot)$, $P'(\cdot)$, $P_\Lambda(\cdot)$ represent probabilities in base model, model Γ' and model Λ separately. The variation of the misclassification function is computed as:

$$\begin{aligned} \Delta d(S; w_x w_y) &= d(S; \Lambda, \Gamma_B) - d(S; \Lambda, \Gamma') \\ &= [\log P'(W_F^*) - \log P_B(W_F^*)] \\ &\quad - [\log P'(W_B^*) - \log P_B(W_B^*)] \end{aligned} \quad (11)$$

Because the only difference between base model and model Γ' is that model Γ' involves the bigram probability $P'(w_y|w_x)$, we have:

$$\begin{aligned} &\log P'(W_F^*) - \log P_B(W_F^*) \\ &= \sum_i [\log P'(w_{F(i)}^* | w_{F(i-1)}^*) - \log P_B(w_{F(i)}^* | w_{F(i-1)}^*)] \\ &= n(W_F^*, w_x w_y) [\log P'(w_y | w_x) - \log P_B(w_y) - \log \alpha_B(w_x)] \end{aligned} \quad (12)$$

Where $n(W_F^*, w_x w_y)$ denotes the number of times the bigram $w_x w_y$ appears in sequence W_F^* . Note that in equation (12), base model is treated as a bigram model instead of a unigram model. The reason lies in two respects. First, the unigram model can be regarded as a particular bigram model by setting all backoff coefficients to 1. Second, the base model is not always a unigram model during the step-by-step growing algorithm, which will be discussed in the next subsection.

In fact, bigram probability $P'(w_y|w_x)$ is extracted from full-bigram model, so $P'(w_y|w_x) = P_F(w_y|w_x)$. In addition, similar deductions can be conducted to the second bracket in equation (11). Thus, we have:

$$\begin{aligned} \Delta d(S; w_x w_y) &= [n(W_F^*, w_x w_y) - n(W_B^*, w_x w_y)] \\ &\quad \times [\log P_F(w_y | w_x) - \log P_B(w_y) - \log \alpha_B(w_x)] \end{aligned} \quad (13)$$

Note that $d(S; \Lambda, \Gamma)$ approximately indicates the likelihood that S will be incorrectly segmented, so $\Delta d(S; w_x w_y)$ represents the performance im-

provement caused by adding $w_x w_y$. Thus, "importance" of bigram $w_x w_y$ on S is computed as:

$$imp(w_x w_y; S) = \Delta d(S; w_x w_y) \quad (14)$$

Case 2: $W_F^* \neq W_0$ and $W_B^* = W_0$

Here, it is just contrary to case 1. In this way, we have:

$$imp(w_x w_y; S) = -\Delta d(S; w_x w_y) \quad (15)$$

Case 3: $W_F^* \neq W_0 \neq W_B^*$

In case 1 and 2, bigrams are added so that discriminant function of correct word sequence becomes bigger, and that of incorrect word sequence becomes smaller. In case 3, both W_F^* and W_B^* are incorrect. Thus, the misclassification function in equation (8) does not represent the likelihood that S will be incorrectly segmented. Therefore, variation of the misclassification function in equation (13) can not be used to measure the "importance" of a bigram. Here, sentence S is ignored, and the "importance" of all bigrams on S are zero.

The above three cases are designed for one sentence. The "importance" of each bigram on the whole training corpus is the sum of its "importance" on each single sentence, as equation (16) shows.

$$imp(w_x w_y) = \sum_S imp(w_x w_y; S) \quad (16)$$

To sum up, the "importance" of each bigram is computed as Figure 3 shows.

1. For each $w_x w_y$, set $imp(w_x w_y) = 0$;
2. For each sentence in training corpus:
For each $w_x w_y$:
if $W_F^* = W_0$ and $W_B^* \neq W_0$:
 $imp(w_x w_y) += \Delta d(S; w_x w_y)$;
else if $W_F^* \neq W_0$ and $W_B^* = W_0$:
 $imp(w_x w_y) -= \Delta d(S; w_x w_y)$;

Figure 3. Calculation of "Importance" of Bigrams

We illustrate the process of computing "importance" of bigrams with a simple example. Suppose S is "这(zhe4)样(yang4)才(cai2)能(neng2)更(geng4)方(fang1)便(bian4)". The segmented result using full-bigram model is "这样(zhe4yang4)/才(cai2)/能(neng2)/更(geng4)/方便(fang1bian4)", which is the correct word sequence. The segmented result using base model

is "这样(zhe4yang4)/才能(cai2neng2)/更(geng4)/方便(fang1bian4)". Obviously, it matches case 1. For bigram "这样(zhe4yang4)才(cai2)", it occurs in W_F^* once, and does not occur in W_B^* . According to equation (13), its "importance" on sentence S is:

$$\begin{aligned} & \text{imp}(\text{这样(zhe4yang4)才(cai2)}; S) \\ &= \log P_F(\text{才(cai2)} | \text{这样(zhe4yang4)}) - \\ & \quad [\log P_B(\text{才(cai2)}) + \log \alpha_B(\text{这样(zhe4yang4)})] \end{aligned}$$

For bigram "更(geng4)方便(fang1bian4)", since it occurs once both in W_F^* and W_B^* , its "importance" on S is zero.

3.3 Step-by-step Growing

Given the target model size, we can add exact number of bigrams to the base model at one time by using the growing algorithm illustrated in Figure 2. But it is more suitable to adopt a step-by-step growing algorithm illustrated in Figure 4.

As shown in equation (13), the "importance" of each bigram depends on the base model. Initially, the base model is set to the unigram model. With bigrams added in, it becomes a growing bigram model. Thus, W_B^* and $\log \alpha_B(w_x)$ will change. So, the added bigrams will affect the calculation of "importance" of bigrams to be added. Generally, adding more bigrams at one time will lead to more negative impacts. Thus, it is expected that models produced by step-by-step growing algorithm may achieve better performance than growing algorithm, and smaller step size will lead to even better performance.

1. Given step size s ;
2. Set the base model to be the unigram model;
3. Segment corpus with full-bigram model;
4. Segment corpus with base model;
5. Compute "importance" of each bigram included in the full-bigram model but excluded from the base model;
6. Sort the bigrams according to their "importance";
7. Add s bigrams with the biggest "importance" to the base model;
8. Re-compute backoff coefficients in the base model;
9. If the base model is still smaller than the desired size, go to step 4; otherwise, stop.

Figure 4. Step-by-step Growing Algorithm

4 Experiments

4.1 Experiment Settings

The training corpus comes from *People's daily 2000*, containing about 25 million Chinese characters. It is manually segmented into word sequences, according to the word segmentation specification of Peking University (Yu et al., 2003). The testing text that is provided by Peking University comes from the second international Chinese word segmentation bakeoff organized by SIGHAN. The testing text is a part of *People's daily 2001*, consisting of about 170K Chinese characters.

The vocabulary is automatically extracted from the training corpus, and the words occurring only once are removed. Finally, about 67K words are included in the vocabulary. The full-bigram model and the unigram model are trained by CMU language model toolkit (Clarkson and Rosenfeld, 1997). Without any count cut-off, the full-bigram model contains about 2 million bigrams.

The word segmentation system is developed based on a source-channel model similar to that described in (Gao et al., 2003). Viterbi algorithm is applied to find the best word segmentation path.

4.2 Evaluation Metrics

The language models built in our experiments are evaluated by two metrics. One is F-Measure of the word segmentation result; the other is language model perplexity.

For F-Measure evaluation, we firstly segment the raw testing text using the model to be evaluated. Then, the segmented result is evaluated by comparing with the gold standard set. The evaluation tool is also from the word segmentation bakeoff. F-Measure is calculated as:

$$\text{F-Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (17)$$

For perplexity evaluation, the language model to be evaluated is used to provide the bigram probabilities for each word in the testing text. The perplexity is the mean logarithm probability as shown in equation (18):

$$PP(M) = 2^{-\frac{1}{N} \sum_{i=1}^N \log_2 P(w_i | w_{i-1})} \quad (18)$$

4.3 Comparison of Pruning Methods

The Kullback-Leibler Distance (KLD) based method is the state-of-the-art method, and is

taken as the baseline¹. Pruning algorithm illustrated in Figure 1 is used for KLD based pruning. Growing algorithms illustrated in Figure 2 and Figure 4 are used for discriminative pruning method. Growing algorithms are not applied to KLD based pruning, because the computation of KLD is independent of the base model.

At step 1 for KLD based pruning, m is set to produce ten models containing 10K, 20K, ..., 100K bigrams. We apply each of the models to the word segmentation system, and evaluate the segmented results with the evaluation tool. The F-Measures of the ten models are illustrated in Figure 5, denoted by "KLD".

For the discriminative pruning criterion, the growing algorithm illustrated in Figure 2 is firstly used. Unigram model acts as the base model. At step 1, n is set to 10K, 20K, ..., 100K separately. At step 2, "importance" of each bigram is computed following Figure 3. Ten models are produced and evaluated. The F-Measures are also illustrated in Figure 5, denoted by "Discrim".

By adding bigrams step by step as illustrated in Figure 4, and setting step size to 10K, 5K, and 2K separately, we obtain other three series of models, denoted by "Step-10K", "Step-5K" and "Step-2K" in Figure 5.

We also include in Figure 5 the performance of the count cut-off method. Obviously, it is inferior to other methods.

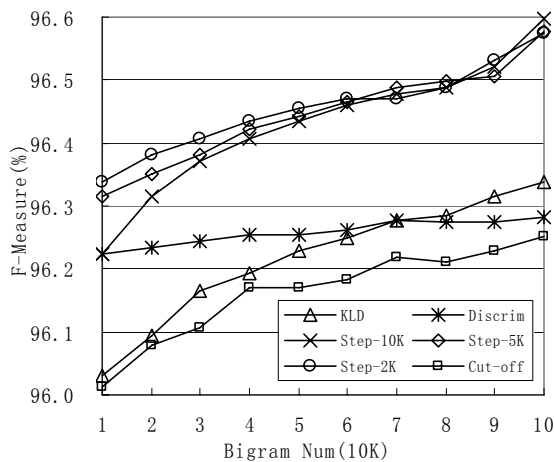


Figure 5. Performance Comparison of Different Pruning Methods

First, we compare the performance of "KLD" and "Discrim". When the model size is small,

such as those models containing less than 70K bigrams, the performance of "Discrim" is better than "KLD". For the models containing more than 70K bigrams, "KLD" gets better performance than "Discrim". The reason is that the added bigrams affect the calculation of "importance" of bigrams to be added, which has been discussed in section 3.3.

If we add the bigrams step by step, better performance is achieved. From Figure 5, it can be seen that all of the models generated by step-by-step growing algorithm outperform "KLD" and "Discrim" consistently. Compared with the baseline KLD based method, step-by-step growing methods result in at least 0.2 percent improvement for each model size.

Comparing "Step-10K", "Step-5K" and "Step-2K", they perform differently before the 60K-bigram point, and perform almost the same after that. The reason is that they are approaching their saturation states, which will be discussed in section 4.5. Before 60K-bigram point, smaller step size yields better performance.

An example of detailed comparison result is shown in Table 1, where the F-Measure is 96.33%. The last column shows the relative model sizes with respect to the KLD pruned model. It shows that with the F-Measure of 96.33%, number of bigrams decreases by up to 90%.

	# of bigrams	% of KLD
KLD	100,000	100%
Step-10K	25,000	25%
Step-5K	15,000	15%
Step-2K	10,000	10%

Table 1. Comparison of Number of Bigrams at F-Measure 96.33%

4.4 Correlation between Perplexity and F-Measure

Perplexities of the models built above are evaluated over the gold standard set. Figure 6 shows how the perplexities vary with the bigram numbers in models. Here, we notice that the KLD models achieve the lowest perplexities. It is not a surprising result, because the goal of KLD based pruning is to minimize the Kullback-Leibler distance that can be interpreted as a relative change of perplexity (Stolcke, 1998).

Now we compare Figure 5 and Figure 6. Perplexities of KLD models are much lower than that of the other models, but their F-Measures are much worse than that of step-by-step growing

¹ Our pilot study shows that the method based on Kullback-Leibler distance outperforms methods based on other criteria introduced in section 2.

models. It implies that lower perplexity does not always lead to higher F-Measure.

However, when the comparison is restricted in a single pruning method, the case is different. For each pruning method, as more bigrams are included in the model, the perplexity curve falls, and the F-Measure curve rises. It implies there are correlations between them. We compute the Pearson product-moment correlation coefficient for each pruning method, as listed in Table 2. It shows that the correlation between perplexity and F-Measure is very strong.

To sum up, the correlation between language model perplexity and system performance (here represented by F-Measure) depends on whether the models come from the same pruning method. If so, the correlation is strong. Otherwise, the correlation is weak.

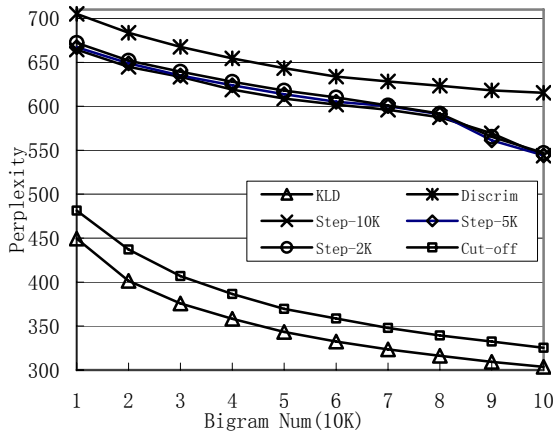


Figure 6. Perplexity Comparison of Different Pruning Methods

Pruning Method	Correlation
Cut-off	-0.990
KLD	-0.991
Discrim	-0.979
Step-10K	-0.985
Step-5K	-0.974
Step-2K	-0.995

Table 2. Correlation between Perplexity and F-Measure

4.5 Combination of Saturated Model and KLD

The above experimental results show that step-by-step growing models achieve the best performance when less than 100K bigrams are added in. Unfortunately, they can not grow up into any desired size. A bigram has no chance to be added into the base model, unless it appears in

the mis-aligned part of the segmented corpus, where $W_F^* \neq W_B^*$. It is likely that not all bigrams have the opportunity. As more and more bigrams are added into the base model, the segmented training corpus using the current base model approaches to that using the full-bigram model. Gradually, none bigram can be added into the current base model. At that time, the model stops growing, and reaches its saturation state. The model that reaches its saturation state is named as *saturated model*. In our experiments, three step-by-step growing models reach their saturation states when about 100K bigrams are added in.

By combining with the baseline KLD based method, we obtain models that outperform the baseline for any model size. We combine them as follows. If the desired model size is smaller than that of the saturated model, step-by-step growing is applied. Otherwise, Kullback-Leibler distance is used for further growing over the saturated model. For instance, by growing over the saturated model of "Step-2K", we obtain combined models containing from 100K to 2 million bigrams. The performance of the combined models and that of the baseline KLD models are illustrated in Figure 7. It shows that the combined model performs consistently better than KLD model over all of bigram numbers. Finally, the two curves converge at the performance of the full-bigram model.

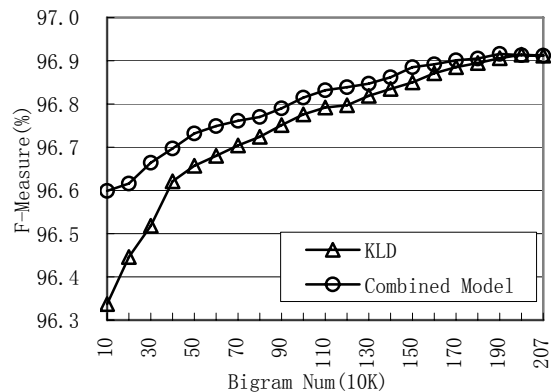


Figure 7. Performance Comparison of Combined Model and KLD Model

5 Conclusions and Future Work

A discriminative pruning criterion of n -gram language model for Chinese word segmentation was proposed in this paper, and a step-by-step growing algorithm was suggested to generate the model of desired size based on a full-bigram model and a base model. Experimental results

showed that the discriminative pruning method achieves significant improvements over the baseline KLD based method. At the same F-measure, the number of bigrams can be reduced by up to 90%. By combining the saturated model and the baseline KLD based method, we achieved better performance for any model size. Analysis shows that, if the models come from the same pruning method, the correlation between perplexity and performance is strong. Otherwise, the correlation is weak.

The pruning methods discussed in this paper focus on bigram pruning, keeping unigram probabilities unchanged. The future work will attempt to prune bigrams and unigrams simultaneously, according to a same discriminative pruning criterion. And we will try to improve the efficiency of the step-by-step growing algorithm. In addition, the method described in this paper can be extended to other applications, such as IME and speech recognition, where language models are applied in a similar way.

References

- Philip Clarkson and Ronald Rosenfeld. 1997. Statistical Language Modeling Using the CMU-Cambridge Toolkit. In *Proc. of the 5th European Conference on Speech Communication and Technology (Eurospeech-1997)*, pages 2707-2710.
- Michael Collins. 2000. Discriminative Reranking for Natural Language Parsing. In *Machine Learning: Proc. of 17th International Conference (ICML-2000)*, pages 175-182.
- Jianfeng Gao and Kai-Fu Lee. 2000. Distribution-based pruning of backoff language models. In *Proc. of the 38th Annual Meeting of Association for Computational Linguistics (ACL-2000)*, pages 579-585.
- Jianfeng Gao, Mu Li, and Chang-Ning Huang. 2003. Improved Source-channel Models for Chinese Word Segmentation. In *Proc. of the 41st Annual Meeting of Association for Computational Linguistics (ACL-2003)*, pages 272-279.
- Jianfeng Gao, Mu Li, Andi Wu, and Chang-Ning Huang. 2005. Chinese Word Segmentation and Named Entity Recognition: A Pragmatic Approach. *Computational Linguistics*, 31(4): 531-574.
- Jianfeng Gao and Min Zhang. 2002. Improving Language Model Size Reduction using Better Pruning Criteria. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 176-182.
- Fredrick Jelinek. 1990. Self-organized language modeling for speech recognition. In Alexander Waibel and Kai-Fu Lee (Eds.), *Readings in Speech Recognition*, pages 450-506.
- Hong-Kwang Jeff Kuo, Eric Fosler-Lussier, Hui Jiang, and Chin-Hui Lee. 2002. Discriminative Training of Language Models for Speech Recognition. In *Proc. of the 27th International Conference On Acoustics, Speech and Signal Processing (ICASSP-2002)*, pages 325-328.
- Franz Josef Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 295-302.
- Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese Segmentation and New Word Detection using Conditional Random Fields. In *Proc. of the 20th International Conference on Computational Linguistics (COLING-2004)*, pages 562-568.
- Brian Roark, Murat Saraclar, Michael Collins, and Mark Johnson. 2004. Discriminative Language Modeling with Conditional Random Fields and the Perceptron Algorithm. In *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-2004)*, pages 47-54.
- Kristie Seymore and Ronald Rosenfeld. 1996. Scalable Backoff Language Models. In *Proc. of the 4th International Conference on Spoken Language Processing (ICSLP-1996)*, pages 232-235.
- Richard Sproat, Chilin Shih, William Gale, and Nancy Chang. 1996. A Stochastic Finite-state Word-segmentation Algorithm for Chinese. *Computational Linguistics*, 22(3): 377-404.
- Andreas Stolcke. 1998. Entropy-based Pruning of Backoff Language Models. In *Proc. of DARPA News Transcription and Understanding Workshop*, pages 270-274.
- Maosong Sun and Benjamin K. Tsou. 2001. A Review and Evaluation on Automatic Segmentation of Chinese. *Contemporary Linguistics*, 3(1): 22-32.
- Shiwen Yu, Huiming Duan, Xuefeng Zhu, Bin Swen, and Baobao Chang. 2003. Specification for Corpus Processing at Peking University: Word Segmentation, POS Tagging and Phonetic Notation. *Journal of Chinese Language and Computing*, 13(2): 121-158.
- Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. 2003. HHMM-based Chinese Lexical Analyzer ICTCLAS. In *Proc. of the ACL-2003 Workshop on Chinese Language Processing (SIGHAN)*, pages 184-187.

Novel Association Measures Using Web Search with Double Checking

Hsin-Hsi Chen

Ming-Shun Lin

Yu-Chuan Wei

Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan

hhchen@csie.ntu.edu.tw; {mslin, ycwei}@nlg.csie.ntu.edu.tw

Abstract

A web search with double checking model is proposed to explore the web as a live corpus. Five association measures including variants of *Dice*, *Overlap Ratio*, *Jaccard*, and *Cosine*, as well as *Co-Occurrence Double Check (CODC)*, are presented. In the experiments on Rubenstein-Goodenough's benchmark data set, the *CODC* measure achieves correlation coefficient 0.8492, which competes with the performance (0.8914) of the model using WordNet. The experiments on link detection of named entities using the strategies of *direct association*, *association matrix* and *scalar association matrix* verify that the double-check frequencies are reliable. Further study on named entity clustering shows that the five measures are quite useful. In particular, *CODC* measure is very stable on word-word and name-name experiments. The application of *CODC* measure to expand community chains for personal name disambiguation achieves 9.65% and 14.22% increase compared to the system without community expansion. All the experiments illustrate that the novel model of web search with double checking is feasible for mining associations from the web.

1 Introduction

In statistical natural language processing, resources used to compute the statistics are indispensable. Different kinds of corpora have made available and many language models have been experimented. One major issue behind the corpus-based approaches is: if corpora adopted can

reflect the up-to-date usage. As we know, languages are live. New terms and phrases are used in daily life. How to capture the new usages is an important research topic.

The Web is a heterogeneous document collection. Huge-scale and dynamic nature are characteristics of the Web. Regarding the Web as a live corpus becomes an active research topic recently. How to utilize the huge volume of web data to measure association of information is an important issue. Resnik and Smith (2003) employ the Web as parallel corpora to provide bilingual sentences for translation models. Keller and Lapata (2003) show that bigram statistics for English language is correlated between corpus and web counts. Besides, how to get the word counts and the word association counts from the web pages without scanning over the whole collections is indispensable. Directly managing the web pages is not an easy task when the Web grows very fast.

Search engine provides some way to return useful information. Page counts for a query denote how many web pages containing a specific word or a word pair roughly. Page count is different from word frequency, which denotes how many occurrences a word appear. Lin and Chen (2004) explore the use of the page counts provided by different search engines to compute the statistics for Chinese segmentation. In addition to the page counts, snippets returned by web search, are another web data for training. A snippet consists of a title, a short summary of a web page and a hyperlink to the web page. Because of the cost to retrieve the full web pages, short summaries are always adopted (Lin, Chen, and Chen, 2005).

Various measures have been proposed to compute the association of objects of different granularity like terms and documents. Rodríguez and Egenhofer (2003) compute the semantic

similarity from WordNet and SDTS ontology by word matching, feature matching and semantic neighborhood matching. Li et al. (2003) investigate how information sources could be used effectively, and propose a new similarity measure combining the shortest path length, depth and local density using WordNet. Matsuo et al. (2004) exploit the Jaccard coefficient to build “Web of Trust” on an academic community.

This paper measures the association of terms using snippets returned by web search. A web search with double checking model is proposed to get the statistics for various association measures in Section 2. Common words and personal names are used for the experiments in Sections 3 and 4, respectively. Section 5 demonstrates how to derive communities from the Web using association measures, and employ them to disambiguate personal names. Finally, Section 6 concludes the remarks.

2 A Web Search with Double Checking Model

Instead of simple web page counts and complex web page collection, we propose a novel model, a *Web Search with Double Checking (WSDC)*, to analyze snippets. In *WSDC* model, two objects X and Y are postulated to have an association if we can find Y from X (a forward process) and find X from Y (a backward process) by web search. The forward process counts the total occurrences of Y in the top N snippets of query X , denoted as $f(Y@X)$. Similarly, the backward process counts the total occurrences of X in the top N snippets of query Y , denoted as $f(X@Y)$. The forward and the backward processes form a double check operation.

Under *WSDC* model, the association scores between X and Y are defined by various formulas as follows.

$$\text{VariantDice}(X,Y) = \begin{cases} 0 & \text{if } f(Y@X) = 0 \text{ or } f(X@Y) = 0 \\ \frac{f(Y@X) + f(X@Y)}{f(X) + f(Y)} & \text{Otherwise} \end{cases} \quad (1)$$

$$\text{VariantCosine}(X,Y) = \frac{\min(f(Y@X), f(X@Y))}{\sqrt{f(X) \times f(Y)}} \quad (2)$$

$$\text{VariantJaccard}(X,Y) = \frac{\min(f(Y@X), f(X@Y))}{f(X) + f(Y) - \max(f(Y@X), f(X@Y))} \quad (3)$$

$$\text{VariantOverlap}(X,Y) = \frac{\min\{f(Y@X), f(X@Y)\}}{\min\{f(X), f(Y)\}} \quad (4)$$

$$\text{CODC}(X,Y) = \begin{cases} 0 & \text{if } f(Y@X) = 0 \text{ or } f(X@Y) = 0 \\ e^{\log\left(\frac{f(Y@X)}{f(X)} \times \frac{f(X@Y)}{f(Y)}\right)} & \text{Otherwise} \end{cases} \quad (5)$$

Where $f(X)$ is the total occurrences of X in the top N snippets of query X , and, similarly, $f(Y)$ is the total occurrences of Y in the top N snippets of query Y . Formulas (1)-(4) are variants of the *Dice*, *Cosine*, *Jaccard*, and *Overlap Ratio* association measure. Formula (5) is a function *CODC* (*Co-Occurrence Double-Check*), which measures the association in an interval $[0,1]$. In the extreme cases, when $f(Y@X)=0$ or $f(X@Y)=0$, $\text{CODC}(X,Y)=0$; and when $f(Y@X)=f(X)$ and $f(X@Y)=f(Y)$, $\text{CODC}(X,Y)=1$. In the first case, X and Y are of no association. In the second case, X and Y are of the strongest association.

3 Association of Common Words

We employ Rubenstein-Goodenough's (1965) benchmark data set to compare the performance of various association measures. The data set consists of 65 word pairs. The similarities between words, called Rubenstein and Goodenough rating (RG rating), were rated on a scale of 0.0 to 4.0 for “semantically unrelated” to “highly synonymous” by 51 human subjects. The Pearson product-moment correlation coefficient, r_{xy} , between the RG ratings X and the association scores Y computed by a model shown as follows measures the performance of the model.

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y} \quad (6)$$

Where $\bar{x}$ and $\bar{y}$ are the sample means of x_i and y_i , and s_x and s_y are sample standard deviations of x_i and y_i and n is total samples.

Most approaches (Resnik, 1995; Lin, 1998; Li et al., 2003) used 28 word pairs only. Resnik (1995) obtained information content from WordNet and achieved correlation coefficient 0.745. Lin (1998) proposed an information-theoretic similarity measure and achieved a correlation coefficient of 0.8224. Li et al. (2003) combined semantic density, path length and depth effect from WordNet and achieved the correlation coefficient 0.8914.

	100	200	300	400	500	600	700	800	900
<i>VariantDice</i>	0.5332	0.5169	0.5352	0.5406	0.5306	0.5347	0.5286	0.5421	0.5250
<i>VariantOverlap</i>	0.5517	0.6516	0.6973	0.7173	0.6923	0.7259	0.7473	0.7556	0.7459
<i>VariantJaccard</i>	0.5533	0.6409	0.6993	0.7229	0.6989	0.738	0.7613	0.7599	0.7486
<i>VariantCosine</i>	0.5552	0.6459	0.7063	0.7279	0.6987	0.7398	0.7624	0.7594	0.7501
<i>CODC</i> ($\alpha=0.15$)	0.5629	0.6951	0.8051	0.8473	0.8438	0.8492	0.8222	0.8291	0.8182
<i>Jaccard Coeff</i> *	0.5847	0.5933	0.6099	0.5807	0.5463	0.5202	0.4855	0.4549	0.4622

Table 1. Correlation Coefficients of WSDC Model on Word-Word Experiments

Model	RG Rating	Resnik (1995)	Lin (1998)	Li et al (2003)	<i>VariantCosine</i> (#snippets=700) WSDC	<i>CODC</i> ($\alpha=0.15$, #snippets=600) WSDC
Correlation Coefficient	-	0.7450	0.8224	0.8914	0.7624	0.8492
chord-smile	0.02	1.1762	0.20	0	0	0
rooster-voyage	0.04	0	0	0	0	0
noon-string	0.04	0	0	0	0	0
glass-magician	0.44	1.0105	0.06	0	0	0
monk-slave	0.57	2.9683	0.18	0.350	0	0
coast-forest	0.85	0	0.16	0.170	0.0019	0.1686
monk-oracle	0.91	2.9683	0.14	0.168	0	0
lad-wizard	0.99	2.9683	0.20	0.355	0	0
forest-graveyard	1	0	0	0.132	0	0
food-rooster	1.09	1.0105	0.04	0	0	0
coast-hill	1.26	6.2344	0.58	0.366	0	0
car-journey	1.55	0	0	0	0.0014	0.2049
crane-implement	2.37	2.9683	0.39	0.366	0	0
brother-lad	2.41	2.9355	0.20	0.355	0.0027	0.1811
bird-crane	2.63	9.3139	0.67	0.472	0	0
bird-cock	2.63	9.3139	0.83	0.779	0.0058	0.2295
food-fruit	2.69	5.0076	0.24	0.170	0.0025	0.2355
brother-monk	2.74	2.9683	0.16	0.779	0.0027	0.1956
asylum-madhouse	3.04	15.666	0.97	0.779	0.0015	0.1845
furnace-stove	3.11	1.7135	0.18	0.585	0.0035	0.1982
magician-wizard	3.21	13.666	1	0.999	0.0031	0.2076
journey-voyage	3.58	6.0787	0.89	0.779	0.0086	0.2666
coast-shore	3.6	10.808	0.93	0.779	0.0139	0.2923
implement-tool	3.66	6.0787	0.80	0.778	0.0033	0.2506
boy-lad	3.82	8.424	0.85	0.778	0.0101	0.2828
Automobile-car	3.92	8.0411	1	1	0.0144	0.4229
Midday-noon	3.94	12.393	1	1	0.0097	0.2994
gem-jewel	3.94	14.929	1	1	0.0107	0.3530

Table 2. Comparisons of WSDC with Models in Previous Researches

In our experiments on the benchmark data set, we used information from the Web rather than WordNet. Table 1 summarizes the correlation coefficients between the RG rating and the association scores computed by our *WSDC* model. We consider the number of snippets from 100 to 900. The results show that *CODC* > *VariantCo-*

sine > *VariantJaccard* > *VariantOverlap* > *VariantDice*. *CODC* measure achieves the best performance 0.8492 when $\alpha=0.15$ and total snippets to be analyzed are 600. Matsuo et al. (2004) used *Jaccard* coefficient to calculate similarity between personal names using the Web. The coefficient is defined as follows.

$$JaccardCoff(X, Y) = \frac{f(X \cap Y)}{f(X \cup Y)} \quad (7)$$

Where $f(X \cap Y)$ is the number of pages including X 's and Y 's homepages when query “ X and Y ” is submitted to a search engine; $f(X \cup Y)$ is the number of pages including X 's or Y 's homepages when query “ X or Y ” is submitted to a search engine. We revised this formula as follows and evaluated it with Rubenstein-Goodenough's benchmark.

$$JaccardCoff(X, Y)^* = \frac{f_s(X \cap Y)}{f_s(X \cup Y)} \quad (8)$$

Where $f_s(X \cap Y)$ is the number of snippets in which X and Y co-occur in the top N snippets of query “ X and Y ”; $f_s(X \cup Y)$ is the number of snippets containing X or Y in the top N snippets of query “ X or Y ”. We test the formula on the same benchmark. The last row of Table 1 shows that *Jaccard Coeff*^{*} is worse than other models when the number of snippets is larger than 100.

Table 2 lists the results of previous researches (Resink, 1995; Lin, 1998; Li et al., 2003) and our *WSDC* models using *VariantCosine* and *CODC* measures. The 28 word pairs used in the experiments are shown. *CODC* measure can compete with Li et al. (2003). The word pair “car-journey” whose similarity value is 0 in the papers (Resink, 1995; Lin, 1998; Li et al., 2003) is captured by our model. In contrast, our model cannot deal with the two word pairs “crane-implement” and “bird-crane”.

4 Association of Named Entities

Although the correlation coefficient of *WSDC* model built on the web is a little worse than that of the model built on WordNet, the Web provides live vocabulary, in particular, named entities. We will demonstrate how to extend our *WSDC* method to mine the association of personal names. That will be difficult to resolve with previous approaches. We design two experiments – say, link detection test and named entity clustering, to evaluate the association of named entities.

Given a named-entity set L , we define a *link detection test* to check if any two named entities NE_i and NE_j ($i \neq j$) in L have a relationship R using the following three strategies.

- **Direct Association:** If the double check frequency of NE_i and NE_j is larger than 0,

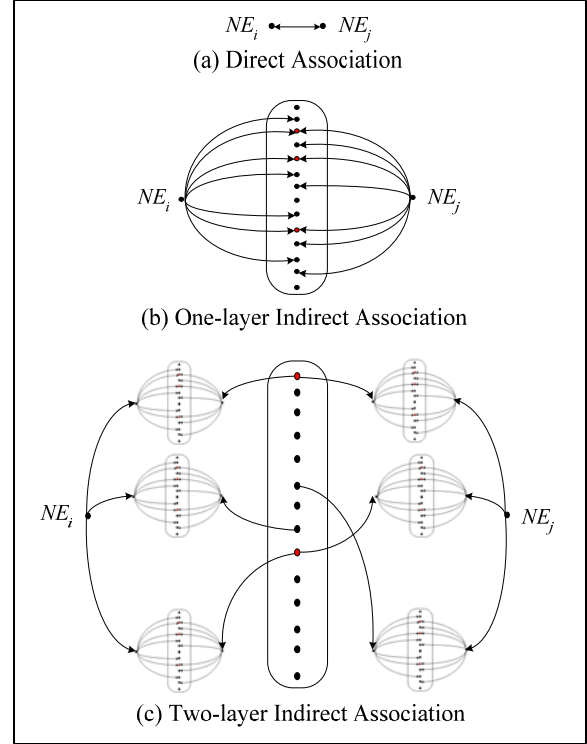


Figure 1. Three Strategies for Link Detection

i.e., $f(NE_j@NE_i) > 0$ and $f(NE_i@NE_j) > 0$, then the link detection test says “yes”, i.e., NE_i and NE_j have *direct association*. Otherwise, the test says “no”. Figure 1(a) shows the direct association.

- **Association Matrix:** Compose an $n \times n$ binary matrix $M = (m_{ij})$, where $m_{ij} = 1$ if $f(NE_j@NE_i) > 0$ and $f(NE_i@NE_j) > 0$; $m_{ij} = 0$ if $f(NE_j@NE_i) = 0$ or $f(NE_i@NE_j) = 0$; and n is total number of named entities in L . Let M' be a transpose matrix of M . The matrix $A = M \times M'$ is an association matrix. Here the element a_{ij} in A means that total a_{ij} common named entities are associated with both NE_i and NE_j directly. Figure 1(b) shows a one-layer indirect association. Here, $a_{ij} = 3$. We can define NE_i and NE_j have an indirect association if a_{ij} is larger than a threshold λ . That is, NE_i and NE_j should associate with at least λ common named entities directly. The strategy of association matrix specifies: if $a_{ij} \geq \lambda$, then the link detection test says “yes”, otherwise it says “no”. In the example shown in Figure 1(b), NE_i and NE_j are indirectly associated when $0 < \lambda \leq 3$.
- **Scalar Association Matrix:** Compose a binary association matrix B from the association matrix A as: $b_{ij} = 1$ if $a_{ij} > 0$ and $b_{ij} = 0$ if $a_{ij} = 0$. The matrix $S = B \times B'$ is a scalar as-

sociation matrix. NE_i and NE_j may indirectly associate with a common named entity NE_k . Figure 1(c) shows a two-layer indirect association. The $s_{ij} = \sum_{k=1}^n b_{ik} \times b_{kj}$ denotes how many such an NE_k there are. In the example of Figure 1(c), two named entities indirectly associate NE_i and NE_j at the same time. We can define NE_i and NE_j have an indirect association if s_{ij} is larger than a threshold δ . In other words, if $s_{ij} > \delta$, then the link detection test says “yes”, otherwise it says “no”.

To evaluate the performance of the above three strategies, we prepare a test set extracted from domz web site (<http://dmoz.org>), the most comprehensive human-edited directory of the Web. The test data consists of three communities: actor, tennis player, and golfer, shown in Table 3. Total 220 named entities are considered. The golden standard of link detection test is: we compose 24,090 ($=220 \times 219/2$) named entity pairs, and assign “yes” to those pairs belonging to the same community.

Category Path in domz.org	# of Person Names
Top: Sports: Golf: Golfers	10
Top: Sports: Tennis: Players: Female (+Male)	90
Top: Arts: People: Image Galleries: Female (+Male): Individual	120

Table 3. Test Set for Association Evaluation of Named Entities

When collecting the related values for computing the double check frequencies for any named entity pair (NE_i and NE_j), i.e., $f(NE_j@NE_i)$, $f(NE_i@NE_j)$, $f(NE_i)$, and $f(NE_j)$, we consider naming styles of persons. For example, “Alba, Jessica” have four possible writing: “Alba, Jessica”, “Jessica Alba”, “J. Alba” and “Alba, J.” We will get top N snippets for each naming style, and filter out duplicate snippets as well as snippets of ULRs including dmoz.org and

google.com. Table 4 lists the experimental results of link detection on the test set. The precisions of two baselines are: guessing all “yes” (46.45%) and guessing all “no” (53.55%). All the three strategies are better than the two baselines and the performance becomes better when the numbers of snippets increase. The strategy of direct association shows that using double checks to measure the association of named entities also gets good effects as the association of common words. For the strategy of association matrix, the best performance 90.14% occurs in the case of 900 snippets and $\lambda=6$. When larger number of snippets is used, a larger threshold is necessary to achieve a better performance. Figure 2(a) illustrates the relationship between precision and threshold (λ). The performance decreases when $\lambda > 6$. The performance of the strategy of scalar association matrix is better than that of the strategy of association matrix in some λ and δ . Figure 2(b) shows the relationship between precision and threshold δ for some number of snippets and λ .

In link detection test, we only consider the binary operation of double checks, i.e., $f(NE_j@NE_i) > 0$ and $f(NE_i@NE_j) > 0$, rather than utilizing the magnitudes of $f(NE_j@NE_i)$ and $f(NE_i@NE_j)$. Next we employ the five formulas proposed in Section 2 to cluster named entities. The same data set as link detection test is adopted. An agglomerative average-link clustering algorithm is used to partition the given 220 named entities based on Formulas (1)-(5). Four-fold cross-validation is employed and B-CUBED metric (Bagga and Baldwin, 1998) is adopted to evaluate the clustering results. Table 5 summarizes the experimental results. CODC (Formula 5), which behaves the best in computing association of common words, still achieves the better performance on different numbers of snippets in named entity clustering. The F-scores of the other formulas are larger than 95% when more snippets are considered to compute the double check frequencies.

Strategies	100	200	300	400	500	600	700	800	900
Direct Association	59.20%	62.86%	65.72%	67.88%	69.83%	71.35%	72.05%	72.46%	72.55%
Association Matrix	71.53% ($\lambda=1$)	79.95% ($\lambda=1$)	84.00% ($\lambda=2$)	86.08% ($\lambda=3$)	88.13% ($\lambda=4$)	89.67% ($\lambda=5$)	89.98% ($\lambda=5$)	90.09% ($\lambda=6$)	90.14% ($\lambda=6$)
Scalar Association Matrix	73.93% ($\lambda=1$, $\delta=6$)	82.69% ($\lambda=2$, $\delta=9$)	86.70% ($\lambda=4$, $\delta=9$)	88.61% ($\lambda=5$, $\delta=10$)	90.90% ($\lambda=6$, $\delta=12$)	91.93% ($\lambda=7$, $\delta=12$)	91.90% ($\lambda=7$, $\delta=18$)	92.20% ($\lambda=10$, $\delta=16$)	92.35% ($\lambda=10$, $\delta=18$)

Table 4. Performance of Link Detection of Named Entities

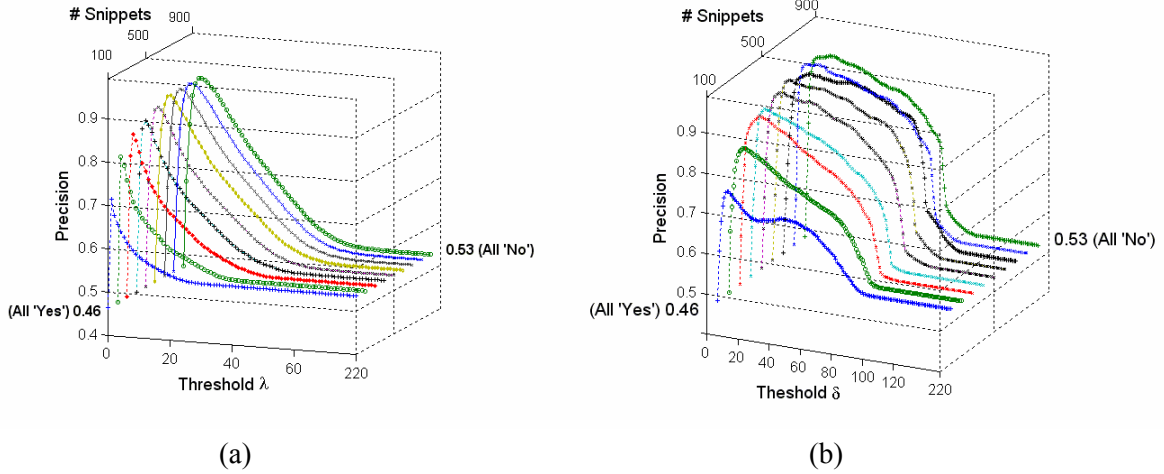


Figure 2. (a) Performance of association matrix strategy. (b) Performance of scalar association matrix strategy (where λ is fixed and its values reference to scalar association matrix in Table 4)

		100	200	300	400	500	600	700	800	900
<i>VariantDice</i>	P	91.70%	88.71%	87.02%	87.49%	96.90%	100.00%	100.00%	100.00%	100.00%
	R	55.80%	81.10%	87.70%	93.00%	89.67%	93.61%	94.42%	94.88%	94.88%
	F	69.38%	84.73%	87.35%	90.16%	93.14%	96.69%	97.12%	97.37%	97.37%
<i>VariantOverlap</i>	P	99.13%	87.04%	85.35%	85.17%	88.16%	88.16%	88.16%	97.59%	98.33%
	R	52.16%	81.10%	86.24%	93.45%	92.03%	93.64%	92.82%	90.82%	93.27%
	F	68.35%	83.96%	85.79%	89.11%	90.05%	90.81%	90.43%	94.08%	95.73%
<i>VariantJaccard</i>	P	99.13%	97.59%	98.33%	95.42%	97.59%	88.16%	95.42%	100.00%	100.00%
	R	55.80%	77.53%	84.91%	88.67%	87.18%	90.58%	88.67%	93.27%	91.64%
	F	71.40%	86.41%	91.12%	91.92%	92.09%	89.35%	91.92%	96.51%	95.63%
<i>VariantCosine</i>	P	84.62%	97.59%	85.35%	85.17%	88.16%	88.16%	88.16%	98.33%	98.33%
	R	56.22%	78.92%	86.48%	93.45%	92.03%	93.64%	93.64%	93.27%	93.27%
	F	67.55%	87.26%	85.91%	89.11%	90.05%	90.81%	90.81%	95.73%	95.73%
<i>CODC</i> ($\alpha=0.15$)	P	91.70%	87.04%	87.02%	95.93%	98.33%	95.93%	95.93%	94.25%	94.25%
	R	55.80%	81.10%	90.73%	94.91%	94.91%	96.52%	98.24%	98.24%	98.24%
	F	69.38%	83.96%	88.83%	95.41%	96.58%	96.22%	97.07%	96.20%	96.20%

Table 5. Performance of Various Scoring Formulas on Named Entity Clustering

5 Disambiguation Using Association of Named Entities

This section demonstrates how to employ association mined from the Web to resolve the ambiguities of named entities. Assume there are n named entities, $NE_1, NE_2, \dots$, and NE_n , to be disambiguated. A named entity NE_j has m accompanying names, called *cue names* later, $CN_{j1}, CN_{j2}, \dots, CN_{jm}$. We have two alternatives to use the cue names. One is using them directly, i.e., NE_j is represented as a *community* of cue names $Community(NE_j) = \{CN_{j1}, CN_{j2}, \dots, CN_{jm}\}$. The

other is to expand the cue names $CN_{j1}, CN_{j2}, \dots, CN_{jm}$ for NE_j using the web data as follows. Let CN_{j1} be an initial seed. Figure 3 sketches the concept of community expansion.

- (1) **Collection:** We submit a seed to Google, and select the top N returned snippets. Then, we use suffix trees to extract possible patterns (Lin and Chen, 2006).
- (2) **Validation:** We calculate *CODC* score of each extracted pattern (denoted B_i) with the seed A . If $CODC(A, B_i)$ is strong enough, i.e., larger than a

threshold θ , we employ B_i as a new seed and repeat steps (1) and (2). This procedure stops either expected number of nodes is collected or maximum number of layers is reached.

- (3) **Union:** The community initiated by the seed CN_{ji} is denoted by $Community(CN_{ji}) = \{B_{ji1}, B_{ji2}, \dots, B_{jik}\}$, where B_{jik} is a new seed. The $Cscore$ score, *community score*, of B_{jik} is the $CODC$ score of B_{jik} with its parent divided by the layer it is located. We repeat *Collection* and *Validation* steps until all the cue names CN_{ji} ($1 \leq i \leq m$) of NE_j are processed. Finally, we have

$$Community(NE_j) = \cup_{i=1}^m Community(CN_{ji})$$

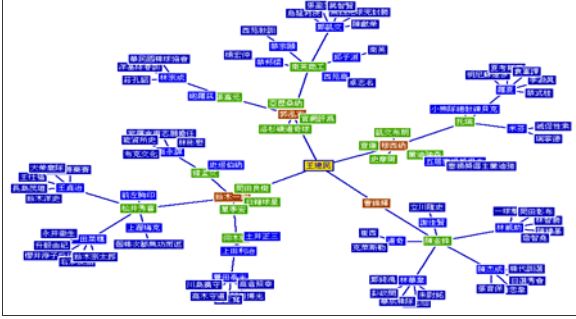


Figure 3. A Community for a Seed “王建民” (“Chien-Ming Wang”)

In a cascaded personal name disambiguation system (Wei, 2006), association of named entities is used with other cues such as titles, common terms, and so on. Assume k clusters, $c_1 c_2 \dots c_k$, have been formed using title cue, and we try to place $NE_1, NE_2, \dots$, and NE_l into a suitable cluster. The cluster $\bar{c}$ is selected by the similarity measure defined below.

$$\begin{aligned} score(NE_j, c_q) \\ = \frac{1}{r} \sum_{i=1}^s count(pn_i) \times Cscore(pn_i) \end{aligned} \quad (9)$$

$$\bar{c} = \arg \max_{c_q (1 \leq q \leq k)} score(NE_j, c_q) \quad (10)$$

Where $pn_1, pn_2, \dots, pn_s$ are names which appear in both $Community(NE_j)$ and $Community(c_q)$; $count(pn_i)$ is total occurrences of pn_i in $Community(c_q)$; r is total occurrences of names in $Community(NE_j)$; $Cscore(pn_i)$ is community score of pn_i .

If $score(NE_j, \bar{c})$ is larger than a threshold, then NE_j is placed into cluster $\bar{c}$. In other words,

NE_j denotes the same person as those in $\bar{c}$. We let the new $Community(\bar{c})$ be the old $Community(\bar{c}) \cup \{CN_{j1}, CN_{j2}, \dots, CN_{jm}\}$. Otherwise, NE_j is left undecided.

To evaluate the personal name disambiguation, we prepare three corpora for an ambiguous name “王建民” (Chien-Ming Wang) from United Daily News Knowledge Base (UDN), Google Taiwan (TW), and Google China (CN). Table 6 summarizes the statistics of the test data sets. In UDN news data set, 37 different persons are mentioned. Of these, 13 different persons occur more than once. The most famous person is a pitcher of New York Yankees, which occupies 94.29% of 2,205 documents. In TW and CN web data sets, there are 24 and 107 different persons. The majority in TW data set is still the New York Yankees’s “Chien-Ming Wang”. He appears in 331 web pages, and occupies 88.03%. Comparatively, the majority in CN data set is a research fellow of Chinese Academy of Social Sciences, and he only occupies 18.29% of 421 web pages. Total 36 different “Chien-Ming Wang”’s occur more than once. Thus, CN is an unbiased corpus.

	UDN	TW	CN
# of documents	2,205	376	421
# of persons	37	24	107
# of persons of occurrences>1	13	9	36
Majority	94.29%	88.03%	18.29%

Table 6. Statistics of Test Corpora

		M1	M2
UDN	P	0.9742	0.9674 (↓0.70%)
	R	0.9800	0.9677 (↓1.26%)
	F	0.9771	0.9675 (↓0.98%)
TW	P	0.8760	0.8786 (↑0.07%)
	R	0.6207	0.7287 (↑17.40%)
	F	0.7266	0.7967 (↑9.65%)
CN	P	0.4910	0.5982 (↑21.83%)
	R	0.8049	0.8378 (↑4.09%)
	F	0.6111	0.6980 (↑14.22%)

Table 7. Disambiguation without/with Community Expansion

Table 7 shows the performance of a personal name disambiguation system without (M1)/with (M2) community expansion. In the news data set (i.e., UDN), M1 is a little better than M2. Compared to M1, M2 decreases 0.98% of F-score. In contrast, in the two web data sets (i.e., TW and CN), M2 is much better than M1. M2 has 9.65% and 14.22% increases compared to M1. It shows that mining association of named entities from the Web is very useful to disambiguate ambiguous names. The application also confirms the effectiveness of the proposed association measures indirectly.

6 Concluding Remarks

This paper introduces five novel association measures based on web search with double checking (*WSDC*) model. In the experiments on association of common words, *Co-Occurrence Double Check* (*CODC*) measure competes with the model trained from WordNet. In the experiments on the association of named entities, which is hard to deal with using WordNet, *WSDC* model demonstrates its usefulness. The strategies of direct association, association matrix, and scalar association matrix detect the link between two named entities. The experiments verify that the double-check frequencies are reliable.

Further study on named entity clustering shows that the five measures – say, *VariantDice*, *VariantOverlap*, *VariantJaccard*, *VariantCosine* and *CODC*, are quite useful. In particular, *CODC* is very stable on word-word and name-name experiments. Finally, *WSDC* model is used to expand community chains for a specific personal name, and *CODC* measures the association of community member and the personal name. The application on personal name disambiguation shows that 9.65% and 14.22% increase compared to the system without community expansion.

Acknowledgements

Research of this paper was partially supported by National Science Council, Taiwan, under the contracts 94-2752-E-001-001-PAE and 95-2752-E-001-001-PAE.

References

A. Bagga and B. Baldwin. 1998. Entity-Based Cross-Document Coreferencing Using the Vector Space

Model. *Proceedings of 36th COLING-ACL Conference*, 79-85.

F. Keller and M. Lapata. 2003. Using the Web to Obtain Frequencies for Unseen Bigrams. *Computational Linguistics*, 29(3): 459-484.

Y. Li, Z.A. Bandar and D. McLean. 2003. An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4): 871-882.

D. Lin. 1998. An Information-Theoretic Definition of Similarity. *Proceedings of the Fifteenth International Conference on Machine Learning*, 296-304.

H.C. Lin and H.H. Chen. 2004. Comparing Corpus-based Statistics and Web-based Statistics: Chinese Segmentation as an Example. *Proceedings of 16th ROCLING Conference*, 89-100.

M.S. Lin, C.P. Chen and H.H. Chen. 2005. An Approach of Using the Web as a Live Corpus for Spoken Transliteration Name Access. *Proceedings of 17th ROCLING Conference*, 361-370.

M.S. Lin and H.H. Chen. 2006. Constructing a Named Entity Ontology from Web Corpora. *Proceedings of 5th International Conference on Language Resources and Evaluation*.

Y. Matsuo, H. Tomobe, K. Hasida, and M. Ishizuka. 2004. Finding Social Network for Trust Calculation. *Proceedings of 16th European Conference on Artificial Intelligence*, 510-514.

P. Resnik. 1995. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 448-453.

P. Resnik and N.A. Smith. 2003. The Web as a Parallel Corpus. *Computational Linguistics*, 29(3): 349-380.

M.A. Rodríguez and M.J. Egenhofer. 2003. Determining Semantic Similarity among Entity Classes from Different Ontologies. *IEEE Transactions on Knowledge and Data Engineering*, 15(2): 442-456.

H. Rubenstein and J.B. Goodenough. 1965. Contextual Correlates of Synonymy. *Communications of the ACM*, 8(10): 627-633.

Y.C. Wei. 2006. *A Study of Personal Name Disambiguation*. Master Thesis, Department of Computer Science and Information Engineering, National Taiwan University, Taiwan.

Semantic Retrieval for the Accurate Identification of Relational Concepts in Massive Textbases

Yusuke Miyao* Tomoko Ohta* Katsuya Masuda* Yoshimasa Tsuruoka†
Kazuhiro Yoshida* Takashi Ninomiya‡ Jun'ichi Tsujii*†

*Department of Computer Science, University of Tokyo

†School of Informatics, University of Manchester

‡Information Technology Center, University of Tokyo

Hongo 7-3-1, Bunkyo-ku, Tokyo 113-0033 JAPAN

{yusuke, okap, kmasuda, tsuruoka, kyoshida, ninomi, tsujii}@is.s.u-tokyo.ac.jp

Abstract

This paper introduces a novel framework for the accurate retrieval of relational concepts from huge texts. Prior to retrieval, all sentences are annotated with predicate argument structures and ontological identifiers by applying a deep parser and a term recognizer. During the run time, user requests are converted into queries of region algebra on these annotations. Structural matching with pre-computed semantic annotations establishes the accurate and efficient retrieval of relational concepts. This framework was applied to a text retrieval system for MEDLINE. Experiments on the retrieval of biomedical correlations revealed that the cost is sufficiently small for real-time applications and that the retrieval precision is significantly improved.

1 Introduction

Rapid expansion of text information has motivated the development of efficient methods of accessing information in huge texts. Furthermore, user demand has shifted toward the retrieval of more precise and complex information, including *relational concepts*. For example, biomedical researchers deal with a massive quantity of publications; MEDLINE contains approximately 15 million references to journal articles in life sciences, and its size is rapidly increasing, at a rate of more than 10% yearly (National Library of Medicine, 2005). Researchers would like to be able to search this huge textbase for biomedical correlations such as protein-protein or gene-disease associations (Blaschke and Valencia, 2002; Hao et al., 2005; Chun et al., 2006). However, the framework

of traditional information retrieval (IR) has difficulty with the accurate retrieval of such relational concepts because relational concepts are essentially determined by semantic relations between words, and keyword-based IR techniques are insufficient to describe such relations precisely.

The present paper demonstrates a framework for the accurate real-time retrieval of relational concepts from huge texts. Prior to retrieval, we prepare a semantically annotated textbase by applying NLP tools including deep parsers and term recognizers. That is, all sentences are annotated in advance with semantic structures and are stored in a structured database. User requests are converted on the fly into patterns of these semantic annotations, and texts are retrieved by matching these patterns with the pre-computed semantic annotations. The accurate retrieval of relational concepts is attained because we can precisely describe relational concepts using semantic annotations. In addition, real-time retrieval is possible because semantic annotations are computed in advance.

This framework has been implemented for a text retrieval system for MEDLINE. We first apply a deep parser (Miyao and Tsujii, 2005) and a dictionary-based term recognizer (Tsuruoka and Tsujii, 2004) to MEDLINE and obtain annotations of predicate argument structures and ontological identifiers of genes, gene products, diseases, and events. We then provide a search engine for these annotated sentences. User requests are converted into queries of region algebra (Clarke et al., 1995) extended with variables (Masuda et al., 2006) on these annotations. A search engine for the extended region algebra efficiently finds sentences having semantic annotations that match the input queries. In this paper, we evaluate this system with respect to the retrieval of biomedical correlations

Symbol	CRP
Name	C-reactive protein, pentraxin-related
Species	Homo sapiens
Synonym	MGC88244, PTX1
Product	C-reactive protein precursor, C-reactive protein, pentraxin-related protein
External links	EntrezGene:1401, GDB:119071, ...

Table 1: An example GENA entry

and examine the effects of using predicate argument structures and ontological identifiers.

The need for the discovery of relational concepts has been investigated intensively in Information Extraction (IE). However, little research has targeted on-demand retrieval from huge texts. One difficulty is that IE techniques such as pattern matching and machine learning require heavier processing in order to be applied on the fly. Another difficulty is that target information must be formalized beforehand and each system is designed for a specific task. For instance, an IE system for protein-protein interactions is not useful for finding gene-disease associations. Apart from IE research, enrichment of texts with various annotations has been proposed and is becoming a new research area for information management (IBM, 2005; TEI, 2004). The present study basically examines this new direction in research. The significant contribution of the present paper, however, is to provide the first empirical results of this framework for a real task with a huge textbase.

2 Background: Resources and Tools for Semantic Annotations

The proposed system for the retrieval of relational concepts is a product of recent developments in NLP resources and tools. In this section, ontology databases, deep parsers, and search algorithms for structured data are introduced.

2.1 Ontology databases

Ontology databases are collections of words and phrases in specific domains. Such databases have been constructed extensively for the systematic management of domain knowledge by organizing textual expressions of ontological entities that are detached from actual sentences.

For example, GENA (Koike and Takagi, 2004) is a database of genes and gene products that is semi-automatically collected from well-known databases, including HUGO, OMIM, Genatlas, Locuslink, GDB, MGI, FlyBase, WormBase,

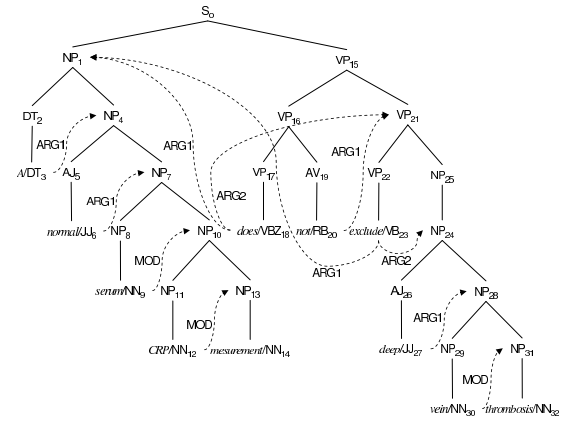


Figure 1: An output of HPSG parsing



Figure 2: A predicate argument structure

CYGD, and SGD. Table 1 shows an example of a GENA entry. “Symbol” and “Name” denote short forms and nomenclatures of genes, respectively. “Species” represents the organism species in which this gene is observed. “Synonym” is a list of synonyms and name variations. “Product” gives a list of products of this gene, such as proteins coded by this gene. “External links” provides links to other databases, and helps to obtain detailed information from these databases. For biomedical terms other than genes/gene products, the Unified Medical Language System (UMLS) meta-thesaurus (Lindberg et al., 1993) is a large database that contains various names of biomedical and health-related concepts.

Ontology databases provide mappings between textual expressions and entities in the real world. For example, Table 1 indicates that CRP, MGC88244, and PTX1 denote the same gene conceptually. Hence, these resources enable us to canonicalize variations of textual expressions of ontological entities.

2.2 Parsing technologies

Recently, state-of-the-art CFG parsers (Charniak and Johnson, 2005) can compute phrase structures of natural sentences at fairly high accuracy. These parsers have been used in various NLP tasks including IE and text mining. In addition, parsers that compute deeper analyses, such as predicate argument structures, have become available for

the processing of real-world sentences (Miyao and Tsujii, 2005). Predicate argument structures are canonicalized representations of sentence meanings, and express the semantic relations of words explicitly. Figure 1 shows an output of an HPSG parser (Miyao and Tsujii, 2005) for the sentence “A normal serum CRP measurement does not exclude deep vein thrombosis.” The dotted lines express predicate argument relations. For example, the ARG1 arrow coming from “exclude” points to the noun phrase “A normal serum CRP measurement”, which indicates that the subject of “exclude” is this noun phrase, while such relations are not explicitly represented by phrase structures.

Predicate argument structures are beneficial for our purpose because they can represent relational concepts in an abstract manner. For example, the relational concept of “CRP excludes thrombosis” can be represented as a predicate argument structure, as shown in Figure 2. This structure is universal in various syntactic expressions, such as passivization (e.g., “thrombosis is excluded by CRP”) and relativization (e.g., “thrombosis that CRP excludes”). Hence, we can abstract surface variations of sentences and describe relational concepts in a canonicalized form.

2.3 Structural search algorithms

Search algorithms for structured texts have been studied extensively, and examples include XML databases with XPath (Clark and DeRose, 1999) and XQuery (Boag et al., 2005), and region algebra (Clarke et al., 1995). The present study focuses on region algebra extended with variables (Masuda et al., 2006) because it provides an efficient search algorithm for tags with cross boundaries. When we annotate texts with various levels of syntactic/semantic structures, cross boundaries are inherently nonnegligible. In fact, as described in Section 3, our system exploits annotations of predicate argument structures and ontological entities, which include substantial cross boundaries.

Region algebra is defined as a set of operators on *regions*, i.e., word sequences. Table 2 shows operators of the extended region algebra, where A and B denote regions, and results of operations are also regions. For example, “A & B” denotes a region that includes both A and B. Four containment operators, >, >>, <, and <<, represent ancestor/descendant relations in XML. For example, “A > B” indicates that A is an ancestor of B. In

[tag]	Region covered with “<tag>”
A > B	A containing B
A >> B	A containing B (A is not nested)
A < B	A contained by B
A << B	A contained by B (B is not nested)
A - B	Starting with A and ending with B
A & B	A and B
A B	A or B

Table 2: Operators of the extended region algebra

```
[sentence] >>
(( [word arg1="$subject"] > exclude) &
  ([phrase id="$subject"] > CRP))
```

Figure 3: A query of the extended region algebra

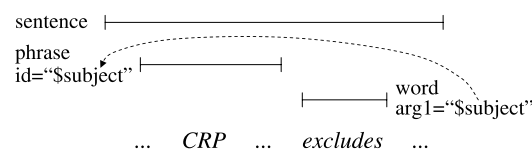


Figure 4: Matching with the query in Figure 3

search algorithms for region algebra, the cost of retrieving the first answer is constant, and that of an exhaustive search is bounded by the lowest frequency of a word in a query (Clarke et al., 1995).

Variables in the extended region algebra allow us to express shared structures and are necessary in order to describe predicate argument structures. For example, Figure 3 shows a formula in the extended region algebra that represents the predicate argument structure of “CRP excludes something.” This formula indicates that a sentence contains a region in which the word “exclude” exists, the first argument (“arg1”) phrase of which includes the word “CRP.” A predicate argument relation is expressed by the variable, “\$subject.” Figure 4 shows a situation in which this formula is satisfied. Three horizontal bars describe regions covered by <sentence>, <phrase>, and <word> tags, respectively. The dotted line denotes the relation expressed by this variable. Given this formula as a query, a search engine can retrieve sentences having semantic annotations that satisfy this formula.

3 A Text Retrieval System for MEDLINE

While the above resources and tools have been developed independently, their collaboration opens up a new framework for the retrieval of relational concepts, as described below (Figure 5).

Off-line processing: Prior to retrieval, a deep parser is applied to compute predicate argument

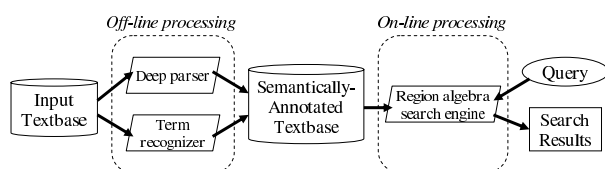


Figure 5: Framework of semantic retrieval

structures, and a term recognizer is applied to create mappings from textual expressions into identifiers in ontology databases. Semantic annotations are stored and indexed in a structured database for the extended region algebra.

On-line processing: User input is converted into queries of the extended region algebra. A search engine retrieves sentences having semantic annotations that match the queries.

This framework is applied to a text retrieval engine for MEDLINE. MEDLINE is an exhaustive database covering nearly 4,500 journals in the life sciences and includes the bibliographies of articles, about half of which have abstracts. Research on IE and text mining in biomedical science has focused mainly on MEDLINE. In the present paper, we target all articles indexed in MEDLINE at the end of 2004 (14,785,094 articles). The following sections explain in detail off-/on-line processing for the text retrieval system for MEDLINE.

3.1 Off-line processing: HPSG parsing and term recognition

We first parsed all sentences using an HPSG parser (Miyao and Tsujii, 2005) to obtain their predicate argument structures. Because our target is biomedical texts, we re-trained a parser (Hara et al., 2005) with the GENIA treebank (Tateisi et al., 2005), and also applied a bidirectional part-of-speech tagger (Tsuruoka and Tsujii, 2005) trained with the GENIA treebank as a preprocessor.

Because parsing speed is still unrealistic for parsing the entire MEDLINE on a single machine, we used two geographically separated computer clusters having 170 nodes (340 Xeon CPUs). These clusters are separately administered and not dedicated for use in the present study. In order to effectively use such an environment, GXP (Taura, 2004) was used to connect these clusters and distribute the load among them. Our processes were given the lowest priority so that our task would not disturb other users. We finished parsing the entire MEDLINE in nine days (Ninomiya et al., 2006).

# entries (genes)	517,773
# entries (gene products)	171,711
# entries (diseases)	148,602
# expanded entries	4,467,855

Table 3: Sizes of ontologies used for term recognition

Event type	Expressions
<i>influence</i>	effect, affect, role, response, ...
<i>regulation</i>	mediate, regulate, regulation, ...
<i>activation</i>	induce, activate, activation, ...

Table 4: Event expression ontology

Next, we annotated technical terms, such as genes and diseases, to create mappings to ontological identifiers. A dictionary-based term recognition algorithm (Tsuruoka and Tsujii, 2004) was applied for this task. First, an expanded term list was created by generating name variations of terms in GENA and the UMLS meta-thesaurus¹. Table 3 shows the size of the original database and the number of entries expanded by name variations. Terms in MEDLINE were then identified by the longest matching of entries in this expanded list with words/phrases in MEDLINE.

The necessity of ontologies is not limited to nominal expressions. Various verbs are used for expressing events. For example, activation events of proteins can be expressed by “activate,” “enhance,” and other event expressions. Although the numbers of verbs and their event types are much smaller than those of technical terms, verbal expressions are important for the description of relational concepts. Since ontologies of event expressions in this domain have not yet been constructed, we developed an ontology from scratch. We investigated 500 abstracts extracted from MEDLINE, and classified 167 frequent expressions, including verbs and their nominalized forms, into 18 event types. Table 4 shows a part of this ontology. These expressions in MEDLINE were automatically annotated with event types.

As a result, we obtained semantically annotated MEDLINE. Table 5 shows the size of the original MEDLINE and semantic annotations. Figure 6 shows semantic annotations for the sentence in Figure 1, where “-” indicates nodes of XML,²

¹We collected disease names by specifying a query with the semantic type as “Disease or Syndrome.”

²Although this example is shown in XML, this textbase contains tags with cross boundaries because tags for predicate argument structures and technical terms may overlap.

# papers	14,785,094
# abstracts	7,291,857
# sentences	70,935,630
# words	1,462,626,934
# successfully parsed sentences	69,243,788
# predicate argument relations	1,510,233,701
# phrase tags	3,094,105,383
# terms (genes)	84,998,621
# terms (gene products)	27,471,488
# terms (diseases)	19,150,984
# terms (event expressions)	51,810,047
Size of the original MEDLINE	9.3 GByte
Size of the semantic annotations	292 GByte
Size of the index file for region algebra	954 GByte

Table 5: Sizes of the original and semantically annotated MEDLINE textbases

```

- <sentence sentence_id="e6e525">
- <phrase id="0" cat="S" head="15" lex_head="18">
- <phrase id="1" cat="NP" head="4" lex_head="14">
- <phrase id="2" cat="DT" head="3" lex_head="3">
- <word id="3" pos="DT" cat="DT" base="a" arg1="4">
- A
- <phrase id="4" cat="NP" head="7" lex_head="14">
- <phrase id="5" cat="AJ" head="6" lex_head="6">
- <word id="6" pos="JJ" cat="AJ" base="normal" arg1="7">
- normal
- <phrase id="7" cat="NP" head="10" lex_head="14">
- <phrase id="8" cat="NP" head="9" lex_head="9">
- <word id="9" pos="NN" cat="NP" base="serum" mod="10">
- serum
- <phrase id="10" cat="NP" head="13" lex_head="14">
- <phrase id="11" cat="NP" head="12" lex_head="12">
- <entity_name id="entity-1" type="gene"
  gene_id="GHS003134" gene_symbol="CRP"
  gene_name="C-reactive protein, pentraxin-related"
  species="Homo sapiens"
  db_site="EntrezGene:1401|GDB:119071|GenAtlas:CRP">
- <word id="12" pos="NN" cat="NP" base="crp" mod="13">
- CRP
- <phrase id="13" cat="NP" head="14" lex_head="14">
- <word id="14" pos="NN" cat="NP" base="measurement">
- measurement
- <phrase id="15" cat="VP" head="16" lex_head="18">
- <phrase id="16" cat="VP" head="17" lex_head="18">
- <phrase id="17" cat="VP" head="18" lex_head="18">
- <word id="18" pos="VBZ" cat="VP" base="do"
  arg1="1" arg2="21">
- does
- <phrase id="19" cat="AV" head="20" lex_head="20">
- <word id="20" pos="RB" cat="AV" base="not" arg1="21">
- not
- <phrase id="21" cat="VP" head="22" lex_head="23">
- <phrase id="22" cat="VP" head="23" lex_head="23">
- <word id="23" pos="VB" cat="VP" base="exclude"
  arg1="1" arg2="24">
- exclude
...

```

Figure 6: A semantically annotated sentence

although the latter half of the sentence is omitted because of space limitations. Sentences are annotated with four tags,³ “phrase,” “word,” “sentence,” and “entity_name,” and their attributes as given in Table 6. Predicate argument structures are annotated as attributes, “mod” and “argX,” which point to the IDs of the argument phrases. For example, in Figure 6, the <word> tag for “exclude” has the attributes `arg1="1"` and `arg2="24"`, which denote the IDs of the subject and object phrases, respectively.

³Additional tags exist for representing document structures such as “title” (details omitted).

Tag	Attributes
phrase	id, cat, head, lex_head
word	id, cat, pos, base, mod, argX, rel_type
sentence	sentence_id
entity_name	id, type, gene_id/disease_id, gene_symbol, gene_name, species, db_site

Attribute	Description
id	unique identifier
cat	syntactic category
head	head daughter’s ID
lex_head	lexical head’s ID
pos	part-of-speech
base	base form of the word
mod	ID of modifying phrase
argX	ID of the X-th argument of the word
rel_type	event type
sentence_id	sentence’s ID
type	whether gene, gene_prod, or disease
gene_id	ID in GENA
disease_id	ID in the UMLS meta-thesaurus
gene_symbol	short form of the gene
gene_name	nomenclature of the gene
species	species that have this gene
db_site	links to external databases

Table 6: Tags (upper) and attributes (lower) for semantic annotations

3.2 On-line processing

The off-line processing described above results in much simpler on-line processing. User input is converted into queries of the extended region algebra, and the converted queries are entered into a search engine for the extended region algebra. The implementation of a search engine is described in detail in Masuda et al. (2006).

Basically, given subject x , object y , and verb v , the system creates the following query:

```

[sentence] >>
  ([word arg1="$subject" arg2="$object"
    base="$v"] &
   ([phrase id="$subject"] > x) &
   ([phrase id="$object"] > y))

```

Ontological identifiers are substituted for x , y , and v , if possible. Nominal keywords, i.e., x and y , are replaced by `[entity_name gene_id="n"]` or `[entity_name disease_id="n"]`, where n is the ontological identifier of x or y . For verbal keywords, `base="$v"` is replaced by `rel_type="r"`, where r is the event type of v .

4 Evaluation

Our system is evaluated with respect to speed and accuracy. Speed is indispensable for real-time interactive text retrieval systems, and accuracy is key for the motivation of semantic retrieval. That is, our motivation for employing semantic retrieval

Query No.	User input
1	<i>something</i> inhibit ERK2
2	<i>something</i> trigger diabetes
3	adiponectin increase <i>something</i>
4	TNF activate IL6
5	dystrophin cause <i>disease</i>
6	macrophage induce <i>something</i>
7	<i>something</i> suppress MAP phosphorylation
8	<i>something</i> enhance p53 (negative)

Table 7: Queries for experiments

```
[sentence] >>
([word rel_type="activation"] &
 [entity_name type="gene" gene_id="GHS019685"] &
 [entity_name type="gene" gene_id="GHS009426"])

[sentence] >>
([word arg1="$subject" arg2="$object"
 rel_type="activation"] &
 ([phrase id="$subject"] >
 [entity_name type="gene" gene_id="GHS019685"]) &
 ([phrase cat="np" id="$object"] >
 [entity_name type="gene" gene_id="GHS009426"])]
```

Figure 7: Queries of the extended region algebra for Query 4-3 (upper: keyword search, lower: semantic search)

was to provide a device for the accurate identification of relational concepts. In particular, high precision is desired in text retrieval from huge texts because users want to extract relevant information, rather than collect exhaustive information.

We have two parameters to vary: whether to use predicate argument structures and whether to use ontological identifiers. The effect of using predicate argument structures is evaluated by comparing “keyword search” with “semantic search.” The former is a traditional style of IR, in which sentences are retrieved by matching words in a query with words in sentences. The latter is a new feature of the present system, in which sentences are retrieved by matching predicate argument relations in a query with those in a semantically annotated textbase. The effect of using ontological identifiers is assessed by changing queries of the extended region algebra. When we use the term ontology, nominal keywords in queries are replaced with ontological identifiers in GENA and the UMLS meta-thesaurus. When we use the event expression ontology, verbal keywords in queries are replaced with event types.

Table 7 is a list of queries used in the following experiments. Words in italics indicate a class of words: “*something*” indicates that any word can appear, and *disease* indicates that any disease expression can appear. These queries were

selected by a biologist, and express typical relational concepts that a biologist may wish to find. Queries 1, 3, and 4 represent relations of genes/proteins, where ERK2, adiponectin, TNF, and IL6 are genes/proteins. Queries 2 and 5 describe relations concerning diseases, and Query 6 is a query that is not relevant to genes or diseases. Query 7 expresses a complex relation concerning a specific phenomena, i.e., phosphorylation, of MAP. Query 8 describes a relation concerning a gene, i.e., p53, while “(negative)” indicates that the target of retrieval is negative mentions. This is expressed by “not” modifying a predicate.

For example, Query 4 attempts to retrieve sentences that mention the protein-protein interaction “*TNF* activates *IL6*.” This is converted into queries of the extended region algebra given in Figure 7. The upper query is for keyword search and only specifies the appearances of the three words. Note that the keywords are translated into the ontological identifiers, “activation,” “GHS019685,” and “GHS009426.” The lower query is for semantic search. The variables in “arg1” and “arg2” indicate that “GHS019685” and “GHS009426” are the subject and object, respectively, of “activation”.

Table 8 summarizes the results of the experiments. The postfixes of query numbers denote whether ontological identifiers are used. *X-1* used no ontologies, and *X-2* used only the term ontology. *X-3* used both the term and event expression ontologies⁴. Comparison of *X-1* and *X-2* clarifies the effect of using the term ontology. Comparison of *X-2* and *X-3* shows the effect of the event expression ontology. The results for *X-3* indicate the maximum performance of the current system. This table shows that the time required for the semantic search for the first answer, shown as “time (first)” in seconds, was reasonably short. Thus, the present framework is acceptable for real-time text retrieval. The numbers of answers increased when we used the ontologies, and this result indicates the efficacy of both ontologies for obtaining relational concepts written in various expressions.

Accuracy was measured by judgment by a biologist. At most 100 sentences were retrieved for each query, and the results of keyword search and semantic search were merged and shuffled. A biologist judged the shuffled sentences (1,839 sentences in total) without knowing whether the sen-

⁴Query 5-1 is not tested because “*disease*” requires the term ontology, and Query 6-2 is not tested because “macrophage” is not assigned an ontological identifier.

Query No.	Keyword search					Semantic search				
	# ans.	time (first/all)	precision	<i>n</i> -precision		# ans.	time (first/all)	precision	relative recall	
1-1	252	0.00/ 1.5	74/100 (74%)	74/100 (74%)		143	0.01/ 2.5	96/100 (96%)	51/74 (69%)	
1-2	348	0.00/ 1.9	61/100 (61%)	61/100 (61%)		174	0.01/ 3.1	89/100 (89%)	42/61 (69%)	
1-3	884	0.00/ 3.2	50/100 (50%)	50/100 (50%)		292	0.01/ 5.3	91/100 (91%)	21/50 (42%)	
2-1	125	0.00/ 1.8	45/100 (45%)	9/ 27 (33%)		27	0.02/ 2.9	23/ 27 (85%)	17/45 (38%)	
2-2	113	0.00/ 2.9	40/100 (40%)	10/ 26 (38%)		26	0.06/ 4.0	22/ 26 (85%)	19/40 (48%)	
2-3	6529	0.00/ 12.1	42/100 (42%)	42/100 (42%)		662	0.01/1527.4	76/100 (76%)	8/42 (19%)	
3-1	287	0.00/ 1.5	20/100 (20%)	4/ 30 (13%)		30	0.05/ 2.4	23/ 30 (80%)	6/20 (30%)	
3-2	309	0.01/ 2.1	21/100 (21%)	4/ 32 (13%)		32	0.10/ 3.5	26/ 32 (81%)	6/21 (29%)	
3-3	338	0.01/ 2.2	24/100 (24%)	8/ 39 (21%)		39	0.05/ 3.6	32/ 39 (82%)	8/24 (33%)	
4-1	4	0.26/ 1.5	0/ 4 (0%)	0/ 0 (—)		0	2.44/ 2.4	0/ 0 (—)	0/ 0 (—)	
4-2	195	0.01/ 2.5	9/100 (9%)	1/ 6 (17%)		6	0.09/ 4.1	5/ 6 (83%)	2/ 9 (22%)	
4-3	2063	0.00/ 7.5	5/100 (5%)	5/ 94 (5%)		94	0.02/ 10.5	89/ 94 (95%)	2/ 5 (40%)	
5-2	287	0.08/ 6.3	73/100 (73%)	73/100 (73%)		116	0.05/ 14.7	97/100 (97%)	37/73 (51%)	
5-3	602	0.01/ 15.9	50/100 (50%)	50/100 (50%)		122	0.05/ 14.2	96/100 (96%)	23/50 (46%)	
6-1	10698	0.00/ 42.8	14/100 (14%)	14/100 (14%)		1559	0.01/3014.5	65/100 (65%)	10/14 (71%)	
6-3	42106	0.00/3379.5	11/100 (11%)	11/100 (11%)		2776	0.01/5100.1	61/100 (61%)	5/11 (45%)	
7	87	0.04/ 2.7	34/ 87 (39%)	7/ 15 (47%)		15	0.05/ 4.2	10/ 15 (67%)	10/34 (29%)	
8	1812	0.01/ 7.6	19/100 (19%)	17/ 84 (20%)		84	0.20/ 29.2	73/ 84 (87%)	7/19 (37%)	

Table 8: Number of retrieved sentences, retrieval time, and accuracy

tence was retrieved by keyword search or semantic search. Without considering which words actually matched the query, a sentence is judged to be correct when any part of the sentence expresses all of the relations described by the query. The modality of sentences was not distinguished, except in the case of Query 8. These evaluation criteria may be disadvantageous for the semantic search because its ability to exactly recognize the participants of relational concepts is not evaluated. Table 8 shows the precision attained by keyword/semantic search and *n*-precision, which denotes the precision of the keyword search, in which the same number, *n*, of outputs is taken as the semantic search. The table also gives the *relative recall* of the semantic search, which represents the ratio of sentences that are correctly output by the semantic search among those correctly output by the keyword search. This does not necessarily represent the true recall because sentences not output by keyword search are excluded. However, this is sufficient for the comparison of keyword search and semantic search.

The results show that the semantic search exhibited impressive improvements in precision. The precision was over 80% for most queries and was nearly 100% for Queries 4 and 5. This indicates that predicate argument structures are effective for representing relational concepts precisely, especially for relations in which two entities are involved. Relative recall was approximately 30–50%, except for Query 2. In the following, we will investigate the reasons for the residual errors.

Table 9 shows the classifications of the errors of

Disregarding of noun phrase structures	45
Term recognition errors	33
Parsing errors	11
Other reasons	8
Incorrect human judgment	7
Nominal expressions	41
Phrasal verb expressions	26
Inference required	24
Coreference resolution required	19
Parsing errors	16
Other reasons	15
Incorrect human judgment	10

Table 9: Error analysis (upper: 104 false positives, lower: 151 false negatives)

semantic retrieval. The major reason for false positives was that our queries ignore internal structures of noun phrases. The system therefore retrieved noun phrases that do not directly mention target entities. For example, “*the increased mortality in patients with diabetes was caused by ...*” does not indicate the trigger of diabetes. Another reason was term recognition errors. For example, the system falsely retrieved sentences containing “*p40*,” which is sometimes, but not necessarily used as a synonym for “*ERK2*.” Machine learning-based term disambiguation will alleviate these errors. False negatives were caused mainly by nominal expressions such as “*the inhibition of ERK2*.” This is because the present system does not convert user input into queries on nominal expressions. Another major reason, phrasal verb expressions such as “*lead to*,” is also a shortage of our current strategy of query creation. Because semantic annotations already in-

clude linguistic structures of these expressions, the present system can be improved further by creating queries on such expressions.

5 Conclusion

We demonstrated a text retrieval system for MEDLINE that exploits pre-computed semantic annotations⁵. Experimental results revealed that the proposed system is sufficiently efficient for real-time text retrieval and that the precision of retrieval was remarkably high. Analysis of residual errors showed that the handling of noun phrase structures and the improvement of term recognition will increase retrieval accuracy. Although the present paper focused on MEDLINE, the NLP tools used in this system are domain/task independent. This framework will thus be applicable to other domains such as patent documents.

The present framework does not conflict with conventional IR/IE techniques, and integration with these techniques is expected to improve the accuracy and usability of the proposed system. For example, query expansion and relevancy feedback can be integrated in a straightforward way in order to improve accuracy. Document ranking is useful for the readability of retrieved results. IE systems can be applied off-line, in the manner of the deep parser in our system, for annotating sentences with target information of IE. Such annotations will enable us to retrieve higher-level concepts, such as relationships among relational concepts.

Acknowledgment

This work was partially supported by Grant-in-Aid for Scientific Research on Priority Areas "Systems Genomics" (MEXT, Japan), Genome Network Project (NIG, Japan), and Solution-Oriented Research for Science and Technology (JST, Japan).

References

- C. Blaschke and A. Valencia. 2002. The frame-based module of the SUISEKI information extraction system. *IEEE Intelligent Systems*, 17(2):14–20.
- S. Boag, D. Chamberlin, M. F. Fernández, D. Florescu, J. Robie, and J. Siméon. 2005. XQuery 1.0: An XML query language.
- E. Charniak and M. Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proc. ACL 2005*.

⁵A web-based demo of our system is available on-line at: <http://www-tsujii.is.s.u-tokyo.ac.jp/medie/>

- H.-W. Chun, Y. Tsuruoka, J.-D. Kim, R. Shiba, N. Nagata, T. Hishiki, and J. Tsujii. 2006. Extraction of gene-disease relations from MedLine using domain dictionaries and machine learning. In *Proc. PSB 2006*, pages 4–15.
- J. Clark and S. DeRose. 1999. XML Path Language (XPath) version 1.0.
- C. L. A. Clarke, G. V. Cormack, and F. J. Burkowski. 1995. An algebra for structured text search and a framework for its implementation. *The Computer Journal*, 38(1):43–56.
- Y. Hao, X. Zhu, M. Huang, and M. Li. 2005. Discovering patterns to extract protein-protein interactions from the literature: Part II. *Bioinformatics*, 21(15):3294–3300.
- T. Hara, Y. Miyao, and J. Tsujii. 2005. Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In *Proc. IJCNLP 2005*.
- IBM, 2005. *Unstructured Information Management Architecture (UIMA) SDK User's Guide and Reference*.
- A. Koike and T. Takagi. 2004. Gene/protein/family name recognition in biomedical literature. In *Proc. Biolink 2004*, pages 9–16.
- D. A. Lindberg, B. L. Humphreys, and A. T. McCray. 1993. The Unified Medical Language System. *Methods in Inf. Med.*, 32(4):281–291.
- K. Masuda, T. Ninomiya, Y. Miyao, T. Ohta, and J. Tsujii. 2006. Nested region algebra extended with variables. In Preparation.
- Y. Miyao and J. Tsujii. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proc. 43rd ACL*, pages 83–90.
- National Library of Medicine. 2005. Fact Sheet MEDLINE. Available at <http://www.nlm.nih.gov/pubs/factsheets/medline.html>.
- T. Ninomiya, Y. Tsuruoka, Y. Miyao, K. Taura, and J. Tsujii. 2006. Fast and scalable HPSG parsing. *Traitement automatique des langues (TAL)*, 46(2).
- Y. Tateisi, A. Yakushiji, T. Ohta, and J. Tsujii. 2005. Syntax annotation for the GENIA corpus. In *Proc. IJCNLP 2005, Companion volume*, pages 222–227.
- K. Taura. 2004. GXP : An interactive shell for the grid environment. In *Proc. IWIA2004*, pages 59–67.
- TEI Consortium, 2004. *Text Encoding Initiative*.
- Y. Tsuruoka and J. Tsujii. 2004. Improving the performance of dictionary-based approaches in protein name recognition. *Journal of Biomedical Informatics*, 37(6):461–470.
- Y. Tsuruoka and J. Tsujii. 2005. Bidirectional inference with the easiest-first strategy for tagging sequence data. In *Proc. HLT/EMNLP 2005*, pages 467–474.

Exploring Distributional Similarity Based Models for Query Spelling Correction

Mu Li

Microsoft Research Asia
5F Sigma Center
Zhichun Road, Haidian District
Beijing, China, 100080
mul@microsoft.com

Yang Zhang

School of
Computer Science and Technology
Tianjin University
Tianjin, China, 300072
yangzhang@tju.edu.cn

Muhua Zhu

School of
Information Science and Engineering
Northeastern University
Shenyang, Liaoning, China, 110004
zhumh@ics.neu.edu.cn

Ming Zhou

Microsoft Research Asia
5F Sigma Center
Zhichun Road, Haidian District
Beijing, China, 100080
mingzhou@microsoft.com

Abstract

A query speller is crucial to search engine in improving web search relevance. This paper describes novel methods for use of distributional similarity estimated from query logs in learning improved query spelling correction models. The key to our methods is the property of distributional similarity between two terms: it is high between a frequently occurring misspelling and its correction, and low between two irrelevant terms only with similar spellings. We present two models that are able to take advantage of this property. Experimental results demonstrate that the distributional similarity based models can significantly outperform their baseline systems in the web query spelling correction task.

1 Introduction

Investigations into query log data reveal that more than 10% of queries sent to search engines contain misspelled terms (Cucerzan and Brill, 2004). Such statistics indicate that a good query speller is crucial to search engine in improving web search relevance, because there is little opportunity that a search engine can retrieve many relevant contents with misspelled terms.

The problem of designing a spelling correction program for web search queries, however, poses special technical challenges and cannot be well solved by general purpose spelling correction methods. Cucerzan and Brill (2004) discussed in detail specialties and difficulties of a query spell checker, and illustrated why the existing methods could not work for query spelling correction. They also identified that no single evidence, either a conventional spelling lexicon or term frequency in the query logs, can serve as criteria for validate queries.

To address these challenges, we concentrate on the problem of learning improved query spelling correction model by integrating distributional similarity information automatically derived from query logs. The key contribution of our work is identifying that we can successfully use the evidence of distributional similarity to achieve better spelling correction accuracy. We present two methods that are able to take advantage of distributional similarity information. The first method extends a string edit-based error model with confusion probabilities within a generative source channel model. The second method explores the effectiveness of our approach within a discriminative maximum entropy model framework by integrating distributional similarity-based features. Experimental results demonstrate that both methods can significantly outperform their baseline systems in the spelling correction task for web search queries.

The rest of the paper is structured as follows: after a brief overview of the related work in Section 2, we discuss the motivations for our approach, and describe two methods that can make use of distributional similarity information in Section 3. Experiments and results are presented in Section 4. The last section contains summaries and outlines promising future work.

2 Related Work

The method for web query spelling correction proposed by Cucerzan and Brill (2004) is essentially based on a source channel model, but it requires iterative running to derive suggestions for very-difficult-to-correct spelling errors. Word bigram model trained from search query logs is used as the source model, and the error model is approximated by inverse weighted edit distance of a correction candidate from its original term. The weights of edit operations are interactively optimized based on statistics from the query logs. They observed that an edit distance-based error model only has less impact on the overall accuracy than the source model. The paper reports that un-weighted edit distance will cause the overall accuracy of their speller’s output to drop by around 2%. The work of Ahmad and Kondrak (2005) tried to employ an unsupervised approach to error model estimation. They designed an EM (Expectation Maximization) algorithm to optimize the probabilities of edit operations over a set of search queries from the query logs, by exploiting the fact that there are more than 10% misspelled queries scattered throughout the query logs. Their method is concerned with single character edit operations, and evaluation was performed on an isolated word spelling correction task.

There are two lines of research in conventional spelling correction, which deal with non-word errors and real-word errors respectively. Non-word error spelling correction is concerned with the task of generating and ranking a list of possible spelling corrections for each query word not found in a lexicon. While traditionally candidate ranking is based on manually tuned scores such as assigning weights to different edit operations or leveraging candidate frequencies, some statistical models have been proposed for this ranking task in recent years. Brill and Moore (2000) presented an improved error model over the one proposed by Kernighan et al. (1990) by allowing generic string-to-string edit operations, which helps with modeling major cognitive errors such

as the confusion between *le* and *al*. Toutanova and Moore (2002) further explored this via explicit modeling of phonetic information of English words. Both these two methods require misspelled/correct word pairs for training, and the latter also needs a pronunciation lexicon. Real-word spelling correction is also referred to as context sensitive spelling correction, which tries to detect incorrect usage of valid words in certain contexts (Golding and Roth, 1996; Mangu and Brill, 1997).

Distributional similarity between words has been investigated and successfully applied in many natural language tasks such as automatic semantic knowledge acquisition (Dekang Lin, 1998) and language model smoothing (Essen and Steinbiss, 1992; Dagan et al., 1997). An investigation on distributional similarity functions can be found in (Lillian Lee, 1999).

3 Distributional Similarity-Based Models for Query Spelling Correction

3.1 Motivation

Most of the previous work on spelling correction concentrates on the problem of designing better error models based on properties of character strings. This direction ever evolves from simple Damerau-Levenshtein distance (Damerau, 1964; Levenshtein, 1966) to probabilistic models that estimate string edit probabilities from corpus (Church and Gale, 1991; Mayes et al, 1991; Ristad and Yianilos, 1997; Brill and Moore, 2000; and Ahmad and Kondrak, 2005). In the mentioned methods, however, the similarities between two strings are modeled on the average of many misspelling-correction pairs, which may cause many idiosyncratic spelling errors to be ignored. Some of those are typical word-level cognitive errors. For instance, given the query term *adventura*, a character string-based error model usually assigns similar similarities to its two most probable corrections *adventure* and *aventura*. Taking into account that *adventure* has a much higher frequency of occurring, it is most likely that *adventure* would be generated as a suggestion. However, our observation into the query logs reveals that *adventura* in most cases is actually a common misspelling of *aventura*. Two annotators were asked to judge 36 randomly sampled queries that contain more than one term, and they agreed upon that 35 of them should be *aventura*.

To solve this problem, we consider alternative methods to make use of the information beyond a

term's character strings. Distributional similarity provides such a dimension to view the possibility that one word can be replaced by another based on the statistics of words co-occurring with them. Distributional similarity has been proposed to perform tasks such as language model smoothing and word clustering, but to the best of our knowledge, it has not been explored in estimating similarities between misspellings and their corrections. In this section, we will only involve the cosine metric for illustration purpose.

Query logs can serve as an excellent corpus for distributional similarity estimation. This is because query logs are not only an up-to-date term base, but also a comprehensive spelling error repository (Cucerzan and Brill, 2004; Ahmad and Kondrak, 2005). Given enough size of query logs, some misspellings, such as *adventura*, will occur so frequently that we can obtain reliable statistics of their typical usage. Essential to our method is the observation of high distributional similarity between frequently occurring spelling errors and their corrections, but low between irrelevant terms. For example, we observe that *adventura* occurred more than 3,300 times in a set of logged queries that spanned three months, and its context was similar to that of *aventura*. Both of them usually appeared after words like *peurto* and *lyrics*, and were followed by *mall*, *palace* and *resort*. Further computation shows that, in the *tf* (term frequency) vector space based on surrounding words, the cosine value between them is approximately 0.8, which indicates these two terms are used in a very similar way among all the users trying to search *aventura*. The cosine between *adventura* and *adventure* is less than 0.03 and basically we can conclude that they are two irrelevant terms, although their spellings are similar.

Distributional similarity is also helpful to address another challenge for query spelling correction: differentiating valid OOV terms from frequently occurring misspellings.

	InLex	Freq	Cosine
<i>vaccum</i>	No	18,430	0.99
<i>vacuum</i>	Yes	158,428	
<i>seraphin</i>	No	1,718	0.30
<i>seraphim</i>	Yes	14,407	

Table 1. Statistics of two word pairs with similar spellings

Table 1 lists detailed statistics of two word pairs, each of pair of words have similar spelling, lexicon and frequency properties. But the distri-

butional similarity between each pair of words provides the necessary information to make correction classification that *vacuum* is a spelling error while *seraphin* is a valid OOV term.

3.2 Problem Formulation

In this work, we view the query spelling correction task as a statistical sequence inference problem. Under the probabilistic model framework, it can be conceptually formulated as follows. Given a correction candidate set C for a query string q :

$$C = \{c \mid \text{EditDist}(q, c) < \delta\}$$

in which each correction candidate c satisfies the constraint that the edit distance between c and q is less than a given threshold δ , the model is to find c^* in C with the highest probability:

$$c^* = \arg \max_{c \in C} P(c \mid q) \quad (1)$$

In practice, the correction candidate set C is not generated from the entire query string directly. Correction candidates are generated for each term of a query first, and then C is constructed by composing the candidates of individual terms. The edit distance threshold δ is set for each term proportionally to the length of the term.

3.3 Source Channel Model

Source channel model has been widely used for spelling correction (Kernighan et al., 1990; Mayes, Damerau et al., 1991; Brill and More, 2000; Ahmad and Kondrak, 2005). Instead of directly optimize (1), source channel model tries to solve an equivalent problem by applying Bayes's rule and dropping the constant denominator:

$$c^* = \arg \max_{c \in C} P(q \mid c)P(c) \quad (2)$$

In this approach, two component generative models are involved: source model $P(c)$ that generates the user's intended query c and error model $P(q \mid c)$ that generates the real query q given c . These two component models can be independently estimated.

In practice, for a multi-term query, the source model can be approximated with an n -gram statistical language model, which is estimated with tokenized query logs. Taking bigram model for example, c is a correction candidate containing n terms, $c = c_1 c_2 \dots c_n$, then $P(c)$ can be written as the product of consecutive bigram probabilities:

$$P(c) = \prod P(c_i \mid c_{i-1})$$

Similarly, the error model probability of a query is decomposed into generation probabilities of individual terms which are assumed to be independently generated:

$$P(q | c) = \prod P(q_i | c_i)$$

Previous proposed methods for error model estimation are all based on the similarity between the character strings of q_i and c_i as described in 3.1. Here we describe a distributional similarity-based method for this problem. Essentially there are different ways to estimate distributional similarity between two words (Dagan et al., 1997), and the one we propose to use is *confusion probability* (Essen and Steinbiss, 1992). Formally, confusion probability P_c estimates the possibility that one word w_1 can be replaced by another word w_2 :

$$P_c(w_2 | w_1) = \sum_w \frac{P(w | w_1)}{P(w)} P(w | w_2) P(w_2) \quad (3)$$

where w belongs to the set of words that co-occur with both w_1 and w_2 .

From the spelling correction point of view, given w_1 to be a valid word and w_2 one of its spelling errors, $P_c(w_2 | w_1)$ actually estimates opportunity that w_1 is misspelled as w_2 in query logs. Compared to other similarity measures such as cosine or Euclidean distance, confusion probability is of interest because it defines a probabilistic distribution rather than a generic measure. This property makes it more theoretically sound to be used as error model probability in the Bayesian framework of the source channel model. Thus it can be applied and evaluated independently. However, before using confusion probability as our error model, we have to solve two problems: probability renormalization and smoothing.

Unlike string edit-based error models, which distribute a major portion of probability over terms with similar spellings, confusion probability distributes probability over the entire vocabulary in the training data. This property may cause the problem of unfair comparison between different correction candidates if we directly use (3) as the error model probability. This is because the synonyms of different candidates may share different portion of confusion probabilities. This problem can be solved by re-normalizing the probabilities only over a term's possible correction candidates and itself. To obtain better estimation, here we also require that the frequency

of a correction candidate should be higher than that of the query term, based on the observation that correct spellings generally occur more often in query logs. Formally, given a word w and its correction candidate set C , the confusion probability of a word w' conditioned on w can be redefined as

$$P_c(w' | w) = \begin{cases} \frac{P'_c(w' | w)}{\sum_{c \in C} P'_c(c | w)} & w' \in C \\ 0 & w' \notin C \end{cases} \quad (4)$$

where $P'_c(w' | w)$ is the original definition of confusion probability.

In addition, we might also have the zero-probability problem when the query term has not appeared or there are few context words for it in the query logs. In such cases there is no distributional similarity information available to any known terms. To solve this problem, we define the final error model probability as the linear combination of confusion probability and a string edit-based error model probability $P_{ed}(q | c)$:

$$P(q | c) = \lambda P_c(q | c) + (1 - \lambda) P_{ed}(q | c) \quad (5)$$

where λ is the interpolation parameter between 0 and 1 that can be experimentally optimized on a development data set.

3.4 Maximum Entropy Model

Theoretically we are more interested in building a unified probabilistic spelling correction model that is able to leverage all available features, which could include (but not limited to) traditional character string-based typographical similarity, phonetic similarity and distributional similarity proposed in this work. The maximum entropy model (Berger et al., 1996) provides us with a well-founded framework for this purpose, which has been extensively used in natural language processing tasks ranging from part-of-speech tagging to machine translation.

For our task, the maximum entropy model defines a posterior probabilistic distribution $P(c | q)$ over a set of feature functions $f_i(q, c)$ defined on an input query q and its correction candidate c :

$$P(c | q) = \frac{\exp \sum_{i=1}^N \lambda_i f_i(c, q)}{\sum_c \exp \sum_{i=1}^N \lambda_i f_i(c, q)} \quad (6)$$

where λ s are feature weights, which can be optimized by maximizing the posterior probability on the training set:

$$\lambda^* = \arg \max_{\lambda} \sum_{(t,q) \in TD} \log P_{\lambda}(t|q)$$

where TD denotes the set of training samples in the form of query-truth pairs presented to the training algorithm.

We use the Generalized Iterative Scaling (GIS) algorithm (Darroch and Ratcliff, 1972) to learn the model parameter λ s of the maximum entropy model. GIS training requires normalization over all possible prediction classes as shown in the denominator in equation (6). Since the potential number of correction candidates may be huge for multi-term queries, it would not be practical to perform the normalization over the entire search space. Instead, we use a method to approximate the sum over the n -best list (a list of most probable correction candidates). This is similar to what Och and Ney (2002) used for their maximum entropy-based statistical machine translation training.

3.4.1 Features

Features used in our maximum entropy model are classified into two categories I) baseline features and II) features supported by distributional similarity evidence. Below we list the feature templates.

Category I:

1. *Language model probability feature*. This is the only real-valued feature with feature value set to the logarithm of source model probability:

$$f_{prob}(q, c) = \log P(c)$$

2. *Edit distance-based features*, which are generated by checking whether the weighted Levenshtein edit distance between a query term and its correction is in certain range;

All the following features, including this one, are binary features, and have the feature function of the following form:

$$f_n(q, c) = \begin{cases} 1 & \text{constraint satisfied} \\ 0 & \text{otherwise} \end{cases}$$

in which the feature value is set to 1 when the constraints described in the template are satisfied; otherwise the feature value is set to 0.

3. *Frequency-based features*, which are generated by checking whether the frequencies of a

query term and its correction candidate are above certain thresholds;

4. *Lexicon-based features*, which are generated by checking whether a query term and its correction candidate are in a conventional spelling lexicon;

5. *Phonetic similarity-based features*, which are generated by checking whether the edit distance between the metaphones (Philips, 1990) of a query term and its correction candidate is below certain thresholds.

Category II:

6. *Distributional similarity based term features*, which are generated by checking whether a query term's frequency is higher than certain thresholds but there are no candidates for it with higher frequency and high enough distributional similarity. This is usually an indicator that the query term is valid and not covered by the spelling lexicon. The frequency thresholds are enumerated from 10,000 to 50,000 with the interval 5,000.

7. *Distributional similarity based correction candidate features*, which are generated by checking whether a correction candidate's frequency is higher than the query term or the correction candidate is in the lexicon, and at the same time the distributional similarity is higher than certain thresholds. This generally gives the evidence that the query term may be a common misspelling of the current candidate. The distributional similarity thresholds are enumerated from 0.6 to 1 with the interval 0.1.

4 Experimental Results

4.1 Dataset

We randomly sampled 7,000 queries from daily query logs of MSN Search and they were manually labeled by two annotators. For each query identified to contain spelling errors, corrections were given by the annotators independently. From the annotation results that both annotators agreed upon 3,061 queries were extracted, which were further divided into a test set containing 1,031 queries and a training set containing 2,030 queries. In the test set there are 171 queries identified containing spelling errors with an error rate of 16.6%. The numbers on the training set is 312 and 15.3%, respectively. The average length of queries on training set is 2.8 terms and on test set it is 2.6.

In our experiments, a term bigram model is used as the source model. The bigram model is trained with query log data of MSN Search during the period from October 2004 to June 2005. Correction candidates are generated from a term base extracted from the same set of query logs.

For each of the experiments, the performance is evaluated by the following metrics:

Accuracy: The number of correct outputs generated by the system divided by the total number of queries in the test set;

Recall: The number of correct suggestions for misspelled queries generated by the system divided by the total number of misspelled queries in the test set;

Precision: The number of correct suggestions for misspelled queries generated by the system divided by the total number of suggestions made by the system.

4.2 Results

We first investigated the impact of the interpolation parameter λ in equation (5) by applying the confusion probability-based error model on training set. For the string edit-based error model probability $P_{ed}(q|c)$, we used a heuristic score computed as the inverse of weighted edit distance, which is similar to the one used by Cucerzan and Brill (2004).

Figure 1 shows the accuracy metric at different settings of λ . The accuracy generally gains improvements before λ reaches 0.9. This shows that confusion probability plays a more important role in the combination. As a result, we empirically set $\lambda = 0.9$ in the following experiments.

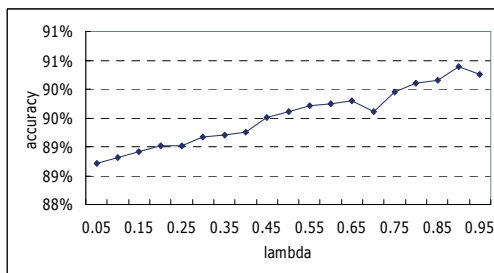


Figure 1. Accuracy with different λ s

To evaluate whether the distributional similarity can contribute to performance improvements, we conducted the following experiments. For source channel model, we compared the confusion probability-based error model (SC-SimCM) against two baseline error model settings, which are source model only (SC-NoCM) and the heuristic

string edit-based error model (SC-EdCM) we just described. Two maximum entropy models were trained with different feature sets. ME-NoSim is the model trained only with baseline features. It serves as the baseline for ME-Full, which is trained with all the features described in 3.4.1. In training ME-Full, cosine distance is used as the similarity measure examined by feature functions.

In all the experiments we used the standard viterbi algorithm to search for the best output of source channel model. The n -best list for maximum entropy model training and testing is generated based on language model scores of correction candidates, which can be easily obtained by running the forward-viterbi backward-A* algorithm. On a 3.0GHZ Pentium4 personal computer, the system can process 110 queries per second for source channel model and 86 queries per second for maximum entropy model, in which 20 best correction candidates are used.

Model	Accuracy	Recall	Precision
SC-NoCM	79.7%	63.3%	40.2%
SC-EdCM	84.1%	62.7%	47.4%
SC-SimCM	88.2%	57.4%	58.8%
ME-NoSim	87.8%	52.0%	60.0%
ME-Full	89.0%	60.4%	62.6%

Table 2. Performance results for different models

Table 2 details the performance scores for the experiments, which shows that both of the two distributional similarity-based models boost accuracy over their baseline settings. SC-SimCM achieves 26.3% reduction in error rate over SC-EdCM, which is significant to the 0.001 level (paired t-test). ME-Full outperforms ME-NoSim in all three evaluation measures, with 9.8% reduction in error rate and 16.2% improvement in recall, which is significant to the 0.01 level.

It is interesting to note that the accuracy of SC-SimCM is slightly better than ME-NoSim, although ME-NoSim makes use of a rich set of features. ME-NoSim tends to keep queries with frequently misspelled terms unchanged (e.g. *caffeine extractions from soda*) to reduce false alarms (e.g. *bicycle* suggested for *biocycle*).

We also investigated the performance of the models discussed above at different recall. Figure 2 and Figure 3 show the precision-recall curves and accuracy-recall curves of different models. We observed that the performance of SC-SimCM and ME-NoSim are very close to each other and ME-Full consistently yields better performance over the entire P-R curve.

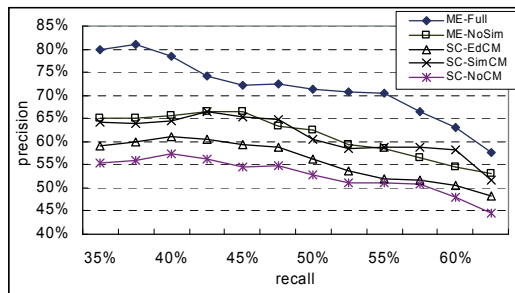


Figure 2. Precision-recall curve of different models

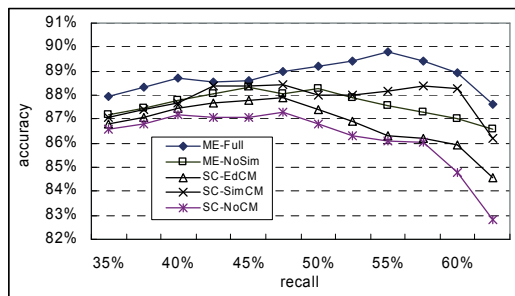


Figure 3. Accuracy-recall curve of different models

We performed a study on the impact of training size to ensure all models are trained with enough data.

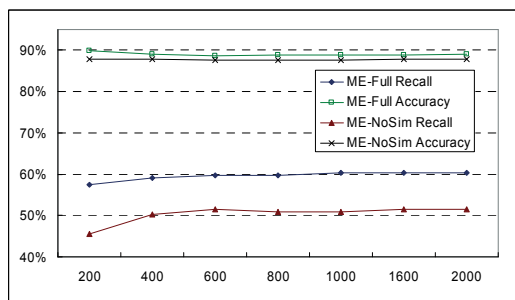


Figure 4. Accuracy of maximum entropy models trained with different number of samples

Figure 4 shows the accuracy of the two maximum entropy models as functions of number of training samples. From the results we can see that after the number of training samples reaches 600 there are only subtle changes in accuracy and recall. Therefore basically it can be concluded that 2,000 samples are sufficient to train a maximum entropy model with the current feature sets.

5 Conclusions and Future Work

We have presented novel methods to learn better statistical models for the query spelling correction task by exploiting distributional similarity information. We explained the motivation of our methods with the statistical evidence distilled from query log data. To evaluate our proposed methods, two probabilistic models that can take

advantage of such information are investigated. Experimental results show that both methods can achieve significant improvements over their baseline settings.

A subject of future research is exploring more effective ways to utilize distributional similarity even beyond query logs. Currently for low-frequency terms in query logs there are no reliable distribution similarity evidence available for them. A promising method of dealing with this in next steps is to explore information in the resulting page of a search engine, since the snippets in the resulting page can provide far greater detailed information about terms in a query.

References

- Farooq Ahmad and Grzegorz Kondrak. 2005. Learning a spelling error model from search query logs. *Proceedings of EMNLP 2005*, pages 955-962.
- Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computation Linguistics*, 22(1):39-72.
- Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. *Proceedings of 38th annual meeting of the ACL*, pages 286-293.
- Kenneth W. Church and William A. Gale. 1991. Probability scoring for spelling correction. In *Statistics and Computing*, volume 1, pages 93-103.
- Silviu Cucerzan and Eric Brill. 2004. Spelling correction as an iterative process that exploits the collective knowledge of web users. *Proceedings of EMNLP'04*, pages 293-300.
- Ido Dagan, Lillian Lee and Fernando Pereira. 1997. Similarity-Based Methods for Word Sense Disambiguation. *Proceedings of the 35th annual meeting of ACL*, pages 56-63.
- Fred Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communication of the ACM* 7(3):659-664.
- J. N. Darroch and D. Ratcliff. 1972. Generalized iterative scaling for long-linear models. *Annals of Mathematical Statistics*, 43:1470-1480.
- Ute Essen and Volker Steinbiss. 1992. Co-occurrence smoothing for stochastic language modeling. *Proceedings of ICASSP*, volume 1, pages 161-164.
- Andrew R. Golding and Dan Roth. 1996. Applying winnow to context-sensitive spelling correction. *Proceedings of ICML 1996*, pages 182-190.
- Mark D. Kernighan, Kenneth W. Church and William A. Gale. 1990. A spelling correction program

- based on a noisy channel model. *Proceedings of COLING 1990*, pages 205-210.
- Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys*. 24(4): 377-439
- Lillian Lee. 1999. Measures of distributional similarity. *Proceedings of the 37th annual meeting of ACL*, pages 25-32.
- V. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physice – Doklady* 10: 707-710.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. *Proceedings of COLING-ACL 1998*, pages 768-774.
- Lidia Mangu and Eric Brill. 1997. Automatic rule acquisition for spelling correction. *Proceedings of ICML 1997*, pages 734-741.
- Eric Mayes, Fred Damerau and Robert Mercer. 1991. Context based spelling correction. *Information processing and management* 27(5): 517-522.
- Franz Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. *Proceedings of the 40th annual meeting of ACL*, pages 295-302.
- Lawrence Philips. 1990. Hanging on the metaphone. *Computer Language Magazine*, 7(12): 39.
- Eric S. Ristad and Peter N. Yianilos. 1997. Learning string edit distance. *Proceedings of ICML 1997*. pages 287-295
- Kristina Toutanova and Robert Moore. 2002. Pronunciation modeling for improved spelling correction. *Proceedings of the 40th annual meeting of ACL*, pages 144-151.

Robust PCFG-Based Generation using Automatically Acquired LFG Approximations

Aoife Cahill¹ and Josef van Genabith^{1,2}

¹ National Centre for Language Technology (NCLT)
School of Computing, Dublin City University, Dublin 9, Ireland

² Center for Advanced Studies, IBM Dublin, Ireland
{acahill,josef}@computing.dcu.ie

Abstract

We present a novel PCFG-based architecture for robust probabilistic generation based on wide-coverage LFG approximations (Cahill et al., 2004) automatically extracted from treebanks, maximising the probability of a tree given an f-structure. We evaluate our approach using string-based evaluation. We currently achieve coverage of 95.26%, a BLEU score of 0.7227 and string accuracy of 0.7476 on the Penn-II WSJ Section 23 sentences of length ≤ 20 .

1 Introduction

Wide coverage grammars automatically extracted from treebanks are a corner-stone technology in state-of-the-art probabilistic parsing. They achieve robustness and coverage at a fraction of the development cost of hand-crafted grammars. It is surprising to note that to date, such grammars do not usually figure in the complementary operation to parsing – natural language surface realisation.

Research on statistical natural language surface realisation has taken three broad forms, differing in where statistical information is applied in the generation process. Langkilde (2000), for example, uses n-gram word statistics to rank alternative output strings from symbolic hand-crafted generators to select paths in parse forest representations. Bangalore and Rambow (2000) use n-gram word sequence statistics in a TAG-based generation model to rank output strings and additional statistical and symbolic resources at intermediate generation stages. Ratnaparkhi (2000) uses maximum entropy models to drive generation with word bigram or dependency representations taking into account (unrealised) semantic features. Valldal and Oepen (2005) present a discriminative disambiguation model using a hand-crafted HPSG

grammar for generation. Belz (2005) describes a method for building statistical generation models using an automatically created *generation* treebank for weather forecasts. None of these probabilistic approaches to NLG uses a full treebank grammar to drive generation.

Bangalore et al. (2001) investigate the effect of training size on performance while using grammars automatically extracted from the Penn-II Treebank (Marcus et al., 1994) for generation. Using an automatically extracted XTAG grammar, they achieve a string accuracy of 0.749 on their test set. Nakanishi et al. (2005) present probabilistic models for a chart generator using a HPSG grammar acquired from the Penn-II Treebank (the Enju HPSG). They investigate discriminative disambiguation models following Valldal and Oepen (2005) and their best model achieves coverage of 90.56% and a BLEU score of 0.7723 on Penn-II WSJ Section 23 sentences of length ≤ 20 .

In this paper we present a novel PCFG-based architecture for probabilistic generation based on wide-coverage, robust Lexical Functional Grammar (LFG) approximations automatically extracted from treebanks (Cahill et al., 2004). In Section 2 we briefly describe LFG (Kaplan and Bresnan, 1982). Section 3 presents our generation architecture. Section 4 presents evaluation results on the Penn-II WSJ Section 23 test set using string-based metrics. Section 5 compares our approach with alternative approaches in the literature. Section 6 concludes and outlines further research.

2 Lexical Functional Grammar

Lexical Functional Grammar (LFG) (Kaplan and Bresnan, 1982) is a constraint-based theory of grammar. It (minimally) posits two levels of representation, c(onstituent)-structure and f(unctional)-structure. C-structure is represented by context-free phrase-structure trees, and captures surface

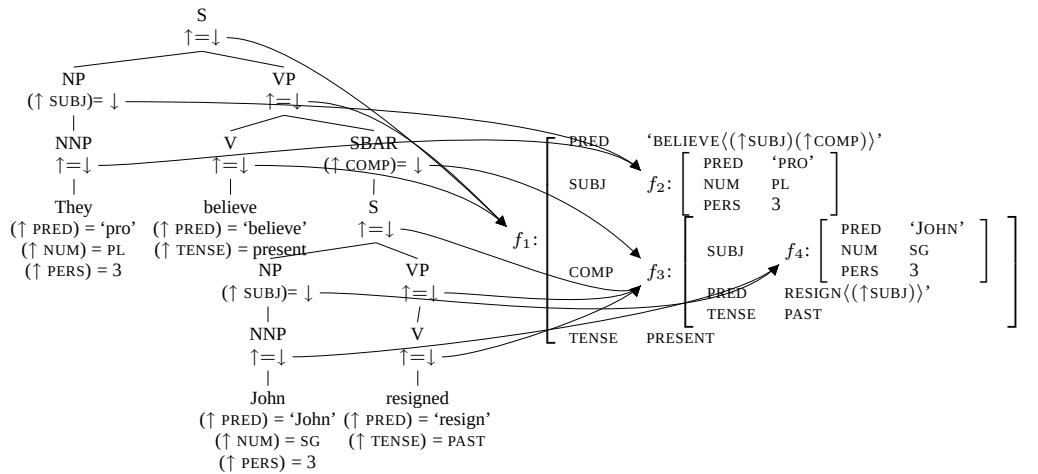


Figure 1: C- and f-structures for the sentence *They believe John resigned.*

grammatical configurations such as word order. The nodes in the trees are annotated with functional equations (attribute-value structure constraints) which are resolved to produce an f-structure. F-structures are recursive attribute-value matrices, representing abstract syntactic functions. F-structures approximate to basic predicate-argument-adjunct structures or dependency relations. Figure 1 shows the c- and f-structures for the sentence “*They believe John resigned*”.

3 PCFG-Based Generation for Treebank-Based LFG Resources

Cahill et al. (2004) present a method to automatically acquire wide-coverage robust probabilistic LFG approximations¹ from treebanks. The method is based on an automatic f-structure annotation algorithm that associates nodes in treebank trees with f-structure equations. For each tree, the equations are collected and passed on to a constraint solver which produces an f-structure for the tree. Cahill et al. (2004) present two parsing architectures: the **pipeline** and the **integrated** parsing architecture. In the **pipeline** architecture, a PCFG (or a history-based lexicalised generative parser) is extracted from the treebank and used to parse unseen text into trees, the resulting trees are annotated with f-structure equations by the f-structure annotation algorithm and a constraint solver produces an f-structure. In the **in-**

tegrated architecture, first the treebank trees are automatically annotated with f-structure information, f-structure annotated PCFGs with rules of the form $NP(\uparrow OBJ = \downarrow) \rightarrow DT(\uparrow = \downarrow) NN(\uparrow = \downarrow)$ are extracted, syntactic categories followed by equations are treated as monadic CFG categories during grammar extraction and parsing, unseen text is parsed into trees with f-structure annotations, the annotations are collected and a constraint solver produces an f-structure.

The generation architecture presented here builds on the integrated parsing architecture resources of Cahill et al. (2004). The generation process takes an f-structure (such as the f-structure on the right in Figure 1) as input and outputs the most likely f-structure annotated tree (such as the tree on the left in Figure 1) given the input f-structure

$$\text{argmax}_{Tree} P(Tree|F\text{-Str})$$

where the probability of a tree given an f-structure is decomposed as the product of the probabilities of all f-structure annotated productions contributing to the tree but where in addition to conditioning on the LHS of the production (as in the integrated parsing architecture of Cahill et al. (2004)) each production $X \rightarrow Y$ is now also conditioned on the set of f-structure features *Feats* ϕ -linked² to the LHS of the rule. For an f-structure annotated tree *Tree* and f-structure *F-Str* with $\Phi(Tree)=F-Str$:³

¹The resources are approximations in that (i) they do not enforce LFG completeness and coherence constraints and (ii) PCFG-based models can only approximate LFG and similar constraint-based formalisms (Abney, 1997).

${}^2\phi$ links LFG's c-structure to f-structure in terms of many-to-one functions from tree nodes into f-structure.

${}^3\Phi$ resolves the equations in *Tree* into *F-Str* (if satisfiable) in terms of the piece-wise function ϕ .

Conditioning F-Structure Features	Grammar Rules	Probability
{PRED, SUBJ, COMP, TENSE}	$VP(\uparrow=\downarrow) \rightarrow VBD(\uparrow=\downarrow) SBAR(\uparrow_{COMP}=\downarrow)$	0.4998
{PRED, SUBJ, COMP, TENSE}	$VP(\uparrow=\downarrow) \rightarrow VBP(\uparrow=\downarrow) SBAR(\uparrow_{COMP}=\downarrow)$	0.0366
{PRED, SUBJ, COMP, TENSE}	$VP(\uparrow=\downarrow) \rightarrow VBD(\uparrow=\downarrow), S(\uparrow_{COMP}=\downarrow)$	6.48e-6
{PRED, SUBJ, COMP, TENSE}	$VP(\uparrow=\downarrow) \rightarrow VBD(\uparrow=\downarrow) S(\uparrow_{COMP}=\downarrow)$	3.88e-6
{PRED, SUBJ, COMP, TENSE}	$VP(\uparrow=\downarrow) \rightarrow VBP(\uparrow=\downarrow), SBARQ(\uparrow_{COMP}=\downarrow)$	7.86e-7
{PRED, SUBJ, COMP, TENSE}	$VP(\uparrow=\downarrow) \rightarrow VBD(\uparrow=\downarrow) SBARQ(\uparrow_{COMP}=\downarrow)$	1.59e-7

Table 1: Example VP Generation rules automatically extracted from Sections 02–21 of the Penn-II Treebank

$$P(Tree|F-Str) := \prod_{\substack{X \rightarrow Y \text{ in Tree} \\ \phi(X) = Feats}} P(X \rightarrow Y|X, Feats) \quad (1)$$

$$P(X \rightarrow Y|X, Feats) = \frac{P(X \rightarrow Y, X, Feats)}{P(X, Feats)} = \quad (2)$$

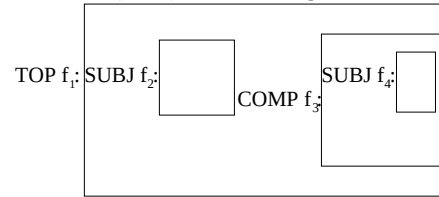
$$\frac{P(X \rightarrow Y, Feats)}{P(X, Feats)} \approx \frac{\#(X \rightarrow Y, Feats)}{\#(X \rightarrow \dots, Feats)} \quad (3)$$

and where probabilities are estimated using a simple MLE and rule counts (#) from the automatically f-structure annotated treebank resource of Cahill et al. (2004). Lexical rules (rules expanding preterminals) are conditioned on the full set of (atomic) feature-value pairs ϕ -linked to the RHS. The intuition for conditioning rules in this way is that local f-structure components of the input f-structure drive the generation process. This conditioning effectively turns the f-structure annotated PCFGs of Cahill et al. (2004) into probabilistic generation grammars. For example, in Figure 1 (where ϕ -links are represented as arrows), we automatically extract the rule $S(\uparrow=\downarrow) \rightarrow NP(\uparrow_{SUBJ}=\downarrow) VP(\uparrow=\downarrow)$ conditioned on the feature set {PRED,SUBJ,COMP,TENSE}. The probability of the rule is then calculated by counting the number of occurrences of that rule (and the associated set of features), divided by the number of occurrences of rules with the same LHS and set of features. Table 1 gives example VP rule expansions with their probabilities when we train a grammar from Sections 02–21 of the Penn Treebank.

3.1 Chart Generation Algorithm

The generation algorithm is based on chart generation as first introduced by Kay (1996) with Viterbi-pruning. The generation grammar is first converted into Chomsky Normal Form (CNF). We recursively build a chart-like data structure in a bottom-up fashion. In contrast to packing of locally equivalent edges (Carroll and Oepen, 2005),

in our approach if two chart items have equivalent rule left-hand sides and lexical coverage, only the most probable one is kept. Each grammatical function-labelled (sub-)f-structure in the overall f-structure indexes a (sub-)chart. The chart for each f-structure generates the most probable tree for that f-structure, given the internal set of conditioning f-structure features and its grammatical function label. At each level, grammatical function indexed charts are initially unordered. Charts are linearised by generation grammar rules once the charts themselves have produced the most probable tree for the chart. Our example in Figure 1 generates the following grammatical function indexed, embedded and (at each level of embedding) unordered (sub-)chart configuration:



For each local subchart, the following algorithm is applied:

```

Add lexical rules
While subchart is Changing
  Apply unary productions
  Apply binary productions
Propagate compatible rules

```

3.2 A Worked Example

As an example, we step through the construction of the COMP-indexed chart at level f_3 of the f-structure in Figure 1. For lexical rules, we check the feature set at the sub-f-structure level and the values of the features. Only features associated with lexical material are considered. The SUBJ-indexed sub-chart f_4 is constructed by first adding the rule $NNP(\uparrow=\downarrow) \rightarrow John(\uparrow_{PRED}='John', \uparrow_{NUM}=pl, \uparrow_{PERS}=3)$. If more than one lexical rule corresponds to a particular set of features and values in the f-structure, we add all rules with different LHS categories. If two or more

rules with equal LHS categories match the feature set, we only add the most probable one.

Unary productions are applied if the RHS of the unary production matches the LHS of an item already in the chart *and* the feature set of the unary production matches the conditioning feature set of the local sub-f-structure. In our example, this results in the rule $\text{NP}(\uparrow\text{SUBJ}=\downarrow) \rightarrow \text{NNP}(\uparrow=\downarrow)$, conditioned on $\{\text{NUM}, \text{PERS}, \text{PRED}\}$, being added to the sub-chart at level f_4 (the probability associated with this item is the probability of the rule multiplied by the probability of the previous chart item which combines with the new rule). When a rule is added to the chart, it is automatically associated with the yield of the rule, allowing us to propagate chunks of generated material upwards in the chart. If two items in the chart have the same LHS (and the same yield independent of word order), only the item with the highest probability is kept. This Viterbi-style pruning ensures that processing is efficient.

At sub-chart f_4 there are no binary rules that can be applied. At this stage, it is not possible to add any more items to the sub-chart, therefore we propagate items in the chart that are compatible with the sub-chart index SUBJ. In our example, only the rule $\text{NP}(\uparrow\text{SUBJ}=\downarrow) \rightarrow \text{NNP}(\uparrow=\downarrow)$ (which yields the string *John*) is propagated to the next level up in the overall chart for consideration in the next iteration. If the yield of an item being propagated upwards in the chart is subsumed by an element already at that level, the subsumed item is removed. This results in efficiently treating the well known problem originally described in Kay (1996), where one unnecessarily retains sub-optimal strings. For example, generating the string “*The very tall strong athletic man*”, one does not want to keep variations such as “*The very tall man*”, or “*The athletic man*”, if one can generate the entire string. Our method ensures that only the most probable tree with the longest yield will be propagated upwards.

The COMP-indexed chart at level f_3 of the f-structure is constructed in a similar fashion. First the lexical rule $\text{V}(\uparrow=\downarrow) \rightarrow \text{resigned}$ is added. Next, conditioning on $\{\text{PRED}, \text{SUBJ}, \text{TENSE}\}$, the unary rule $\text{VP}(\uparrow=\downarrow) \rightarrow \text{V}(\uparrow=\downarrow)$ (with yield *resigned*) is added. We combine the new $\text{VP}(\uparrow=\downarrow)$ rule with the $\text{NP}(\uparrow\text{SUBJ}=\downarrow)$ already present from the previous iteration to enable us to add the rule $\text{S}(\uparrow=\downarrow) \rightarrow \text{NP}(\uparrow\text{SUBJ}=\downarrow) \text{VP}(\uparrow=\downarrow)$, conditioned

on $\{\text{PRED}, \text{SUBJ}, \text{TENSE}\}$. The yield of this rule is *John resigned*. Next, conditioning on the same feature set, we add the rule $\text{SBAR}(\uparrow\text{comp}=\downarrow) \rightarrow \text{S}(\uparrow=\downarrow)$ with yield *John resigned* to the chart. It is not possible to add any more new rules, so at this stage, only the $\text{SBAR}(\uparrow\text{COMP}=\downarrow)$ rule with yield *John resigned* is propagated up to the next level.

The process continues until at the outermost level of the f-structure, there are no more rules to be added to the chart. At this stage, we search for the most probable rule with TOP as its LHS category and return the yield of this rule as the output of the generation process. Generation fails if there is no rule with LHS TOP at this level in the chart.

3.3 Lexical Smoothing

Currently, the only smoothing in the system applies at the lexical level. Our backoff uses the built-in lexical macros⁴ of the automatic f-structure annotation algorithm of Cahill et al. (2004) to identify potential part-of-speech categories corresponding to a particular set of features. Following Baayen and Sproat (1996) we assume that unknown words have a probability distribution similar to hapax legomena. We add a lexical rule for each POS tag that corresponds to the f-structure features at that level to the chart with a probability computed from the original POS tag probability distribution multiplied by a very small constant. This means that lexical rules seen during training have a much higher probability than lexical rules added during the smoothing phase. Lexical smoothing has the advantage of boosting coverage (as shown in Tables 3, 4, 5 and 6 below) but slightly degrades the quality of the strings generated. We believe that the tradeoff in terms of quality is worth the increase in coverage.

Smoothing is *not* carried out when there is no suitable *phrasal* grammar rule that applies during the process of generation. This can lead to the generation of partial strings, since some f-structure components may fail to generate a corresponding string. In such cases, generation outputs the concatenation of the strings generated by the remaining components.

4 Experiments

We train our system on WSJ Sections 02–21 of the Penn-II Treebank and evaluate against the raw

⁴The lexical macros associate POS tags with sets of features, for example the tag NNS (plural noun) is associated with the features $\uparrow\text{PRED}=\$ \text{LEMMA}$ and $\uparrow\text{NUM}=\text{pl}$.

S. length	≤ 20	≤ 25	≤ 30	≤ 40	all
Training	16667	23597	29647	36765	39832
Test	1034	1464	1812	2245	2416

Table 2: Number of training and test sentences per sentence length

strings from Section 23. We use Section 22 as our development set. As part of our evaluation, we experiment with sentences of varying length (20, 25, 30, 40, all), both in training and testing. Table 2 gives the number of training and test sentences for each sentence length. In each case, we use the automatically generated f-structures from Cahill et al. (2004) from the original Section 23 treebank trees as f-structure input to our generation experiments. We automatically mark adjunct and coordination scope in the input f-structure. Notice that these automatically generated f-structures are not “perfect”, i.e. they are not guaranteed to be complete and coherent (Kaplan and Bresnan, 1982): a local f-structure may contain material that is not supposed to be there (incoherence) and/or may be missing material that is supposed to be there (incompleteness). The results presented below show that our method is robust with respect to the quality of the f-structure input and will always attempt to generate partial output rather than fail. We consider this an important property as pristine generation input cannot always be guaranteed in realistic application scenarios, such as probabilistic transfer-based machine translation where generation input may contain a certain amount of noise.

4.1 Pre-Training Treebank Transformations

During the development of the generation system, we carried out error analysis on our development set WSJ Section 22 of the Penn-II Treebank. We identified some initial pre-training transformations to the treebank that help generation.

Punctuation: Punctuation is not usually encoded in f-structure representations. Because our architecture is completely driven by rules conditioned by f-structure information automatically extracted from an f-structure annotated treebank, its placement of punctuation is not principled. This led to anomalies such as full stops appearing mid sentence and quotation marks appearing in undesired locations. One partial solution to this was to reduce the amount of punctuation that the system trained on. We removed all punctuation

apart from commas and full stops from the training data. We did not remove any punctuation from the evaluation test set (Section 23), but our system will ever only produce commas and full stops. In the evaluation (Tables 3, 4, 5 and 6) we are penalised for the missing punctuation. To solve the problem of full stops appearing mid sentence, we carry out a punctuation post-processing step on all generated strings. This removes mid-sentence full stops and adds missing full stops at the end of generated sentences prior to evaluation. We are working on a more appropriate solution allowing the system to generate all punctuation.

Case: English does not have much case marking, and for parsing no special treatment was encoded. However, when generating, it is very important that the first person singular pronoun is *I* in the nominative case and *me* in the accusative. Given the original grammar used in parsing, our generation system was not able to distinguish nominative from accusative contexts. The solution we implemented was to carry out a grammar transformation in a pre-processing step, to automatically annotate personal pronouns with their case information. This resulted in phrasal and lexical rules such as $NP(\uparrow SUBJ) \rightarrow PRP^{nom}(\uparrow = \downarrow)$ and $PRP^{nom}(\uparrow = \downarrow) \rightarrow I$ and greatly improved the accuracy of the pronouns generated.

4.2 String-Based Evaluation

We evaluate the output of our generation system against the raw strings of Section 23 using the Simple String Accuracy and BLEU (Papineni et al., 2002) evaluation metrics. Simple String Accuracy is based on the string edit distance between the output of the generation system and the gold standard sentence. BLEU is the weighted average of n-gram precision against the gold standard sentences. We also measure coverage as the percentage of input f-structures that generate a string. For evaluation, we automatically expand all contracted words. We only evaluate strings produced by the system (similar to Nakanishi et al. (2005)).

We conduct a total of four experiments. The parameters we investigate are lexical smoothing (Section 3.3) and partial output. Partial output is a robustness feature for cases where a sub-f-structure component fails to generate a string and the system outputs a concatenation of the strings generated by the remaining components, rather than fail completely.

Sentence length of Training Data	Evaluation Metric	Section 23 Sentences of length:				
		≤ 20	≤ 25	≤ 30	≤ 40	all
≤ 20	BLEU	0.6812	0.6601	0.6373	0.6013	0.5793
	String Accuracy	0.7274	0.7052	0.6875	0.6572	0.6431
	Coverage	96.52	95.83	94.59	93.76	93.92
≤ 25	BLEU	0.6915	0.6800	0.6696	0.6396	0.6233
	String Accuracy	0.7262	0.7095	0.6983	0.6731	0.6618
	Coverage	96.52	95.83	94.59	93.76	93.92
≤ 30	BLEU	0.6979	0.6881	0.6792	0.6576	0.6445
	String Accuracy	0.7317	0.7169	0.7075	0.6853	0.6749
	Coverage	97.97	97.95	97.41	97.15	97.31
≤ 40	BLEU	0.7045	0.6951	0.6852	0.6715	0.6605
	String Accuracy	0.7349	0.7212	0.7074	0.6881	0.6788
	Coverage	98.45	98.36	98.01	97.82	97.93
all	BLEU	0.7077	0.6974	0.6859	0.6734	0.6651
	String Accuracy	0.7373	0.7221	0.7087	0.6894	0.6808
	Coverage	98.65	98.5	98.12	97.95	98.05

Table 3: Generation +partial output +lexical smoothing

Sentence length of Training Data	Evaluation Metric	Section 23 Sentences of length:				
		≤ 20	≤ 25	≤ 30	≤ 40	all
all	BLEU	0.6253	0.6097	0.5887	0.5730	0.5590
	String Accuracy	0.6886	0.6688	0.6513	0.6317	0.6207
	Coverage	91.20	91.19	90.84	90.33	90.11

Table 4: Generation +partial output -lexical smoothing

Varying the length of the sentences included in the training data (Tables 3 and 5) shows that results improve (both in terms of coverage and string quality) as the length of sentence included in the training data increases.

Tables 3 and 5 give the results for the experiments including lexical smoothing and varying partial output. Table 3 (+partial, +smoothing) shows that training on sentences of all lengths and evaluating all strings (including partial outputs), our system achieves coverage of 98.05%, a BLEU score of 0.6651 and string accuracy of 0.6808. Table 5 (-partial, +smoothing) shows that coverage drops to 89.49%, BLEU score increases to 0.6979 and string accuracy to 0.7012, when the system is trained on sentences of all lengths. Similarly, for strings ≤ 20 , coverage drops from 98.65% to 95.26%, BLEU increases from 0.7077 to 0.7227 and String Accuracy from 0.7373 to 0.7476. Including partial output increases coverage (by more than 8.5 percentage points for all sentences) and hence robustness while slightly decreasing quality.

Tables 3 (+partial, +smoothing) and 4 (+partial, -smoothing) give results for the experiments including partial output but varying lexical smoothing. With no lexical smoothing (Table 4), the system (trained on all sentence lengths) produces

strings for 90.11% of the input f-structures and achieves a BLEU score of 0.5590 and string accuracy of 0.6207. Switching off lexical smoothing has a negative effect on all evaluation metrics (coverage and quality), because many more strings produced are now partial (since for PRED values unseen during training, no lexical entries are added to the chart).

Comparing Tables 5 (-partial, +smoothing) and 6 (-partial, -smoothing), where the system does not produce any partial outputs and lexical smoothing is varied, shows that training on all sentence lengths, BLEU score increases from 0.6979 to 0.7147 and string accuracy increases from 0.7012 to 0.7192. At the same time, coverage drops dramatically from 89.49% (Table 5) to 47.60% (Table 6).

Comparing Tables 4 and 6 shows that while partial output almost doubles coverage, this comes at a price of a severe drop in quality (BLEU score drops from 0.7147 to 0.5590). On the other hand, comparing Tables 5 and 6 shows that lexical smoothing achieves a similar increase in coverage with only a very slight drop in quality.

5 Discussion

Nakanishi et al. (2005) achieve 90.56% coverage and a BLEU score of 0.7723 on Section 23

Sentence length of Training Data	Evaluation Metric	Section 23 Sentences of length:				
		≤ 20	≤ 25	≤ 30	≤ 40	all
≤ 20	BLEU	0.7326	0.7185	0.7165	0.7082	0.7052
	String Accuracy	0.76	0.7428	0.7363	0.722	0.7175
	Coverage	85.49	81.56	77.26	71.94	69.08
≤ 25	BLEU	0.7300	0.7235	0.7218	0.7118	0.7077
	String Accuracy	0.7517	0.7382	0.7315	0.7172	0.7116
	Coverage	89.65	87.77	84.38	80.31	78.56
≤ 30	BLEU	0.7207	0.7125	0.7107	0.6991	0.6946
	String Accuracy	0.747	0.7336	0.7275	0.711	0.7045
	Coverage	93.23	92.14	89.74	86.59	85.18
≤ 40	BLEU	0.7221	0.7140	0.7106	0.7016	0.6976
	String Accuracy	0.746	0.7331	0.7236	0.7072	0.7001
	Coverage	94.58	93.85	91.89	89.62	88.33
all	BLEU	0.7227	0.7145	0.7095	0.7011	0.6979
	String Accuracy	0.7476	0.7331	0.7239	0.7077	0.7012
	Coverage	95.26	94.40	92.55	90.69	89.49

Table 5: Generation -partial output +lexical smoothing

Sentence length of Training Data	Evaluation Metric	Section 23 Sentences of length:				
		≤ 20	≤ 25	≤ 30	≤ 40	all
all	BLEU	0.7272	0.7237	0.7201	0.7160	0.7147
	String Accuracy	0.7547	0.7436	0.7361	0.7237	0.7192
	Coverage	61.99	57.38	53.64	47.60	47.60

Table 6: Generation -partial output -lexical smoothing

sentences, restricted to length ≤ 20 for efficiency reasons. Langkilde-Geary’s (2002) best system achieves 82.8% coverage, a BLEU score of 0.924 and string accuracy of 0.945 against Section 23 sentences of all lengths. Callaway (2003) achieves 98.7% coverage and a string accuracy of 0.6607 on sentences of all lengths. Our best results for sentences of length ≤ 20 are coverage of 95.26%, BLEU score of 0.7227 and string accuracy of 0.7476. For all sentence lengths, our best results are coverage of 89.49%, a BLEU score of 0.6979 and string accuracy of 0.7012.

Using hand-crafted grammar-based generation systems (Langkilde-Geary, 2002; Callaway, 2003), it is possible to achieve very high results. However, hand-crafted systems are expensive to construct and not easily ported to new domains or other languages. Our methodology, on the other hand, is based on resources automatically acquired from treebanks and easily ported to new domains and languages, simply by retraining on suitable data. Recent work on the automatic acquisition of multilingual LFG resources from treebanks for Chinese, German and Spanish (Burke et al., 2004; Cahill et al., 2005; O’Donovan et al., 2005) has shown that given a suitable treebank, it is possible to automatically acquire high quality LFG re-

sources in a very short space of time. The generation architecture presented here is easily ported to those different languages and treebanks.

6 Conclusion and Further Work

We present a new architecture for stochastic LFG surface realisation using the automatically annotated treebanks and extracted PCFG-based LFG approximations of Cahill et al. (2004). Our model maximises the probability of a tree given an f-structure, supporting a simple and efficient implementation that scales to wide-coverage treebank-based resources. An improved model would maximise the probability of a string given an f-structure by summing over trees with the same yield. More research is required to implement such a model efficiently using packed representations (Carroll and Oepen, 2005). Simple PCFG-based models, while effective and computationally efficient, can only provide approximations to LFG and similar constraint-based formalisms (Abney, 1997). Research on discriminative disambiguation methods (Valldal and Oepen, 2005; Nakanishi et al., 2005) is important. Kaplan and Wedekind (2000) show that for certain linguistically interesting classes of LFG (and PATR etc.) grammars, generation from f-structures yields a context free language. Their proof involves the notion of a

“refinement” grammar where f-structure information is compiled into CFG rules. Our probabilistic generation grammars bear a conceptual similarity to Kaplan and Wedekind’s “refinement” grammars. It would be interesting to explore possible connections between the treebank-based empirical work presented here and the theoretical constructs in Kaplan and Wedekind’s proofs.

We presented a full set of generation experiments on varying sentence lengths training on Sections 02–21 of the Penn Treebank and evaluating on Section 23 strings. Sentences of length ≤ 20 achieve coverage of 95.26%, BLEU score of 0.7227 and string accuracy of 0.7476 against the raw Section 23 text. Sentences of all lengths achieve coverage of 89.49%, BLEU score of 0.6979 and string accuracy of 0.7012. Our method is robust and can cope with noise in the f-structure input to generation and will attempt to produce partial output rather than fail.

Acknowledgements

We gratefully acknowledge support from Science Foundation Ireland grant 04/BR/CS0370 for the research reported in this paper.

References

- Stephen Abney. 1997. Stochastic Attribute-Value Grammars. *Computational Linguistics*, **23**(4):597–618.
- Harald Baayen and Richard Sproat. 1996. Estimating lexical priors for low-frequency morphologically ambiguous forms. *Computational Linguistics*, **22**(2):155–166.
- Srinivas Bangalore and Owen Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. In *Proceedings of COLING 2000*, pages 42–48, Saarbrücken, Germany.
- Srinivas Bangalore, John Chen, and Owen Rambow. 2001. Impact of quality and quantity of corpora on stochastic generation. In *Proceedings of EMNLP 2001*, pages 159–166.
- Anja Belz. 2005. Statistical generation: Three methods compared and evaluated. In *Proceedings of the 10th European Workshop on Natural Language Generation (ENLG’ 05)*, pages 15–23, Aberdeen, Scotland.
- Michael Burke, Olivia Lam, Rowena Chan, Aoife Cahill, Ruth O’Donovan, Adams Bodomo, Josef van Genabith, and Andy Way. 2004. Treebank-Based Acquisition of a Chinese Lexical-Functional Grammar. In *Proceedings of the 18th Pacific Asia Conference on Language, Information and Computation*, pages 161–172, Tokyo, Japan.
- 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 *Proceedings of ACL-04*, pages 320–327, Barcelona, Spain.
- Aoife Cahill, Martin Forst, Michael Burke, Mairead McCarthy, Ruth O’Donovan, Christian Rohrer, Josef van Genabith, and Andy Way. 2005. Treebank-based acquisition of multilingual unification grammar resources. *Journal of Research on Language and Computation; Special Issue on “Shared Representations in Multilingual Grammar Engineering”*, pages 247–279.
- Charles B. Callaway. 2003. Evaluating coverage for large symbolic NLG grammars. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 811–817, Acapulco, Mexico.
- John Carroll and Stephan Oepen. 2005. High efficiency realization for a wide-coverage unification grammar. In *Proceedings of IJCNLP05*, pages 165–176, Jeju Island, Korea.
- Ron Kaplan and Joan Bresnan. 1982. Lexical Functional Grammar, a Formal System for Grammatical Representation. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations*, pages 173–281. MIT Press, Cambridge, MA.
- Ron Kaplan and Juergen Wedekind. 2000. LFG Generation produces Context-free languages. In *Proceedings of COLING 2000*, pages 141–148, Saarbrücken, Germany.
- Martin Kay. 1996. Chart Generation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 200–204, Santa Cruz, CA.
- Irene Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Second International Natural Language Generation Conference*, pages 17–24, Harriman, NY.
- Irene Langkilde. 2000. Forest-based statistical sentence generation. In *Proceedings of NAACL 2000*, pages 170–177, Seattle, WA.
- Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 110–115, Princeton, NJ.
- Hiroko Nakanishi, Yusuke Miyao, and Jun’ichi Tsujii. 2005. Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of the International Workshop on Parsing Technology*, Vancouver, Canada.
- Ruth O’Donovan, Aoife Cahill, Josef van Genabith, and Andy Way. 2005. Automatic Acquisition of Spanish LFG Resources from the CAST3LB Treebank. In *Proceedings of LFG 05*, pages 334–352, Bergen, Norway.
- Kishore Papineni, Salim Roukos, Todd Ward, and WeiJing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL 2002*, pages 311–318, Philadelphia, PA.
- Adwait Ratnaparkhi. 2000. Trainable methods for natural language generation. In *Proceedings of NAACL 2000*, pages 194–201, Seattle, WA.
- Erik Valldal and Stephan Oepen. 2005. Maximum Entropy Models for Realization Reranking. In *Proceedings of the 10th Machine Translation Summit*, pages 109–116, Phuket, Thailand.

Incremental generation of spatial referring expressions in situated dialog*

John D. Kelleher

Dublin Institute of Technology
Dublin, Ireland

john.kelleher@comp.dit.ie

Geert-Jan M. Kruijff

DFKI GmbH
Saarbrücken, Germany
gj@dfki.de

Abstract

This paper presents an approach to incrementally generating locative expressions. It addresses the issue of combinatorial explosion inherent in the construction of relational context models by: (a) contextually defining the set of objects in the context that may function as a landmark, and (b) sequencing the order in which spatial relations are considered using a cognitively motivated hierarchy of relations, and visual and discourse salience.

1 Introduction

Our long-term goal is to develop conversational robots with whom we can interact through natural, fluent, visually situated dialog. An inherent aspect of visually situated dialog is reference to objects located in the physical environment (Moratz and Tenbrink, 2006). In this paper, we present a computational approach to the generation of spatial locative expressions in such situated contexts.

The simplest form of locative expression is a prepositional phrase, modifying a noun phrase to locate an object. (1) illustrates the type of locative we focus on generating. In this paper we use the term **target** (T) to refer to the object that is being located by a spatial expression and the term **landmark** (L) to refer to the object relative to which the target's location is described.

- (1) a. the book [T] on the table [L]

Generating locative expressions is part of the general field of generating referring expressions (GRE). Most GRE algorithms deal with the same problem: given a domain description and a **target object**, generate a description of the target object that distinguishes it from the other objects in the domain. We use **distractor objects** to indicate the

objects in the context excluding the target that at a given point in processing fulfill the description of the target object that has been generated. The description generated is said to be **distinguishing** if the set of distractor objects is empty.

Several GRE algorithms have addressed the issue of generating locative expressions (Dale and Haddock, 1991; Horacek, 1997; Gardent, 2002; Krahmer and Theune, 2002; Varges, 2004). However, all these algorithms assume the GRE component has access to a predefined scene model. For a conversational robot operating in dynamic environments this assumption is unrealistic. If a robot wishes to generate a contextually appropriate reference it cannot assume the availability of a fixed scene model, rather it must dynamically construct one. However, constructing a model containing all the relationships between all the entities in the domain is prone to combinatorial explosion, both in terms of the number objects in the context (the location of each object in the scene must be checked against all the other objects in the scene) and number of inter-object spatial relations (as a greater number of spatial relations will require a greater number of comparisons between each pair of objects).¹ Also, the context free *a priori* construction of such an exhaustive scene model is cognitively implausible. Psychological research indicates that spatial relations are not preattentively perceptually available (Treisman and Gormican, 1988), their perception requires attention (Logan, 1994; Logan, 1995). Subjects appear to construct contextually dependent reduced relational scene models, not exhaustive context free models.

Contributions We present an approach to in-

The research reported here was supported by the CoSy project, EU FP6 IST "Cognitive Systems" FP6-004250-IP.

¹In English, the vast majority of spatial locatives are binary, some notable exceptions include: *between*, *amongst* etc. However, we will not deal with these exceptions in this paper.

crementally generating locative expressions. It addresses the issue of combinatorial explosion inherent in relational scene model construction by incrementally creating a series of reduced scene models. Within each scene model only one spatial relation is considered and only a subset of objects are considered as candidate landmarks. This reduces both the number of relations that must be computed over each object pair and the number of object pairs. The decision as to which relations should be included in each scene model is guided by a cognitively motivated hierarchy of spatial relations. The set of candidate landmarks in a given scene is dependent on the set of objects in the scene that fulfil the description of the target object and the relation that is being considered.

Overview §2 presents some relevant background data. §3 presents our GRE approach. §4 illustrates the framework on a worked example and expands on some of the issues relevant to the framework. We end with conclusions.

2 Data

If we consider that English has more than eighty spatial prepositions (omitting compounds such as *right next to*) (Landau, 1996), the combinatorial aspect of relational scene model construction becomes apparent. It should be noted that for our purposes, the situation is somewhat easier because a distinction can be made between static and dynamic prepositions: static prepositions primarily² denote the location of an object, dynamic prepositions primarily denote the path of an object (Jackendoff, 1983; Herskovits, 1986), see (2). However, even focusing just on the set of static prepositions does not remove the combinatorial issues effecting the construction of a scene model.

- (2) a. the tree is behind [static] the house
- b. the man walked across [dyn.] the road

In general, static prepositions can be divided into two sets: **topological** and **projective**. Topological prepositions are the category of prepositions referring to a region that is proximal to the landmark; e.g., *at*, *near*, etc. Often, the distinctions between the semantics of the different topological prepositions is based on pragmatic constraints, e.g. the use of *at* licences the target to be

²Static prepositions can be used in dynamic contexts, e.g. *the man ran behind the house*, and dynamic prepositions can be used in static ones, e.g. *the tree lay across the road*.

in contact with the landmark, whereas the use of *near* does not. Projective prepositions describe a region projected from the landmark in a particular direction; e.g., *to the right of*, *to the left of*. The specification of the direction is dependent on the frame of reference being used (Herskovits, 1986).

Static prepositions have both qualitative and quantitative semantic properties. The qualitative aspect is evident when they are used to denote an object by contrasting its location with that of the distractor objects. Using Figure 1 as visual context, the locative expression *the circle on the left of the square* illustrates the contrastive semantics of a projective preposition, as only one of the circles in the scene is located in that region. Taking Figure 2, the locative expression *the circle near the black square* shows the contrastive semantics of a topological preposition. Again, of the two circles in the scene only one of them may be appropriately described as being *near the black square*, the other circle is more appropriately described as being *near the white square*. The quantitative aspect is evident when a static preposition denotes an object using a relative scale. In Figure 3 the locative *the circle to the right of the square* shows the relative semantics of a projective preposition. Although both the circles are located *to the right of the square* we can distinguish them based on their location in the region. Figure 3 also illustrates the relative semantics of a topological preposition Figure 3. We can apply a description like *the circle near the square* to either circle if none other were present. However, if both are present we can interpret the reference based on relative proximity to the landmark *the square*.



Figure 1: Visual context illustrating contrastive semantics of projective prepositions



Figure 2: Visual context illustrating contrastive semantics of topological prepositions



Figure 3: Visual context illustrating relative semantics of topological and projective prepositions

3 Approach

We base our GRE approach on an extension of the incremental algorithm (Dale and Reiter, 1995). The motivation for basing our approach on this algorithm is its polynomial complexity. The algorithm iterates through the properties of the target and for each property computes the set of distractor objects for which (a) the conjunction of the properties selected so far, and (b) the current property hold. A property is added to the list of selected properties if it reduces the size of the distractor object set. The algorithm succeeds when all the distractors have been ruled out, it fails if all the properties have been processed and there are still some distractor objects. The algorithm can be refined by ordering the checking of properties according to fixed preferences, e.g. first a taxonomic description of the target, second an absolute property such as colour, third a relative property such as size. (Dale and Reiter, 1995) also stipulate that the type description of the target should be included in the description even if its inclusion does not make the target distinguishable.

We extend the original incremental algorithm in two ways. First we integrate a model of object salience by modifying the condition under which a description is deemed to be distinguishing: it is, if all the distractors have been ruled out or if the salience of the target object is greater than the highest salience score ascribed to any of the current distractors. This is motivated by the observation that people can easily resolve underdetermined references using salience (Duwe and Strohner, 1997). We model the influence of visual and discourse salience using a function $salience(L)$, Equation 1. The function returns a value between 0 and 1 to represent the relative salience of a landmark L in the scene. The relative salience of an object is the average of its visual salience (S_{vis}) and discourse salience (S_{disc}),

$$salience(L) = (S_{vis}(L) + S_{disc}(L))/2 \quad (1)$$

Visual salience S_{vis} is computed using the algorithm of (Kelleher and van Genabith, 2004). Computing a relative salience for each object in a scene is based on its perceivable size and its centrality relative to the viewer focus of attention, returning scores in the range of 0 to 1. The discourse salience (S_{disc}) of an object is computed based on recency of mention (Hajicová, 1993) except we represent the maximum overall salience in the

scene as 1, and use 0 to indicate that the landmark is not salient in the current context. Algorithm 1 gives the basic algorithm with salience.

Algorithm 1 The Basic Incremental Algorithm

Require: T = target object; D = set of distractor objects.

Initialise: $P = \{type, colour, size\}$; $DESC = \{\}$

for $i = 0$ to $|P|$ **do**

if $T_{salience()} > \text{MAXDISTRACTORSALIENCE}$ **then**

Distinguishing description generated

if $type(x) \notin DESC$ **then**

$DESC = DESC \cup type(x)$

end if

 return $DESC$

else

$D' = \{x : x \in D, P_i(x) = P_i(T)\}$

if $|D'| < |D|$ **then**

$DESC = DESC \cup P_i(T)$

$D = \{x : x \in D, P_i(x) = P_i(T)\}$

end if

end if

end for

Failed to generate distinguishing description

 return $DESC$

Secondly, we extend the incremental algorithm in how we construct the context model used by the algorithm. The context model determines to a large degree the output of the incremental algorithm. However, Dale and Reiter do not define how this set should be constructed, they only write: “[w]e define the context set to be the set of entities that the hearer is currently assumed to be attending to” (Dale and Reiter, 1995, pg. 236).

Before applying the incremental algorithm we must construct a context model in which we can check whether or not the description generated distinguishes the target object. To constrain the combinatorial explosion in relational scene model construction we construct a series of reduced scene models, rather than one complex exhaustive model. This construction is driven by a hierarchy of spatial relations and the partitioning of the context model into objects that may and may not function as landmarks. These two components are developed below. §3.1 discusses a hierarchy of spatial relations, and §3.2 presents a classification of landmarks and uses these groupings to create a definition of a distinguishing locative description. In §3.3 we give the generation algorithm integrating these components.

3.1 Cognitive Ordering of Contexts

Psychological research indicates that spatial relations are not preattentively perceptually available (Treisman and Gormican, 1988). Rather, their perception requires attention (Logan, 1994;

Logan, 1995). These findings point to subjects constructing contextually dependent reduced relational scene models, rather than an exhaustive context free model. Mimicking this, we have developed an approach to context model construction that constrains the combinatorial explosion inherent in the construction of relational context models by incrementally building a series of reduced context models. Each context model focuses on a different spatial relation. The ordering of the spatial relations is based on the cognitive load of interpreting the relation. Below we motivate and develop the ordering of relations used.

We can reasonably assume that it takes less effort to describe one object than two. Following the Principle of Minimal Cooperative Effort (Clark and Wilkes-Gibbs, 1986), one should only use a locative expression when there is no distinguishing description of the target object using a simple feature based approach. Also, the Principle of Sensitivity (Dale and Reiter, 1995) states that when producing a referring expression, one should prefer features the hearer is known to be able to interpret and see. This points to a preference, due to cognitive load, for descriptions that identify an object using purely physical and easily perceivable features ahead of descriptions that use spatial expressions. Experimental results support this (van der Sluis and Krahmer, 2004).

Similarly, we can distinguish between the cognitive loads of processing different forms of spatial relations. In comparing the cognitive load associated with different spatial relations it is important to recognize that they are represented and processed at several levels of abstraction. For example, the **geometric level**, where metric properties are dealt with, the **functional level**, where the specific properties of spatial entities deriving from their functions in space are considered, and the **pragmatic level**, which gathers the underlying principles that people use in order to discard wrong relations or to deduce more information (Edwards and Moulin, 1998). Our discussion is grounded at the geometric level.

Focusing on static prepositions, we assume topological prepositions have a lower perceptual load than projective ones, as perceiving two objects being close to each other is easier than the processing required to handle frame of reference ambiguity (Carlson-Radvansky and Irwin, 1994; Carlson-Radvansky and Logan,

1997). Figure 4 lists the preferences, further discerning objects type as the easiest to process, before absolute gradable predicates (e.g. color), which is still easier than relative gradable predicates (e.g. size) (Dale and Reiter, 1995).

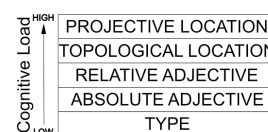


Figure 4: Cognitive load

We can refine the topological versus projective preference further if we consider their contrastive and relative uses of these relations (§2). Perceiving and interpreting a contrastive use of a spatial relation is computationally easier than judging a relative use. Finally, within projective prepositions, psycholinguistic data indicates a perceptually based ordering of the relations: *above/below* are easier to perceive and interpret than *in front of/behind* which in turn are easier than *to the right of/to the left of* (Bryant et al., 1992; Gapp, 1995). In sum, we propose the following ordering: *topological contrastive* < *topological relative* < *projective contrastive* < *projective relative*.

For each level of this hierarchy we require a computational model of the semantics of the relation at that level that accommodates both contrastive and relative representations. In §2 we noted that the distinctions between the semantics of the different topological prepositions is often based on functional and pragmatic issues.³ Currently, however, more psycholinguistic data is required to distinguish the cognitive load associated with the different topological prepositions. We use the model of topological proximity developed in (Kelleher et al., 2006) to model all the relations at this level. Using this model we can define the extent of a region proximal to an object. If the target or one of the distractor objects is the only object within the region of proximity around a given landmark this is taken to model a contrastive use of a topological relation relative to that landmark. If the landmark's region of proximity contains more than one object from the target and distractor object set then it is a relative use of a topological relation. We handle the issue of frame of reference ambiguity and model the semantics of projective prepositions using the framework developed in (Kelleher et al., 2006). Here again, the contrastive-relative distinc-

³See *inter alia* (Talmy, 1983; Herskovits, 1986; Vandeloise, 1991; Fillmore, 1997; Garrod et al., 1999) for more discussion on these differences

tion is dependent on the number of objects within the region of space defined by the preposition.

3.2 Landmarks and Descriptions

If we want to use a locative expression, we must choose another object in the scene to function as landmark. An implicit assumption in selecting a landmark is that the hearer can easily identify and locate the object within the context. A landmark can be: the speaker (3)a, the hearer (3)b, the scene (3)c, an object in the scene (3)d, or a group of objects in the scene (3)e.⁴

- (3) a. the ball on *my* right [speaker]
- b. the ball to *your* left [hearer]
- c. the ball on the right [scene]
- d. the ball to the left of *the box* [an object in the scene]
- e. the ball in the middle [group of objects]

Currently, we need new empirical research to see if there is a preference order between these landmark categories. Intuitively, in most situations, either of the interlocutors are ideal landmarks because the speaker can naturally assume that the hearer is aware of the speaker's location and their own. Focusing on instances where an object in the scene is used as a landmark, several authors (Talmy, 1983; Landau, 1996; Gapp, 1995) have noted a target-landmark asymmetry: generally, the landmark object is more permanently located, larger, and taken to have greater geometric complexity. These characteristics are indicative of salient objects and empirical results support this correlation between object salience and landmark selection (Beun and Cremers, 1998). However, the salience of an object is intrinsically linked to the context it is embedded in. For example, in Figure 5 the ball has a relatively high salience, because it is a singleton, despite the fact that it is smaller and geometrically less complex than the other figures. Moreover, in this scene it is the only object that can function as a landmark without recourse to using the scene itself or a grouping of objects.

Clearly, deciding which objects in a given context are suitable to function as landmarks is a complex and contextually dependent process. Some of the factors effecting this decision are object



Figure 5: Landmark salience

salience and the functional relationships between objects. However, one basic constraint on landmark selection is that the landmark should be distinguishable from the target. For example, given the context in Figure 5 and all other factors being equal, using a locative such as *the man to the left of the man* would be much less helpful than using *the man to the right of the ball*. Following this observation, we treat an object as a **candidate landmark** if the following conditions are met: (1) the object is not the target, and (2) it is not in the distractor set either.

Furthermore, a **target landmark** is a member of the candidate landmark set that stands in relation to the target. A **distractor landmark** is a member of the candidate landmark set that stands in the considered relation to a distractor object. We then define a **distinguishing locative description** as a locative description where there is target landmark that can be distinguished from all the members of the set of distractor landmarks under the relation used in the locative.

3.3 Algorithm

We first try to generate a distinguishing description using Algorithm 1. If this fails, we divide the context into three components: the target, the distractor objects, and the set of candidate landmarks. We then iterate through the set of candidate landmarks (using a salience ordering if there is more than one, cf. Equation 1) and try to create a distinguishing locative description. The salience ordering of the landmarks is inspired by (Conklin and McDonald, 1982) who found that the higher the salience of an object the more likely it appears in the description of the scene it was embedded in. For each candidate landmark we iterate through the hierarchy of relations, checking for each relation whether the candidate can function as a target landmark under that relation. If so we create a context model that defines the set of target and distractor landmarks. We create a distinguishing locative description by using the basic incremental algorithm to distinguish the target landmark from the distractor landmarks. If we succeed in generating a distinguishing locative description we return

⁴See (Gorniak and Roy, 2004) for further discussion on the use of spatial extrema of the scene and groups of objects in the scene as landmarks

the description and stop.

Algorithm 2 The Locative Incremental Algorithm

```

DESC = Basic-Incremental-Algorithm(T,D)
if  $DESC \neq \text{Distinguishing}$  then
  create  $CL$  the set of candidate landmarks
   $CL = \{x : x \neq T, DESC(x) = \text{false}\}$ 
  for  $i = 0$  to  $|CL|$  by  $\text{salience}(CL)$  do
    for  $j = 0$  to  $|R|$  do
      if  $R_j(T, CL_i) = \text{true}$  then
         $TL = \{CL_i\}$ 
         $DL = \{z : z \in CL, R_j(D, z) = \text{true}\}$ 
         $\text{LANDDESC} = \text{Basic-Incremental-Algorithm}(TL, DL)$ 
        if  $\text{LANDDESC} = \text{Distinguishing}$  then
          Distinguishing locative generated
          return  $\{DESC, R_j, \text{LANDDESC}\}$ 
        end if
      end if
    end for
  end for
end if
return FAIL

```

If we cannot create a distinguishing locative description we face two choices: (1) iterate on to the next relation in the hierarchy, (2) create an embedded locative description distinguishing the landmark. We adopt (1) over (2), preferring *the dog to the right of the car* over *the dog near the car to the right of the house*. However, we can generate these longer embedded descriptions if needed, by replacing the call to the basic incremental algorithm for the landmark object with a call to the whole locative expression generation algorithm, using the target landmark as the target object and the set of distractor landmarks as the distractors.

An important point in this context is the issue of infinite regression (Dale and Haddock, 1991). A compositional GRE system may in certain contexts generate an infinite description, trying to distinguish the landmark in terms of the target, and the target in terms of the landmark, cf. (4). But, this infinite recursion can only occur if the context is not modified between calls to the algorithm. This issue does not affect Algorithm 2 as each call to the algorithm results in the domain being partitioned into those objects we can and cannot use as landmarks. This not only reduces the number of object pairs that relations must be computed for, but also means that we need to create a distinguishing description for a landmark on a context that is a strict subset of the context the target description was generated in. This way the algorithm *cannot* distinguish a landmark using its target.

- (4) the bowl on the table supporting the bowl
on the table supporting the bowl ...

3.4 Complexity

The computational complexity of the incremental algorithm is $O(n_d * n_l)$, with n_d the number of distractors, and n_l the number of attributes in the final referring description (Dale and Reiter, 1995). This complexity is independent of the number of attributes to be considered. Algorithm 2 is bound by the same complexity. For the average case, however, we see the following. For one, with every increase in n_l , we see a strict decrease in n_d : the more attributes we need, the fewer distractors we strictly have due to the partitioning into distractor and target landmarks. On the other hand, we have the dynamic construction of a context model. This latter factor is not considered in (Dale and Reiter, 1995), meaning we would have to multiply $O(n_d * n_l)$ with a constant K_{ctx} for context construction. Depending on the size of this constant, we may see an advantage of our algorithm in that we only consider a single spatial relation each time we construct a context model, we avoid an exponential number of comparisons: we need to make at most $n_d * (n_d - 1)$ comparisons (and only n_d if relations are symmetric).

4 Discussion

We exemplify the approach on the visual scene on the left of Figure 6. This context consists of two red boxes R1 and R2 and two blue balls B1 and B2. Imagine that we want to refer to B1. We begin by calling Algorithm 2. This in turn calls Algorithm 1, returning the property *ball*. This is not sufficient to create a distinguishing description as B2 is also a ball. In this context the set of candidate landmarks equals $\{R1, R2\}$. We take R1 as first candidate landmark, and check for topological proximity in the scene as modeled in (Keller et al., 2006). The image on the right of Figure 6 illustrates the resulting scene analysis: the green region on the left defines the area deemed to be proximal to R1, and the yellow region on the right defines the area proximal to R2. Clearly, B1 is in the area proximal to R1, making R1 a target landmark. As none of the distractors (i.e., B2) are located in a region that is proximal to a candidate landmark there are no distractor landmarks. As a result when the basic incremental algorithm is called to create a distinguishing description for the target landmark R1 it will return *box* and this will be deemed to be a distinguishing locative description. The overall algorithm will then return

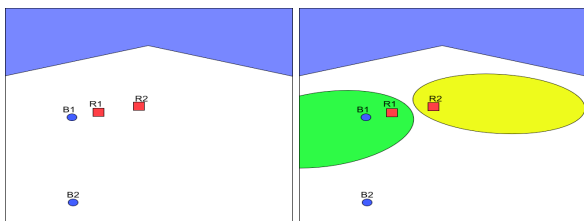


Figure 6: A visual scene and the topological analysis of R1 and R2

the vector $\{ball, proximal, box\}$ which would result in the realiser generating a reference of the form: *the ball near the box*.⁵

The relational hierarchy used by the framework has some commonalities with the relational subsumption hierarchy proposed in (Krahmer and Theune, 2002). However, there are two important differences between them. First, an implication of the subsumption hierarchy proposed in (Krahmer and Theune, 2002) is that the semantics of the relations at lower levels in the hierarchy are subsumed by the semantics of their parent relations. For example, in the portion of the subsumption hierarchy illustrated in (Krahmer and Theune, 2002) the relation *next to* subsumes the relations *left of* and *right of*. By contrast, the relational hierarchy developed here is based solely on the relative cognitive load associated with the semantics of the spatial relations and makes no claims as to the semantic relationships between the semantics of the spatial relations. Secondly, (Krahmer and Theune, 2002) do not use their relational hierarchy to guide the construction of domain models.

By providing a basic contextual definition of a landmark we are able to partition the context in an appropriate manner. This partitioning has two advantages. One, it reduces the complexity of the context model construction, as the relationships between the target and the distractor objects or between the distractor objects themselves do not need to be computed. Two, the context used during the generation of a landmark description is always a subset of the context used for a target (as the target, its distractors and the other objects in the domain that do not stand in relation to the target or distractors under the relation being considered are excluded). As a result the framework avoids the issue of infinite recursion. Furthermore, the target-landmark relationship is automat-

ically included as a property of the landmark as its feature based description need only distinguish it from objects that stand in relation to one of the distractor objects under the same spatial relationship.

In future work we will focus on extending the framework to handle some of the issues affecting the incremental algorithm, see (van Deemter, 2001). For example, generating locative descriptions containing negated relations, conjunctions of relations and involving sets of objects (sets of targets and landmarks).

5 Conclusions

We have argued that if a conversational robot functioning in dynamic partially known environments needs to generate contextually appropriate locative expressions it must be able to construct a context model that explicitly marks the spatial relations between objects in the scene. However, the construction of such a model is prone to the issue of combinatorial explosion both in terms of the number of objects in the context (the location of each object in the scene must be checked against all the other objects in the scene) and number of inter-object spatial relations (as a greater number of spatial relations will require a greater number of comparisons between each pair of objects).

We have presented a framework that addresses this issue by: (a) contextually defining the set of objects in the context that may function as a landmark, and (b) sequencing the order in which spatial relations are considered using a cognitively motivated hierarchy of relations. Defining the set of objects in the scene that may function as a landmark reduces the number of object pairs that a spatial relation must be computed over. Sequencing the consideration of spatial relations means that in each context model only one relation needs to be checked and in some instances the agent need not compute some of the spatial relations, as it may have succeeded in generating a distinguishing locative using a relation earlier in the sequence.

A further advantage of our approach stems from the partitioning of the context into those objects that may function as a landmark and those that may not. As a result of this partitioning the algorithm avoids the issue of infinite recursion, as the partitioning of the context stops the algorithm from distinguishing a landmark using its target.

We have employed the approach in a system for Human-Robot Interaction, in the setting of object

⁵For more examples, see the videos available at <http://www.dfki.de/cosy/media/>.

manipulation in natural scenes. For more detail, see (Kruijff et al., 2006a; Kruijff et al., 2006b).

References

- R.J. Beun and A. Cremers. 1998. Object reference in a shared domain of conversation. *Pragmatics and Cognition*, 6(1/2):121–152.
- D.J. Bryant, B. Tversky, and N. Franklin. 1992. Internal and external spatial frameworks representing described scenes. *Journal of Memory and Language*, 31:74–98.
- L.A. Carlson-Radvansky and D. Irwin. 1994. Reference frame activation during spatial term assignment. *Journal of Memory and Language*, 33:646–671.
- L.A. Carlson-Radvansky and G.D. Logan. 1997. The influence of reference frame selection on spatial template construction. *Journal of Memory and Language*, 37:411–437.
- H. Clark and D. Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22:1–39.
- E. Jeffrey Conklin and David D. McDonald. 1982. Saliency: the key to the selection problem in natural language generation. In *ACL Proceedings, 20th Annual Meeting*, pages 129–135.
- R. Dale and N. Haddock. 1991. Generating referring expressions involving relations. In *Proceeding of the Fifth Conference of the European ACL*, pages 161–166, Berlin, April.
- R. Dale and E. Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.
- I. Duwe and H. Strohner. 1997. Towards a cognitive model of linguistic reference. Report: 97/1 - Situierete Künstliche Kommunikatoren 97/1, Universität Bielefeld.
- G. Edwards and B. Moulin. 1998. Towards the simulation of spatial mental images using the voronoi model. In P. Oliver and K.P. Gapp, editors, *Representation and processing of spatial expressions*, pages 163–184. Lawrence Erlbaum Associates.
- C. Fillmore. 1997. *Lecture on Deixis*. CSLI Publications.
- K.P. Gapp. 1995. Angle, distance, shape, and their relationship to projective relations. In *Proceedings of the 17th Conference of the Cognitive Science Society*.
- C. Gardent. 2002. Generating minimal definite descriptions. In *Proceedings of the 40th International Conference of the Association of Computational Linguistics (ACL-02)*, pages 96–103.
- S. Garrod, G. Ferrier, and S. Campbell. 1999. In and on: investigating the functional geometry of spatial prepositions. *Cognition*, 72:167–189.
- P. Gorniak and D. Roy. 2004. Grounded semantic composition for visual scenes. *Journal of Artificial Intelligence Research*, 21:429–470.
- E. Hajicová. 1993. Issues of sentence structure and discourse patterns. In *Theoretical and Computational Linguistics*, volume 2, Charles University, Prague.
- A. Herskovits. 1986. *Language and spatial cognition: An interdisciplinary study of prepositions in English*. Studies in Natural Language Processing. Cambridge University Press.
- H. Horacek. 1997. An algorithm for generating referential descriptions with flexible interfaces. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, Madrid.
- R. Jackendoff. 1983. *Semantics and Cognition*. Current Studies in Linguistics. The MIT Press.
- J. Kelleher and J. van Genabith. 2004. A false colouring real time visual saliency algorithm for reference resolution in simulated 3d environments. *AI Review*, 21(3-4):253–267.
- J.D. Kelleher, G.J.M. Kruijff, and F. Costello. 2006. Proximity in context: An empirically grounded computational model of proximity for processing topological spatial expressions. In *Proceedings ACL/COLING 2006*.
- E. Krahmer and M. Theune. 2002. Efficient context-sensitive generation of referring expressions. In K. van Deemter and R. Kibble, editors, *Information Sharing: Reference and Presupposition in Language Generation and Interpretation*. CSLI Publications, Stanford.
- G.J.M. Kruijff, J.D. Kelleher, G. Berginc, and A. Leonardis. 2006a. Structural descriptions in human-assisted robot visual learning. In *Proceedings of the 1st Annual Conference on Human-Robot Interaction (HRI'06)*.
- G.J.M. Kruijff, J.D. Kelleher, and Nick Hawes. 2006b. Information fusion for visual reference resolution in dynamic situated dialogue. In E. André, L. Dybkjaer, W. Minker, H. Neumann, and M. Weber, editors, *Perception and Interactive Technologies (PIT 2006)*. Springer Verlag.
- B. Landau. 1996. Multiple geometric representations of objects in language and language learners. In P. Bloom, M. Peterson, L. Nadel, and M. Garrett, editors, *Language and Space*, pages 317–363. MIT Press, Cambridge.
- G. D. Logan. 1994. Spatial attention and the apprehension of spatial realtions. *Journal of Experimental Psychology: Human Perception and Performance*, 20:1015–1036.
- G.D. Logan. 1995. Linguistic and conceptual control of visual spatial attention. *Cognitive Psychology*, 12:523–533.
- R. Moratz and T. Tenbrink. 2006. Spatial reference in linguistic human-robot interaction: Iterative, empirically supported development of a model of projective relations. *Spatial Cognition and Computation*.
- L. Talmy. 1983. How language structures space. In H.L. Pick, editor, *Spatial orientation. Theory, research and application*, pages 225–282. Plenum Press.
- A. Treisman and S. Gormican. 1988. Feature analysis in early vision: Evidence from search asymmetries. *Psychological Review*, 95:15–48.
- K. van Deemter. 2001. Generating referring expressions: Beyond the incremental algorithm. In *4th Int. Conf. on Computational Semantics (IWCS-4)*, Tilburg.
- I van der Sluis and E Krahmer. 2004. The influence of target size and distance on the production of speech and gesture in multimodal referring expressions. In *Proceedings of International Conference on Spoken Language Processing (ICSLP04)*.
- C. Vandeloise. 1991. *Spatial Prepositions: A Case Study From French*. The University of Chicago Press.
- S. Varges. 2004. Overgenerating referring expressions involving relations and booleans. In *Proceedings of the 3rd International Conference on Natural Language Generation*, University of Brighton.

Learning to Predict Case Markers in Japanese

Hisami Suzuki Kristina Toutanova¹

Microsoft Research

One Microsoft Way, Redmond WA 98052 USA

{hisamis,kristout}@microsoft.com

Abstract

Japanese case markers, which indicate the grammatical relation of the complement NP to the predicate, often pose challenges to the generation of Japanese text, be it done by a foreign language learner, or by a machine translation (MT) system. In this paper, we describe the task of predicting Japanese case markers and propose machine learning methods for solving it in two settings: (i) *monolingual*, when given information only from the Japanese sentence; and (ii) *bilingual*, when also given information from a corresponding English source sentence in an MT context. We formulate the task after the well-studied task of English semantic role labelling, and explore features from a syntactic dependency structure of the sentence. For the monolingual task, we evaluated our models on the Kyoto Corpus and achieved over 84% accuracy in assigning correct case markers for each phrase. For the bilingual task, we achieved an accuracy of 92% per phrase using a bilingual dataset from a technical domain. We show that in both settings, features that exploit dependency information, whether derived from gold-standard annotations or automatically assigned, contribute significantly to the prediction of case markers.

1 Introduction: why predict case?

Generation of grammatical elements such as inflectional endings and case markers has become an important component technology, particularly in the context of machine translation (MT). In an English-to-Japanese MT system, for example, Japanese case markers, which indicate grammatical relations (e.g., subject, object, location) of the complement noun phrase to the predicate, are among the most difficult to generate appropriately. This is because the case markers often do not correspond to any word in the source language as many grammatical relations are expressed via word order in English. It is also difficult because the mapping between the case markers and the grammatical

relations they express is very complex. For the same reasons, generation of case markers is challenging to foreign language learners. This difficulty in generation, however, does not mean the choice of case markers is insignificant: when a generated sentence contains mistakes in grammatical elements, they often lead to severe unintelligibility, sometimes resulting in a different semantic interpretation from the intended one. Therefore, having a model that makes reasonable predictions about which case marker to generate given the content words of a sentence, is expected to help MT and generation in general, particularly when the source (or native) and the target languages are morphologically divergent.

But how reliably can we predict case markers in Japanese using the information that exists only in the sentence? Consider the example in Figure 1. This sentence contains two case markers, *kara* 'from' and *ni*, the latter not corresponding to any word in English. If we were to predict the case markers in this sentence, there are multiple valid answers for each decision, many of which correspond to different semantic relations. For example, for the first case marker slot in Figure 1 filled by *kara*, *wa* (topic marker), *ni* 'in' or no case marker at all are all reasonable choices, while other markers such as *wo* (object marker), *de* 'at', *made* 'until', etc. are not considered reasonable. For the second slot filled by *ni*, *ga* (subject marker) is also a grammatically reasonable choice, making *Einstein* the subject of *idolize*, thus changing the meaning of the sentence. As is obvious in this example, the choice among the correct answers is determined by the speaker's intent in uttering the sentence, and is therefore impossible to recover from the content words or the sentence structure alone. At the same time, many impossible or unlikely case marking decisions can be eliminated by a case prediction model. Combined with an external component (for example an MT component) that can resolve semantic and intentional ambiguity, a case prediction model can be quite useful in sentence generation.

This paper discusses the task of case marker assignment in two distinct but related settings. After defining the task in Section 2 and describing our models in Section 3, we first discuss the *monolingual* task in Sections 4, whose goal is to predict the case markers

¹ Author names arranged alphabetically

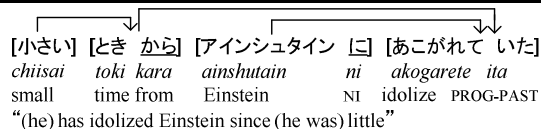


Figure 1. Example of case markers in Japanese (taken from the Kyoto Corpus). Square brackets indicate bunsetsu (phrase) boundaries, to be discussed below. Arrows between phrases indicate dependency relations.

using Japanese sentences and their dependency structure alone. We formulated this task after the well-studied task of semantic role labeling in English (e.g., Gildea and Jurafsky, 2002; Carreras and Màrques, 2005), whose goal is to assign one of 20 semantic role labels to each phrase in a sentence with respect to a given predicate, based on the annotations provided by PropBank (Palmer et al., 2005). Though the task of case marker prediction is more ambiguous and subject to uncertainty than the semantic role labeling task, we obtained some encouraging results which we present in Section 4. Next, in Section 5, we describe the *bilingual* task, in which information about case assignment can be extracted from a corresponding source language sentence. Though the process of MT introduces uncertainties in generating the features we use, we show that the benefit of using dependency structure in our models is far greater than not using it even when the assigned structure is not perfect.

2 The task of case prediction

In this section, we define the task of case prediction. We start with the description of the case markers we used in this study.

2.1 Nominal particles in Japanese

Traditionally, Japanese nominal postpositions are classified into the following three categories (e.g., Tera-mura, 1991; Masuoka and Takubo, 1992):

Case particles (or case markers). They indicate grammatical relations of the complement NP to the predicate. As they are jointly determined by the NP and the predicate, case markers often do not allow a simple mapping to a word in another language, which makes their generation more difficult. The relationship between the case marker and the grammatical relation it indicates is not straightforward either: a case marker can (and often does) indicate multiple grammatical relations as in *Ainshutain-ni akogareru* "idolize Einstein" where *ni* marks the Object relation, and in *Tokyo-ni sumu* "live in Tokyo" where *ni* indicates Location. Conversely, the same grammatical relation may be indicated by different case markers: both *ni* and *de* in *Tokyo-ni sumu* "live in Tokyo" and *Tokyo-de au* "meet in Tokyo" indicate the Location relation. We

included 10 case markers as the primary target of prediction, as shown in the first 10 lines of Table 1.

Conjunctive particles. These particles are used to conjoin words and phrases, corresponding to English "and" and "or". As their occurrence is not predictable from the sentence structure alone, we did not include them in the current prediction task.

Focus particles. These particles add focus to a phrase against a given background or contextual knowledge, for example *shika* and *mo* in *pasuta-shika tabenakatta* "ate only pasta" and *pasuta-mo tabeta* "also ate pasta", corresponding to *only* and *also* respectively. Note that they often replace case markers: in the above examples, the object marker *wo* is no longer present when *shika* or *mo* is used. As they add information to the predicate-argument structure and are in principle not predictable given the sentence structure alone, we did not consider them as the target of our task. One exception is the topic marker *wa*, which we included as a target of prediction for the following reasons:

- Some linguists recognize *wa* as a topic marker, separately from other focus particles (e.g. Masuoka and Takubo, 1992). The main function of *wa* is to introduce a topic in the sentence, which is to some extent predictable from the structure of the sentence.
- *wa* is extremely frequent in Japanese text. For example, it accounts for 13.2% of all postpositions in Kyoto University Text Corpus (henceforth Kyoto Corpus, Kurohashi and Nagao, 1997), making it the third most frequent postposition after *no* (20.57%) and *wo* (13.5%). Generating *wa* appropriately thus greatly enhances the readability of the text.
- Unlike other focus particles such as *shika* and *mo*, *wa* does not translate into any word in English, which makes it difficult to generate by using the information from the source language.

Therefore, in addition to the 10 true case markers, we also included *wa* as a case marker in our study.² Furthermore, we also included the combination of case particles plus *wa* as a secondary target of prediction. The case markers that can appear followed by *wa* are indicated by a check mark in the column "+wa" in Table 1. Thus there are seven secondary targets: *niwa*, *karawa*, *towa*, *dewa*, *ewa*, *madewa*, *yorowa*. Therefore, we have in total 18 case particles to assign to phrases.

2.2 Task definition

The case prediction task we are solving is as follows. We are given a sentence as a list of *bunsetsu* together

² This set comprises the majority (92.5%) of the nominal particles, while conjunctive and focus particles account for only 7.5% of the nominal particles in Kyoto Corpus.

case markers		grammatical functions (e.g.)	+wa
が	<i>ga</i>	subject; object	
を	<i>wo</i>	object; path	
の ⁴	<i>no</i>	genitive; subject	
に	<i>ni</i>	dative object, location	✓
から	<i>kara</i>	source	✓
と	<i>to</i>	quotative, reciprocal, as	✓
で	<i>de</i>	location, instrument, cause	✓
へ	<i>e</i>	goal, direction	✓
まで	<i>made</i>	goal (up to, until)	✓
より	<i>yor</i>	source, object of comparison	✓
は	<i>wa</i>	topic	

Table 1. Case markers included in this study

with a dependency structure. For our monolingual experiments, we used the dependency structure annotation in the Kyoto Corpus; for our bilingual experiments, we used automatically derived dependency structure (Quirk et al., 2005). Each *bunsetsu* (or simply *phrase* in this paper) is defined as consisting of one content word (or n-content words in the case of compounds with n-components) plus any number of function words (including particles, auxiliaries and affixes). Case markers are classified as function words, and there is at most one case marker per phrase.³ In testing, the case marker for each phrase is hidden; the task is to assign to each phrase one of the 18 case markers defined above or NONE; NONE indicates that the phrase does not have a case marker.

2.3 Related work

Though the task of case marker prediction as formulated in this paper is novel, similar tasks have been defined in the past. The semantic role labeling task mentioned in Section 1 is one example; the task of function tag assignment in English (e.g., Blaheta and Charniak, 2000) is another. These tasks are similar to the case prediction task in that they try to assign semantic or function tags to a parsed structure. However, there is one major difference between these tasks and the current task: semantic role labels and function tags can for the most part be uniquely determined given the sentence and its parse structure; decisions about case markers, on the other hand, are highly ambiguous given the sentence structure alone, as mentioned in Section 1. This makes our task more ambiguous than the related tasks. As a concrete comparison, the two most frequent semantic role labels (ARG0 and ARG1) account for 60% of the labeled arguments in PropBank

(Carreras and Màrquez, 2005), whereas our 2 most frequent case markers (*no* and *wo*) account for only 43% of the case-marked phrases. We should also note that semantic role labels and function tags have been artificially defined in accordance with theoretical decisions about what annotations should be useful for natural language understanding tasks; in contrast, the case markers are part of the surface sentence string and do not reflect any theoretical decisions.

The task of case prediction in Japanese has previously focused on recovering *implicit* case relations, which result when noun phrases are relativized or topicalized (e.g., Baldwin, 2000; Kawahara et al., 2004; Murata and Isahara, 2005). Their goal is different from ours, as we aim to generate surface forms of case markers rather than recover deeper case relations for which surface case marker are often used as a proxy.

In the context of sentence generation, Gamon et al. (2002) used a decision tree to classify nouns into one of the four cases in German, as part of their sentence realization from a semantic representation, achieving high accuracy (87% to 93.5%). Again, this is a substantially easier task than ours, because there are only four classes and one of them (nominative) accounts for 70% of all cases. Uchimoto et al. (2002), which is the work most related to ours, propose a model of generating function words (not limited to case markers) from "keywords" or headwords of phrases in Japanese. The components of their model are based on n-gram language models using the surface word strings and *bunsetsu* dependency information, and the results they report are not comparable to ours, as they limit their test sentences to the ones consisting only of two or three content words. We will see in the next section that our models are also quite different from theirs as we employ a much richer set of features.

3 Classifiers for case prediction

We implemented two types of models for the task of case prediction: *local models*, which choose the case marker of each phrase independently of the case markers of other phrases, and *joint models*, which incorporate dependencies among the case markers of dependents of the same head phrase. We describe the two types of models in turn.

3.1 Local classifiers

Following the standard practice in semantic role labeling, we divided the case prediction task into the tasks of *identification* and *classification* (Gildea and Jurafsky, 2002; Pradhan et al., 2004). In the identification task, we assign to each phrase one of two labels: HAS-CASE, meaning that the phrase has a case marker, or NONE, meaning that it does not have a case. In the

³ One exception is that *no* can appear after certain case markers; in such cases, we considered *no* to be the case for the phrase.

⁴ *no* is typically not considered as a case marker but rather as a conjunctive particle indicating adnominal relation; however, as *no* can also be used to indicate the subject in a relative clause, we included it in our study.

Basic features for phrases (self, parent)
HeadPOS, PrevHeadPOS, NextHeadPOS
PrevPOS, Prev2POS, NextPOS, Next2POS
HeadNounSubPos: time, formal nouns, adverbial
HeadLemma
HeadWord, PrevHeadWord, NextHeadWord
PrevWord, Prev2Word, NextWord, Next2Word
LastWordLemma (excluding case markers)
LastWordInfl (excluding case markers)
IsFiniteClause
IsDateExpression
IsNumberExpression
HasPredicateNominal
HasNominalizer
HasPunctuation: comma, period
HasFiniteClausalModifier
RelativePosition: sole, first, mid, last
NSiblings (number of siblings)
Position (absolute position among siblings)
Voice: pass, caus, passcaus
Negation
Basic features for phrase relations (parent-child pair)
DependencyType: D,P,A,I
Distance: linear distance in bunsetsu, 1, 2-5, >6
Subcat: POS tag of parent + POS tag of all children + indication for current
Combined features (selected)
HeadPOS + HeadLemma
ParentLemma + HeadLemma
Position + NSiblings
IsFiniteClause + GrandparentNounSubPos

Table 2: Basic and combined features for local classifiers

classification task, we assign one of the 18 case markers to each phrase that has been labeled with HASCASE by the identification model.

We train a binary classifier for identification and a multi-class classifier (with 18 classes) for classification. We obtain a classifier for the complete task by chaining the two classifiers. Let $P_{ID}(c|b)$ and $P_{CLS}(c|b)$ denote the probability of class c for bunsetsu b according to the identification and classification models, respectively. We define the probability distribution over classes of the complete model for case assignment as follows:

$$P_{CaseAssign}(NONE|b) = P_{ID}(NONE|b)$$

$$P_{CaseAssign}(l|b) = P_{ID}(HASCASE|b) * P_{CLS}(l|b)$$

Here, l denotes one of the 18 case markers.

We employ this decomposition mainly for efficiency in training: that is, the decomposition allows us to train the classification models on a subset of training examples consisting only of those phrases that have a case marker, following Toutanova et al. (2005). Among various machine learning methods that can be used to train the classifiers, we chose log-linear models for both identification and classification tasks, as they

produce probability distributions which allows chaining of the two component models and easy integration into an MT system.

3.2 Joint classifiers

Toutanova et al. (2005) report a substantial improvement in performance on the semantic role labeling task by building a joint classifier, which takes the labels of other phrases into account when classifying a given phrase. This is motivated by the fact that the argument structure is a joint structure, with strong dependencies among arguments. Since the case markers also reflect the argument structure to some extent, we implemented a joint classifier for the case prediction task as well.

We applied the joint classifiers in the framework of N -best reranking (Collins, 2000), following Toutanova et al. (2005). That is, we produced N -best ($N=5$ in our experiments) case assignment sequence candidates for a set of sister phrases using the local models, and trained a joint classifier that learns to choose the best candidate from the set of sisters. The oracle accuracy of the 5-best candidate list was 95.9% per phrase.

4 Monolingual case prediction task

In this section we describe our models trained and evaluated using the gold-standard dependency annotations provided by the Kyoto Corpus. These annotations allow us to define a rich set of features exploring the syntactic structure.

4.1 Features

The basic local model features we used for the identification and classification models are listed in Table 2. They consist of features for a phrase, for its parent phrase and for their relations. Only one feature (GrandparentNounSubPos) currently refers to the grandparent of the phrase; all other features are between the phrase, its parent and its sibling nodes, and are a superset of the dependency-based features used by Hacioglu (2004) for the semantic labeling task. In addition to these basic features, we added 20 combined features, some of which are shown at the bottom of Table 2.

For the joint model, we implemented only two types of features: *sequence* of non-NONE case markers for a set of sister phrases, and *repetition* of non-NONE case markers. These features are intended to capture regularities in the sequence of case markers of phrases that modify the same head phrase.

All of these features are represented as binary features: that is, when the value of a feature is not binary, we have treated the combination of the feature name plus the value as a unique feature. With a count cut-off of 2 (i.e., features must occur at least twice to be in the model), we have 724,264 features in the identification

model, and 3,963,096 features in the classification model. The number of joint features in the joint model is 3,808. All models are trained using a Gaussian prior.

4.2 Data and baselines

We divided the Kyoto Corpus (version 3.0) into the following three sections:

- Training: contains news articles of January 1, 3-11 and editorial articles of January-August; 24,263 sentences, 234,474 phrases.
- Devtest: contains news articles of January 12-13 and editorial articles of September. 4,833 sentences, 47,580 phrases.
- Test: contains news articles of January 14-17 and editorial articles of October-December. 9,287 sentences, 89,982 phrases.

The devtest set was used only for tuning model parameters and for performing error analysis.

As no previous work exists on the task of predicting case markers on the Kyoto Corpus, it is important to establish a good baseline. The simplest baseline of always selecting the most frequent label (NONE) gives us an accuracy of 47.5% on the test set. Out of the non-NONE case markers, the most frequent is *no*, which occurs in 26.6% of all case-marked phrases.

A more reasonable baseline is to use a language model to predict case. We trained and tested two language models: the first model, called KCLM, is trained on the same data as our log-linear models (24,263 sentences); the second model, called BigCLM, is trained on much more data from the same domain (826,373 sentences), taking advantage of the fact that language models do not require dependency annotation for training. The language models were trained using the CMU language modeling toolkit with default parameter settings (Clarkson and Rosenfeld, 1997).

We tested the language model baselines using the same task set-up as for our classifier: for each phrase, each of the 18 possible case markers and NONE is evaluated. The position for insertion of a case marker in each phrase is given according to our task set-up, i.e., at the end of a phrase preceding any punctuation. We choose the case assignment of the sequence of phrases in the sentence that maximizes the language model probability of the resulting sentence. We computed the most likely case assignment sequence using a dynamic programming algorithm.

4.3 Results and discussion

The results of running our models on case marker prediction are shown in Table 3. The first three rows correspond to the components of the local model: the identification task (*Id*, for all phrases), the classification task (*Cls*, only for case-marked phrases) and the complete task (*Both*, for all phrases). The accuracy on

Models	Task	Training	Test
log-linear	Id	99.8	96.9
log-linear	Cls	96.6	74.3
log-linear (local)	Both	98.0	83.9
log-linear(joint)	Both	97.8	84.3
baseline (frequency)	Both	48.2	47.5
baseline (KCLM)	Both	93.9	67.0
baseline (BigCLM)	Both	—	78.0

Table 3: Accuracy of case prediction models (%)

the complete task using the local model is 83.9%; the joint model improves it to 84.3%.

The improvement due to the joint model is small in absolute percentage points (0.4%), but is statistically significant according to a test for the difference of proportions ($p < 0.05$). The use of a joint classifier did not lead to as large an improvement over the local classifier as for the semantic role labeling task. There are several reasons for that we can think of. First, we have only used a limited set of features for the joint model, i.e., case sequence and repetition features. A more extensive use of global features might lead to a larger improvement. Secondly, unlike the task of semantic role labeling, where there are about 20 phrases that need to be labeled with respect to a predicate, about 50% of all phrases in the Kyoto Corpus do not have sister nodes. This means that these phrases cannot take advantage of the joint classifier using the current model formulation. Finally, case markers are much shallower than semantic role labels in the level of linguistic analysis, and so are inherently subject to more variations, including missing arguments (so called zero pronouns) and repeated case markers corresponding to different semantic roles.

From Table 3, it is clear that our models outperform the baseline model significantly. The language model trained on the same data has much lower performance (67.0% vs. 84.3%), which shows that our system is exploiting the training data much more efficiently by looking at the dependency and other syntactic features. An inspection of the 500 most highly weighted features also indicates that phrase dependency-based features are very useful for both identification and classification. Given much more data, though, the language model improves significantly to 78%, but our classifier still achieves a 29% error reduction over it. The differences between the language models and the log-linear models are statistically significant at level $p < 0.01$ according to a test for the difference of proportions.

Figure 2 plots the recall and precision for the frequently occurring (>500) cases. We achieve good results on NONE and *no*, which are the least ambiguous decisions. Cases such as *ni*, *wa*, *ga*, and *de* are highly confusable with other markers as they indicate multiple grammatical relations, and the performance of our

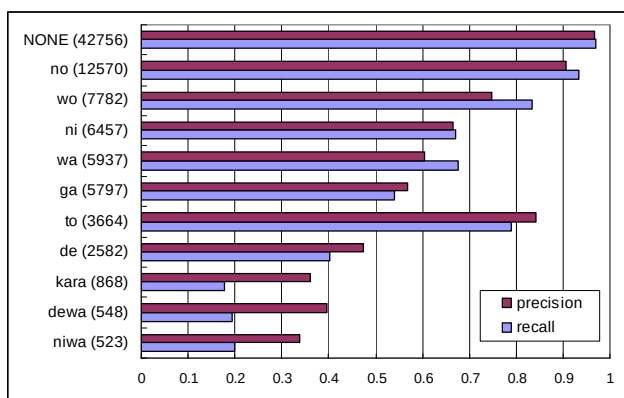


Figure 2: Precision and recall per case marker (frequency in parentheses)

models on them is therefore limited. As expected, performance (especially recall) on secondary targets (*dewa*, *niwa*) suffers greatly due to the ambiguity with their primary targets.

5 Bilingual case prediction task: simulating case prediction in MT

Incorporating a case prediction model into MT requires taking additional factors into consideration, compared to the monolingual task described above. On the one hand, we need to extend our model to handle the additional knowledge source, i.e., the source sentence. This can potentially provide very useful features to our model, which are not available in the monolingual task. On the other hand, since gold-standard dependency annotation is not available in the MT context, we must deal with the imperfections in structural annotations.

In this section, we describe our case prediction models in the context of English-to-Japanese MT. In this setting, dependency information for the target language (Japanese) is available only through projection of a dependency structure from the source language (English) in a tree-to-string-based statistical MT system (Quirk et al., 2005). We conducted experiments using the English source sentences and the reference translations in Japanese: that is, our task is to predict the case markers of the Japanese reference translations correctly using all other words in the reference sentence, information from the source sentence through word alignment, and the Japanese dependency structure projected via an MT component. Ultimately, our goal is to improve the case marker assignment of a candidate translation using a case prediction model; the experiments described in this section on reference translations serve as an important preliminary step toward achieving that final goal. We will show in this section that even the automatically derived syntactic information is very useful in assigning case markers in

the target language, and that utilizing the information from the source language also greatly contributes to reducing case marking errors.

5.1 Data and task set-up

The dataset we used is a collection of parallel English-Japanese sentences from a technical (computer) domain. We used 15,000 sentence pairs for training, 5,000 for development, and 4,241 for testing.

The parallel sentences were word-aligned using GIZA++ (Och and Ney, 2000), and submitted to a tree-to-string-based MT system (Quirk et al., 2005) which utilizes the dependency structure of the source language and projects dependency structure to the target language. Figure 3 shows an example of an aligned sentence pair: on the source (English) side, part-of-speech (POS) tags and word dependency structure are assigned (solid arcs). The alignments between English and Japanese words are indicated by the dotted lines. In order to create phrase-level dependency structures like the ones utilized in the Kyoto Corpus monolingual task, we derived some additional information for the Japanese sentence in the following manner.

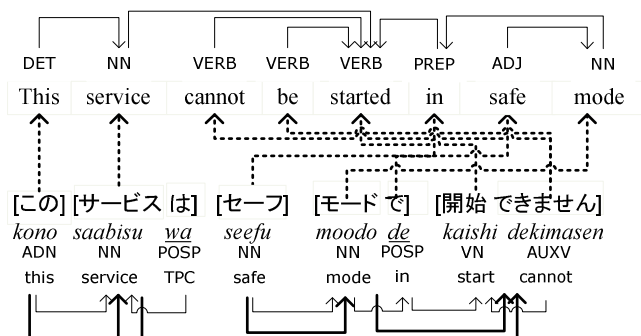


Figure 3. Aligned English-Japanese sentence pair

First, we tagged the sentence using an automatic tagger with a set of 19 POS tags. We used these POS tags to parse the words into phrases (bunsetsu): each bunsetsu consists of one content word plus any number of function words, where content and function words are defined via POS. We then constructed a phrase-level dependency structure using a breadth-first traversal of the word dependency structure projected from English. These phrase dependencies are indicated by bold arcs in Figure 3. The case markers to be predicted (*wa* and *de* in this case) are underlined.

The task of case marker prediction is the same as described in Section 2: to assign one of the 18 case markers described in Section 2 or NONE to each phrase.

5.2 Baseline models

We implemented the baseline models discussed in Section 4.2 for this domain as well. The most frequent

Monolingual features	
Feature	Example
HeadWord /HeadPOS	<i>saabisu/NN</i>
PrevWord/PrevPOS	<i>kono/AND</i>
Prev2Word/Prev2WordPOS	<i>none/none</i>
NextWord/NextPOS	<i>seefu/NN</i>
Next2Word/Net2POS	<i>moodo/NN</i>
PrevHeadWord/PrevHeadPOS	<i>kono/AND</i>
NextHeadWord/NextHeadPOS	<i>seefu/NN</i>
ParentHeadWord/ParentHeadPOS	<i>kaishi/VN</i>
Subcat: POS tags of all sisters and parent	<i>NN-c,NN,VN-h</i>
NSiblings (including self)	2
Distance	1
Direction	<i>left</i>
Alternative Parent Word /POS	<i>saabisu/NN</i>
Bilingual features	
Feature	Example
Word/POS of source words aligned to the head of the phrase	<i>service/NN</i>
Word/POS of all source words aligned to any word in the phrase	<i>service/NN</i>
Word/POS of all source words aligned to the head word of the parent phrase	<i>started/VERB</i>
Word/POS of all source words aligned to alternative parent words of the phrase	<i>service/NN, started/VERB</i>
All source preposition words	<i>in</i>
Word/POS of parent of source word aligned to any word in the phrase	<i>started/VERB</i>
Aligned Subcat	<i>NN-c,VERB,VERB,VERB-h,PREP</i>
Aligned NSiblings	4
Aligned Distance	2
Aligned Direction	<i>left</i>

Table 4: Monolingual and bilingual features

case assignment is again NONE, which accounts for 62.0% of the test set. The frequency of NONE is higher in this task than in the Kyoto Corpus, because our bunsetsu-parsing algorithm prefers to err on the side of making too many rather than too few phrases. This is because our final goal is to generate all case markers, and if we mistakenly joined two bunsetsu into one, our case assigner would be able to propose only one case marker for the resulting bunsetsu, which would be necessarily wrong if both bunsetsu had case markers. The most frequent case marker is again *no*, which occurs in 29.4% of all case-marked phrases. As in the monolingual task, we trained two trigram language models: one was trained on the training set of our case prediction models (15,000 sentences); another was trained on a much larger set of 450,000 sentences from the same domain. The results of these baselines are discussed in Section 5.4.

5.3 Log-linear models

The models we built for this task are log-linear models as described in Section 3. In order to isolate the impact of information from the source language available for the case prediction task, we built two kinds of models:

Model	Test data
baseline (frequency)	62.0
baseline (15kLM)	79.0
baseline (450kLM)	83.6
log-linear monolingual	85.3
log-linear bilingual	92.3

Table 5: Accuracy of bilingual case prediction (%)

monolingual models, which do not use any information from the source English sentences, and *bilingual* models, which use information from the source. Both models are local models in the sense discussed in Section 3.

Table 4 shows the features used in the monolingual and bilingual models, along with the examples (the value of the feature for the phrase [*saabisu wa*] in Figure 3); in addition to these, we also provided some feature combinations for both monolingual and bilingual models. Many of the monolingual features (i.e., first 11 lines in Table 4) are also present in Table 2. Note that lexically based features are of greater importance for this task, as the dependency information available in this context is of much poorer quality than that provided by the Kyoto Corpus. In addition to the features in Table 2, we added a Direction feature (with values *left* and *right*), and an Alternative Parent feature. Alternative parents are all words which are the parents of any word in the phrase, according to the word-based dependency tree, with the constraint that case markers cannot be alternative parents. This feature captures the information that is potentially lost in the process of building a phrase dependency structure from word dependency information in the target language.

The bottom half of Table 4 shows bilingual features. The features of the source sentence are obtained through word alignments. We create features from the source words aligned to the head of the phrase, to the head of the parent phrase, or to any alternative parents. If any word in the phrase is aligned to a preposition in the source language, our model can use the information as well. In addition to word- and POS-features for aligned source words, we also refer to the corresponding dependency between the phrase and its parent phrase in the English source. If the head of the Japanese phrase is aligned to a single source word s_1 , and the head of its parent phrase is aligned to a single source word s_2 , we extract the relationship between s_1 and s_2 , and define subcategorization, direction, distance, and number of siblings features, in order to capture the grammatical relation in the source, which is more reliable than in the projected target dependency structure.

5.4 Results and discussion

Table 5 summarizes the results on the complete case assignment task in the MT context. Compared to the language model trained on the same data (15kLM), our

monolingual model performs significantly better, achieving a 30% error reduction (85.3% vs. 79.0%). Our monolingual model outperforms even the language model trained on 30 times more data (85.3% vs. 83.6%), with an error reduction of 10%. The difference is statistically significant at level $p < 0.01$ according to a test for the difference of proportions. This means that even though the projected dependency information is not perfect, it is still useful for the case prediction task.

When we add the bilingual features, the error rate of our model is cut almost in half: the bilingual model achieves an error reduction of 48% over the monolingual model (92.3% vs. 85.3%, statistically significant at level $p < 0.01$). This result is very encouraging: it indicates that information from the source sentence can be exploited very effectively to improve the accuracy of case assignment. The usefulness of the source language information is also obvious when we inspect which case markers had the largest gains in accuracy due to this information: the top three cases were *kara* (0.28 to 0.65, a 57% gain), *dewa* (0.44 to 0.65, a 32% gain) and *to* (0.64 to 0.85, a 24% gain), all of which have translations as English prepositions. Markers such as *ga* (subject marker, 0.68 to 0.74, a 8% gain) and *wo* (object marker, 0.83 to 0.86, a 3.5% gain), on the other hand, showed only a limited gain.

6 Conclusion and future directions

This paper described the task of predicting case markers in Japanese, and reported results in a monolingual and a bilingual settings. The results show that the models we proposed, which explore syntax-based features and features from the source language in the bilingual task, can effectively predict case markers.

There are a number of extensions and next steps we can think of at this point, the most immediate and important one of which is to incorporate the proposed model in an end-to-end MT system to make improvements in the output of MT. We would also like to perform a more extensive analysis of features and feature ablation experiments. Finally, we would also like to extend the proposed model to include languages with inflectional morphology and the prediction of grammatical elements in general.

Acknowledgements

We would like to thank the anonymous reviewers for their comments, and Bob Moore, Arul Menezes, Chris Quirk, and Lucy Vanderwende for helpful discussions.

References

Baldwin, T. 2004. Making Sense of Japanese Relative Clause Constructions, In *Proceedings of the 2nd Workshop on Text Meaning and Interpretation*.

- Blaheta, D. and E. Charniak. 2000. Assigning function tags to parsed text. In *Proceedings of NAACL*, pp.234-240.
- Carreras, X. and L. Màrquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *Proceedings of CoNLL-2005*.
- Clarkson, P.R. and R. Rosenfeld. 1997. Statistical Language Modeling Using the CMU-Cambridge Toolkit. In *Proceedings of ESCA Eurospeech*, pp. 2007-2010.
- Collins, M. 2000. Discriminative reranking for natural language parsing. In *Proceedings of ICML*.
- Gamon, M., E. Ringger, S. Corston-Oliver and R. Moore. 2002. Machine-learned Context for Linguistic Operations in German Sentence Realization. In *Proceeding of ACL*.
- Gildea, D. and D. Jurafsky. 2002. Automatic Labeling of Semantic Roles. In *Computational Linguistics* 28(3): 245-288.
- Hacioglu, K. 2004. Semantic Role Labeling using Dependency Trees. In *Proceedings of COLING 2004*.
- Kawahara, D., N. Kaji and S. Kurohashi. 2000. Japanese Case Structure Analysis by Unsupervised Construction of a Case Frame Dictionary. In *Proceedings of COLING*, pp. 432-438.
- Kurohashi, S. and M.Nagao. 1997. Kyoto University Text Corpus Project. In *Proceedings of ANLP*, pp.115-118.
- Masuoka, T. and Y. Takubo. 1992. *Kiso Nihongo Bunpou* (Fundamental Japanese grammar), revised version. Kuroshio Shuppan, Tokyo.
- Murata, M., and H. Isahara. 2005. Japanese Case Analysis Based on Machine Learning Method that Uses Borrowed Supervised Data. In *Proceedings of IEEE NLP-KE-2005*, pp.774-779.
- Och, F.J. and H. Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL*: pp.440-447.
- Palmer, M., D. Gildea and P. Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. In *Computational Linguistics* 31(1).
- Pradhan, S., W. Ward, K. Hacioglu, L. Martin, D. Jurafsky. 2004. Shallow Semantic Parsing Using Support Vector Machines. In *Proceedings of HLT/NAACL*.
- Quirk, C., A. Menezes and C. Cherry. 2005. Dependency Tree Translation: Syntactically Informed Phrasal SMT. In *Proceedings of ACL*.
- Teramura, H. 1991. *Nihongo-no shintakusu-to imi* (Japanese syntax and meaning). Volume III. Kuroshio Shuppan, Tokyo.
- Toutanova, K., A. Haghighi and C. D. Manning. 2005. Joint Learning Improves Semantic Role Labeling. In *Proceeding of ACL*, pp.589-596.
- Uchimoto, K., S. Sekine and H. Isahara. 2002. Text Generation from Keywords. In *Proceedings of COLING 2002*, pp.1037-1043.

Are These Documents Written from Different Perspectives? A Test of Different Perspectives Based On Statistical Distribution Divergence

Wei-Hao Lin

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213 U.S.A.
whlin@cs.cmu.edu

Alexander Hauptmann

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213 U.S.A.
alex@cs.cmu.edu

Abstract

In this paper we investigate how to automatically determine if two document collections are written from different *perspectives*. By perspectives we mean a point of view, for example, from the perspective of Democrats or Republicans. We propose a test of different perspectives based on distribution divergence between the statistical models of two collections. Experimental results show that the test can successfully distinguish document collections of different perspectives from other types of collections.

1 Introduction

Conflicts arise when two groups of people take very different perspectives on political, socio-economical, or cultural issues. For example, here are the answers that two presidential candidates, John Kerry and George Bush, gave during the third presidential debate in 2004 in response to a question on abortion:

- (1) Kerry: What is an article of faith for me is not something that I can legislate on somebody who doesn't share that article of faith. I believe that choice is a woman's choice. It's between a woman, God and her doctor. And that's why I support that.
- (2) Bush: I believe the ideal world is one in which every child is protected in law and welcomed to life. I understand there's great differences on this issue of abortion, but I believe reasonable people can come together and put good law in place that will help reduce the number of abortions.

After reading the above transcripts some readers may conclude that one takes a "pro-choice" perspective while the other takes a "pro-life" perspective, the two dominant perspectives in the abortion controversy.

Perspectives, however, are not always manifested when two pieces of text together are put together. For example, the following two sentences are from Reuters newswire:

- (3) Gold output in the northeast China province of Heilongjiang rose 22.7 pct in 1986 from 1985's level, the New China News Agency said.
- (4) Exco Chairman Richard Lacy told Reuters the acquisition was being made from Bank of New York Co Inc, which currently holds a 50.1 pct, and from RMJ partners who hold the remainder.

A reader would not from this pair of examples perceive as strongly contrasting perspectives as the Kerry-Bush answers. Instead, as the Reuters annotators did, one would label Example 3 as "gold" and Example 4 as "acquisition", that is, as two *topics* instead of two perspectives.

Why does the contrast between Example 1 and Example 2 convey different perspectives, but the contrast between Example 3 and Example 4 result in different topics? How can we define the impalpable "different perspectives" anyway? The definition of "perspective" in the dictionary is "subjective evaluation of relative significance,"¹ but can we have a computable definition to test the existence of different perspectives?

¹The American Heritage Dictionary of the English Language, 4th ed. We are interested in identifying "ideological perspectives" (Verdonk, 2002), not first-person or second-person "perspective" in narrative.

The research question about the definition of different perspectives is not only scientifically intriguing, it also enables us to develop important natural language processing applications. Such a computational definition can be used to detect the emergence of contrasting perspectives. Media and political analysts regularly monitor broadcast news, magazines, newspapers, and blogs to see if there are public opinion splitting. The huge number of documents, however, make the task extremely daunting. Therefore an automated test of different perspectives will be very valuable to information analysts.

We first review the relevant work in Section 2. We take a model-based approach to develop a computational definition of different perspectives. We first develop statistical models for the two document collections, $\mathcal{A}$ and $\mathcal{B}$, and then measure the degree of contrast by calculating the “distance” between $\mathcal{A}$ and $\mathcal{B}$. How document collections are statistically modeled and how distribution difference is estimated are described in Section 3. The document corpora are described in Section 4. In Section 5, we evaluate how effective the proposed test of difference perspectives based on statistical distribution. The experimental results show that the distribution divergence can successfully separate document collections of different perspectives from other kinds of collection pairs. We also investigate if the pattern of distribution difference is due to personal writing or speaking styles.

2 Related Work

There has been interest in understanding how beliefs and ideologies can be represented in computers since mid-sixties of the last century (Abelson and Carroll, 1965; Schank and Abelson, 1977). The Ideology Machine (Abelson, 1973) can simulate a right-wing ideologue, and POLITICS (Carbonell, 1978) can interpret a text from conservative or liberal ideologies. In this paper we take a statistics-based approach, which is very different from previous work that rely very much on manually-constructed knowledge base.

Note that what we are interested in is to determine if two document collections are written from different perspectives, not to model individual perspectives. We aim to capture the characteristics, specifically the statistical regularities of any pairs of document collections with opposing perspectives. Given a pair of document collections $\mathcal{A}$ and

$\mathcal{B}$, our goal is not to construct classifiers that can predict if a document was written from the perspective of $\mathcal{A}$ or $\mathcal{B}$ (Lin et al., 2006), but to determine if the document collection pair $(\mathcal{A}, \mathcal{B})$ convey opposing perspectives.

There has been growing interest in subjectivity and sentiment analysis. There are studies on learning subjective language (Wiebe et al., 2004), identifying opinionated documents (Yu and Hatzivasiloglou, 2003) and sentences (Riloff et al., 2003; Riloff and Wiebe, 2003), and discriminating between positive and negative language (Turney and Littman, 2003; Pang et al., 2002; Dave et al., 2003; Nasukawa and Yi, 2003; Morinaga et al., 2002). There are also research work on automatically classifying movie or product reviews as positive or negative (Nasukawa and Yi, 2003; Mullen and Collier, 2004; Beineke et al., 2004; Pang and Lee, 2004; Hu and Liu, 2004).

Although we expect by its very nature much of the language used when expressing a perspective to be subjective and opinionated, the task of labeling a document or a sentence as subjective is orthogonal to the test of different perspectives. A subjectivity classifier may successfully identify all subjective sentences in the document collection pair $\mathcal{A}$ and $\mathcal{B}$, but knowing the number of subjective sentences in $\mathcal{A}$ and $\mathcal{B}$ does not necessarily tell us if they convey opposing perspectives. We utilize the subjectivity patterns automatically extracted from foreign news documents (Riloff and Wiebe, 2003), and find that the percentages of the subjective sentences in the bitterlemons corpus (see Section 4) are similar (65.6% in the Palestinian documents and 66.2% in the Israeli documents). The high but almost equivalent number of subjective sentences in two perspectives suggests that perspective is largely expressed in subjective language but subjectivity ratio is not enough to tell if two document collections are written from the same (Palestinian v.s. Palestinian) or different perspectives (Palestinian v.s. Israeli)².

3 Statistical Distribution Divergence

We take a model-based approach to measure to what degree, if any, two document collections are different. A document is represented as a point

²However, the close subjectivity ratio doesn’t mean that subjectivity can never help identify document collections of opposing perspectives. For example, the accuracy of the test of different perspectives may be improved by focusing on only subjective sentences.

in a V -dimensional space, where V is vocabulary size. Each coordinate is the frequency of a word in a document, i.e., term frequency. Although vector representation, commonly known as a bag of words, is oversimplified and ignores rich syntactic and semantic structures, more sophisticated representation requires more data to obtain reliable models. Practically, bag-of-word representation has been very effective in many tasks, including text categorization (Sebastiani, 2002) and information retrieval (Lewis, 1998).

We assume that a collection of N documents, $y_1, y_2, \dots, y_N$ are sampled from the following process,

$$\begin{aligned}\theta &\sim \text{Dirichlet}(\alpha) \\ y_i &\sim \text{Multinomial}(n_i, \theta).\end{aligned}$$

We first sample a V -dimensional vector θ from a Dirichlet prior distribution with a hyperparameter α , and then sample a document y_i repeatedly from a Multinomial distribution conditioned on the parameter θ , where n_i is the document length of the i th document in the collection and assumed to be known and fixed.

We are interested in comparing the parameter θ after observing document collections $\mathcal{A}$ and $\mathcal{B}$:

$$\begin{aligned}p(\theta|\mathcal{A}) &= \frac{p(\mathcal{A}|\theta)p(\theta)}{p(\mathcal{A})} \\ &= \text{Dirichlet}(\theta|\alpha + \sum_{y_i \in \mathcal{A}} y_i).\end{aligned}$$

The posterior distribution $p(\theta|\cdot)$ is a Dirichlet distribution since a Dirichlet distribution is a conjugate prior for a Multinomial distribution.

How should we measure the difference between two posterior distributions $p(\theta|\mathcal{A})$ and $p(\theta|\mathcal{B})$? One common way to measure the difference between two distributions is Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951), defined as follows,

$$\begin{aligned}D(p(\theta|\mathcal{A})||p(\theta|\mathcal{B})) \\ = \int p(\theta|\mathcal{A}) \log \frac{p(\theta|\mathcal{A})}{p(\theta|\mathcal{B})} d\theta.\end{aligned}\quad (5)$$

Directly calculating KL divergence according to (5) involves a difficult high-dimensional integral. As an alternative, we approximate KL divergence using Monte Carlo methods as follows,

1. Sample $\theta_1, \theta_2, \dots, \theta_M$ from $\text{Dirichlet}(\theta|\alpha + \sum_{y_i \in \mathcal{A}} y_i)$.

2. Return $\hat{D} = \frac{1}{M} \sum_{i=1}^M \log \frac{p(\theta_i|\mathcal{A})}{p(\theta_i|\mathcal{B})}$ as a Monte Carlo estimate of $D(p(\theta|\mathcal{A})||p(\theta|\mathcal{B}))$.

Algorithms of sampling from Dirichlet distribution can be found in (Ripley, 1987). As $M \rightarrow \infty$, the Monte Carlo estimate will converge to true KL divergence by the Law of Large Numbers.

4 Corpora

To evaluate how well KL divergence between posterior distributions can discern a document collection pair of different perspectives, we collect two corpora of documents that were written or spoken from different perspectives and one newswire corpus that covers various topics, as summarized in Table 1. No stemming algorithms is performed; no stopwords are removed.

Corpus	Subset	$ \mathcal{D} $	$ \bar{d} $	V
bitterlemons	Palestinian	290	748.7	10309
	Israeli	303	822.4	11668
	Pal. Editor	144	636.2	6294
	Pal. Guest	146	859.6	8661
	Isr. Editor	152	819.4	8512
	Isr. Guest	151	825.5	8812
2004 Presidential Debate	Kerry	178	124.7	2554
	Bush	176	107.8	2393
	1st Kerry	33	216.3	1274
	1st Bush	41	155.3	1195
	2nd Kerry	73	103.8	1472
	2nd Bush	75	89.0	1333
	3rd Kerry	72	104.0	1408
	3rd Bush	60	98.8	1281
Reuters-21578	ACQ	2448	124.7	14293
	CRUDE	634	214.7	9009
	EARN	3987	81.0	12430
	GRAIN	628	183.0	8236
	INTEREST	513	176.3	6056
	MONEY-FX	801	197.9	8162
	TRADE	551	255.3	8175

Table 1: The number of documents $|\mathcal{D}|$, average document length $|\bar{d}|$, and vocabulary size V of the three corpora.

The first perspective corpus consists of articles published on the bitterlemons website³ from late 2001 to early 2005. The website is set up to “contribute to mutual understanding [between Palestinians and Israelis] through the open exchange of ideas”⁴. Every week an issue about the Israeli-Palestinian conflict is selected for discussion (e.g., “Disengagement: unilateral or coordinated?”), and a Palestinian editor and an Israeli editor each contribute one article addressing the

³<http://www.bitterlemons.org/>

⁴<http://www.bitterlemons.org/about/about.html>

issue. In addition, the Israeli and Palestinian editors interview a guest to express their views on the issue, resulting in a total of four articles in a weekly edition. The perspective from which each article is written is labeled as either Palestinian or Israeli by the editors.

The second perspective corpus consists of the transcripts of the three Bush-Kerry presidential debates in 2004. The transcripts are from the website of the Commission on Presidential Debates⁵. Each spoken document is roughly an answer to a question or a rebuttal. The transcript are segmented by the speaker tags already in the transcripts. All words from moderators are discarded.

The topical corpus contains newswire from Reuters in 1987. Reuters-21578⁶ is one of the most common testbeds for text categorization. Each document belongs to none, one, or more of the 135 categories (e.g., “Mergers” and “U.S. Dollars”.) The number of documents in each category is not evenly distributed (median 9.0, mean 105.9). To estimate statistics reliably, we only consider categories with more than 500 documents, resulting in a total of seven categories (ACQ, CRUDE, EARN, GRAIN, INTEREST, MONEY-FX, and TRADE).

5 Experiments

A test of different perspectives is acute when it can draw distinctions between document collection pairs of different perspectives and document collection pairs of the same perspective and others. We thus evaluate the proposed test of different perspectives in the following four types of document collection pairs $(\mathcal{A}, \mathcal{B})$:

Different Perspectives (DP) $\mathcal{A}$ and $\mathcal{B}$ are written from different perspectives. For example, $\mathcal{A}$ is written from the Palestinian perspective and $\mathcal{B}$ is written from the Israeli perspective in the bitterlemons corpus.

Same Perspective (SP) $\mathcal{A}$ and $\mathcal{B}$ are written from the same perspective. For example, $\mathcal{A}$ and $\mathcal{B}$ consist of the words spoken by Kerry.

Different Topics (DT) $\mathcal{A}$ and $\mathcal{B}$ are written on different topics. For example, $\mathcal{A}$ is about

acquisition (ACQ) and $\mathcal{B}$ is about crude oil (CRUDE).

Same Topic (ST) $\mathcal{A}$ and $\mathcal{B}$ are written on the same topic. For example, $\mathcal{A}$ and $\mathcal{B}$ are both about earnings (EARN).

The effectiveness of the proposed test of different perspectives can thus be measured by how the distribution divergence of DP document collection pairs is separated from the distribution divergence of SP, DT, and ST document collection pairs. The little the overlap of the range of distribution divergence, the sharper the test of different perspectives.

To account for large variation in the number of words and vocabulary size across corpora, we normalize the total number of words in a document collection to be the same K , and consider only the top $C\%$ frequent words in the document collection pair. We vary the values of K and C , and find that K changes the absolute scale of KL divergence but does not change the rankings of four conditions. Rankings among four conditions is consistent when C is small. We only report results of $K = 1000$, $C = 10$ in the paper due to space limit.

There are two kinds of variances in the estimation of divergence between two posterior distribution and should be carefully checked. The first kind of variance is due to Monte Carlo methods. We assess the Monte Carlo variance by calculating a 100α percent confidence interval as follows,

$$[\hat{D} - \Phi^{-1}\left(\frac{\alpha}{2}\right)\frac{\hat{\sigma}}{\sqrt{M}}, \hat{D} + \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\frac{\hat{\sigma}}{\sqrt{M}}]$$

where $\hat{\sigma}^2$ is the sample variance of $\theta_1, \theta_2, \dots, \theta_M$, and $\Phi(\cdot)^{-1}$ is the inverse of the standard normal cumulative density function. The second kind of variance is due to the intrinsic uncertainties of data generating processes. We assess the second kind of variance by collecting 1000 bootstrapped samples, that is, sampling with replacement, from each document collection pair.

5.1 Quality of Monte Carlo Estimates

The Monte Carlo estimates of the KL divergence from several document collection pair are listed in Table 2. A complete list of the results is omitted due to the space limit. We can see that the 95% confidence interval captures well the Monte Carlo estimates of KL divergence. Note that KL divergence is not symmetric. The KL divergence

⁵<http://www.debates.org/pages/debtrans.html>

⁶<http://www.ics.uci.edu/~kdd/databases/reuters21578/reuters21578.html>

$\mathcal{A}$	$\mathcal{B}$	$\hat{D}$	95% CI
ACQ	ACQ	2.76	[2.62, 2.89]
Palestinian	Palestinian	3.00	[3.54, 3.85]
Palestinian	Israeli	27.11	[26.64, 27.58]
Israeli	Palestinian	28.44	[27.97, 28.91]
Kerry	Bush	58.93	[58.22, 59.64]
ACQ	EARN	615.75	[610.85, 620.65]

Table 2: The Monte Carlo estimate $\hat{D}$ and 95% confidence interval (CI) of the Kullback-Leibler divergence of several document collection pairs $(\mathcal{A}, \mathcal{B})$ with the number of Monte Carlo samples $M = 1000$.

of the pair (Israeli, Palestinian) is not necessarily the same as (Palestinian, Israeli). KL divergence is greater than zero (Cover and Thomas, 1991) and equal to zero only when document collections $\mathcal{A}$ and $\mathcal{B}$ are exactly the same. Here (ACQ, ACQ) is close to but not exactly zero because they are different samples of documents in the ACQ category. Since the CIs of Monte Carlo estimates are reasonably tight, we assume them to be exact and ignore the errors from Monte Carlo methods.

5.2 Test of Different Perspectives

We now present the main result of the paper. We calculate the KL divergence between posterior distributions of document collection pairs in four conditions using Monte Carlo methods, and plot the results in Figure 1. The test of different perspectives based on statistical distribution divergence is shown to be very acute. The KL divergence of the document collection pairs in the DP condition fall mostly in the middle range, and is well separated from the high KL divergence of the pairs in DT condition and from the low KL divergence of the pairs in SP and ST conditions. Therefore, by simply calculating the KL divergence of a document collection pair, we can reliably predict that they are written from different perspectives if the value of KL divergence falls in the middle range, from different topics if the value is very large, from the same topic or perspective if the value is very small.

5.3 Personal Writing Styles or Perspectives?

One may suspect that the mid-range distribution divergence is attributed to personal speaking or writing styles and has nothing to do with different perspectives. The doubt is expected because

half of the bitterlemons corpus are written by one Palestinian editor and one Israeli editor (see Table 1), and the debate transcripts come from only two candidates.

We test the hypothesis by computing the distribution divergence of the document collection pair (Israeli Guest, Palestinian Guest), that is, a Different Perspectives (DP) pair. There are more than 200 different authors in the Israeli Guest and Palestinian Guest collection. If the distribution divergence of the pair with diverse authors falls out of the middle range, it will support that mid-range divergence is due to writing styles. On the other hand, if the distribution divergence still fall in the middle range, we are more confident the effect is attributed to different perspectives. We compare the distribution divergence of the pair (Israeli Guest, Palestinian Guest) with others in Figure 2.

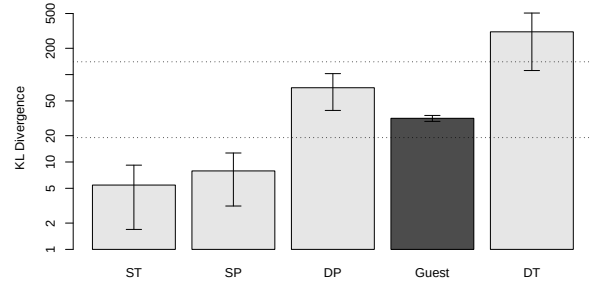


Figure 2: The average KL divergence of document collection pairs in the bitterlemons Guest subset (Israeli Guest vs. Palestinian Guest), ST, SP, DP, DT conditions. The horizontal lines are the same as those in Figure 1.

The results show that the distribution divergence of the (Israeli Guest, Palestinian Guest) pair, as other pairs in the DP condition, still falls in the middle range, and is well separated from SP and ST in the low range and DT in the high range. The decrease in KL divergence due to writing or speaking styles is noticeable, and the overall effect due to different perspectives is strong enough to make the test robust. We thus conclude that the test of different perspectives based on distribution divergence indeed captures different perspectives, not personal writing or speaking styles.

5.4 Origins of Differences

While the effectiveness of the test of different perspectives is demonstrated in Figure 1, one may

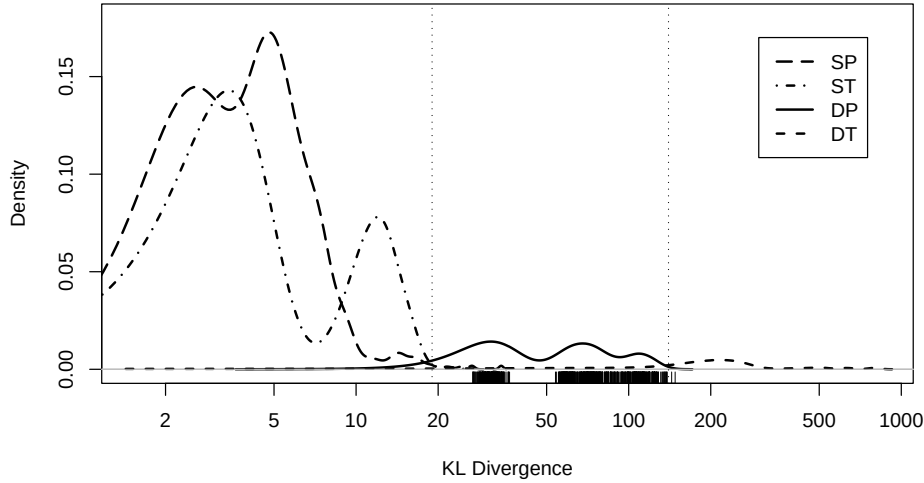


Figure 1: The KL divergence of the document collection pairs in four conditions: Different Perspectives (DP), Same Perspective (SP), Different Topics (DT), and Same Topic (ST). Note that the x axis is in log scale. The Monte Carlo estimates $\hat{D}$ of the pairs in DP condition are plotted as rugs. $\hat{D}$ of the pairs in other conditions are omitted to avoid clutter and summarized in one-dimensional density using Kernel Density Estimation. The vertical lines are drawn at the points with equivalent densities.

wonder why the distribution divergence of the document collection pair with different perspectives falls in the middle range and what causes the large and small divergence of the document collection pairs with different topics (DT) and the same topic (ST) or perspective (SP), respectively. In other words where do the differences result from?

We answer the question by taking a closer look at the causes of the distribution divergence in our model. We compare the expected marginal difference of θ between two posterior distributions $p(\theta|\mathcal{A})$ and $p(\theta|\mathcal{B})$. The marginal distribution of the i -th coordinate of θ , that is, the i -th word in the vocabulary, is a Beta distribution, and thus the expected value can be easily calculated. We plot the $\Delta\theta = E[\theta_i|\mathcal{A}] - E[\theta_i|\mathcal{B}]$ against $E[\theta_i|\mathcal{A}]$ for each condition in Figure 3.

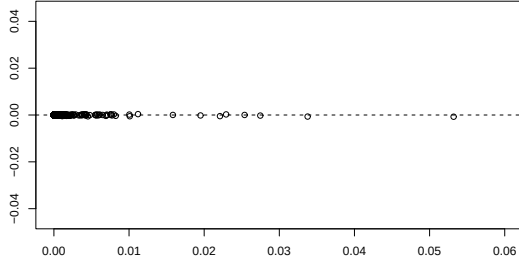
How $\Delta\theta$ is deviated from zero partially explains different patterns of distribution divergence in Figure 1. In Figure 3d we see that the $\Delta\theta$ increases as θ increases, and the deviance from zero is much greater than those in the Same Perspective (Figure 3b) and Same Topic (Figure 3a) conditions. The large $\Delta\theta$ not only accounts for large distribution divergence of the document pairs in DT conditions, but also shows that words in different topics that is frequent in one topic are less likely to be

frequent in the other topic. At the other extreme, document collection pairs of the Same Perspective (SP) or Same Topic (ST) show very little difference in θ , which matches our intuition that documents of the same perspective or the same topic use the same vocabulary in a very similar way.

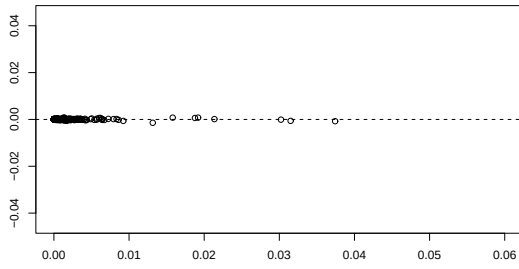
The manner in which $\Delta\theta$ is varied with the value of θ in the Different Perspective (DP) condition is very unique. The $\Delta\theta$ in Figure 3c is not as small as those in the SP and ST conditions, but at the same time not as large as those in DT conditions, resulting in mid-range distribution divergence in Figure 1. Why do document collections of different perspectives distribute this way? Partly because articles from different perspectives focus on the closely related issues (the Palestinian-Israeli conflict in the bitterlemons corpus, or the political and economical issues in the debate corpus), the authors of different perspectives write or speak in a similar vocabulary, but with emphasis on different words.

6 Conclusions

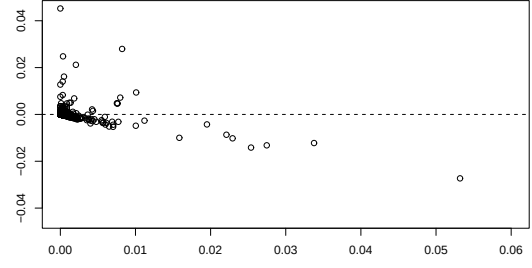
In this paper we develop a computational test of different perspectives based on statistical distribution divergence between the statistical models of document collections. We show that the pro-



(a) Same Topic (ST)

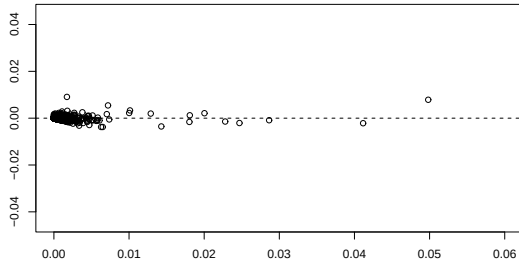


(b) Same Topic (SP)



(d) Two examples of Different Topics (DT)

Figure 3: Cont'd



(c) Two examples of Different Perspective (DP)

Figure 3: The $\Delta\theta$ vs. θ plots of the typical document collection pairs in four conditions. The horizontal line is $\Delta\theta = 0$.

posed test can successfully separate document collections of different perspectives from other types of document collection pairs. The distribution divergence falling in the middle range can not simply be attributed to personal writing or speaking styles. From the plot of multinomial parameter difference we offer insights into where the different patterns of distribution divergence come from.

Although we validate the test of different perspectives by comparing the DP condition with DT, SP, and ST conditions, the comparisons are by no means exhaustive, and the distribution divergence of some document collection pairs may also fall in the middle range. We plan to investigate more types of document collections pairs, e.g., the document collections from different text genres (Kessler et al., 1997).

Acknowledgment

We would like to thank the anonymous reviewers for useful comments and suggestions. This material is based on work supported by the Advanced Research and Development Activity (ARDA) under contract number NBCHC040037.

References

- Robert P. Abelson and J. Douglas Carroll. 1965. Computer simulation of individual belief systems. *The American Behavioral Scientist*, 8:24–30, May.
- Robert P. Abelson, 1973. *Computer Models of Thought and Language*, chapter The Structure of Belief Systems, pages 287–339. W. H. Freeman and Company.
- Philip Beineke, Trevor Hastie, and Shivakumar Vaithyanathan. 2004. The sentimental factor: Improving review classification via human-provided information. In *Proceedings of the Association for Computational Linguistics (ACL-2004)*.
- Jaime G. Carbonell. 1978. POLITICS: Automated ideological reasoning. *Cognitive Science*, 2(1):27–51.
- Thomas M. Cover and Joy A. Thomas. 1991. *Elements of Information Theory*. Wiley-Interscience.
- Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th International World Wide Web Conference (WWW2003)*.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Brett Kessler, Geoffrey Nunberg, and Hinrich Schütze. 1997. Automatic detection of text genre. In *Proceedings of the 35th Conference on Association for Computational Linguistics*, pages 32–38.
- S. Kullback and R. A. Leibler. 1951. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, March.
- David D. Lewis. 1998. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of the 9th European Conference on Machine Learning (ECML)*.
- Wei-Hao Lin, Theresa Wilson, Janyce Wiebe, and Alexander Hauptmann. 2006. Which side are you on? identifying perspectives at the document and sentence levels. In *Proceedings of Tenth Conference on Natural Language Learning (CoNLL)*.
- S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima. 2002. Mining product reputations on the web. In *Proceedings of the 2002 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Tony Mullen and Nigel Collier. 2004. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*.
- T. Nasukawa and J. Yi. 2003. Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd International Conference on Knowledge Capture (K-CAP 2003)*.
- Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the Association for Computational Linguistics (ACL-2004)*.
- Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*.
- Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*.
- Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the 7th Conference on Natural Language Learning (CoNLL-2003)*.
- B. D. Ripley. 1987. *Stochastic Simulation*. Wiley.
- Roger C. Schank and Robert P. Abelson. 1977. *Scripts, plans, goals, and understanding: an inquiry into human knowledge structures*. Lawrence Erlbaum Associates.
- Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March.
- Peter Turney and Michael L. Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, 21(4):315–346.
- Peter Verdonk. 2002. *Stylistics*. Oxford University Press.
- Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. 2004. Learning subjective language. *Computational Linguistics*, 30(3).
- Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*.

Word Sense and Subjectivity

Janyce Wiebe

Department of Computer Science
University of Pittsburgh
wiebe@cs.pitt.edu

Rada Mihalcea

Department of Computer Science
University of North Texas
rada@cs.unt.edu

Abstract

Subjectivity and meaning are both important properties of language. This paper explores their interaction, and brings empirical evidence in support of the hypotheses that (1) subjectivity is a property that can be associated with word senses, and (2) word sense disambiguation can directly benefit from subjectivity annotations.

1 Introduction

There is growing interest in the automatic extraction of opinions, emotions, and sentiments in text (*subjectivity*), to provide tools and support for various NLP applications. Similarly, there is continuous interest in the task of *word sense disambiguation*, with sense-annotated resources being developed for many languages, and a growing number of research groups participating in large-scale evaluations such as SENSEVAL.

Though both of these areas are concerned with the *semantics* of a text, over time there has been little interaction, if any, between them. In this paper, we address this gap, and explore possible interactions between subjectivity and word sense.

There are several benefits that would motivate such a joint exploration. First, at the resource level, the augmentation of lexical resources such as WordNet (Miller, 1995) with subjectivity labels could support better subjectivity analysis tools, and principled methods for refining word senses and clustering similar meanings. Second, at the tool level, an explicit link between subjectivity and word sense could help improve methods for each, by integrating features learned from one into the other in a pipeline approach, or through joint simultaneous learning.

In this paper we address two questions about word sense and subjectivity. First, can subjectivity labels be assigned to word senses? To address this question, we perform two studies. The first (Section 3) investigates agreement between annotators who manually assign the labels *subjective*, *objective*, or *both* to WordNet senses. The second study (Section 4) evaluates a method for automatic assignment of subjectivity labels to word senses. We devise an algorithm relying on distributionally similar words to calculate a subjectivity score, and show how it can be used to automatically assess the subjectivity of a word sense.

Second, can automatic subjectivity analysis be used to improve word sense disambiguation? To address this question, the output of a subjectivity sentence classifier is input to a word-sense disambiguation system, which is in turn evaluated on the nouns from the SENSEVAL-3 English lexical sample task (Section 5). The results of this experiment show that a subjectivity feature can significantly improve the accuracy of a word sense disambiguation system for those words that have both subjective and objective senses.

A third obvious question is, can word sense disambiguation help automatic subjectivity analysis? However, due to space limitations, we do not address this question here, but rather leave it for future work.

2 Background

Subjective expressions are words and phrases being used to express opinions, emotions, evaluations, speculations, etc. (Wiebe et al., 2005). A general covering term for such states is *private state*, “a state that is not open to objective obser-

vation or verification” (Quirk et al., 1985).¹ There are three main types of subjective expressions:²

(1) references to private states:

His **alarm** grew.
He **absorbed** the information quickly.
He was **boiling with anger**.

(2) references to speech (or writing) events expressing private states:

UCC/Disciples leaders **roundly condemned** the Iranian President’s **verbal assault** on Israel.

The editors of the left-leaning paper **attacked** the new House Speaker.

(3) expressive subjective elements:

He would be **quite a catch**.
What’s the catch?
That doctor is a **quack**.

Work on automatic subjectivity analysis falls into three main areas. The first is identifying words and phrases that are associated with subjectivity, for example, that *think* is associated with private states and that *beautiful* is associated with positive sentiments (e.g., (Hatzivassiloglou and McKeown, 1997; Wiebe, 2000; Kamps and Marx, 2002; Turney, 2002; Esuli and Sebastiani, 2005)). Such judgments are made for words. In contrast, our end task (in Section 4) is to assign subjectivity labels to word *senses*.

The second is subjectivity classification of sentences, clauses, phrases, or word instances in the context of a particular text or conversation, either subjective/objective classifications or positive/negative sentiment classifications (e.g., (Riloff and Wiebe, 2003; Yu and Hatzivassiloglou, 2003; Dave et al., 2003; Hu and Liu, 2004)).

The third exploits automatic subjectivity analysis in applications such as review classification (e.g., (Turney, 2002; Pang and Lee, 2004)), mining texts for product reviews (e.g., (Yi et al., 2003; Hu and Liu, 2004; Popescu and Etzioni, 2005)), summarization (e.g., (Kim and Hovy, 2004)), information extraction (e.g., (Riloff et al., 2005)),

and question answering (e.g., (Yu and Hatzivassiloglou, 2003; Stoyanov et al., 2005)).

Most manual subjectivity annotation research has focused on annotating words, out of context (e.g., (Heise, 2001)), or sentences and phrases in the context of a text or conversation (e.g., (Wiebe et al., 2005)). The new annotations in this paper are instead targeting the annotation of word *senses*.

3 Human Judgment of Word Sense Subjectivity

To explore our hypothesis that subjectivity may be associated with word senses, we developed a manual annotation scheme for assigning subjectivity labels to WordNet senses,³ and performed an inter-annotator agreement study to assess its reliability. Senses are classified as *S(ubjective)*, *O(bjective)*, or *B(oth)*. Classifying a sense as *S* means that, when the sense is used in a text or conversation, we expect it to express subjectivity; we also expect the phrase or sentence containing it to be subjective.

We saw a number of subjective expressions in Section 2. A subset is repeated here, along with relevant WordNet senses. In the display of each sense, the first part shows the synset, gloss, and any examples. The second part (marked with =>) shows the immediate hypernym.

His **alarm** grew.
alarm, dismay, consternation – (fear resulting from the awareness of danger)

=> fear, fearfulness, fright – (an emotion experienced in anticipation of some specific pain or danger (usually accompanied by a desire to flee or fight))

He was **boiling** with anger.
seethe, boil – (be in an agitated emotional state; “The customer was seething with anger”)

=> be – (have the quality of being; (copula, used with an adjective or a predicate noun); “John is rich”; “This is not a good answer”)

What’s the **catch**?
catch – (a hidden drawback; “it sounds good but what’s the catch?”)

=> drawback – (the quality of being a hindrance; “he pointed out all the drawbacks to my plan”)

That doctor is a **quack**.
quack – (an untrained person who pretends to be a physician and who dispenses medical advice)

=> doctor, doc, physician, MD, Dr., medico

Before specifying what we mean by an objective sense, we give examples.

¹Note that *sentiment*, the focus of much recent work in the area, is a type of subjectivity, specifically involving positive or negative opinion, emotion, or evaluation.

²These distinctions are not strictly needed for this paper, but may help the reader appreciate the examples given below.

³All our examples and data used in the experiments are from WordNet 2.0.

The **alarm** went off.

alarm, warning device, alarm system – (a device that signals the occurrence of some undesirable event)

=> device – (an instrumentality invented for a particular purpose; “the device is small enough to wear on your wrist”; “a device intended to conserve water”)

The water **boiled**.

boil – (come to the boiling point and change from a liquid to vapor; “Water boils at 100 degrees Celsius”)

=> change state, turn – (undergo a transformation or a change of position or action; “We turned from Socialism to Capitalism”; “The people turned against the President when he stole the election”)

He sold his **catch** at the market.

catch, haul – (the quantity that was caught; “the catch was only 10 fish”)

=> indefinite quantity – (an estimated quantity)

The duck’s **quack** was loud and brief.

quack – (the harsh sound of a duck)

=> sound – (the sudden occurrence of an audible event; “the sound awakened them”)

While we expect phrases or sentences containing subjective senses to be subjective, we do not necessarily expect phrases or sentences containing objective senses to be objective. Consider the following examples:

Will someone shut that damn **alarm** off?
Can’t you even **boil** water?

While these sentences contain objective senses of *alarm* and *boil*, the sentences are subjective nonetheless. But they are not subjective due to *alarm* and *boil*, but rather to punctuation, sentence forms, and other words in the sentence. Thus, classifying a sense as *O* means that, when the sense is used in a text or conversation, we do not expect it to express subjectivity and, if the phrase or sentence containing it is subjective, the subjectivity is due to something else.

Finally, classifying a sense as *B* means it covers both subjective and objective usages, e.g.:

absorb, suck, imbibe, soak up, sop up, suck up, draw, take in, take up – (take in, also metaphorically; “The sponge absorbs water well”; “She drew strength from the minister’s words”)

Manual subjectivity judgments were added to a total of 354 senses (64 words). One annotator, Judge 1 (a co-author), tagged all of them. A second annotator (Judge 2, who is not a co-author) tagged a subset for an agreement study, presented next.

3.1 Agreement Study

For the agreement study, Judges 1 and 2 independently annotated 32 words (138 senses). 16 words

have both *S* and *O* senses and 16 do not (according to Judge 1). Among the 16 that do not have both *S* and *O* senses, 8 have only *S* senses and 8 have only *O* senses. All of the subsets are balanced between nouns and verbs. Table 1 shows the contingency table for the two annotators’ judgments on this data. In addition to *S*, *O*, and *B*, the annotation scheme also permits *U(ncertain)* tags.

	S	O	B	U	Total
S	39	0	0	4	43
O	3	73	2	4	82
B	1	0	3	1	5
U	3	2	0	3	8
Total	46	75	5	12	138

Table 1: Agreement on balanced set (Agreement: 85.5%, κ : 0.74)

Overall agreement is 85.5%, with a Kappa (κ) value of 0.74. For 12.3% of the senses, at least one annotator’s tag is *U*. If we consider these cases to be borderline and exclude them from the study, percent agreement increases to 95% and κ rises to 0.90. Thus, annotator agreement is especially high when both are certain.

Considering only the 16-word subset with both *S* and *O* senses (according to Judge 1), κ is .75, and for the 16-word subset for which Judge 1 gave only *S* or only *O* senses, κ is .73. Thus, the two subsets are of comparable difficulty.

The two annotators also independently annotated the 20 ambiguous nouns (117 senses) of the SENSEVAL-3 English lexical sample task used in Section 5. For this tagging task, *U* tags were not allowed, to create a definitive gold standard for the experiments. Even so, the κ value for them is 0.71, which is not substantially lower. The distributions of Judge 1’s tags for all 20 words can be found in Table 3 below.

We conclude this section with examples of disagreements that illustrate sources of uncertainty. First, uncertainty arises when subjective senses are missing from the dictionary. The labels for the senses of noun *assault* are (*O:O,O:O,O:O,O:UO*).⁴ For verb *assault* there is a subjective sense:

attack, round, assail, lash out, snipe, assault (attack in speech or writing) “The editors of the left-leaning paper attacked the new House Speaker”

However, there is no corresponding sense for

⁴I.e., the first three were labeled *O* by both annotators. For the fourth sense, the second annotator was not sure but was leaning toward *O*.

noun *assault*. A missing sense may lead an annotator to try to see subjectivity in an objective sense.

Second, uncertainty can arise in weighing hypernym against sense. It is fine for a synset to imply just *S* or *O*, while the hypernym implies both (the synset specializes the more general concept). However, consider the following, which was tagged (*O:UB*).

attack – (a sudden occurrence of an uncontrollable condition; “an attack of diarrhea”)

=> affliction – (a cause of great suffering and distress)

While the sense is only about the condition, the hypernym highlights subjective reactions to the condition. One annotator judged only the sense (giving tag *O*), while the second considered the hypernym as well (giving tag *UB*).

4 Automatic Assessment of Word Sense Subjectivity

Encouraged by the results of the agreement study, we devised a method targeting the automatic annotation of word senses for subjectivity.

The main idea behind our method is that we can derive information about a word sense based on information drawn from words that are distributionally similar to the given word sense. This idea relates to the unsupervised word sense ranking algorithm described in (McCarthy et al., 2004). Note, however, that (McCarthy et al., 2004) used the information about distributionally similar words to approximate corpus frequencies for word senses, whereas we target the estimation of a *property* of a given word sense (the “subjectivity”).

Starting with a given ambiguous word w , we first find the distributionally similar words using the method of (Lin, 1998) applied to the automatically parsed texts of the British National Corpus. Let $DSW = dsw_1, dsw_2, \dots, dsw_n$ be the list of top-ranked distributionally similar words, sorted in decreasing order of their similarity.

Next, for each sense ws_i of the word w , we determine the similarity with each of the words in the list DSW , using a WordNet-based measure of semantic similarity ($wnss$). Although a large number of such word-to-word similarity measures exist, we chose to use the (Jiang and Conrath, 1997) measure, since it was found both to be efficient and to provide the best results in previous experiments involving word sense ranking (McCarthy et al., 2004)⁵. For distributionally similar words

⁵Note that unlike the above measure of *distributional* sim-

Algorithm 1 Word Sense Subjectivity Score

Input: Word sense w_i

Input: Distributionally similar words $DSW = \{dsw_j | j = 1..n\}$

Output: Subjectivity score $\text{subj}(w_i)$

```

1:  $\text{subj}(w_i) = 0$ 
2:  $\text{total}_{sim} = 0$ 
3: for  $j = 1$  to  $n$  do
4:    $Insts_j =$  all instances of  $dsw_j$  in the MPQA corpus
5:   for  $k$  in  $Insts_j$  do
6:     if  $k$  is in a subj. expr. in MPQA corpus then
7:        $\text{subj}(w_i) += \text{sim}(w_i, dsw_j)$ 
8:     else if  $k$  is not in a subj. expr. in MPQA corpus then
9:        $\text{subj}(w_i) -= \text{sim}(w_i, dsw_j)$ 
10:    end if
11:     $\text{total}_{sim} += \text{sim}(w_i, dsw_j)$ 
12:  end for
13: end for
14:  $\text{subj}(w_i) = \text{subj}(w_i) / \text{total}_{sim}$ 

```

that are themselves ambiguous, we use the sense that maximizes the similarity score. The similarity scores associated with each word dsw_j are normalized so that they add up to one across all possible senses of w , which results in a score described by the following formula:

$$\text{sim}(ws_i, dsw_j) = \frac{wnss(ws_i, dsw_j)}{\sum_{i' \in \text{senses}(w)} wnss(ws_{i'}, dsw_j)}$$

where

$$wnss(ws_i, dsw_j) = \max_{k \in \text{senses}(dsw_j)} wnss(ws_i, dsw_j^k)$$

A selection process can also be applied so that a distributionally similar word belongs only to one sense. In this case, for a given sense w_i we use only those distributionally similar words with whom w_i has the highest similarity score across all the senses of w . We refer to this case as *similarity-selected*, as opposed to *similarity-all*, which refers to the use of all distributionally similar words for all senses.

Once we have a list of similar words associated with each sense ws_i and the corresponding similarity scores $\text{sim}(ws_i, dsw_j)$, we use an annotated corpus to assign subjectivity scores to the senses. The corpus we use is the MPQA Opinion Corpus, which consists of over 10,000 sentences from the world press annotated for subjective expressions (all three types of subjective expressions described in Section 2).⁶

ilarity which measures similarity between words, rather than word senses, here we needed a similarity measure that also takes into account word senses as defined in a sense inventory such as WordNet.

⁶The MPQA corpus is described in (Wiebe et al., 2005) and available at www.cs.pitt.edu/mpqa/database/release/.

Algorithm 1 is our method for calculating sense subjectivity scores. The subjectivity score is a value in the interval $[-1, +1]$ with $+1$ corresponding to highly subjective and -1 corresponding to highly objective. It is a sum of sim scores, where $sim(w_i, dsw_j)$ is added for each instance of dsw_j that is in a subjective expression, and subtracted for each instance that is not in a subjective expression.

Note that the annotations in the MPQA corpus are for subjective expressions in context. Thus, the data is somewhat noisy for our task, because, as discussed in Section 3, objective senses may appear in subjective expressions. Nonetheless, we hypothesized that subjective senses tend to appear more often in subjective expressions than objective senses do, and use the appearance of words in subjective expressions as evidence of sense subjectivity.

(Wiebe, 2000) also makes use of an annotated corpus, but in a different approach: given a word w and a set of distributionally similar words DSW , that method assigns a subjectivity score to w equal to the conditional probability that any member of DSW is in a subjective expression. Moreover, the end task of that work was to annotate words, while our end task is the more difficult problem of annotating word senses for subjectivity.

4.1 Evaluation

The evaluation of the algorithm is performed against the gold standard of 64 words (354 word senses) using Judge 1’s annotations, as described in Section 3.

For each sense of each word in the set of 64 ambiguous words, we use Algorithm 1 to determine a subjectivity score. A subjectivity label is then assigned depending on the value of this score with respect to a pre-selected threshold. While a threshold of 0 seems like a sensible choice, we perform the evaluation for different thresholds ranging across the $[-1, +1]$ interval, and correspondingly determine the precision of the algorithm at different points of recall⁷. Note that the word senses for which none of the distributionally similar words are found in the MPQA corpus are not

⁷Specifically, in the list of word senses ranked by their subjectivity score, we assign a subjectivity label to the top N word senses. The precision is then determined as the number of correct subjectivity label assignments out of all N assignments, while the recall is measured as the correct subjective senses out of all the subjective senses in the gold standard data set. By varying the value of N from 1 to the total number of senses in the corpus, we can derive precision and recall curves.

included in this evaluation (excluding 82 senses), since in this case a subjectivity score cannot be calculated. The evaluation is therefore performed on a total of 272 word senses.

As a baseline, we use an “informed” random assignment of subjectivity labels, which randomly assigns S labels to word senses in the data set, such that the maximum number of S assignments equals the number of correct S labels in the gold standard data set. This baseline guarantees a maximum recall of 1 (which under true random conditions might not be achievable). Correspondingly, given the controlled distribution of S labels across the data set in the baseline setting, the precision is equal for all eleven recall points, and is determined as the total number of correct subjective assignments divided by the size of the data set⁸.

Algorithm	Number of DSW	Break-even point
similarity-all	100	0.41
similarity-selected	100	0.50
similarity-all	160	0.43
similarity-selected	160	0.50
baseline	-	0.27

Table 2: Break-even point for different algorithm and parameter settings

There are two aspects of the sense subjectivity scoring algorithm that can influence the label assignment, and correspondingly their evaluation.

First, as indicated above, after calculating the semantic similarity of the distributionally similar words with each sense, we can either use *all* the distributionally similar words for the calculation of the subjectivity score of each sense (*similarity-all*), or we can use only those that lead to the highest similarity (*similarity-selected*). Interestingly, this aspect can drastically affect the algorithm accuracy. The setting where a distributionally similar word can belong only to one sense significantly improves the algorithm performance. Figure 1 plots the interpolated precision for eleven points of recall, for *similarity-all*, *similarity-selected*, and *baseline*. As shown in this figure, the precision-recall curves for our algorithm are clearly above the “informed” baseline, indicating the ability of our algorithm to automatically identify subjective word senses.

Second, the number of distributionally similar words considered in the first stage of the algorithm can vary, and might therefore influence the

⁸In other words, this fraction represents the probability of making the correct subjective label assignment by chance.

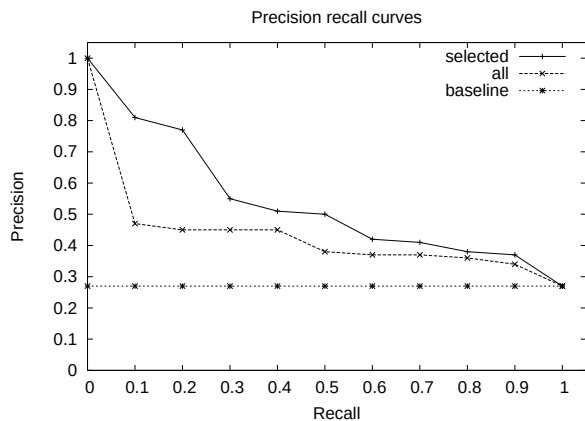


Figure 1: Precision and recall for automatic subjectivity annotations of word senses (DSW=160).

output of the algorithm. We experiment with two different values, namely 100 and 160 top-ranked distributionally similar words. Table 2 shows the break-even points for the four different settings that were evaluated,⁹ with results that are almost double compared to the informed baseline. As it turns out, for weaker versions of the algorithm (i.e., *similarity-all*), the size of the set of distributionally similar words can significantly impact the performance of the algorithm. However, for the already improved *similarity-selected* algorithm version, this parameter does not seem to have influence, as similar results are obtained regardless of the number of distributionally similar words. This is in agreement with the finding of (McCarthy et al., 2004) that, in their word sense ranking method, a larger set of neighbors did not influence the algorithm accuracy.

5 Automatic Subjectivity Annotations for Word Sense Disambiguation

The final question we address is concerned with the potential impact of subjectivity on the quality of a word sense classifier. To answer this question, we augment an existing data-driven word sense disambiguation system with a feature reflecting the subjectivity of the examples where the ambiguous word occurs, and evaluate the performance of the new subjectivity-aware classifier as compared to the traditional context-based sense classifier.

We use a word sense disambiguation system that integrates both local and topical features.

⁹The break-even point (Lewis, 1992) is a standard measure used in conjunction with precision-recall evaluations. It represents the value where precision and recall become equal.

Specifically, we use the current word and its part-of-speech, a local context of three words to the left and right of the ambiguous word, the parts-of-speech of the surrounding words, and a global context implemented through sense-specific keywords determined as a list of at most five words occurring at least three times in the contexts defining a certain word sense. This feature set is similar to the one used by (Ng and Lee, 1996), as well as by a number of SENSEVAL systems. The parameters for sense-specific keyword selection were determined through cross-fold validation on the training set. The features are integrated in a Naive Bayes classifier, which was selected mainly for its performance in previous work showing that it can lead to a state-of-the-art disambiguation system given the features we consider (Lee and Ng, 2002).

The experiments are performed on the set of ambiguous nouns from the SENSEVAL-3 English lexical sample evaluation (Mihalcea et al., 2004). We use the rule-based subjective sentence classifier of (Riloff and Wiebe, 2003) to assign an *S*, *O*, or *B* label to all the training and test examples pertaining to these ambiguous words. This subjectivity annotation tool targets sentences, rather than words or paragraphs, and therefore the tool is fed with sentences. We also include a surrounding context of two additional sentences, because the classifier considers some contextual information.

Our hypothesis motivating the use of a sentence-level subjectivity classifier is that instances of subjective *senses* are more likely to be in subjective *sentences*, and thus that sentence subjectivity is an informative feature for the disambiguation of words having both subjective and objective senses.

For each ambiguous word, we perform two separate runs: one using the basic disambiguation system described earlier, and another using the subjectivity-aware system that includes the additional subjectivity feature. Table 3 shows the results obtained for these 20 nouns, including word sense disambiguation accuracy for the two different systems, the most frequent sense baseline, and the subjectivity/objectivity split among the word senses (according to Judge 1). The words in the top half of the table are the ones that have both *S* and *O* senses, and those in the bottom are the ones that do not. If we were to use Judge 2’s tags instead of Judge 1’s, only one word would change: *source* would move from the top to the bottom of the table.

Word	Senses	Sense subjectivity	Data		Baseline	Classifier	
			train	test		basic	+ subj.
Words with subjective senses							
argument	5	3-S 2-O	221	111	49.4%	51.4%	54.1%
atmosphere	6	2-S 4-O	161	81	65.4%	65.4%	66.7%
difference	5	2-S 3-O	226	114	40.4%	54.4%	57.0%
difficulty	4	2-S 2-O	46	23	17.4%	47.8%	52.2%
image	7	2-S 5-O	146	74	36.5%	41.2%	43.2%
interest	7	1-S 5-O 1-B	185	93	41.9%	67.7%	68.8%
judgment	7	5-S 2-O	62	32	28.1%	40.6%	43.8%
plan	3	1-S 2-O	166	84	81.0%	81.0%	81.0%
sort	4	1-S 2-O 1-B	190	96	65.6%	66.7%	67.7%
source	9	1-S 8-O	64	32	40.6%	40.6%	40.6%
Average					46.6%	55.6%	57.5%
Words with no subjective senses							
arm	6	6-O	266	133	82.0%	85.0%	84.2%
audience	4	4-O	200	100	67.0%	74.0%	74.0%
bank	10	10-O	262	132	62.6%	62.6%	62.6%
degree	7	5-O 2-B	256	128	60.9%	71.1%	71.1%
disc	4	4-O	200	100	38.0%	65.6%	66.4%
organization	7	7-O	112	56	64.3%	64.3%	64.3%
paper	7	7-O	232	117	25.6%	49.6%	48.0%
party	5	5-O	230	116	62.1%	62.9%	62.9%
performance	5	5-O	172	87	26.4%	34.5%	34.5%
shelter	5	5-O	196	98	44.9%	65.3%	65.3%
Average					53.3%	63.5%	63.3%
Average for all words					50.0%	59.5%	60.4%

Table 3: Word Sense Disambiguation with and without subjectivity information, for the set of ambiguous nouns in SENSEVAL-3

For the words that have both *S* and *O* senses, the addition of the subjectivity feature alone can bring a significant error rate reduction of 4.3% ($p < 0.05$ paired t-test). Interestingly, no improvements are observed for the words with no subjective senses; on the contrary, the addition of the subjectivity feature results in a small degradation. Overall for the entire set of ambiguous words, the error reduction is measured at 2.2% (significant at $p < 0.1$ paired t-test).

In almost all cases, the words with both *S* and *O* senses show improvement, while the others show small degradation or no change. This suggests that if a subjectivity label is available for the words in a lexical resource (e.g. using Algorithm 1 from Section 4), such information can be used to decide on using a subjectivity-aware system, thereby improving disambiguation accuracy.

One of the exceptions is *disc*, which had a small benefit, despite not having any subjective senses. As it happens, the first sense of *disc* is *phonograph record*.

phonograph record, phonograph recording, record, disk, disc, platter – (sound recording consisting of a disc with continuous grooves; formerly used to reproduce music by rotating while a phonograph needle tracked in the grooves)

The improvement can be explained by observing that many of the training and test sentences containing this sense are labeled subjective by the classifier, and indeed this sense frequently occurs in subjective sentences such as “This is anyway a stunning **disc**.”

Another exception is the noun *plan*, which did not benefit from the subjectivity feature, although it does have a subjective sense. This can perhaps be explained by the data set for this word, which seems to be particularly difficult, as the basic classifier itself could not improve over the most frequent sense baseline.

The other word that did not benefit from the subjectivity feature is the noun *source*, for which its only subjective sense did not appear in the sense-annotated data, leading therefore to an “objective only” set of examples.

6 Conclusion and Future Work

The questions posed in the introduction concerning the possible interaction between subjectivity and word sense found answers throughout the paper. As it turns out, a correlation can indeed be established between these two semantic properties of language.

Addressing the first question of whether subjectivity is a property that can be assigned to word senses, we showed that good agreement ($\kappa=0.74$) can be achieved between human annotators labeling the subjectivity of senses. When uncertain cases are removed, the κ value is even higher (0.90). Moreover, the automatic subjectivity scoring mechanism that we devised was able to successfully assign subjectivity labels to senses, significantly outperforming an “informed” baseline associated with the task. While much work remains to be done, this first attempt has proved the feasibility of correctly assigning subjectivity labels to the fine-grained level of word senses.

The second question was also positively answered: the quality of a word sense disambiguation system can be improved with the addition of subjectivity information. Section 5 provided evidence that automatic subjectivity classification may improve word sense disambiguation performance, but mainly for words with both subjective and objective senses. As we saw, performance may even degrade for words that do not. Tying the pieces of this paper together, once the senses in a dictionary have been assigned subjectivity labels, a word sense disambiguation system could consult them to decide whether it should consider or ignore the subjectivity feature.

There are several other ways our results could impact future work. Subjectivity labels would be a useful source of information when manually augmenting the lexical knowledge in a dictionary,

e.g., when choosing hypernyms for senses or deciding which senses to eliminate when defining a coarse-grained sense inventory (if there is a subjective sense, at least one should be retained).

Adding subjectivity labels to WordNet could also support automatic subjectivity analysis. First, the input corpus could be sense tagged and the subjectivity labels of the assigned senses could be exploited by a subjectivity recognition tool. Second, a number of methods for subjectivity or sentiment analysis start with a set of seed words and then search through WordNet to find other subjective words (Kamps and Marx, 2002; Yu and Hatzivassiloglou, 2003; Hu and Liu, 2004; Kim and Hovy, 2004; Esuli and Sebastiani, 2005). However, such searches may veer off course down objective paths. The subjectivity labels assigned to senses could be consulted to keep the search traveling along subjective paths.

Finally, there could be different strategies for exploiting subjectivity annotations and word sense. While the current setting considered a pipeline approach, where the output of a subjectivity annotation system was fed to the input of a method for semantic disambiguation, future work could also consider the role of word senses as a possible way of improving subjectivity analysis, or simultaneous annotations of subjectivity and word meanings, as done in the past for other language processing problems.

Acknowledgments We would like to thank Theresa Wilson for annotating senses, and the anonymous reviewers for their helpful comments. This work was partially supported by ARDA AQUAINT and by the NSF (award IIS-0208798).

References

- K. Dave, S. Lawrence, and D. Pennock. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proc. WWW-2003*, Budapest, Hungary. Available at <http://www2003.org>.
- A. Esuli and F. Sebastiani. 2005. Determining the semantic orientation of terms through gloss analysis. In *Proc. CIKM-2005*.
- V. Hatzivassiloglou and K. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proc. ACL-97*, pages 174–181.
- D. Heise. 2001. Project magellan: Collecting cross-cultural affective meanings via the internet. *Electronic Journal of Sociology*, 5(3).
- M. Hu and B. Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of ACM SIGKDD*.
- J. Jiang and D. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, Taiwan.
- J. Kamps and M. Marx. 2002. Words with attitude. In *Proc. 1st International WordNet Conference*.
- S.M. Kim and E. Hovy. 2004. Determining the sentiment of opinions. In *Proc. Coling 2004*.
- Y.K. Lee and H.T. Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *Proc. EMNLP 2002*.
- D. Lewis. 1992. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of ACM SIGIR*.
- D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL*, Montreal, Canada.
- D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant senses in untagged text. In *Proc. ACL 2004*.
- R. Mihalcea, T. Chklovski, and A. Kilgariff. 2004. The Senseval-3 English lexical sample task. In *Proc. ACL/SIGLEX Senseval-3*.
- G. Miller. 1995. Wordnet: A lexical database. *Communication of the ACM*, 38(11):39–41.
- H.T. Ng and H.B. Lee. 1996. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In *Proc. ACL 1996*.
- B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proc. ACL 2004*.
- A. Popescu and O. Etzioni. 2005. Extracting product features and opinions from reviews. In *Proc. of HLT/EMNLP 2005*.
- R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman, New York.
- E. Riloff and J. Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proc. EMNLP 2003*.
- E. Riloff, J. Wiebe, and W. Phillips. 2005. Exploiting subjectivity classification to improve information extraction. In *Proc. AAAI 2005*.
- V. Stoyanov, C. Cardie, and J. Wiebe. 2005. Multi-perspective question answering using the opqa corpus. In *Proc. HLT/EMNLP 2005*.
- P. Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proc. ACL 2002*.
- J. Wiebe, T. Wilson, and C. Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 1(2).
- J. Wiebe. 2000. Learning subjective adjectives from corpora. In *Proc. AAAI 2000*.
- J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack. 2003. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proc. ICDM 2003*.
- H. Yu and V. Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proc. EMNLP 2003*.

Improving QA Accuracy by Question Inversion

John Prager

IBM T.J. Watson Res. Ctr.

Yorktown Heights

N.Y. 10598

jprager@us.ibm.com

Pablo Duboue

IBM T.J. Watson Res. Ctr.

Yorktown Heights

N.Y. 10598

duboue@us.ibm.com

Jennifer Chu-Carroll

IBM T.J. Watson Res. Ctr.

Yorktown Heights

N.Y. 10598

jenc@us.ibm.com

Abstract

This paper demonstrates a conceptually simple but effective method of increasing the accuracy of QA systems on factoid-style questions. We define the notion of an inverted question, and show that by requiring that the answers to the original and inverted questions be mutually consistent, incorrect answers get demoted in confidence and correct ones promoted. Additionally, we show that lack of validation can be used to assert no-answer (**nil**) conditions. We demonstrate increases of performance on TREC and other question-sets, and discuss the kinds of future activities that can be particularly beneficial to approaches such as ours.

1 Introduction

Most QA systems nowadays consist of the following standard modules: QUESTION PROCESSING, to determine the bag of words for a query and the desired answer type (the type of the entity that will be offered as a candidate answer); SEARCH, which will use the query to extract a set of documents or passages from a corpus; and ANSWER SELECTION, which will analyze the returned documents or passages for instances of the answer type in the most favorable contexts. Each of these components implements a set of heuristics or hypotheses, as devised by their authors (cf. Clarke et al. 2001, Chu-Carroll et al. 2003).

When we perform failure analysis on questions incorrectly answered by our system, we find that there are broadly speaking two kinds of failure. There are errors (we might call them *bugs*) on the implementation of the said heuristics: errors in tagging, parsing, named-entity recognition; omissions in synonym lists; missing patterns, and just plain programming errors. This class can be characterized by being fixable by identifying incorrect code and fixing it, or adding more items, either explicitly or through training. The other class of errors (what we might call *unlucky*) are at the boundaries of the heuristics;

situations where the system did not do anything “wrong,” in the sense of *bug*, but circumstances conspired against finding the correct answer.

Usually when *unlucky* errors occur, the system generates a reasonable query and an appropriate answer type, and at least one passage containing the right answer is returned. However, there may be returned passages that have a larger number of query terms and an incorrect answer of the right type, or the query terms might just be physically closer to the incorrect answer than to the correct one. ANSWER SELECTION modules typically work either by trying to prove the answer is correct (Moldovan & Rus, 2001) or by giving them a weight produced by summing a collection of heuristic features (Radev et al., 2000); in the latter case candidates having a larger number of matching query terms, even if they do not exactly match the context in the question, might generate a larger score than a correct passage with fewer matching terms.

To be sure, *unlucky* errors are usually *bugs* when considered from the standpoint of a system with a more sophisticated heuristic, but any system at any point in time will have limits on what it tries to do; therefore the distinction is not absolute but is relative to a heuristic and system.

It has been argued (Prager, 2002) that the success of a QA system is proportional to the impedance match between the question and the knowledge sources available. We argue here similarly. Moreover, we believe that this is true not only in terms of the correct answer, but the distracters,¹ or incorrect answers too. In QA, an *unlucky* incorrect answer is not usually predictable in advance; it occurs because of a coincidence of terms and syntactic contexts that cause it to be preferred over the correct answer. It has no connection with the correct answer and is only returned because its enclosing passage so happens to exist in the same corpus as the correct answer context. This would lead us to believe that if a

¹ We borrow the term from multiple-choice test design.

different corpus containing the correct answer were to be processed, while there would be no guarantee that the correct answer would be found, it would be unlikely (i.e. *very unlucky*) if the same incorrect answer as before were returned.

We have demonstrated elsewhere (Prager et al. 2004b) how using multiple corpora can improve QA performance, but in this paper we achieve similar goals without using additional corpora. We note that factoid questions are usually about relations between entities, e.g. “What is the capital of France?”, where one of the arguments of the relationship is sought and the others given. We can *invert* the question by substituting the candidate answer back into the question, while making one of the given entities the so-called wh-word, thus “Of what country is Paris the capital?” We hypothesize that asking this question (and those formed from other candidate answers) will locate a largely different set of passages in the corpus than the first time around. As will be explained in Section 3, this can be used to decrease the confidence in the incorrect answers, and also increase it for the correct answer, so that the latter becomes the answer the system ultimately proposes.

This work is part of a continuing program of demonstrating how meta-heuristics, using what might be called “collateral” information, can be used to constrain or adjust the results of the primary QA system.

In the next Section we review related work. In Section 3 we describe our algorithm in detail, and in Section 4 present evaluation results. In Section 5 we discuss our conclusions and future work.

2 Related Work

Logic and inferencing have been a part of Question-Answering since its earliest days. The first such systems were natural-language interfaces to expert systems, e.g., SHRDLU (Winograd, 1972), or to databases, e.g., LIFER/LADDER (Hendrix et al. 1977). CHAT-80 (Warren & Pereira, 1982), for instance, was a DCG-based NL-query system about world geography, entirely in Prolog. In these systems, the NL question is transformed into a semantic form, which is then processed further. Their overall architecture and system operation is very different from today’s systems, however, primarily in that there was no text corpus to process.

Inferencing is a core requirement of systems that participate in the current PASCAL Recognizing Textual Entailment (RTE) challenge (see <http://www.pascal-network.org/Challenges/RTE> and <http://www.pascal-network.org/Challenges/RTE2>). It is also used in at least two of the more

visible end-to-end QA systems of the present day. The LCC system (Moldovan & Rus, 2001) uses a *Logic Prover* to establish the connection between a candidate answer passage and the question. Text terms are converted to logical forms, and the question is treated as a goal which is “proven”, with real-world knowledge being provided by Extended WordNet. The IBM system PIQUANT (Chu-Carroll et al., 2003) used Cyc (Lenat, 1995) in answer verification. Cyc can in some cases confirm or reject candidate answers based on its own store of instance information; in other cases, primarily of a numerical nature, Cyc can confirm whether candidates are within a reasonable range established for their subtype.

At a more abstract level, the use of inversions discussed in this paper can be viewed as simply an example of finding support (or lack of it) for candidate answers. Many current systems (see, e.g. (Clarke et al., 2001; Prager et al. 2004b)) employ redundancy as a significant feature of operation: if the same answer appears multiple times in an internal *top-n* list, whether from multiple sources or multiple algorithms/agents, it is given a confidence boost, which will affect whether and how it gets returned to the end-user.

The work here is a continuation of previous work described in (Prager et al. 2004a,b). In the former we demonstrated that for a certain kind of question, if the inverted question were given, we could improve the F-measure of accuracy on a question set by 75%. In this paper, by contrast, we do not manually provide the inverted question, and in the second evaluation presented here we do not restrict the question type.

3 Algorithm

3.1 System Architecture

A simplified block-diagram of our PIQUANT system is shown in Figure 1. The outer block on the left, QS1, is our basic QA system, in which the QUESTION PROCESSING (QP), SEARCH (S) and ANSWER SELECTION (AS) subcomponents are indicated. The outer block on the right, QS2, is another QA-System that is used to answer the inverted questions. In principle QS2 could be QS1 but parameterized differently, or even an entirely different system, but we use another instance of QS1, as-is. The block in the middle is our *Constraints Module* CM, which is the subject of this paper.

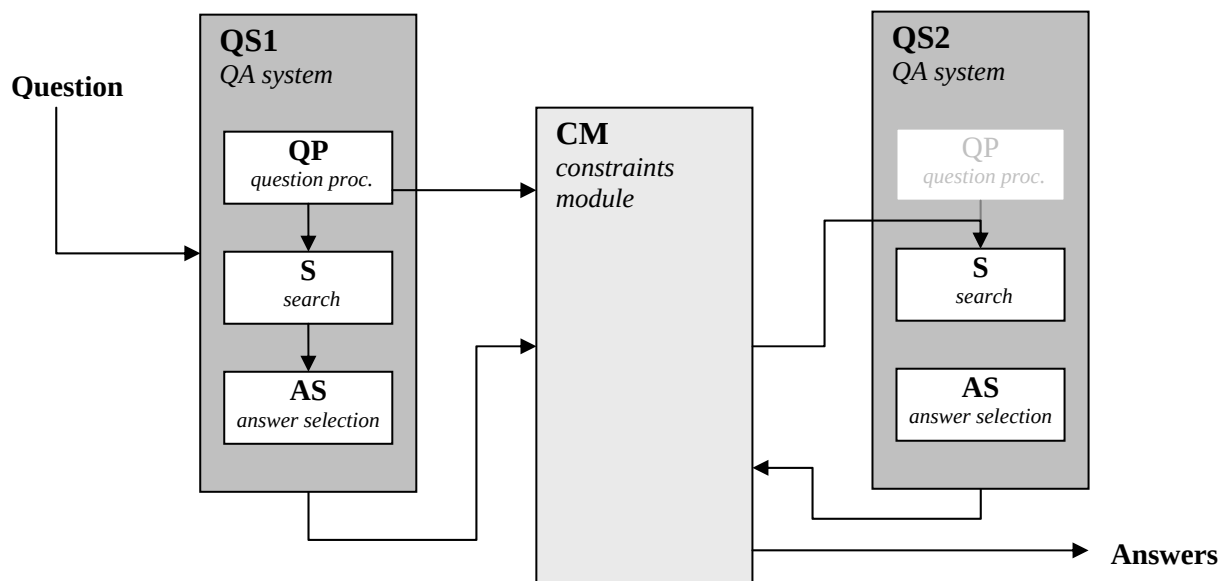


Figure 1. Constraints Architecture. QS1 and QS2 are (possibly identical) QA systems.

The Question Processing component of QS2 is not used in this context since CM simulates its output by modifying the output of QP in QS1, as described in Section 3.3.

3.2 Inverting Questions

Our open-domain QA system employs a named-entity recognizer that identifies about a hundred types. Any of these can be answer types, and there are corresponding sets of patterns in the QUESTION PROCESSING module to determine the answer type sought by any question. When we wish to invert a question, we must find an entity in the question whose type we recognize; this entity then becomes the sought answer for the inverted question. We call this entity the inverted or *pivot* term.

Thus for the question:

- (1) "What was the capital of Germany in 1985?"

Germany is identified as a term with a known type (COUNTRY). Then, given the candidate answer <CANDANS>, the inverted question becomes

- (2) "Of what country was <CANDANS> the capital in 1985?"

Some questions have more than one invertible term. Consider for example:

- (3) "Who was the 33rd president of the U.S.?"

This question has 3 inversion points:

- (4) "What number president of the U.S. was <CANDANS>?"
- (5) "Of what country was <CANDANS> the 33rd president?"

- (6) "<CANDANS> was the 33rd what of the U.S.?"

Having more than one possible inversion is in theory a benefit, since it gives more opportunity for enforcing consistency, but in our current implementation we just pick one for simplicity. We observe on training data that, in general, the smaller the number of unique instances of an answer type, the more likely it is that the inverted question will be correctly answered. We generated a set NELIST of the most frequently-occurring named-entity types in questions; this list is sorted in order of estimated cardinality.

It might seem that the question inversion process can be quite tricky and can generate possibly unnatural phrasings, which in turn can be difficult to reparse. However, the examples given above were simply English renditions of internal inverted structures – as we shall see the system does not need to use a natural language representation of the inverted questions.

Some questions are either not invertible, or, like "How did *X* die?" have an inverted form ("Who died of cancer?") with so many correct answers that we know our algorithm is unlikely to benefit us. However, as it is constituted it is unlikely to hurt us either, and since it is difficult to automatically identify such questions, we don't attempt to intercept them. As reported in (Prager et al. 2004a), an estimated 79% of the questions in TREC question sets can be inverted meaningfully. This places an upper limit on the gains to be achieved with our algorithm, but is high enough to be worth pursuing.

3.3 Inversion Algorithm

As shown in the previous section, not all questions have easily generated inverted forms (even by a human). However, we do not need to explicate the inverted form in natural language in order to process the inverted question.

In our system, a question is processed by the QUESTION PROCESSING module, which produces a structure called a *QFrame*, which is used by the subsequent SEARCH and ANSWER SELECTION modules. The *QFrame* contains the list of terms and phrases in the question, along with their properties, such as POS and NE-type (if it exists), and a list of syntactic relationship tuples. When we have a candidate answer in hand, we do not need to produce the inverted English question, but merely the *QFrame* that would have been generated from it. Figure 1 shows that the CONSTRAINTS MODULE takes the *QFrame* as one of its inputs, as shown by the link from QP in QS1 to CM. This *inverted QFrame* can be generated by a set of simple transformations, substituting the pivot term in the bag of words with a candidate answer $\langle \text{CANDANS} \rangle$, the original answer type with the type of the pivot term, and in the relationships the pivot term with its type and the original answer type with $\langle \text{CANDANS} \rangle$. When relationships are evaluated, a type token will match any instance of that type. Figure 2 shows a simplified view of the original *QFrame* for “What was the capital of Germany in 1945?”, and Figure 3 shows the corresponding Inverted *QFrame*. COUNTRY is determined to be a better type to invert than YEAR, so “Germany” becomes the pivot. In Figure 3, the token $\langle \text{CANDANS} \rangle$ might take in turn “Berlin”, “Moscow”, “Prague” etc.

Keywords: {1945, Germany, capital} AnswerType: CAPITAL Relationships: {(Germany, capital), (capital, CAPITAL), (capital, 1945)}

Figure 2. Simplified *QFrame*

Keywords: {1945, $\langle \text{CANDANS} \rangle$, capital} AnswerType: COUNTRY Relationships: {(COUNTRY, capital), (capital, $\langle \text{CANDANS} \rangle$), (capital, 1945)}

Figure 3. Simplified Inverted *QFrame*.

The output of QS2 after processing the inverted *QFrame* is a list of answers to the inverted question, which by extension of the nomenclature we call “inverted answers.” If no term in the question has an identifiable type, inversion is not possible.

3.4 Profiting From Inversions

Broadly speaking, our goal is to keep or re-rank the candidate answer hit-list on account of inversion results. Suppose that a question Q is inverted around pivot term T , and for each candidate answer C_i , a list of “inverted” answers $\{C_{ij}\}$ is generated as described in the previous section. If T is on one of the $\{C_{ij}\}$, then we say that C_i is *validated*. Validation is not a guarantee of keeping or improving C_i ’s position or score, but it helps. Most cases of failure to validate are called *refutation*; similarly, refutation of C_i is not a guarantee of lowering its score or position.

It is an open question how to adjust the results of the initial candidate answer list in light of the results of the inversion. If the scores associated with candidate answers (in both directions) were true probabilities, then a Bayesian approach would be easy to develop. However, they are not in our system. In addition, there are quite a few parameters that describe the inversion scenario.

Suppose Q generates a list of the top- N candidates $\{C_i\}$, with scores $\{S_i\}$. If this inversion method were not to be used, the top candidate on this list, C_1 , would be the emitted answer. The question generated by inverting about T and substituting C_i is QT_i . The system is fixed to find the top 10 passages responsive to QT_i , and generates an ordered list C_{ij} of candidate answers found in this set.

Each inverted question QT_i is run through our system, generating inverted answers $\{C_{ij}\}$, with scores $\{S_{ij}\}$, and whether and where the pivot term T shows up on this list, represented by a list of positions $\{P_i\}$, where P_i is defined as:

$$P_i = j \quad \text{if } C_{ij} = T, \text{ for some } j$$

$$P_i = -1 \text{ otherwise}$$

We added to the candidate list the special answer **nil**, representing “no answer exists in the corpus.”

As described earlier, we had observed from training data that failure to validate candidates of certain types (such as **PERSON**) would not necessarily be a real refutation, so we established a set of types **SOFTREFUTATION** which would contain the broadest of our types. At the other end of the spectrum, we observed that certain narrow candidate types such as **UsState** would definitely be refuted if validation didn’t occur. These are put in set **MUSTCONSTRAIN**. Our goal was to develop an algorithm for recomputing all the original scores $\{S_i\}$ from some combination (based on either arithmetic or decision-trees) of

$\{S_i\}$ and $\{S_{ij}\}$ and membership of SOFTREFUTATION and MUSTCONSTRAIN. Reliably learning all those weights, along with set membership, was not possible given only several hundred questions of training data. We therefore focused on a reduced problem.

We observed that when run on TREC question sets, the frequency of the rank of our top answer fell off rapidly, except with a second mode when the tail was accumulated in a single bucket. Our numbers for TRECs 11 and 12 are shown in Table 1.

Top answer rank	TREC11	TREC12
1	170	108
2	35	32
3	23	14
4	7	7
5	14	9
elsewhere	251	244
% correct	34	26

Table 1. Baseline statistics for TREC11-12.

We decided to focus on those questions where we got the right answer in second place (for brevity, we’ll call these second-place questions). Given that TREC scoring only rewards first-place answers, it seemed that with our incremental approach we would get most benefit there. Also, we were keen to limit the additional response time incurred by our approach. Since evaluating the top N answers to the original question with the Constraints process requires calling the QA system another N times per question, we were happy to limit N to 2. In addition, this greatly reduced the number of parameters we needed to learn.

For the evaluation, which consisted of determining if the resulting top answer was right or wrong, it meant ultimately deciding on one of three possible outcomes: the original top answer, the original second answer, or **nil**. We hoped to promote a significant number of second-place finishers to top place and introduce some **nils**, with minimal disturbance of those already in first place.

We used TREC11 data for training, and established a set of thresholds for a decision-tree approach to determining the answer, using Weka (Witten & Frank, 2005). We populated sets SOFTREFUTATION and MUSTCONSTRAIN by manual inspection.

The result is Algorithm A, where ($i \in \{1,2\}$) and

- o The C_i are the original candidate answers
- o The a_k are learned parameters ($k \in \{1..13\}$)
- o V_i means the i th answer was validated

- o P_i was the rank of the validating answer to question QT_i
- o A_i was the score of the validating answer to QT_i .

Algorithm A. Answer re-ranking using constraints validation data.

1. **If** $C_1 = \text{nil}$ **and** V_2 , **return** C_2
2. **If** V_1 **and** $A_1 > a_1$, **return** C_1
3. **If not** V_1 **and not** V_2 **and**
 $\text{type}(T) \in \text{MUSTCONSTRAIN}$,
return nil
4. **If not** V_1 **and not** V_2 **and**
 $\text{type}(T) \notin \text{SOFTREFUTATION}$,
if $S_1 > a_2$, **return** C_1 **else nil**
5. **If not** V_2 , **return** C_1
6. **If not** V_1 **and** V_2 **and**
 $A_2 > a_3$ **and** $P_2 < a_4$ **and**
 $S_1 - S_2 < a_5$ **and** $S_2 > a_6$, **return** C_2
7. **If** V_1 **and** V_2 **and**
 $(A_2 - P_2/a_7) > (A_1 - P_1/a_7)$ **and**
 $A_1 < a_8$ **and** $P_1 > a_9$ **and**
 $A_2 < a_{10}$ **and** $P_2 > a_{11}$ **and**
 $S_1 - S_2 < a_{12}$ **and** $(S_2 - P_2/a_7) > a_{13}$,
return C_2
8. **else return** C_1

4 Evaluation

Due to the complexity of the learned algorithm, we decided to evaluate in stages. We first performed an evaluation with a fixed question type, to verify that the purely arithmetic components of the algorithm were performing reasonably. We then evaluated on the entire TREC12 factoid question set.

4.1 Evaluation 1

We created a fixed question set of 50 questions of the form “What is the capital of X ?”, for each state in the U.S. The inverted question “What state is Z the capital of?” was correctly generated in each case. We evaluated against two corpora: the AQUAINT corpus, of a little over a million news-wire documents, and the CNS corpus, with about 37,000 documents from the Center for Nonproliferation Studies in Monterey, CA. We expected there to be answers to most questions in the former corpus, so we hoped there our method would be useful in converting 2nd place answers to first place. The latter corpus is about WMDs, so we expected there to be holes in the state capital coverage², for which **nil** identification would be useful.³

² We manually determined that only 23 state capitals were attested to in the CNS corpus, compared with all in AQUAINT.

³ We added Tbilisi to the answer key for “What is the capital of Georgia?”, since there was nothing in the question to disambiguate Georgia.

The baseline is our regular search-based QA-System without the Constraint process. In this baseline system there was no special processing for **nil** questions, other than if the search (which always contained some required terms) returned no documents. Our results are shown in Table 2.

	AQUAINT baseline	AQUAINT w/con- straints	CNS baseline	CNS w/con- straints
Firsts (non-nil)	39/50	43/50	7/23	4/23
Total nils	0/0	0/0	0/27	16/27
Total firsts	39/50	43/50	7/50	20/50
% correct	78	86	14	40

Table 2. Evaluation on AQUAINT and CNS corpora.

On the AQUAINT corpus, four out of seven 2nd place finishers went to first place. On the CNS corpus 16 out of a possible 26 correct no-answer cases were discovered, at a cost of losing three previously correct answers. The percentage correct score increased by a relative 10.3% for AQUAINT and 186% for CNS. In both cases, the error rate was reduced by about a third.

4.2 Evaluation 2

For the second evaluation, we processed the 414 factoid questions from TREC12. Of special interest here are the questions initially in first and second places, and in addition any questions for which **nils** were found.

As seen in Table 1, there were 32 questions which originally evaluated in rank 2. Of these, four questions were not invertible because they had no terms that were annotated with any of our named-entity types, e.g. #2285 “How much does it cost for gastric bypass surgery?”

Of the remaining 28 questions, 12 were promoted to first place. In addition, two new **nils** were found. On the down side, four out of 108 previous first place answers were lost. There was of course movement in the ranks two and beyond whenever **nils** were introduced in first place, but these do not affect the current TREC-QA factoid correctness measure, which is whether the top answer is correct or not. These results are summarized in Table 3.

While the overall percentage improvement was small, note that only second-place answers were candidates for re-ranking, and 43% of these were

promoted to first place and hence judged correct. Only 3.7% of originally correct questions were casualties. To the extent that these percentages are stable across other collections, as long as the size of the set of second-place answers is at least about 1/10 of the set of first-place answers, this form of the Constraint process can be applied effectively.

	Baseline	Constraints
Firsts (non-nil)	105	113
nils	3	5
Total firsts	108	118
% correct	26.1	28.5

Table 3. Evaluation on TREC12 Factoids.

5 Discussion

The experiments reported here pointed out many areas of our system which previous failure analysis of the basic QA system had not pinpointed as being too problematic, but for which improvement should help the Constraints process. In particular, this work brought to light a matter of major significance, *term equivalence*, which we had not previously focused on too much (and neither had the QA community as a whole). We will discuss that in Section 5.4.

Quantitatively, the results are very encouraging, but it must be said that the number of questions that we evaluated were rather small, as a result of the computational expense of the approach.

From Table 1, we conclude that the most mileage is to be achieved by our QA-System as a whole by addressing those questions which did not generate a correct answer in the first one or two positions. We have performed previous analyses of our system’s failure modes, and have determined that the passages that are output from the SEARCH component contain the correct answer 70-75% of the time. The ANSWER SELECTION module takes these passages and proposes a candidate answer list. Since the CONSTRAINTS MODULE’s operation can be viewed as a re-ranking of the output of ANSWER SELECTION, it could in principle boost the system’s accuracy up to that 70-75% level. However, this would either require a massive training set to establish all the parameters and weights required for all the possible re-ranking decisions, or a new model of the answer-list distribution.

5.1 Probability-based Scores

Our ANSWER SELECTION component assigns scores to candidate answers on the basis of the number of terms and term-term syntactic relationships from the

original question found in the answer passage (where the candidate answer and wh-word(s) in the question are identified terms). The resulting numbers are in the range 0-1, but are not true probabilities (e.g. where answers with a score of 0.7 would be correct 70% of the time). While the generated scores work well to rank candidates for a given question, inter-question comparisons are not generally meaningful. This made the learning of a decision tree (Algorithm A) quite difficult, and we expect that when addressed, will give better performance to the Constraints process (and maybe a simpler algorithm). This in turn will make it more feasible to re-rank the top 10 (say) original answers, instead of the current 2.

5.2 Better confidences

Even if no changes to the ranking are produced by the Constraints process, then the mere act of validation (or not) of existing answers can be used to adjust confidence scores. In TREC2002 (Voorhees, 2003), there was an evaluation of responses according to systems' confidences in their own answers, using the Average Precision (AP) metric. This is an important consideration, since it is generally better for a system to say "I don't know" than to give a wrong answer. On the TREC12 questions set, our AP score increased 2.1% with Constraints, using the algorithm we presented in (Chu-Carroll et al. 2002).

5.3 More complete NER

Except in pure pattern-based approaches, e.g. (Brill, 2002), answer types in QA systems typically correspond to the types identifiable by their named-entity recognizer (NER). There is no agreed-upon number of classes for an NER system, even approximately. It turns out that for best coverage by our CONSTRAINTS MODULE, it is advantageous to have a relatively large number of types. It was mentioned in Section 4.2 that certain questions were not invertible because no terms in them were of a recognizable type. Even when questions did have typed terms, if the types were very high-level then creating a meaningful inverted question was problematic. For example, for QA without Constraints it is not necessary to know the type of "MTV" in "When was MTV started?", but if it is only known to be a Name then the inverted question "What <Name> was started in 1980?" could be too general to be effective.

5.4 Establishing Term Equivalence

The somewhat surprising condition that emerged from this effort was the need for a much more complete ability than had previously been recognized for the system to establish the equivalence of two terms. Redundancy has always played a large role in QA

systems – the more occurrences of a candidate answer in retrieved passages the higher the answer's score is made to be. Consequently, at the very least, a string-matching operation is needed for checking equivalence, but other techniques are used to varying degrees.

It has long been known in IR that stemming or lemmatization is required for successful term matching, and in NLP applications such as QA, resources such as WordNet (Miller, 1995) are employed for checking synonym and hypernym relationships; Extended WordNet (Moldovan & Novischi, 2002) has been used to establish lexical chains between terms. However, the Constraints work reported here has highlighted the need for more extensive equivalence testing.

In direct QA, when an ANSWER SELECTION module generates two (or more) equivalent correct answers to a question (e.g. "*Ferdinand Marcos*" vs. "*President Marcos*"; "*French*" vs. "*France*"), and fails to combine them, it is observed that as long as either one is in first place then the question is correct and might not attract more attention from developers. It is only when neither is initially in first place, but combining the scores of correct candidates boosts one to first place that the failure to merge them is relevant. However, in the context of our system, we are comparing the pivot term from the original question to the answers to the inverted questions, and failure here will directly impact validation and hence the usefulness of the entire approach.

As a consequence, we have identified the need for a component whose sole purpose is to establish the equivalence, or generally the kind of relationship, between two terms. It is clear that the processing will be very type-dependent – for example, if two populations are being compared, then a numerical difference of 5% (say) might not be considered a difference at all; for "Where" questions, there are issues of granularity and physical proximity, and so on. More examples of this problem were given in (Prager et al. 2004a). Moriceau (2006) reports a system that addresses part of this problem by trying to rationalize different but "similar" answers to the user, but does not extend to a general-purpose equivalence identifier.

6 Summary

We have extended earlier Constraints-based work through the method of question inversion. The approach uses our QA system recursively, by taking candidate answers and attempts to validate them through asking the inverted questions. The outcome

is a re-ranking of the candidate answers, with the possible insertion of **nil** (no answer in corpus) as the top answer.

While we believe the approach is general, and can work on any question and arbitrary candidate lists, due to training limitations we focused on two restricted evaluations. In the first we used a fixed question type, and showed that the error rate was reduced by 36% and 30% on two very different corpora. In the second evaluation we focused on questions whose direct answers were correct in the second position. 43% of these questions were subsequently judged correct, at a cost of only 3.7% of originally correct questions. While in the future we would like to extend the Constraints process to the entire answer candidate list, we have shown that applying it only to the top two can be beneficial as long as the second-place answers are at least a tenth as numerous as first-place answers. We also showed that the application of Constraints can improve the system's confidence in its answers.

We have identified several areas where improvement to our system would make the Constraints process more effective, thus getting a double benefit. In particular we feel that much more attention should be paid to the problem of determining if two entities are the same (or "close enough").

7 Acknowledgments

This work was supported in part by the Disruptive Technology Office (DTO)'s Advanced Question Answering for Intelligence (AQUAINT) Program under contract number H98230-04-C-1577. We would like to thank the anonymous reviewers for their helpful comments.

References

- Brill, E., Dumais, S. and Banko M. "An analysis of the AskMSR question-answering system." In *Proceedings of EMNLP 2002*.
- Chu-Carroll, J., J. Prager, C. Welty, K. Czuba and D. Ferrucci. "A Multi-Strategy and Multi-Source Approach to Question Answering", *Proceedings of the 11th TREC*, 2003.
- Clarke, C., Cormack, G., Kisman, D. and Lynam, T. "Question answering by passage selection (Multitext experiments for TREC-9)" in *Proceedings of the 9th TREC*, pp. 673-683, 2001.
- Hendrix, G., Sacerdoti, E., Sagalowicz, D., Slocum J.: *Developing a Natural Language Interface to Complex Data*. VLDB 1977: 292
- Lenat, D. 1995. "Cyc: A Large-Scale Investment in Knowledge Infrastructure." *Communications of the ACM* 38, no. 11.
- Miller, G. "WordNet: A Lexical Database for English", *Communications of the ACM* 38(11) pp. 39-41, 1995.
- Moldovan, D. and Novischi, A, "Lexical Chains for Question Answering", *COLING 2002*.
- Moldovan, D. and Rus, V., "Logic Form Transformation of WordNet and its Applicability to Question Answering", *Proceedings of the ACL*, 2001.
- Moriceau, V. "Numerical Data Integration for Cooperative Question-Answering", in *EACL Workshop on Knowledge and Reasoning for Language Processing (KRAQ'06)*, Trento, Italy, 2006.
- Prager, J.M., Chu-Carroll, J. and Czuba, K. "Question Answering using Constraint Satisfaction: QA-by-Dossier-with-Constraints", *Proc. 42nd ACL*, pp. 575-582, Barcelona, Spain, 2004(a).
- Prager, J.M., Chu-Carroll, J. and Czuba, K. "A Multi-Strategy, Multi-Question Approach to Question Answering" in *New Directions in Question-Answering*, Maybury, M. (Ed.), AAAI Press, 2004(b).
- Prager, J., "A Curriculum-Based Approach to a QA Roadmap" *LREC 2002 Workshop on Question Answering: Strategy and Resources*, Las Palmas, May 2002.
- Radev, D., Prager, J. and Samn, V. "Ranking Suspected Answers to Natural Language Questions using Predictive Annotation", *Proceedings of ANLP 2000*, pp. 150-157, Seattle, WA.
- Voorhees, E. "Overview of the TREC 2002 Question Answering Track", *Proceedings of the 11th TREC*, Gaithersburg, MD, 2003.
- Warren, D., and F. Pereira "An efficient easily adaptable system for interpreting natural language queries," *Computational Linguistics*, 8:3-4, 110-122, 1982.
- Winograd, T. Procedures as a representation for data in a computer program for understanding natural language. *Cognitive Psychology*, 3(1), 1972.
- Witten, I.H. & Frank, E. *Data Mining. Practical Machine Learning Tools and Techniques*. Elsevier Press, 2005.

Reranking Answers for Definitional QA Using Language Modeling

Yi Chen

School of Software Engineering
Chongqing University
Chongqing, China, 400044
126cy@126.com

Ming Zhou

Microsoft Research Asia
5F Sigma Center, No.49 Zhichun
Road, Haidian
Beijing, China, 100080
mingzhou@microsoft.com

Shilong Wang

College of Mechanical Engineering
Chongqing University
Chongqing, China, 400044
slwang@cqu.edu.cn

Abstract*

Statistical ranking methods based on centroid vector (profile) extracted from external knowledge have become widely adopted in the top definitional QA systems in TREC 2003 and 2004. In these approaches, terms in the centroid vector are treated as a bag of words based on the independent assumption. To relax this assumption, this paper proposes a novel language model-based answer reranking method to improve the existing bag-of-words model approach by considering the dependence of the words in the centroid vector. Experiments have been conducted to evaluate the different dependence models. The results on the TREC 2003 test set show that the reranking approach with biterm language model, significantly outperforms the one with the bag-of-words model and unigram language model by 14.9% and 12.5% respectively in F-Measure(5).

1 Introduction

In recent years, QA systems in TREC (Text REtrieval Conference) have made remarkable progress (Voorhees, 2002). The task of TREC QA before 2003 has mainly focused on the *factoid* questions, in which the answer to the question is a number, a person name, or an organization name, or the like.

Questions like “Who is Colin Powell?” or “What is mold?” are *definitional* questions

(Voorhees, 2003). Statistics from 2,516 Frequently Asked Questions (FAQ) extracted from *Internet FAQ Archives*¹ show that around 23.6% are definitional questions. This indicates that definitional questions occur frequently and are important question types. TREC started the evaluation for definitional QA in 2003. The definitional QA systems in TREC are required to extract definitional nuggets/sentences that contain the highly descriptive information about the question target from a given large corpus.

For definitional question, statistical ranking methods based on centroid vector (profile) extracted from external resources, such as the online encyclopedia, are widely adopted in the top systems in TREC 2003 and 2004 (Xu et al., 2003; Blair-Goldensohn et al., 2003; Wu et al., 2004). In these systems, for a given question, a vector is formed consisting of the most frequent co-occurring terms with the question target as the question profile. Candidate answers extracted from a given large corpus are ranked based on their similarity to the question profile. The similarity is normally the *TFIDF* score in which both the candidate answer and the question profile are treated as a bag of words in the framework of Vector Space Model (VSM).

VSM is based on an independence assumption, which assumes that terms in a vector are statistically independent from one another. Although this assumption makes the development of retrieval models easier and the retrieval operation tractable, it does not hold in textual data. For example, for question “Who is Bill Gates?” words “born” and “1955” in the candidate answer are not independent.

In this paper, we are interested in considering the term dependence to improve the answer reranking for definitional QA. Specifically, the

*This work was finished while the first author was visiting Microsoft Research Asia during March 2005-March 2006 as a component of the project of AskBill Chatbot led by Dr. Ming Zhou.

¹ <http://www.faqs.org/faqs/>

language model is utilized to capture the term dependence. A language model is a probability distribution that captures the statistical regularities of natural language use. In a language model, key elements are the probabilities of word sequences, denoted as $P(w_1, w_2, \dots, w_n)$ or $P(w_{1:n})$ for short. Recently, language model has been successfully used for information retrieval (IR) (Ponte and Croft, 1998; Song and Croft, 1998; Lafferty et al., 2001; Gao et al., 2004; Cao et al., 2005). Our natural thinking is to apply language model to rank the candidate answers as it has been applied to rank search results in IR task.

The basic idea of our research is that, given a definitional question q , an ordered centroid OC which is learned from the web and a language model $LM(OC)$ which is trained with it. Candidate answers can be ranked by probability estimated by $LM(OC)$. A series of experiments on standard TREC 2003 collection have been conducted to evaluate bigram and biterm language models. Results show that both these two language models produce promising results by capturing the term dependence and biterm model achieves the best performance. Biterm language model interpolating with unigram model significantly improves the VSM and unigram model by 14.9% and 12.5% in F-Measure(5).

In the rest of this paper, Section 2 reviews related work. Section 3 presents details of the proposed method. Section 4 introduces the structure of our experimental system. We show the experimental results in Section 5, and conclude the paper in Section 6.

2 Related Work

Web information has been widely used for answer reranking and validation. For factoid QA task, *AskMSR* (Brill et al., 2001) ranks the answers by counting the occurrences of candidate answers returned from a search engine. Similarly, *DIOGENE* (Magnini et al., 2002) applies search engines to validate candidate answers.

For definitional QA task, Lin (2002) presented an approach in which web-based answer reranking is combined with dictionary-based (e.g., WordNet) reranking, which leads to a 25% increase in mean reciprocal rank (MRR). Xu et al. (2003) proposed a statistical ranking method based on centroid vector (i.e., vector of words and frequencies) learned from the online encyclopedia (i.e., *Wikipedia*²) and the web. Candi-

date answers were reranked based on their similarity (*TFIDF* score) to the centroid vector. Similar techniques were explored in (Blair-Goldensohn et al., 2003). In this paper, we explore the dependence among terms in centroid vector for improving the answer reranking for definitional QA.

In recent years, language modeling has been widely employed in IR (Ponte and Croft, 1998; Song and Croft, 1998; Miller and Zhai, 1999; Lafferty and Zhai, 2001). The basic idea is to compute the conditional probability $P(Q|D)$, i.e., the probability of generating a query Q given the observation of a document D . The searched documents are ranked in descending order of this probability.

Song and Croft (1998) proposed a general language model to incorporate word dependence by using bigrams. Srikanth and Srihari (2002) introduced biterm language models similar to the bigram model except that the constraint of order in terms is relaxed and improved performance was observed. Gao et al. (2004) presented a new method of capturing word dependencies, in which they extended state-of-the-art language modeling approaches to information retrieval by introducing a dependence structure that learned from training data. Cao et al. (2005) proposed a novel dependence model to incorporate both relationships of WordNet and co-occurrence with the language modeling framework for IR. In our approach, we propose bigram and biterm models to capture the term dependence in centroid vector.

Applying language modeling for the QA task has not been widely researched. Zhang D. and Lee (2003) proposed a method using language model for passage retrieval for the factoid QA. They trained two language models, in which one was the question-topic language model and the other was passage language model. They utilized the divergence between the two language models to rank passages. In this paper, we focus on reranking answers for definitional questions.

As other ranking approaches, Xu, et al. (2005) formalized ranking definitions as classification problems, and Cui et al. (2004) proposed soft patterns to rank answers for definitional QA.

3 Reranking Answers Using Language Model

3.1 Model background

In practice, language model is often approximated by N-gram models.

Unigram:

² <http://www.wikipedia.org>

$$P(w_{1,n}) = P(w_1)P(w_2)...P(w_n) \quad (1)$$

Bigram:

$$P(w_{1,n}) = P(w_1)P(w_2|w_1)...P(w_n|w_{n-1}) \quad (2)$$

The unigram model makes a strong assumption that each word occurs independently. The bigram model takes the local context into consideration. It has been proved to work better than the unigram language model in IR (e.g., Song and Croft, 1998).

Biterm language models are similar to bigram language models except that the constraint of order in terms is relaxed. Therefore, a document containing *information retrieval* and a document containing *retrieval (of) information* will be assigned the same generation probability. The biterm probabilities can be approximated using the frequency of occurrence of terms.

Three approximation methods were proposed in Srikanth and Srihari (2002). The so-called min-Adhoc approximation truly relaxes the constraint of word order and outperformed other two approximation methods in their experiments.

$$P_{BT}(w_i | w_{i-1}) \approx \frac{C(w_{i-1}, w_i) + C(w_i, w_{i-1})}{\min\{C(w_{i-1}), C(w_i)\}} \quad (3)$$

Equation (3) is the min-Adhoc approximation. Where $C(X)$ gives the occurrences of the string X .

3.2 Reranking based on language model

In our approach, we adopt bigram and biterm language models. As a smoothing approach, linear interpolation of unigrams and bigrams is employed.

Given a candidate answer $A = t_1 t_2 ... t_i ... t_n$ and a bigram or biterm back-off language model OC trained with the ordered centroid, the probability of generating A can be estimated by Equation (4).

$$P(A | OC) = P(t_1, ..., t_n | OC) \quad (4)$$

$$= P(t_1 | OC) \prod_{i=2}^n [\lambda P(t_i | OC) + (1 - \lambda) P(t_i | t_{i-1}, OC)]$$

where OC stands for the language model of the ordered centroid and λ is the mixture weight combining the unigram and bigram (or biterm) probabilities. After taking logarithm and exponential for Equation (4), we get Equation (5).

$$Score(A) = \exp \left(\log P(t_1 | OC) + \sum_{i=2}^n \log [\lambda P(t_i | OC) + (1 - \lambda) P(t_i | t_{i-1}, OC)] \right) \quad (5)$$

We observe that this formula penalizes verbose candidate answers. This can be alleviated by adding a brevity penalty, BP , which is inspired by machine translation evaluation (Papineni et al., 2001).

$$BP = \exp \left(\min \left(1 - \frac{L_{ref}}{L_A}, 1 \right) \right) \quad (6)$$

where L_{ref} is a constant standing for the length of reference answer (i.e., centroid vector). L_A is the length of the candidate answer. By combining Equation (5) and (6), we get the final scoring function.

$$FinalScore(A) = BP \times Score(A) \quad (7)$$

$$= \exp \left(\min \left(1 - \frac{L_{ref}}{L_A}, 1 \right) \right) \times \exp \left(\log P(t_1 | OC) + \sum_{i=2}^n \log [\lambda P(t_i | OC) + (1 - \lambda) P(t_i | t_{i-1}, OC)] \right)$$

3.3 Parameter estimation

In Equation (7), we need to estimate three parameters: $P(t_i | OC)$, $P(t_i | t_{i-1}, OC)$ and λ .

For $P(t_i | OC)$, $P(t_i | t_{i-1}, OC)$, maximum likelihood estimation (MLE) is employed.

$$P(t_i | OC) = \frac{Count_{OC}(t_i)}{N_{OC}} \quad (8)$$

$$P(t_i | t_{i-1}, OC) = \frac{Count_{OC}(t_{i-1}, t_i)}{Count_{OC}(t_{i-1})} \quad (9)$$

where $Count_{OC}(X)$ is the occurrences of the string X in the ordered centroid and N_{OC} stands for the total number of tokens in the ordered centroid.

For biterm language model, we use the above mentioned min-Adhoc approximation (Srikanth and Srihari, 2002).

$$P_{BT}(t_i | t_{i-1}, OC) = \frac{Count_{OC}(t_{i-1}, t_i) + Count_{OC}(t_i, t_{i-1})}{\min\{Count_{OC}(t_{i-1}), Count_{OC}(t_i)\}} \quad (10)$$

For unigram, we do not need smoothing because we only concern terms in the centroid vector. Recall that bigram and biterm probabilities have already been smoothed by interpolation.

The λ can be learned from a training corpus using an Expectation Maximization (EM) algorithm. Specifically, we estimate λ by maximizing the likelihood of all training instances, given the bigram or biterm model:

$$\lambda^* = \arg \max_{\lambda} \sum_{j=1}^{|INS|} P(t_1^{(j)} ... t_{l(j)}^{(j)} | OC) \quad (11)$$

$$= \arg \max_{\lambda} \sum_{j=1}^{|INS|} \left\{ \sum_{i=2}^{l_j} \log [\lambda P(t_i^{(j)}) + (1 - \lambda) P(t_i^{(j)} | t_{i-1}^{(j)})] \right\}$$

BP and $P(t_1)$ are ignored because they do not affect λ . λ can be estimated using EM iterative procedure:

- 1) Initialize λ to a random estimate between 0 and 1, i.e., 0.5;
- 2) Update λ using:

$$\lambda^{(r+1)} = \frac{1}{|INS|} \times \sum_{j=1}^{|INS|} \frac{1}{l_j - 1} \sum_{i=2}^{l_j} \frac{\lambda^{(r)} P(t_i^{(j)})}{\lambda^{(r)} P(t_i^{(j)}) + (1 - \lambda^{(r)}) P(t_i^{(j)} | t_{i-1}^{(j)})} \quad (12)$$

where INS denotes all training instances and $|INS|$ gives the number of training instances which is used as a normalization factor. l_j gives

the number of tokens in the j^{th} instance in the training data;

3) Repeat Step 2 until λ converges.

We use the TREC 2004 test set³ as our training data and we set λ as 0.4 for bigram model and 0.6 for biterm model according to the experimental results.

4 System Architecture

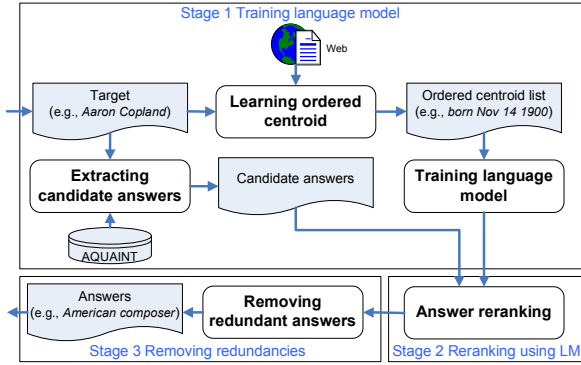


Figure 1. System architecture.

We propose a three-stage approach for answer extraction. It involves: 1) learning a language model from the web; 2) adopting the language model to rerank candidate answers; 3) removing redundancies. Figure 1 shows five main modules.

Learning ordered centroid:

1) Query expansion. Definitional questions are normally short (i.e., who is Bill Gates?). Query expansion is used to refine the query intention. First, reformulate query via simply adding clue words to the questions. i.e., for “Who is ...?” question, we add the word “biography”; and for “What is ...?” question, we add the word “is usually”, “refers to”, etc. We learn these clue words using the similar method proposed in (Ravi-chandran and Hovy, 2002). Second, query a web search engine (i.e., *Google*⁴) with reformulated query and learn top- R (we empirically set $R=5$) most frequent co-occurring terms with the target from returned snippets as query *expansion terms*;

2) Learning centroid vector (profile). We query *Google* again with the target and expanded terms learned in the previous step, download top- N (we empirically set $N=500$ based on the tradeoff between the snippet number and the time complexity) snippets, and split snippets into sentences. Then, we retain the generated sentences that contain the target, denoted as W . Finally, learn top- M (We empirically set $M=350$) most frequent co-

occurring terms (stemmed) from W using Equation (15) (Cui et al., 2004) as the centroid vector.

$$Weight(t) = \frac{\log(Co(t, T) + 1)}{\log(Count(t) + 1) + \log(Count(T) + 1)} \times idf(t) \quad (13)$$

where $Co(t, T)$ denotes the number of sentences in which t co-occurs with the target T , and $Count(t)$ gives the number of sentences containing the word t . We also use the inverse document frequency of t , $idf(t)$ ⁵, as a measurement of the global importance of the word;

3) Extracting ordered centroid. For each sentence in W , we retain the terms in the centroid vector as the ordered centroid list. Words not contained in the centroid vector will be treated as the “stop words” and ignored.

E.g., “Who is Aaron Copland?”, the ordered centroid list is shown below (where italics are extracted and put in the ordered centroid list):

1. Today's Highlight in History: On *No-vember 14, 1900, Aaron Copland*, one of *America's* leading 20th century *com-posers*, was *born* in *New York City*. $\Rightarrow$ *November 14 1900 Aaron Copland America composer born New York City*
2. ...

Extracting candidate answers: We extract candidates from AQUAINT corpus.

- 1) Querying AQUAINT corpus with the target and retrieve relevant documents;
- 2) Splitting documents into sentences and extracting the sentences containing the target. Here in order to improve recall, simple heuristics rules are used to handle the problem of coreference resolution. If a sentence is deemed to contain the target and its next sentence starts with “he”, “she”, “it”, or “they”, then the next sentence is retained.

Training language models: As mentioned above, we train language models using the obtained ordered centroid for each question.

Answer reranking: Once the language models and the candidate answers are ready for a given question, candidate answers are reranked based on the probabilities of the language models generating candidate answers.

Removing redundancies: Repetitive and similar candidate sentences will be removed. Given a reranked candidate answer set CA , redundancy removing is conducted as follows:

³ The test data for TREC-13 includes 65 definition questions. NIST drops one in the official evaluation.

⁴ <http://www.google.com>

⁵ We use the statistics from British National Corpus (BNC) site to approximate words' IDF, <http://www.itri.brighton.ac.uk/~Adam.Kilgariff/bnc-readme.html>.

-
- Step 1: Initially set the result $A=\{\}$, and get top $j=1$ element from CA and then add it to A , $j=2$.
- Step 2: Get the j^{th} element from CA , denoted as CA_j . Compute cosine similarity between CA_j and each element i of A , which is expressed as s_{ij} . Then let $s_{ik}=\max\{s_{1j}, s_{2j}, \dots, s_{ij}\}$, if $s_{ik} < threshold$ (we set it to 0.75), then add j to the set A .
- Step 3: If length of A exceeds a predefined threshold, exit; otherwise, $j=j+1$, go to Step 2.
-

Figure 2. Algorithm for removing redundancy.

5 Experiment & Evaluation

In order to get comparable evaluation, we apply our approach to TREC 2003 definitional QA task. More details will be shown in the following sections.

5.1 Experiment setup

5.1.1 Dataset

We employ the dataset from the TREC 2003 QA task. It includes the AQUAINT corpus of more than 1 million news articles from the *New York Times* (1998-2000), *Associated Press* (1998-2000), *Xinhua News Agency* (1996-2000) and 50 definitional question/answer pairs. In these 50 definitional questions, 30 are for people (e.g., *Aaron Copland*), 10 are for organizations (e.g., *Friends of the Earth*) and 10 are for other entities (e.g., *Quasars*). We employ *Lemur*⁶ to retrieve relevant documents from the AQUAINT corpus. For each query, we return the top 500 documents.

5.1.2 Evaluation metrics

We adopt the evaluation metrics used in the TREC definitional QA task (Voorhees, 2003 and 2004). TREC provides a list of essential and acceptable nuggets for answering each question. We use these nuggets to assess our approach. During this progress, two human assessors examine how many essential and acceptable nuggets are covered in the returned answers. Every question is scored using nugget recall (NR) and an approximation to nugget precision (NP) based on answer length. The final score for a definition response is computed using F-Measure. In TREC 2003, the β parameter was set to 5 indicating that recall is 5 times as important as precision (Voorhees, 2003).

$$F(\beta = 5) = \frac{5^2 * NP * NR}{(5^2 + 1)NP + NR} \quad (14)$$

in which,

$$NR = \frac{\# \text{ essential nuggets returned}}{\# \text{ essential answer nuggets}} \quad (15)$$

$$NP = \begin{cases} 1, & (\text{length} < \text{allowance}) \\ 1 - \frac{(\text{length} - \text{allowance})}{\text{length}}, & (\text{otherwise}) \end{cases} \quad (16)$$

where $\text{allowance} = 100 * (\# \text{ essential} + \# \text{ acceptable nuggets returned})$ and $\text{length} = \# \text{ non-white space characters in strings returned}$.

5.1.3 Baseline system

We employ the *TFIDF* heuristics algorithm-based approach as our baseline system, in which the candidate answers and the centroid are treated as a bag of words.

$$\text{weight}_i = TF_i * IDF_i = TF_i * \ln \frac{N}{DF_i} \quad (17)$$

where TF_i gives the occurrences of term i . DF_i ⁷ is the number of documents containing term i . N gives the total number of documents.

For comparison purpose, the unigram model is adopted and its scoring function is similar with Equation (7). The main difference is that we only concern unigram probability $P(t_i|OC)$ in unigram-based scoring function.

For all systems, we empirically set the threshold of answer length to 12 sentences for people targets (i.e., *Aaron Copland*), and 10 sentences for other targets (i.e., *Quasars*).

5.2 Performance evaluation

As the first evaluation, we assess the performance obtained by our language model method against the baseline system without query expansion (QE). The evaluation results are shown in Table 1.

	Average NR	Average NP	F(5)
Baseline (TFIDF)	0.469	0.221	0.432
Unigram	0.508 (+8.3%)	0.204 (-7.7%)	0.459 (+6.3%)
Bigram	0.554 (+18.1%)	0.234 (+5.9%)	0.505 (+16.9%)
Biterm	0.567 (+20.9%)	0.222 (+0.5%)	0.511 (+18.3%)

Table 1. Comparisons without QE.

From Table 1, it is easy to observe that the unigram, bigram and biterm-based approaches improve the F(5) by 6.3%, 16.9% and 18.3% against the baseline system respectively. At the same time, the bigram and biterm improves the

⁶ A free IR tool, <http://www.lemurproject.org/>

⁷ We also use British National Corpus (BNC) to estimate it.

F(5) by 10.0% and 11.3% against the unigram respectively. The unigram slightly outperform the baseline. We also notice that the biterm model improves slightly over the bigram model since it ignores the order of term-occurrence. This observation coincides with the experimental results of Srikanth and Srihari (2002). These results show that the bigram and biterm models outperform the VSM model and the unigram model dramatically. It is a clear indication that the language model which takes into account the term dependence among centroid vector is an effective way to rerank answers.

As mentioned above, QE is involved in our system. In the second evaluation, we assess the performance obtained by the language model method against the baseline system with QE. We list the evaluation results in Table 2.

	Average NR	Average NP	F(5)
Baseline (QE)	0.508	0.207	0.462
Unigram (QE)	0.518 (+2.0%)	0.223 (+7.7%)	0.472 (+2.2%)
Bigram (QE)	0.573 (+12.8%)	0.228 (+10.1%)	0.518 (+12.1%)
Biterm (QE)	0.582 (+14.6%)	0.240 (+15.9%)	0.531 (+14.9%)

Table 2. Comparisons with QE.

From Table 2, we observe that, with QE, the bigram and biterm still outperform the baseline system (VSM) significantly by 12.1% ($p^8=0.03$) and 14.9% ($p=0.004$) in F(5). Furthermore, the bigram and biterm perform significantly better than the unigram by 9.7% ($p=0.07$) and 12.5% ($p=0.02$) in F(5) respectively. This indicates that the term dependence is effective in keeping improving the performance. It is easy to observe that the baseline is close to the unigram model since both two systems are based on the independent assumption. We also notice that the biterm model improves slightly over the bigram model. At the same time, all of the four systems improve the performance against the corresponding system without QE. The main reason is that the qualities of the centroid vector can be enhanced with QE. We are interested in the performance comparison with or without QE for each system. Through comparison it is found that the baseline system relies on QE more heavily than our approach does. With QE, the baseline system improves the performance by 6.9% and the language model approaches improve the performance by 2.8%, 2.6% and 3.9%, respectively.

⁸ T-Test has been performed.

F(5) performance comparison between the baseline model and the biterm model for each of 50 TREC questions is shown in Figure 3. QE is used in both the baseline system and the biterm system.

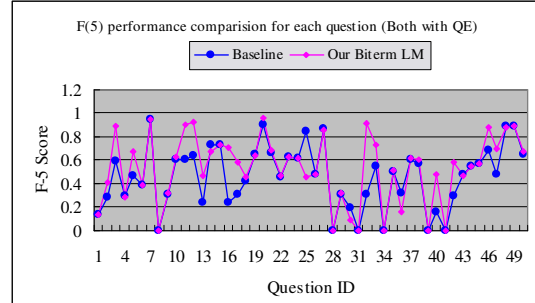


Figure 3. Biterm vs. Baseline.

We are also interested in the comparison with the systems in TREC 2003. The best F(5) score returned by our proposed approach is 0.531, which is close to the top 1 run in TREC 2003 (Voorhees, 2003). The F(5) score of the best system is 0.555, reported by BBN's system (Xu et al., 2003). In BBN's experiments, the centroid vector was learned from the human made external knowledge resources, such as encyclopedia and the web. Table 3 gives the comparison between our biterm model-based system with the BBN's run with different β values.

Run Tag	F(β) Score				
	$\beta=1$	$\beta=2$	$\beta=3$	$\beta=4$	$\beta=5$
BBN	0.310	0.423	0.493	0.532	0.555
Ours	0.288	0.382	0.470	0.509	0.531

Table 3. Comparison with BBN's run.

5.3 Case study

A positive example returned by our proposed approach is given below. For *Qid: 2304*: "Who is Niels Bohr?", the reference answers are given in Table 4 (only vital nuggets are listed):

vital	Danish
vital	Nuclear physicist
vital	Helped create atom bomb
vital	Nobel Prize winner

Table 4. Reference answers for question "Who is Niels Bohr?".

Answers returned by the baseline system and our proposed system are presented in Table 5.

System	Returned answers (Partly)
Baseline system	1. ..., Niels Bohr, the <i>great Danish scientist</i> 2. ...the German physicist Werner Heisenberg and the <i>Danish physicist</i>

	Niels Bohr 3. ...took place between the <i>Danish physicist</i> Niels Bohr and his onetime protege, the German scientist ... 4. ... two <i>great physicists</i> , the Dane Niels Bohr and Werner Heisenberg ... 5. ...
Proposed system	1. ...physicist Werner Heisenberg travel to ... his colleague and old mentor, Niels Bohr, <i>the great Danish scientist</i> 2. ... two <i>great physicists</i> , the Dane Niels Bohr and Werner Heisenberg ... 3. Today's Birthdays: ... <i>Danish nuclear physicist and Nobel Prize winner Niels Bohr (1885-1962)</i> 4. the <i>Danish atomic physicist</i> , and his German pupil, Werner Heisenberg, the author of the uncertainty principle 5. ...

Table 5. Baseline vs. our system for question “Who is Niels Bohr?”.

From Table 5, it can be seen that the baseline system returned only one vital nugget: *Danish* (here we don’t think that *physicist* is equal to *nuclear physicist* semantically). Our proposed system returned three vital nuggets: *Danish*, *Nuclear physicist*, and *Nobel Prize winner*. The answer sentence “Today’s Birthdays: ... Danish nuclear physicist and Nobel Prize winner Niels Bohr (1885-1962)” contains more descriptive information for the question target “Niels Bohr” and is ranked 3rd in the top 12 answers in our proposed system.

5.4 Error analysis

Although we have shown that the language model-based approach significantly improves the system performance, there is still plenty of room for improvement.

- 1) Sparseness of search results derogated the learning of the ordered centroid: E.g.: *Qid* 2348: “What is the medical condition shingles?”, in which we treat the words “medical condition shingles” as the question target. We found that few sentences contain the target “medical condition shingles”. We found utilizing multiple search engines, such as *MSN*⁹, *AltaVista*¹⁰ might alleviate this problem. Besides, more effective smoothing techniques could be promising.
- 2) Term ambiguity: for some queries, the unrelated documents are returned. E.g., for *Qid* 2267: “Who is Alexander Pope?”, all documents returned from the IR tool *Lemur* for

this question are about “Pope John Paul II”, not “Alexander Pope”. This may be caused by the ambiguity of the word “Pope”. In this case, term disambiguation or adding some constraint terms which are learned from the web to the query to the AQUAINT corpus might be helpful.

6 Conclusions and Future Work

In this paper, we presented a novel answer reranking method for definitional question. We use bigram and biterm language models to capture the term dependence. Our contribution can be summarized as follows:

- 1) Word dependence is explored from ordered centroid learned from snippets of a search engine;
- 2) Bigram and biterm models are presented to capture the term dependence and rerank candidate answers for definitional QA;
- 3) Evaluation results show that both bigram and biterm models outperform the VSM and unigram model significantly on TREC 2003 test set.

In our experiments, centroid words were learned from the returned snippets of a web search engine. In the future, we are interested in enhancing the centroid learning using human knowledge sources such as encyclopedia. In addition, we will explore new smoothing techniques to enhance the interpolation method in our current approach.

7 Acknowledgements

The authors are grateful to Dr. Cheng Niu, Yunbo Cao for their valuable suggestions on the draft of this paper. We are indebted to Shiqi Zhao, Shenghua Bao, Wei Yuan for their valuable discussions about this paper. We also thank Dwight for his assistance to polish the English. Thanks also go to anonymous reviewers whose comments have helped improve the final version of this paper.

References

- E. Brill, J. Lin, M. Banko, S. Dumais and A. Ng. 2001. Data-Intensive Question Answering. In *Proceedings of the Tenth Text Retrieval Conference (TREC 2001)*, Gaithersburg, MD, pp. 183-189.
- S. Blair-Goldensohn, K.R. McKeown and A. Hazen Schlaikjer. 2003. A Hybrid Approach for QA Track Definitional Questions. In *Proceedings of the Tenth Text Retrieval Conference (TREC 2003)*, pp. 336-343.

⁹ <http://www.msn.com>

¹⁰ <http://www.altavista.com>

- S. F. Chen and J. T. Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the ACL*, pp. 310-318.
- Hang Cui, Min-Yen Kan and Tat-Seng Chua. 2004. Unsupervised Learning of Soft Patterns for Definitional Question Answering. In *Proceedings of the Thirteenth World Wide Web conference (WWW 2004)*, New York, pp. 90-99.
- Guihong Cao, Jian-Yun Nie, and Jing Bai. 2005. Integrating Word Relationships into Language Models. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development of Information Retrieval (SIGIR 2005)*, Salvador, Brazil.
- Jianfeng Gao, Jian-Yun Nie, Guangyuan Wu and Guihong Cao. 2004. Dependence language model for information retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development of Information Retrieval (SIGIR 2004)*, Sheffield, UK.
- Chin-Yew Lin. 2002. The Effectiveness of Dictionary and Web-Based Answer Reranking. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, Taipei, Taiwan.
- Lafferty, J. and Zhai, C. 2001. Document language models, query models, and risk minimization for information retrieval. In W.B. Croft, D.J. Harper, D.H. Kraft, & J. Zobel (Eds.), In *Proceedings of the 24th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, New Orleans, Louisiana, New York, pp.111-119.
- Magnini, B., Negri, M., Prevete, R., and Tanev, H. 2002. Is It the Right Answer? Exploiting Web Redundancy for Answer Validation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, Philadelphia, PA.
- Miller, D., Leek, T., and Schwartz, R. 1999. A hidden Markov model information retrieval system. In *Proceedings of the 22nd Annual International ACM SIGIR Conference*, pp. 214-221.
- K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2001. Bleu: a Method for Automatic Evaluation of Machine Translation. *IBM Research Report rc22176 (w0109022)*, Thomas J. Watson Research Center.
- Ponte, J., and Croft, W.B. 1998. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, New York, pp.275-281.
- J. Prager, D. Radev, and K. Czuba. 2001. Answering what-is questions by virtual annotation. In *Proceedings of the Human Language Technology Conference (HLT 2001)*, San Diego, CA.
- Deepak Ravichandran and Eduard Hovy. 2002. Learning Surface Text Patterns for a Question Answering System. In *Proceedings of the 40th Annual Meeting of the ACL*, pp. 41-47.
- Song, F., and Croft, W.B. 1999. A general language model for information retrieval. In *Proceedings of the 22nd Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, New York, pp.279-280.
- Srikanth, M. and Srihari, R. 2002. Biterm language models for document retrieval. In *Proceedings of the 2002 ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, Finland.
- Ellen M. Voorhees. 2002. Overview of the TREC 2002 question answering track. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*.
- Ellen M. Voorhees. 2003. Overview of the TREC 2003 question answering track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*.
- Ellen M. Voorhees. 2004. Overview of the TREC 2004 question answering track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2004)*.
- Lide Wu, Xuanjing Huang, Lan You, Zhushuo Zhang, Xin Li, and Yaqian Zhou. 2004. FDUQA on TREC2004 QA Track. In *Proceedings of the Thirteenth Text REtrieval Conference (TREC 2004)*.
- Jinxi Xu, Ana Licuanan, and Ralph Weischedel. 2003. TREC2003 QA at BBN: Answering definitional questions. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*.
- Jun Xu, Yunbo Cao, Hang Li and Min Zhao. 2005. Ranking Definitions with Supervised Learning Methods. In *Proceedings of 14th International World Wide Web Conference (WWW 2005), Industrial and Practical Experience Track*, Chiba, Japan, pp.811-819.
- Zhang D. and Lee WS. 2003. A Language Modeling Approach to Passage Question Answering. In *Proceedings of The 12th Text Retrieval Conference (TREC2003)*, NIST, Gaithersburg.
- Zhai, C, and Lafferty, J. 2001. A Study of Smoothing Methods for Language Models Applied to Information Retrieval. In *Proceedings of the 2001 ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 334-342.

Highly constrained unification grammars

Daniel Feinstein

Department of Computer Science
University of Haifa
31905 Haifa, Israel
daniel@cs.haifa.ac.il

Shuly Wintner

Department of Computer Science
University of Haifa
31905 Haifa, Israel
shuly@cs.haifa.ac.il

Abstract

Unification grammars are widely accepted as an expressive means for describing the structure of natural languages. In general, the recognition problem is undecidable for unification grammars. Even with restricted variants of the formalism, *off-line parsable* grammars, the problem is computationally hard. We present two natural constraints on unification grammars which limit their expressivity. We first show that *non-reentrant* unification grammars generate exactly the class of context-free languages. We then relax the constraint and show that *one-reentrant* unification grammars generate exactly the class of tree-adjoining languages. We thus relate the commonly used and linguistically motivated formalism of unification grammars to more restricted, computationally tractable classes of languages.

1 Introduction

Unification grammars (UG) (Shieber, 1986; Shieber, 1992; Carpenter, 1992) have originated as an extension of context-free grammars, the basic idea being to augment the context-free rules with non context-free annotations (feature structures) in order to express additional information. They can describe phonological, morphological, syntactic and semantic properties of languages simultaneously and are thus linguistically suitable for modeling natural languages. Several formulations of unification grammars have been proposed, and they are used extensively by computational linguists to describe the structure of a variety of natural languages.

Unification grammars are Turing equivalent: determining whether a given string is generated by a given grammar is as hard as deciding whether a Turing machine halts on the empty input (Johnson, 1988). Therefore, the recognition problem for unification grammars is undecidable in the general case. To ensure its decidability, several constraints on unification grammars, commonly known as the *off-line parsability (OLP) constraints*, were suggested, such that the recognition problem is decidable for off-line parsable grammars (Jaeger et al., 2005). The idea behind all the OLP definitions is to rule out grammars which license trees in which unbounded amount of material is generated without expanding the frontier word. This can happen due to two kinds of rules: ϵ -rules (whose bodies are empty) and unit rules (whose bodies consist of a single element). However, even for unification grammars with no such rules the recognition problem is NP-hard (Barton et al., 1987).

In order for a grammar formalism to make predictions about the structure of natural language its generative capacity must be constrained. It is now generally accepted that Context-free Grammars (CFGs) lack the generative power needed for this purpose (Savitch et al., 1987), due to natural language constructions such as reduplication, multiple agreement and crossed agreement. Several linguistic formalisms have been proposed as capable of modeling these phenomena, including Linear Indexed Grammars (LIG) (Gazdar, 1988), Head Grammars (Pollard, 1984), Tree Adjoining Grammars (TAG) (Joshi, 2003) and Combinatory Categorical Grammars (Steedman, 2000). In a seminal work, Vijay-Shanker and Weir (1994) prove that all four formalisms are weakly equivalent. They all generate the class of *mildly context-sensitive languages* (MCSL), all members

of which have recognition algorithms with time complexity $O(n^6)$ (Vijay-Shanker and Weir, 1993; Satta, 1994).¹ As a result of the weak equivalence of four independently developed (and linguistically motivated) extensions of CFG, the class MCSL is considered to be linguistically meaningful, a natural class of languages for characterizing natural languages.

Several authors tried to approximate unification grammars by means of context-free grammars (Rayner et al., 2001; Kiefer and Krieger, 2004) and even finite-state grammars (Pereira and Wright, 1997; Johnson, 1998), but we are not aware of any work which relates unification grammars with the class MCSL. The main objective of this work is to define constraints on UGs which naturally limit their generative capacity. We define two natural and easily testable syntactic constraints on UGs which ensure that grammars satisfying them generate the context-free and the mildly context-sensitive languages, respectively. The contribution of this result is twofold:

- From a theoretical point of view, constraining unification grammars to generate exactly the class MCSL results in a grammatical formalism which is, on one hand, powerful enough for linguists to express linguistic generalizations in, and on the other hand cognitively adequate, in the sense that its generative capacity is constrained;
- Practically, such a constraint can provide efficient recognition algorithms for the limited class of unification grammars.

We define some preliminary notions in section 2 and then show a constrained version of UG which generates the class CFL of context-free languages in section 3. Section 4 presents the main result, namely a restricted version of UG and a mapping of its grammars to LIG, establishing the proposition that such grammars generate exactly the class MCSL. For lack of space, we favor intuitive explanation over rigorous proofs; the full details can be found in Feinstein (2004).

2 Preliminary notions

A CFG is a four-tuple $G^{cf} = \langle V_N, V_t, \mathcal{R}^{cf}, S \rangle$ where V_t is a set of *terminals*, V_N is a set of *non-*

¹The term *mildly context-sensitive* was coined by Joshi (1985), in reference to a less formally defined class of languages. Strictly speaking, what we call MCSL here is also known as the class of *tree-adjoining languages*.

terminals, including the *start symbol* S , and $\mathcal{R}^{cf}$ is a set of productions, assumed to be in a normal form where each rule has either (zero or more) non-terminals or a single terminal in its body, and where the start symbol never occurs in the right hand side of rules. The set of all such context-free grammars is denoted CFGS.

In a linear indexed grammar (LIG),² strings are derived from nonterminals with an associated stack denoted $A[l_1 \dots l_n]$, where A is a nonterminal, each l_i is a stack symbol, and l_1 is the top of the stack. Since stacks can grow to be of unbounded size during a derivation, some way of partially specifying unbounded stacks in LIG productions is needed. We use $A[l_1 \dots l_n \infty]$ to denote the nonterminal A associated with any stack η whose top n symbols are $l_1, l_2, \dots, l_n$. The set of all nonterminals in V_N , associated with stacks whose symbols come from V_s , is denoted $V_N[V_s^*]$.

Definition 1. A **Linear Indexed Grammar** is a five tuple $G^{li} = \langle V_N, V_t, V_s, \mathcal{R}^{li}, S \rangle$ where V_t, V_N and S are as above, V_s is a finite set of indices (stack symbols) and $\mathcal{R}^{li}$ is a finite set of productions in one of the following two forms:

- **fixed stack:** $N_i[p_1 \dots p_n] \rightarrow \alpha$
- **unbounded stack:** $N_i[p_1 \dots p_n \infty] \rightarrow \alpha$ or $N_i[p_1 \dots p_n \infty] \rightarrow \alpha N_j[q_1 \dots q_m \infty] \beta$

where $N_i, N_j \in V_N$, $p_1 \dots p_n, q_1 \dots q_m \in V_s$, $n, m \geq 0$ and $\alpha, \beta \in (V_t \cup V_N[V_s^*])^*$.

A crucial characteristic of LIG is that only *one* copy of the stack can be copied to a *single* element in the body of a rule. If more than one copy were allowed, the expressive power would grow beyond MCSL.

Definition 2. Given a LIG $\langle V_N, V_t, V_s, \mathcal{R}^{li}, S \rangle$, the **derivation relation** $\Rightarrow_{li}$ is defined as follows: for all $\Psi_1, \Psi_2 \in (V_N[V_s^*] \cup V_t)^*$ and $\eta \in V_s^*$,

- If $N_i[p_1 \dots p_n] \rightarrow \alpha \in \mathcal{R}^{li}$ then $\Psi_1 N_i[p_1 \dots p_n] \Psi_2 \Rightarrow_{li} \Psi_1 \alpha \Psi_2$
- If $N_i[p_1 \dots p_n \infty] \rightarrow \alpha \in \mathcal{R}^{li}$ then $\Psi_1 N_i[p_1 \dots p_n \infty] \Psi_2 \Rightarrow_{li} \Psi_1 \alpha \Psi_2$
- If $N_i[p_1 \dots p_n \infty] \rightarrow \alpha N_j[q_1 \dots q_m \infty] \beta \in \mathcal{R}^{li}$ then $\Psi_1 N_i[p_1 \dots p_n \infty] \Psi_2 \Rightarrow_{li} \Psi_1 \alpha N_j[q_1 \dots q_m \infty] \beta \Psi_2$

²The definition is based on Vijay-Shanker and Weir (1994).

The **language** generated by G^{li} is $L(G^{li}) = \{w \in V_t^* \mid S[] \xRightarrow{*}_{li} w\}$, where ' $\xRightarrow{*}_{li}$ ' is the reflexive, transitive closure of ' $\Rightarrow_{li}$ '.

Unification grammars are defined over *feature structures* (FSs) which are directed, connected, rooted, labeled graphs, usually depicted as *attribute-value matrices* (AVM). A feature structure A can be characterized by its set of paths, Π_A , an assignment of atomic values to the ends of some paths, $\Theta_A(\cdot)$, and a reentrancy relation ' $\rightsquigarrow$ ' relating paths which lead to the same node. A sequence of feature structures, where some nodes may be shared by more than one element, is a *multi-rooted structure* (MRS).

Definition 3. *Unification grammars are defined over a signature consisting of a finite set ATOMS of atoms; a finite set FEATS of features and a finite set WORDS of words. A unification grammar is a tuple $G^u = \langle \mathcal{R}^u, A^s, \mathcal{L} \rangle$ where $\mathcal{R}^u$ is a finite set of rules, each of which is an MRS of length $n \geq 1$, $\mathcal{L}$ is a **lexicon**, which associates with every word $w \in \text{WORDS}$ a finite set of feature structures, $\mathcal{L}(w)$, and A^s is a feature structure, the **start symbol**.*

Definition 4. *A unification grammar $\langle \mathcal{R}^u, A^s, \mathcal{L} \rangle$ over the signature $\langle \text{ATOMS}, \text{FEATS}, \text{WORDS} \rangle$ is **non-reentrant** iff for any rule $r^u \in \mathcal{R}^u$, r^u is non-reentrant. It is **one-reentrant** iff for every rule $r^u \in \mathcal{R}^u$, r^u includes at most one reentrancy, between the head of the rule and some element of the body. Let UG_{nr} , UG_{1r} be the sets of all non-reentrant and one-reentrant unification grammars, respectively.*

Informally, a rule is non-reentrant if (on an AVM view) no reentrancy tags occur in it. When the rule is viewed as a (multi-rooted) graph, it is non-reentrant if the in-degree of all nodes is at most 1. A rule is one-reentrant if (on an AVM view) at most one reentrancy tag occurs in it, exactly twice: once in the head of the rule and once in an element of its body. When the rule is viewed as a (multi-rooted) graph, it is one-reentrant if the in-degree of all nodes is at most 1, with the exception of one node whose in-degree can be 2, provided that the only two distinct paths that lead to this node leave from the roots of the head of the rule and an element of the body.

FSs and MRSs are partially ordered by *subsumption*, denoted ' $\sqsubseteq$ '. The least upper bound with respect to subsumption is *unification*, denoted ' $\sqcup$ '. Unification is partial; when $A \sqcup B$ is

undefined we say that the unification *fails* and denote it as $A \sqcup B = \top$. Unification is lifted to MRSs: given two MRSs σ and ρ , it is possible to unify the i -th element of σ with the j -th element of ρ . This operation, called *unification in context* and denoted $(\sigma, i) \sqcup (\rho, j)$, yields two modified variants of σ and ρ : (σ', ρ') .

In unification grammars, *forms* are MRSs. A form $\sigma_A = \langle A_1, \dots, A_k \rangle$ immediately derives another form $\sigma_B = \langle B_1, \dots, B_m \rangle$ (denoted by $\sigma_A \xRightarrow{1}_u \sigma_B$) iff there exists a rule $r^u \in \mathcal{R}^u$ of length n that licenses the derivation. The head of r^u is matched against some element A_i in σ_A using unification in context: $(\sigma_A, i) \sqcup (r^u, 0) = (\sigma'_A, r')$. If the unification does not fail, σ_B is obtained by replacing the i -th element of σ'_A with the body of r' . The reflexive transitive closure of ' $\xRightarrow{1}_u$ ' is denoted by ' $\xRightarrow{*}_u$ '.

Definition 5. *The language of a unification grammar G^u is $L(G^u) = \{w_1 \dots w_n \in \text{WORDS}^* \mid A^s \xRightarrow{*}_u \langle A_1, \dots, A_n \rangle\}$, where $A_i \in \mathcal{L}(w_i)$ for $1 \leq i \leq n$.*

3 Context-free unification grammars

We define a constraint on unification grammars which ensures that grammars satisfying it generate the class CFL. The constraint disallows any reentrancies in the rules of the grammar. When rules are non-reentrant, applying a rule implies that an exact copy of the body of the rule is inserted into the generated (sentential) form, not affecting neighboring elements of the form the rule is applied to. The only difference between rule application in UG_{nr} and the analog operation in CFGS is that the former requires unification whereas the latter only calls for identity check. This small difference does not affect the generative power of the formalisms, since unification can be pre-compiled in this simple case.

The trivial direction is to map a CFG to a non-reentrant unification grammar, since every CFG is, trivially, such a grammar (where terminal and non-terminal symbols are viewed as atomic feature structures). For the inverse direction, we define a mapping from UG_{nr} to CFGS. The non-terminals of the CFG in the image of the mapping are the set of all feature structures defined in the source UG.

Definition 6. *Let $\text{ug2cfg} : \text{UG}_{nr} \mapsto \text{CFGs}$ be a mapping of UG_{nr} to CFGS, such that*

if $G^u = \langle \mathcal{R}^u, A^s, \mathcal{L} \rangle$ is over the signature $\langle \text{ATOMS}, \text{FEATS}, \text{WORDS} \rangle$ then $ug2cfg(G^u) = \langle V_N, V_t, \mathcal{R}^{cf}, S^{cf} \rangle$, where:

- $V_N = \{A_i \mid A_0 \rightarrow A_1 \dots A_n \in \mathcal{R}^u, i \geq 0\} \cup \{A \mid A \in \mathcal{L}(a), a \in \text{ATOMS}\} \cup \{A^s\}$. V_N is the set of all the feature structures occurring in any of the rules or the lexicon of G^u .
- $S^{cf} = A^s$ • $V_t = \text{WORDS}$
- $\mathcal{R}^{cf}$ consists of the following rules:
 1. Let $A_0 \rightarrow A_1 \dots A_n \in \mathcal{R}^u$ and $B \in \mathcal{L}(b)$. If for some $i, 1 \leq i \leq n, A_i \sqcup B \neq \top$, then $A_i \rightarrow b \in \mathcal{R}^{cf}$
 2. If $A_0 \rightarrow A_1 \dots A_n \in \mathcal{R}^u$ and $A^s \sqcup A_0 \neq \top$ then $S^{cf} \rightarrow A_1 \dots A_n \in \mathcal{R}^{cf}$.
 3. Let $r_1^u = A_0 \rightarrow A_1 \dots A_n$ and $r_2^u = B_0 \rightarrow B_1 \dots B_m$, where $r_1^u, r_2^u \in \mathcal{R}^u$. If for some $i, 1 \leq i \leq n, A_i \sqcup B_0 \neq \top$, then the rule $A_i \rightarrow B_1 \dots B_m \in \mathcal{R}^{cf}$

The size of $ug2cfg(G^u)$ is polynomial in the size of G^u . By inductions on the lengths of the derivation sequences, we prove the following theorem:

Theorem 1. If $G^u = \langle \mathcal{R}^u, A^s, \mathcal{L} \rangle$ is a non-reentrant unification grammar and $G^{cf} = ug2cfg(G^u)$, then $L(G^{cf}) = L(G^u)$.

Corollary 2. Non-reentrant unification grammars are weakly equivalent to CFGS.

4 Mildly context-sensitive UG

In this section we show that *one-reentrant unification grammars* generate exactly the class MCSL. In such grammars each rule can have at most one reentrancy, reflecting the LIG situation where stacks can be copied to exactly one daughter in each rule.

4.1 Mapping LIG to UG_{1r}

In order to simulate a given LIG with a unification grammar, a dedicated signature is defined based on the parameters of the LIG.

Definition 7. Given a LIG $\langle V_N, V_t, V_s, \mathcal{R}^{li}, S \rangle$, let τ be $\langle \text{ATOMS}, \text{FEATS}, \text{WORDS} \rangle$, where $\text{ATOMS} = V_N \cup V_s \cup \{\text{elist}\}$, $\text{FEATS} = \{\text{HEAD}, \text{TAIL}\}$, and $\text{WORDS} = V_t$.

We use τ throughout this section as the signature over which UGs are defined. We use FSs over

the signature τ to represent and simulate LIG symbols. In particular, FSs will encode lists in the natural way, hence the features HEAD and TAIL. For the sake of brevity, we use standard list notation when FSs encode lists. LIG symbols are mapped to FSs thus:

Definition 8. Let $toFs$ be a mapping of LIG symbols to feature structures, such that:

1. If $t \in V_t$ then $toFs(t) = \langle t \rangle$
2. If $N \in V_N$ and $p_i \in V_s, 1 \leq i \leq n$, then $toFs(N[p_1, \dots, p_n]) = \langle N, p_1, \dots, p_n \rangle$

The mapping $toFs$ is extended to sequences of symbols by setting $toFs(\alpha\beta) = toFs(\alpha)toFs(\beta)$. Note that $toFs$ is one to one.

When FSs that are images of LIG symbols are concerned, unification is reduced to identity:

Lemma 3. Let $X_1, X_2 \in V_N[V_s^*] \cup V_t$. If $toFs(X_1) \sqcup toFs(X_2) \neq \top$ then $toFs(X_1) = toFs(X_2)$.

When a feature structure which is represented as an unbounded list (a list that is not terminated by *elist*) is unifiable with an image of a LIG symbol, the former is a prefix of the latter.

Lemma 4. Let $C = \langle p_1, \dots, p_n, \boxed{i} \rangle$ be a non-reentrant feature structure, where $p_1, \dots, p_n \in V_s$, and let $X \in V_N[V_s^*] \cup V_t$. Then $C \sqcup toFs(X) \neq \top$ iff $toFs(X) = \langle p_1, \dots, p_n, \alpha \rangle$, for some $\alpha \in V_s^*$.

To simulate LIGs with UGs we represent each symbol in the LIG as a feature structure, encoding the stack of LIG non-terminals as lists. Rules that propagate stacks (from mother to daughter) are simulated by means of reentrancy in the UG.

Definition 9. Let $lig2ug$ be a mapping of LIGs to UG_{1r} , such that if $G^{li} = \langle V_N, V_t, V_s, \mathcal{R}^{li}, S \rangle$ and $G^u = \langle \mathcal{R}^u, A^s, \mathcal{L} \rangle = lig2ug(G^{li})$ then G^u is over the signature τ (definition 7), $A^s = toFs(S[\])$, for all $t \in V_t, \mathcal{L}(t) = \{toFs(t)\}$ and $\mathcal{R}^u$ is defined by:

- A LIG rule of the form $X_0 \rightarrow \alpha$ is mapped to the unification rule $toFs(X_0) \rightarrow toFs(\alpha)$
- A LIG rule of the form $N_i[p_1, \dots, p_n \infty] \rightarrow \alpha \ N_j[q_1, \dots, q_m \infty] \ \beta$ is mapped to the unification rule $\langle N_i, p_1, \dots, p_n, \boxed{1} \rangle \rightarrow toFs(\alpha) \ \langle N_j, q_1, \dots, q_m, \boxed{1} \rangle \rightarrow toFs(\beta)$

Evidently, $lig2ug(G^{li}) \in UG_{1r}$ for any LIG G^{li} .

Theorem 5. If $G^{li} = \langle V_N, V_t, V_s, \mathcal{R}^{li}, S^{li} \rangle$ is a *LIG* and $G^u = \text{lig2ug}(G^{li})$ then $L(G^u) = L(G^{li})$.

4.2 Mapping UG_{1r} to *LIG*

We are now interested in the reverse direction, namely mapping *UGs* to *LIG*. Of course, since *UGs* are more expressive than *LIGs*, only a subset of the former can be correctly simulated by the latter. The differences between the two formalisms can be summarized along three dimensions:

The basic elements *UG* manipulates feature structures, and rules (and forms) are *MRSs*; whereas *LIG* manipulates terminals and non-terminals with stacks of elements, and rules (and forms) are sequences of such symbols.

Rule application In *UG* a rule is applied by *unification in context* of the rule and a sentential form, both of which are *MRSs*, whereas in *LIG*, the head of a rule and the selected element of a sentential form must have the same non-terminal symbol and consistent stacks.

Propagation of information in rules In *UG* information is shared through reentrancies, whereas In *LIG*, information is propagated by copying the stack from the head of the rule to one element of its body.

We show that one-reentrant *UGs* can all be correctly mapped to *LIG*. For the rest of this section we fix a signature $\langle \text{ATOMS}, \text{FEATS}, \text{WORDS} \rangle$ over which *UGs* are defined. Let *NRFSS* be the set of all non-reentrant *FSs* over this signature.

One-reentrant *UGs* induce highly constrained (sentential) forms: in such forms, there are no reentrancies whatsoever, neither between distinct elements nor within a single element. Hence all the *FSs* in forms induced by a one-reentrant *UG* are non-reentrant.

Definition 10. Let *A* be a feature structure with no reentrancies. The **height** of *A*, denoted $|A|$, is the length of the longest path in *A*. This is well-defined since non-reentrant feature structures are acyclic. Let $G^u = \langle \mathcal{R}^u, A^s, \mathcal{L} \rangle \in \text{UG}_{1r}$ be a one-reentrant unification grammar. The **maximum height** of the grammar, $\text{maxHt}(G^u)$, is the height of the highest feature structure in the grammar. This is well defined since all the feature structures of one-reentrant grammars are non-reentrant.

The following lemma indicates an important property of one-reentrant *UGs*. Informally, in any *FS* that is an element of a sentential form induced by such grammars, if two paths are long (specifically, longer than the maximum height of the grammar), they must have a long common prefix.

Lemma 6. Let $G^u = \langle \mathcal{R}^u, A^s, \mathcal{L} \rangle \in \text{UG}_{1r}$ be a one-reentrant unification grammar. Let *A* be an element of a sentential form induced by G^u . If $\pi \cdot \langle F_j \rangle \cdot \pi_1, \pi \cdot \langle F_k \rangle \cdot \pi_2 \in \Pi_A$, where $F_j, F_k \in \text{FEATS}$, $j \neq k$ and $|\pi_1| \leq |\pi_2|$, then $|\pi_1| \leq \text{maxHt}(G^u)$.

Lemma 6 facilitates a view of all the *FSs* induced by such a grammar as (unboundedly long) lists of elements drawn from a finite, predefined set. The set consists of all features in *FEATS* and all the non-reentrant feature structures whose height is limited by the maximal height of the unification grammar. Note that even with one-reentrant *UGs*, feature structures can be unboundedly deep. What lemma 6 establishes is that if a feature structure induced by a one-reentrant unification grammar is deep, then it can be represented as a *single* “core” path which is long, and all the sub-structures which “hang” from this core are depth-bounded. We use this property to encode such feature structures as *cords*.

Definition 11. Let $\Psi : \text{NRFSS} \times \text{PATHS} \mapsto (\text{FEATS} \cup \text{NRFSS})^*$ be a mapping such that if *A* is a non-reentrant *FS* and $\pi = \langle F_1, \dots, F_n \rangle \in \Pi_A$, then the **cord** $\Psi(A, \pi)$ is $\langle A_1, F_1, \dots, A_n, F_n, A_{n+1} \rangle$, where for $1 \leq i \leq n+1$, A_i are non-reentrant *FSs* such that:

- $\Pi_{A_i} = \{ \langle G \rangle \cdot \pi \mid \langle F_1, \dots, F_{i-1}, G \rangle \cdot \pi \in \Pi_A, i \leq n, G \neq F_i \} \cup \{ \varepsilon \}$
- $\Theta_{A_i}(\pi) = \Theta_A(\langle F_1, \dots, F_{i-1} \rangle \cdot \pi)$ (if it is defined).

We also define $\text{last}(\Psi(A, \pi)) = A_{n+1}$. The **height** of a cord is defined as $|\Psi(A, \pi)| = \max_{1 \leq i \leq n+1} (|A_i|)$. For each cord $\Psi(A, \pi)$ we refer to *A* as the **base feature structure** and to π as the **base path**. The **length** of a cord is the length of the base path.

The function Ψ is one to one: given $\Psi(A, \pi)$, both *A* and π are uniquely determined.

Lemma 7. Let G^u be a one-reentrant unification grammar and let *A* be an element of a sentential form induced by G^u . Then there is a path $\pi \in \Pi_A$ such that $|\Psi(A, \pi)| < \text{maxHt}(G^u)$.

Lemma 7 implies that every non-reentrant FS (i.e., FSs induced by one-reentrant grammars) can be represented as a height-limited cord. This mapping resolves the first difference between LIG and UG, by providing a representation of the *basic elements*. We use cords as the stack contents of LIG non-terminals: cords can be unboundedly long, but so can LIG stacks; the crucial point is that cords are height limited, implying that they can be represented using a *finite* number of elements.

We now show how to simulate, in LIG, the unification in context of a rule and a sentential form. The first step is to have exactly one non-terminal symbol (in addition to the start symbol); when all non-terminal symbols are identical, only the content of the stack has to be taken into account. Recall that in order for a LIG rule to be applicable to a sentential form, the stack of the rule's head must be a *prefix* of the stack of the selected element in the form. The only question is whether the two stacks are equal (fixed rule head) or not (unbounded rule head). Since the contents of stacks are cords, we need a property relating two cords, on one hand, with unifiability of their base feature structures, on the other. Lemma 8 establishes such a property. Informally, if the base path of one cord is a prefix of the base path of the other cord and all feature structures along the common path of both cords are unifiable, then the base feature structures of both cords are unifiable. The reverse direction also holds.

Lemma 8. *Let $A, B \in \text{NRFSS}$ be non-reentrant feature structures and $\pi_1, \pi_2 \in \text{PATHS}$ be paths such that $\pi_1 \in \Pi_B$, $\pi_1 \cdot \pi_2 \in \Pi_A$, $\Psi(A, \pi_1 \cdot \pi_2) = \langle t_1, F_1, \dots, F_{|\pi_1|}, t_{|\pi_1|+1}, F_{|\pi_1|+1}, \dots, t_{|\pi_1 \cdot \pi_2|+1} \rangle$, $\Psi(B, \pi_1) = \langle s_1, F_1, \dots, s_{|\pi_1|+1} \rangle$, and $\langle F_{|\pi_1|+1} \rangle \notin \Pi_{s_{|\pi_1|+1}}$. Then $A \sqcup B \neq \top$ iff for all i , $1 \leq i \leq |\pi_1| + 1$, $s_i \sqcup t_i \neq \top$.*

The length of a cord of an element of a sentential form induced by the grammar cannot be bounded, but the length of any cord representation of a rule head is limited by the grammar height. By lemma 8, unifiability of two feature structures can be reduced to a comparison of two cords representing them and only the prefix of the longer cord (as long as the shorter cord) affects the result. Since the cord representation of any grammar rule's head is limited by the height of the grammar we always choose it as the shorter cord in the comparison.

We now define, for a feature structure C (which is a head of a rule) and some path π , the set that

includes all feature structures that are both unifiable with C and can be represented as a cord whose height is limited by the grammar height and whose base path is π . We call this set the *compatibility set* of C and π and use it to define the set of all possible prefixes of cords whose base FSs are unifiable with C (see definition 13). Crucially, the compatibility set of C is finite for any feature structure C since the heights and the lengths of the cords are limited.

Definition 12. *Given a non-reentrant feature structure C , a path $\pi = \langle F_1, \dots, F_n \rangle \in \Pi_C$ and a natural number h , the **compatibility set**, $\Gamma(C, \pi, h)$, is defined as the set of all feature structures A such that $C \sqcup A \neq \top$, $\pi \in \Pi_A$, and $|\Psi(A, \pi)| \leq h$.*

The compatibility set is defined for a feature structure and a given path (when h is taken to be the grammar height). We now define two similar sets, FH and UH, for a given FS, independently of a path. When rules of a one-reentrant unification grammar are mapped to LIG rules (definition 14), FH and UH are used to define heads of fixed and unbounded LIG rules, respectively. A single unification rule is mapped to a set of LIG rules, each with a different head. The stack of the head is some member of the sets FH and UH. Each such member is a prefix of the stack of potential elements of sentential forms that the LIG rule can be applied to.

Definition 13. *Let C be a non-reentrant feature structure and h be a natural number. Then:*

$$\begin{aligned} \text{FH}(C, h) &= \{ \Psi(A, \pi) \mid \pi \in \Pi_C, A \in \Gamma(C, \pi, h) \} \\ \text{UH}(C, h) &= \{ \Psi(A, \pi) \cdot \langle F \rangle \mid \Psi(A, \pi) \in \text{FH}(C, h), \\ &\quad \Theta_C(\pi) \uparrow, F \in \text{FEATS}, \text{val}(\text{last}(\Psi(C \sqcup A, \pi)), \langle F \rangle) \uparrow \} \end{aligned}$$

This accounts for the second difference between LIG and one-reentrant UG, namely *rule application*. We now briefly illustrate our account of the last difference, *propagation of information in rules*. In UG_{1r} information is shared between the rule's head and a single element in its body. Let $r^u = \langle C_0, \dots, C_n \rangle$ be a reentrant unification rule in which the path μ_e , leaving the e -th element of the body, is reentrant with the path μ_0 leaving the head. This rule is mapped to a set of LIG rules, corresponding to the possible rule heads induced by the compatibility set of C_0 . Let r be a member of this set, and let X_0 and X_e be the head and the e -th element of r , respectively. Reentrancy in r^u is modeled in the LIG rule by copying the stack from X_0 to X_e . The major complication is the contents

of this stack, which varies according to the cord representations of C_0 and C_e and to the reentrant paths.

Summing up, in a LIG simulating a one-reentrant UG, FSs are represented as stacks of symbols. The set of stack symbols V_s , therefore, is defined as a set of height bounded non-reentrant FSs. Also, all the features of the UG are stack symbols. V_s is finite due to the restriction on FSs (no reentrancies and height-boundedness). The set of terminals, V_t , is the words of the UG. There are exactly two non-terminal symbols, S (the start symbol) and N .

The set of rules is divided to four. The *start rule* only applies once in a derivation, simulating the situation in UGs of a rule whose head is unifiable with the start symbol. *Terminal rules* are a straight-forward implementation of the lexicon in terms of LIG. *Non-reentrant rules* are simulated in a similar way to how rules of a non-reentrant UG are simulated by CFG (section 3). The major difference is the head of the rule, X_0 , which is defined as explained above. *One-reentrant rules* are simulated similarly to non-reentrant ones, the only difference being the selected element of the rule body, X_e , which is defined as follows.

Definition 14. Let *ug2lig* be a *mapping of* UG_{1r} to LIGs, such that if $G^u = \langle \mathcal{R}^u, A^s, \mathcal{L} \rangle \in UG_{1r}$ then $ug2lig(G^u) = \langle V_N, V_t, V_s, \mathcal{R}^{li}, S \rangle$, where $V_N = \{N, S\}$ (fresh symbols), $V_t = \text{WORDS}$, $V_s = \text{FEATS} \cup \{A \mid A \in \text{NRFSS}, |A| \leq \text{maxHt}(G^u)\}$, and $\mathcal{R}^{li}$ is defined as follows:³

1. $S[] \rightarrow N[\Psi(A^s, \varepsilon)]$
2. For every $w \in \text{WORDS}$ such that $\mathcal{L}(w) = \{C_0\}$ and for every $\pi_0 \in \Pi_{C_0}$, the rule $N[\Psi(C_0, \pi_0)] \rightarrow w$ is in $\mathcal{R}^{li}$.
3. If $\langle C_0, \dots, C_n \rangle \in \mathcal{R}^u$ is a non-reentrant rule, then for every $X_0 \in \text{LIGHEAD}(C_0)$ the rule $X_0 \rightarrow N[\Psi(C_1, \varepsilon)] \dots N[\Psi(C_n, \varepsilon)]$ is in $\mathcal{R}^{li}$.
4. Let $r^u = \langle C_0, \dots, C_n \rangle \in \mathcal{R}^u$ and $(0, \mu_0) \xrightarrow{r^u} (e, \mu_e)$, where $1 \leq e \leq n$. Then for every $X_0 \in \text{LIGHEAD}(C_0)$ the rule

$$\begin{array}{c} X_0 \rightarrow N[\Psi(C_1, \varepsilon)] \dots N[\Psi(C_{e-1}, \varepsilon)] \\ X_e \\ N[\Psi(C_{e+1}, \varepsilon)] \dots N[\Psi(C_n, \varepsilon)] \end{array}$$

³For a non-reentrant FS C_0 , we define: $\text{LIGHEAD}(C_0)$ as $\{N[\eta] \mid \eta \in \text{FH}(C_0, \text{maxHt}(G^u))\} \cup \{N[\eta \infty] \mid \eta \in \text{UH}(C_0, \text{maxHt}(G^u))\}$

is in $\mathcal{R}^{li}$, where X_e is defined as follows. Let π_0 be the base path of X_0 and A be the base feature structure of X_0 . Applying the rule r^u to A , define $(\langle A \rangle, 0) \sqcup (r^u, 0) = (\langle P_0 \rangle, \langle P_0, \dots, P_e, \dots, P_n \rangle)$.

- (a) If μ_0 is not a prefix of π_0 then $X_e = N[\Psi(P_e, \mu_e)]$.
- (b) If $\pi_0 = \mu_0 \cdot \nu$, $\nu \in \text{PATHS}$ then
 - i. If $X_0 = N[\Psi(A, \pi_0)]$ then $X_e = N[\Psi(P_e, \mu_e \cdot \nu)]$.
 - ii. If $X_0 = N[\Psi(A, \pi_0), F \infty]$ then $X_e = N[\Psi(P_e, \mu_e \cdot \nu), F \infty]$.

By inductions on the lengths of the derivations we prove that the mapping is correct:

Theorem 9. If $G^u \in UG_{1r}$, then $L(G^u) = L(ug2lig(G^u))$.

5 Conclusions

The main contribution of this work is the definition of two constraints on unification grammars which dramatically limit their expressivity. We prove that non-reentrant unification grammars generate exactly the class of context-free languages; and that one-reentrant unification grammars generate exactly the class of mildly context-sensitive languages. We thus obtain two linguistically plausible constrained formalisms whose computational processing is tractable.

This main result is primarily a formal grammar result. However, we maintain that it can be easily adapted such that its consequences to (practical) computational linguistics are more evident. The motivation behind this observation is that reentrancy only adds to the expressivity of a grammar formalism when it is potentially *unbounded*, i.e., when infinitely many feature structures can be the possible values at the end of the reentrant paths. It is therefore possible to modestly extend the class of unification grammars which can be shown to generate exactly the class of mildly context-sensitive languages, by allowing also a limited form of multiple reentrancies among the elements in a rule (e.g., to handle agreement phenomena). This can be most useful for grammar writers, and at the same time adds nothing to the expressivity of the formalism. We leave the formal details of such an extension to future work.

This work can also be extended in other directions. The mapping of one-reentrant UGs to LIG is highly verbose, resulting in LIGs with a huge

number of rules. We believe that it should be possible to optimize the mapping such that much smaller grammars are generated. In particular, we are looking into mappings of one-reentrant UGs to other MCSL formalisms, notably TAG.

The two constraints on unification grammars (non-reentrant and one-reentrant) are parallel to the first two classes of the Weir (1992) hierarchy of languages. A possible extension of this work could be a definition of constraints on unification grammars that would generate all the classes of the hierarchy. Another direction is an extension of one-reentrant unification grammars, where the reentrancy does not have to be between the head and one element of the body. Also of interest are two-reentrant unification grammars, possibly with limited kinds of reentrancies.

Acknowledgments

This research was supported by The Israel Science Foundation (grant no. 136/01). We are grateful to Yael Cohen-Sygal, Nissim Francez and James Rogers for their comments and help.

References

- G. Edward Barton, Jr., Robert C. Berwick, and Eric Sven Ristad. 1987. The complexity of LFG. In G. Edward Barton, Jr., Robert C. Berwick, and Eric Sven Ristad, editors, *Computational Complexity and Natural Language*, Computational Models of Cognition and Perception, chapter 3, pages 89–102. MIT Press, Cambridge, MA.
- Bob Carpenter. 1992. *The Logic of Typed Feature Structures*. Cambridge University Press.
- Daniel Feinstein. 2004. Computational investigation of unification grammars. Master's thesis, University of Haifa.
- Gerald Gazdar. 1988. Applicability of indexed grammars to natural languages. In Uwe Reyle and Christian Rohrer, editors, *Natural Language Parsing and Linguistic Theories*, pages 69–94. Reidel.
- Efrat Jaeger, Nissim Francez, and Shuly Wintner. 2005. Unification grammars and off-line parsability. *Journal of Logic, Language and Information*, 14(2):199–234.
- Mark Johnson. 1988. *Attribute-Value Logic and the Theory of Grammar*, volume 16 of *CSLI Lecture Notes*. CSLI, Stanford, California.
- Mark Johnson. 1998. Finite-state approximation of constraint-based grammars using left-corner grammar transforms. In *Proceedings of the 17th international conference on Computational linguistics*, pages 619–623.
- Aravind K. Joshi. 1985. Tree Adjoining Grammars: How much context Sensitivity is required to provide a reasonable structural description. In D. Dowty, I. Karttunen, and A. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge, U.K.
- Aravind K. Joshi. 2003. Tree-adjoining grammars. In Ruslan Mitkov, editor, *The Oxford handbook of computational linguistics*, chapter 26, pages 483–500. Oxford university Press.
- Bernd Kiefer and Hans-Ulrich Krieger. 2004. A context-free superset approximation of unification-based grammars. In Harry Bunt, John Carroll, and Giorgio Satta, editors, *New Developments in Parsing Technology*, pages 229–250. Kluwer Academic Publishers.
- Fernando C. N. Pereira and Rebecca N. Wright. 1997. Finite-state approximation of phrase-structure grammars. In Emmanuel Roche and Yves Schabes, editors, *Finite-State Language Processing*, Language, Speech and Communication, chapter 5, pages 149–174. MIT Press, Cambridge, MA.
- Carl Pollard. 1984. *Generalized phrase structure grammars, head grammars and natural language*. Ph.D. thesis, Stanford University.
- Manny Rayner, John Dowding, and Beth Ann Hockey. 2001. A baseline method for compiling typed unification grammars into context free language models. In *Proceedings of EUROSPEECH 2001*, Aalborg, Denmark.
- Giorgio Satta. 1994. Tree-adjoining grammar parsing and boolean matrix multiplication. In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, volume 20.
- Walter J. Savitch, Emmon Bach, William Marsh, and Gila Safran-Naveh, editors. 1987. *The formal complexity of natural language*, volume 33 of *Studies in Linguistics and Philosophy*. D. Reidel, Dordrecht.
- Stuart M. Shieber. 1986. *An Introduction to Unification Based Approaches to Grammar*. Number 4 in *CSLI Lecture Notes*. CSLI.
- Stuart M. Shieber. 1992. *Constraint-Based Grammar Formalisms*. MIT Press, Cambridge, Mass.
- Mark Steedman. 2000. *The Syntactic Process*. Language, Speech and Communication. The MIT Press, Cambridge, Mass.
- K. Vijay-Shanker and David J. Weir. 1993. Parsing some constrained grammar formalisms. *Computational Linguistics*, 19(4):591 – 636.
- K. Vijay-Shanker and David J. Weir. 1994. The equivalence of four extensions of context-free grammars. *Mathematical systems theory*, 27:511–545.
- David J. Weir. 1992. A geometric hierarchy beyond context-free languages. *Theoretical Computer Science*, 104:235–261.

A polynomial parsing algorithm for the topological model

Synchronizing Constituent and Dependency Grammars, Illustrated by German Word Order Phenomena

Kim Gerdes

ERSS (CNRS) & Signes (Inria),
Université Bordeaux 3

kim.gerdes@u-bordeaux3.fr

Sylvain Kahane

Modyco (CNRS),
Université Paris 10

sk@ccr.jussieu.fr

Abstract

This paper describes a minimal topology driven parsing algorithm for topological grammars that synchronizes a rewriting grammar and a dependency grammar, obtaining two linguistically motivated syntactic structures. The use of non-local *slash* and *visitor* features can be restricted to obtain a CKY type analysis in polynomial time. German long distance phenomena illustrate the algorithm, bringing to the fore the procedural needs of the analyses of syntax-topology mismatches in constraint based approaches like for example HPSG.

1 Introduction

In this paper we investigate the computational problems resulting from the discrepancy between the surface organization of a sentence and its syntactic structure expressing functional relations between words. These syntactic phenomena are addressed under terms like unbounded dependencies or scrambling. This discrepancy is the source of complex mechanisms such as movement in Chomskyan models, functional uncertainty in the functional equations of LFG, or non local features (like slash) in HPSG. Algorithms for these mechanisms are NP-complete, but introducing upper bounds on simultaneous mismatches makes these phenomena parsable in polynomial time. We show that the resulting complexity is of $O(n^{K+5})$ with K being this upper

bound.¹

We consider the so-called topological grammars characterized by the synchronization of two algebraic grammars (equivalent CFG), one generating the surface structure (the topological phrase structure), the other generating the deeper structure (the functional or dependency structure). The parallel construction of the surface and the deeper structure minimally handles the problem of the discrepancy encountered in all formal descriptions of non-local word order phenomena.

The lexicalized version of topological grammars presented in this paper lends itself well to be used in a CKY type algorithm. The description of this algorithm sheds light on the procedural role of the concept underlying the *slash* feature and the dual role of the *visitor* feature introduced here (following Hudson 2000).

Formal topological grammars have been introduced independently by Debusmann & Duchier 2001 and Gerdes & Kahane 2001. Reape 1994 and Kathol 1995 have formalized the classical concept of topology in HPSG without, however, explicitly considering an independent topological phrase structure. More or less complete topological grammars have been developed for Czech, Dutch, German, Modern Greek, Korean, Arabic, and French (Bojar 2004, Yoo 2003, Yoo & Gerdes 2004, El Kassas 2005, Gerdes & Kahane 2006). Different implementations for topological grammars exist: TDG (Debusmann et al. 2003) uses the unspecialized Oz constraint based language, making TDG parsing NP-hard as show Koller & Striegnitz 2002 (although one obtains polynomial time in the average case, Duchier

¹ The linguistic relevance of this upper bound has been discussed for example in Becker *et al.* 1991, arguing that Tree Adjoining Grammar capture the performance limit on extractions in German.

2003); DepLin is a generation system based on a procedural linearization grammar (Gerdes & Yoo 2003), and Clément et al. 2002 and Frank 2003 translate topological grammars into the LFG opposition between c- and f-structures allowing the use of LFG analyzers (for which exist NP hardness results). To our knowledge, the HPSG approach has not been implemented. None of these approaches has explored the theoretical algorithmic properties of an exact topological analysis (leaving aside stochastic approaches). However, the theoretical problems of the upper bounds for the number of extractions have been studied in neighboring dependency-based formalisms by Kahane et al. 1998 and Bröcker 2000.

Our contribution shows how the topology-syntax discrepancy can minimally be seen as two grammars that synchronously construct two independent structures. We believe that the introduction of the dual slash and visitor features allows for a better comprehension of the procedural mechanisms at stake in the unbounded dependencies handling and the (not necessarily apparent) similarities between formalisms such as LFG, HPSG, or dependency grammars.

Section 2 presents the grammar formalism, exemplifying it by a German toy grammar and Section 3 is devoted to the parsing algorithm we propose.

2 The grammar formalism

Our grammar contains three modules: a syntactic grammar, a topological grammar, and the topology-syntax interface. We will present these three modules, exemplifying each of them by a toy grammar for German. Although very simple, this grammar covers a great part of the verbal syntax of German (Bech 1955), including the main scrambling phenomena. For a more complete grammar of German and for grammars of other languages in the same theoretical framework, see the references in our introduction.

It should be noted that the formalism we propose here differs slightly from the previous formal presentation of the topological model, bringing to the fore the synchronization of two grammars and the interface grammar.

2.1 The syntactic grammar

The syntactic module is a classic dependency grammar and generates unordered dependency trees. The parameters to instantiate are the vo-

cabulary V , the set of (lexical) categories C , the initial category IC , the set of syntactic roles R , and the set of lexical rules. A lexical rule assigns a category and a valence list to a word. A valence slot is a couple (r, C) where r is a syntactic role and C a category.² The initial category IC gives the category of the root of the dependency tree.

Example

V = the German words

$C = \{ V_{fin}, V_{zu}, V_{inf}, V_{pp}, N \}$

(V_{fin} = finite verb, V_{zu} = infinitive with *zu*, V_{inf} = bare infinitive, V_{pp} = past participle)³

$R = \{ subj, obj, vcomp \}$

$IC = V_{fin}$

Dependency rules

hat ‘has’: $V_{fin}, val: \langle (subj, N_{nom}), (vcomp, V_{pp}) \rangle$

gelesen ‘read’: $V_{pp}, val: \langle (obj, N_{acc}) \rangle$

The last rule says that *gelesen* is a past participle governing a nominal object at the accusative case. Our grammar generates dependency trees such as Fig. 1 for the sentence (1):

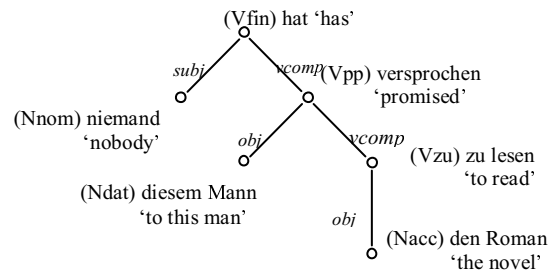


Fig. 1. A dependency tree

- (1) *Den Roman hat diesem Mann niemand zu lesen versprochen*
the novel has-to this man nobody
to read promised
‘Nobody promised to this man to read the novel’.

² We do not present the treatment of modifiers when the governor is selected by the dependent. This does not pose any technical problems but it necessitates particular rules that we will not present here (for the treatment of modifier in a dependency grammar see for example Nasr 1995; various propositions in HPSG can also be adapted here). Neither do we expatiate upon the optionality of some syntactic arguments.

³ For the sake of simplicity we give a very rough presentation of the category. For nouns, cases are added in their names (N_{nom} , N_{gen} , N_{dat} , and N_{acc}).

2.2 The topological grammar

The topological grammar proper generates the topological structures, which are ordered constituent trees. Such a grammar differs slightly from traditional CFGs by distinguishing constituents from positions for constituents, i.e. boxes (= topological constituent) and fields (= positions in a box). A grammar rule indicates for each box its list of fields and how many boxes it can contain for each field. Three values for the filling parameter of a field are possible: exactly one element (!), at most one element (?) and any number of elements (*). A field is called obligatory if its filling parameter is !.

The parameters to instantiate are the set of box names B , the set of field names F , the initial box (ib), and the set of rules.

For the sake of simplicity, in this presentation, we adopt a flat structure, with exactly one box headed by each word of the sentence. We lose a part of the economy of the system (that reuses the same boxes at different levels) but the presentation of the parsing algorithm will be clearer.

Example

$B = \{ \text{md, ed, vc, np} \}$

(md = main domain, ed = embedded domain, vc = verb cluster, np = nominal phrase)

$F = \{ \text{vf, mf, nf, rb, of, } \bullet \}$

(vf = Vorfeld, mf = Mittelfeld, nf = Nachfeld, rb = right bracket, of = Oberfeld, $\bullet$ = head field)

ib = md

Topological rules

$\text{md} \rightarrow \text{vf}^! \bullet \text{mf}^* \text{rb}^? \text{nf}^*$

$\text{ed} \rightarrow \text{mf}^* \text{of}^? \bullet \text{nf}^*$

$\text{vc} \rightarrow \text{of}^? \bullet$

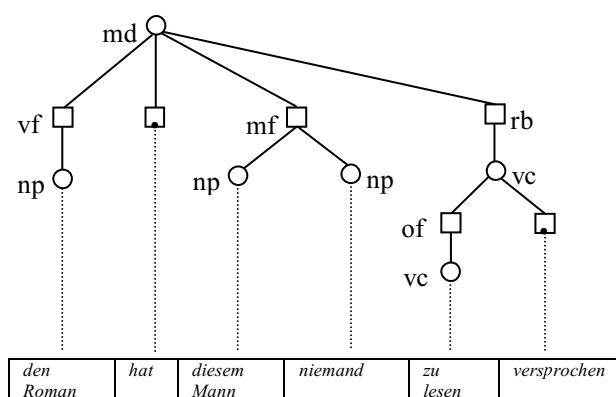


Fig. 2. The topological structure of (1)

Our first rule is the classical topological model of German: a main domain is composed of five

fields and the main verb occupies the second field, the first field vf containing exactly one element. In the embedded domain, the head occupies the right bracket, which is then the head field. A verb in the right bracket offers a place to its left called the Oberfeld (of) for a verbal dependent. Fig. 2 is a graphical representation of the topological structure of the non marked sentence (1). Boxes are represented by circles and fields by squares.

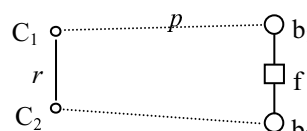
Topological rules only describe the field structure of the boxes. Rules stating which word can head which box in which field are part of the interface grammar. Without these rules our topological grammar overgenerates, allowing every box to go into any field.

2.3 The topology-syntax interface

The topology-syntax interface synchronizes the syntactic grammar and the topological grammar. An interface rule associates the positioning of a dependency node with the positioning of the corresponding topological box.

Each box $b \in B$ is associated to an integer $p(b)$, called its permeability, controlling which constituent can emancipate from it.

An interface rule is a 7-tuple $(C_1, r, C_2, b_1, f, b_2, p)$, where $C_1, C_2 \in C$, $r \in R$, $b_1, b_2 \in B$, $f \in F$, and p is an integer called the permeability level. The rule can be read in two equivalent ways: 1) if a word w_2 of category (cat) C_2 depends on a word w_1 of cat C_1 by a syntactic relation r , then w_2 can head a box b_2 placed in a field f of a box b_1 containing w_1 and separated from b_1 by boxes of permeability $\leq p$ (synthesis reading); 2) if a word w_2 of cat C_2 heads a box b_2 placed in a field f of a box b_1 containing a word w_1 of cat C_1 and separated from b_1 by boxes of permeability $\leq p$, then w_2 can depend on w_1 by a syntactic relation r (analysis reading). The rule is schematized in the following figure:



Dotted lines represent synchronizing links. If a word w labeling a dependency node synchronizes with a box b , we say that w heads b .

Our three grammars conjointly produce a dependency tree and a topological tree whose nodes are synchronized. The two structures are generated parallelly, each one constraining the

other. Synchronizing links are part of the generated language (Kahane 2006).

Example

Permeability

$p(vc)=1, p(ed)=p(xp)=2, p(md)=3$.

Interface rules

For a noun we have one basic rule: a noun depending on a verb can head an NP in any major field (vf/mf/nf) wherever is the verb;⁴ it can cross over vc and ed boxes:

$(V, subj/obj, N, md/ed/vc, vf/mf/nf, np, 2)$

Note that in German, contrarily to English, the placement of a NP does not actually depend on its syntactic role.

For a non-finite verb we have two rules:⁵

- one is similar to the rule for nouns: a verb can head an embedded domain in any major field:
 $(V, vcomp, V\text{-}fin, md/ed/vc, vf/mf/nf, ed, 2)$
- the other is specific to the German(ic) syntax: a non-finite verb can be placed in the right bracket (rb) or at the left of its verbal governor (in the of field) if this governor is already in the right bracket:
 $(V, vcomp, V\text{-}fin, md/ed/vc, of/rb, vc, 0)$

This last rule can be applied recursively, forming a string of verbs called a verb cluster. The dependents of verbs of a same cluster can be freely shared out in the major fields of the same domain. This property produces what is called scrambling. In our model this requires an extensive use of emancipation, allowing any dependent of a verb to be placed in a domain headed by a verbal ancestor. Emancipation is also possible outside the embedded domain, although this would require specific strong information packaging constraints (not reflected in this toy grammar).

3. The parsing algorithm

We begin with a presentation of the algorithm when there is no emancipation. In this case the topological structure and the dependency structure are built in parallel, i.e. each combination of linear segments corresponds to a functional com-

bination. When emancipations are allowed the parsing will be driven by the topological structure only.⁶

3.1 The algorithm without emancipation

The philosophy of a CKY algorithm is to begin parsing one word segments of the sentence, to store the minimum of information in a parse matrix, and to parse bigger and bigger segments by concatenation of segments previously parsed.

In the algorithm for CFG, if we have two consecutive segments from i to j and from $j+1$ to k of cat C_1 and C_2 and if we have a rule $C \rightarrow C_1 C_2$, then we postulate a segment from i to k of cat C . The recurrence step is then:

$$[i,j,cat:C_1] \oplus [j+1,k,cat:C_2] \otimes (C \rightarrow C_1 C_2) = [i,k,cat:C]$$

In our case, the entries of our parse matrix are of the form $[i,j,cat:C, val:X, box:b, fields:Y]$ where i and j delimit the segment, C is the category of the head, X is the list of free valence slots, b is the topological box name of the segment, and Y the list of non-saturated fields of b (including the head field •, indicating which fields are on the left or on the right)

Initialization step

If the i -th word of the sentence we want to parse can have 1) the cat C , 2) the valence X and 3) a word of cat C can head a box b and if 4) there is a topological rule $b \rightarrow Y$, then we store the segment $[i,i,cat:C, val:X, box:b, fields:Y]$.

Recurrence step

We combine two consecutive segments by applying an interface rule. One of the two segments must be saturated which means: 1) all the valence slots of the head have been filled (and thus val is an empty list) ; 2) all the fields of the box are potentially saturated, that is, there is no field with the value ! left.

In the following recurrence step, we suppose that the second segment is saturated and we note *elist* the empty valence list and *sat* any saturated field list.

⁴ We do not develop the nominal topology in this example grammar.

⁵ An initial rule states that the root $Vfin$ heads the initial box md .

⁶ This distinguishes our approach from similar approaches such as Kahane et al. 1998, which do not explicitly consider a separate topological structure. Contrarily to this work, where each emancipation is handled by a dedicated lifting rule, we do not really consider projectivity as the more normal case and we treat emancipated and non-emancipated elements more or less in the same manner.

$[i,j,cat:C_1,val:X,box:b_1,fields:Y]$
 $\oplus [j+1,k,cat:C_2,val:elist,box:b_2,fields:sat]$
 $\otimes (C_1, r, C_2, b_1, f, b_2, p)$
 $= [i,k,cat:C_1,val:X \ominus \langle r,C_2 \rangle, box:b_1, fields:Y \oplus f]$

This step is possible if X contains a valence slot (r,C_2) and then $X \ominus \langle r,C_2 \rangle$ is the list X reduced by (r,C_2) . In the same way, Y must contain a field f at the left of the head field; the fields between $\bullet$ and f must be non obligatory and are suppressed in $Y \oplus f$; moreover, the filling parameter of f is adjusted according to the fact that f now contains a box.

The parsing succeeds if our parse matrix contains at least one segment $[1,n,cat:IC,val:elist,box:ib,fields:sat]$. If we keep backpointers at each step in the algorithm, we have a compact representation of the parse forest.

3.2 The algorithm with emancipation

An emancipated constituent is not in the maximal projection of its governor, i.e. it is not in the box headed by its governor. Consider the following example of partial VP-fronting:

- (2) *Gelesen hat Maria den Roman.*
 read has Maria the novel
 ‘Maria read the novel’

In (2), the past participle *gelesen* heads an embedded domain in vf. Its object *den Roman* is emancipated and placed in the field mf of the main domain. Suppose we want to apply our previous algorithm (the CKY parsing without emancipation). We can easily parse the segments *gelesen*, *hat Maria*, and *den Roman*, but neither *gelesen* and *hat Maria* (*gelesen* not saturated), nor *hat Maria* and *den Roman* (no valence for *den Roman*) can be combined.

Our parsing will be driven by the topological structure and the condition of the topological saturation of the dependent is maintained. Two cases of combination of segments are possible.

The first case is illustrated by the combination between *hat Maria* and *gelesen*, where *gelesen* still expects a dependent. Therefore we do not require the valence of the topological phrase to be saturated and we must percolate it in a special feature similar to the slash feature of G/HPSG (Gazdar et al. 1985, Pollard & Sag 1994)

The second case is illustrated by the combination between *den Roman* and *hat Maria*. In this case we do not trigger a correspondence rule because no dependency must be built. We must

store *den Roman* in a special feature we call *visitor* (see Hudson 2000 for a similar device), which is the converse of the slash feature. The slash feature allows us to lift up a need (a valence slot to be filled), while the visitor feature allows handing down a resource (that will fill a valence slot). Or more precisely in our case: The visitor allows a governor to keep a non-solicited segment while waiting for an element that can take this element in its valence.

Note however that, with our conditions on the saturation of topological constituents, the two strategies are not interchangeable and they are both necessary. Let us consider two new examples.

- (3) *Maria hat den Roman gelesen.*
 Maria has the novel read
 ‘Maria read the novel’

Although the sentence (3) is projective, *den Roman* must be analyzed as an emancipated constituent. Indeed, *gelesen* is in the right bracket of the main domain and the maximal projection of *gelesen*, the verb cluster, does not contain its dependent *den Roman*, which is in the Mittelfeld of the main domain headed by *hat*. From a topological point of view, *den Roman* can only combine with *hat* but it is not in the valence of *hat* and it must be considered a visitor.

- (4) *Ich glaube, dass den Roman Maria*
 I think that the novel Maria
gelesen hat.
 read has

‘I think that Maria read the novel’

In (4), *gelesen hat* forms a verb cluster in the right bracket of a complementizer phrase. The NP *den Roman* is still emancipated and, from the topological viewpoint, it cannot combine with its governor *gelesen*. It also cannot combine with *hat* because they are separated by *Maria* and *gelesen*. The smallest topological phrase containing *den Roman* and *gelesen* also contains *hat*. Therefore the slash strategy is needed and *gelesen* and *hat* must combine before combining with *den Roman* (and *Maria*).

We make two major changes in our previous algorithm. On the one hand we replace the val feature by two new features:

- the vis(itor) feature, which stores triples (C,b,f) indicating that a box b of head C has been placed in the field f ;

- the slash feature, which stores valence slots of the head as well as the non saturated slots of its dependents.

On the other hand we proceed in two steps: first we combine consecutive segments without triggering interface rules, by storing all information in our vis and slash features; second we trigger interface rules to reduce the contents of vis and slash features.

Initialization step

The segment $[i, i, \text{cat}:C, \text{val}:X, \text{box}:b, \text{fields}:Y]$ of the previous algorithm is replaced by $[i, i, \text{cat}:C, \text{vis}:elist, \text{slash}:X', \text{box}:b, \text{fields}:Y]$ where each valence slot (r, C') of X gives a slash slot $(C, r, C', 0)$, with C the cat of the head and 0 indicating that this slash slot has not emancipated across any box.

Combination step

The combination is driven by the topological structure, so one of the two segments combined must be topologically saturated (fields:sat) but we no longer require that the valence of this segment is saturated: its free valence slot will be slashed.

$$\begin{aligned} & [i, j, \text{cat}:C_1, \text{vis}:Z, \text{slash}:X_1, \text{box}:b_1, \text{fields}:Y] \oplus \\ & [j+1, k, \text{cat}:C_2, \text{vis}:elist, \text{slash}:X_2, \text{box}:b_2, \text{fields}:sat] \\ & = [i, k, \text{cat}:C_1, \text{vis}:Z \oplus (C_2, f, b_2), \text{slash}:X_1 \oplus X_2', \\ & \text{box}:b_1, \text{fields}:Y \oplus f] \end{aligned}$$

where each 4-tuple (C, r, C', p) of X_2 gives a 4-tuple (C, r, C', p') with p' the max of p and $p(b_1)$, the permeability of b_1 .

In other words the second segment is placed in the field f and this is stored in the visitor feature (cf. (C_2, f, b_2)). At the same time the slash content X_2 of the second segment is added to the slash content of the first segment, but we must indicate that these valence slots have crossed the box b_1 and this is why the permeability level is adjusted.

Reduction step

$$\begin{aligned} & [i, j, \text{cat}:C, \text{vis}:Z \oplus (C_2, f, b_2), \text{slash}:X \oplus \\ & (C_1, r, C_2, p'), \text{box}:b_1, \text{fields}:Y] \\ & \otimes (C_1, r, C_2, b_1, f, b_2, p) \\ & = [i, j, \text{cat}:C, \text{vis}:Z, \text{val}:X, \text{box}:b_1, \text{fields}:Y] \\ & \text{provided that } p' \leq p. \end{aligned}$$

A reduction is possible if vis and slash contain elements referring to the same category C_2 : the vis element (C_2, f, b_2) says that we have encountered a box b_2 of head C_2 in the field f , while the slash element (C_1, r, C_2, p') indicates that a word of cat C_2 is required to fill the r valence slot of a word of cat C_1 . If furthermore the slashed slot

has not crossed over boxes of permeability greater than p ($p' \leq p$), then the interface rule $(C_1, r, C_2, b_1, f, b_2, p)$ can apply and the segment can be reduced.

The parsing succeeds if the parse matrix contains a segment $[1, n, \text{cat}:IC, \text{vis}:elist, \text{slash}:elist, \text{box}:ib, \text{fields}:sat]$, where n is the length of the sentence.

Example: Parsing of *Den Roman hat Maria zu lesen versprochen* (cf. (1)). We focus on the combination of the segment *den Roman hat Maria* (where *den Roman* is a visitor) with the verb cluster *zu lesen versprochen*.

den Roman hat Maria: $S_1 = [1, 4, \text{cat}:Vfin, \text{vis}:<(Nacc, vf, np)>, \text{slash}:<(Vfin, aux, Vpp, 0)>, \text{box}:md, \text{fields}: \bullet mf^* rb^* nf^*]$

zu lesen versprochen: $S_2 = [5, 7, \text{cat}:Vpp, \text{vis}:elist, \text{slash}:<(Vzu, dobj, Nacc, 1)>, \text{box}:vc, \text{fields}: \bullet]$

The segment S_2 is topologically saturated, so S_1 and S_2 can combine.

$$\begin{aligned} S &= S_1 \oplus S_2 = \\ & [1, 7, \text{cat}:Vfin, \text{vis}:<(Nacc, vf, np), (Vpp, rb, vc)>, \\ & \text{slash}:<(Vfin, aux, Vpp, 0), (Vzu, obj, Nacc, 1)>, \\ & \text{box}:md, \text{fields}: \bullet nf^*] \end{aligned}$$

S can be reduced twice

- by merging $(aux, Vfin, Vpp, 0)$ and (Vpp, rb, vc) using the interface rule $(V, aux, Vfin, md, rb, vc, 0)$
- and by merging $(Vzu, dobj, Nacc, vc)$ and $(Nacc, vf, np)$ using the interface rule $(V, dobj, N, md, vf, vc, 2)$.

After reduction, $S = [1, 7, \text{cat}:Vfin, \text{vis}:elist, \text{slash}:elist, \text{box}:md, \text{fields}: \bullet nf^*]$, proving that (1) is a grammatical sentence of German.

3.3 Complexity

The parse matrix of a CKY algorithm has less than n^2 entries, where n is the length of the sentence parsed. In the CFG case, the number of possible segment descriptions is bounded by the number of categories, which we call C . To fill a new entry in the parse matrix at least n combinations of two entries must be considered and the number of operations is bounded by GC^2n^3 , where G is the number of rules (each combination involves a grammar rule).

In the algorithm without emancipation the number of possible descriptions of segments is still

bounded and the complexity remains $O(n^3)$. However if we want to retrieve the dependency forest (Nasr 2003) we need to store backpointers, that is the place of the head in each segment description; the number of segment descriptions goes up by a factor n and the complexity becomes $O(n^5)$ ⁷.

The slash and visitor features are more expensive: We assume the slash and visitor sets to be bounded by K , i.e. we suppose that we do not need to keep more than K entries in the slash and visitor sets at a time. Consequently the number of segment descriptions remains bounded by CV^K , where V is the number of valence slot descriptions, and the algorithm complexity is still of type $O(n^3)$. But if we introduce backpointers to retrieve the dependency forest, we need to keep them in valence slot descriptions in order to remember which word has a valence slot to fill. The number of segment descriptions is thus bounded by $CV^K n^{K+1}$ and the time complexity of the algorithm is in $O(n^{K+5})$. We avoid exponential growth only because we restrict the number of slash and visitor entries of each configuration.

4 Conclusion

We have proposed a parsing algorithm for the topological model that is minimal in the sense that its additional exponential growth (the factor K) corresponds exactly to the number of mismatches between functional dependency and topological constituency. In different terms, given we want to construct the topological and the dependency structures, two independent and linguistically significant structures, and given two separate grammars expressing the constraints on the construction of these structures, then the cost of adding the interface constraints is exponential precisely in the number of memory positions needed in order to keep track of the differences between the two structures. Precise parsing of these three grammars cannot do with less.

We could hypothesize that the need of differentiating more or less independent levels of syntactic analysis (e.g. surface vs. deep structure, ...), which is at the origin of formalisms like LFG or HPSG, algorithmically boils down to this exponentiality, at least concerning the syntax/surface

expressive needs. It might be interesting to compare these results with efficiency considerations for HPSG as in Nishida et al. 2001 and for restricted graph grammars for dependency-orientated generation (Bohnet & Wanner 2001). Our bottom-up strategy driven by the topological structure forces us to introduce tools equivalent to the slash feature of G/HPSG. We hope that this presentation sheds light on the procedural role of the slash feature, and on the complementary possibility of a linguistic analysis using a visitor feature.

It should be noted that, in spite of its simplicity, the German topological grammar presented as an example allows the control of syntactic constraints on phenomena like scrambling, partial VP fronting, and auxiliary flip, which demonstrates the expressivity of the topological approach. The grammars of languages like Czech and Modern Greek show that the topological approach allows for a straightforward integration of information structure in the interface constraints.

Work is in progress on experimental implementations of the presented algorithm and on choosing useful and linguistically accessible input and output formats. Real values on efficiency will not be available as long as the grammar does not surpass experimental size. A linguistic study on corpora might determine what types of elements are actually emancipated and in particular what types of elements can be emancipated simultaneously, i.e. what list of slashed element are possible, given that this is the main factor of complexity of the algorithm (see Kiefer et al. 1999 for similar heuristic considerations for HPSG parsing).

Acknowledgements

We would like to thank Renaud Marlet and the three anonymous reviewers for their vast comments and constructive criticism.

References

- Bech, Gunnar, 1955, *Studien über das deutsche Verbum infinitum*, 2nd edition 1983, Linguistische Arbeiten 139, Niemeyer, Tübingen.
- Becker, Tilman, Aravind K. Joshi, Owen Rambow, 1991, Long-Distance Scrambling and Tree Adjoining Grammars. *EACL 1991*
- Bohnet, Bernd and Leo Wanner, 2001, On Using a Parallel Graph Rewriting Grammar Formalism in Generation. *Proceedings of the 8th European Na-*

⁷ We do not attempt to further optimize our algorithm. Following Eisner 2000's ideas it seems to be possible to constrain the complexity to $O(n^3)$.

- Natural Language Generation Workshop (at ACL)*, Toulouse.
- Bojar, Ondřej, 2004, Problems of Inducing Large Coverage Constraint-Based Dependency Grammar for Czech. *International Workshop on Constraint Solving and Language Processing*, Universitet, Roskilde, pp. 29-42.
- Bröker, Norbert, 1998, Separating Surface Order and Syntactic Relations in a Dependency Grammars, *COLING-ACL '98*, 174-180.
- Debusmann, Ralph, Denys Duchier and Joachim Niehren, 2004, The XDG Grammar Development Kit, *Second International Mozart/Oz Conference*, Charleroi
- Drach, Erich, 1937, *Grundgedanken der deutschen Satzlehre*, Diesterweg, Frankfurt/M..
- Duchier, Denys, Ralph Debusmann, 2001, Topological Dependency Trees: A Constraint-Based Account of Linear Precedence, *ACL 2001*, 180-87.
- Duchier, Denys, 2003, Configuration of labelled trees under lexicalized constraints and principles, *Journal of Research on Language and Computation*.
- El Kassas, Dina, Sylvain Kahane, 2004, Modélisation de l'ordre des mots en arabe standard, *JEP-TALN, Workshop on Arabic Language Processing*, Fez, 259-264.
- Eisner, Jason (2000). Bilexical grammars and their cubic-time parsing algorithms. In Harry Bunt and Anton Nijholt (eds.), *Advances in Probabilistic and Other Parsing Technologies*, pages 29-62. Kluwer.
- Frank, Anette, 2003, Projecting LFG F-Structures from Chunks. *LFG 2003*, Saratoga Springs, New York, 217-237.
- Gazdar, Gerald, Ewan Klein, Geoffrey Pullum and Ivan Sag, 1985, *Generalized Phrase structure grammar*, Harvard University Press, Cambridge.
- Gerdes, Kim, Sylvain Kahane, 2001, Word Order in German: A Formal Dependency Grammar Using a Topological Hierarchy, *ACL 2001*, 220-27.
- Gerdes, Kim, Sylvain Kahane, 2006, L'amas verbal au cœur d'une modélisation topologique, *Linguisticae Investigationes*, 29:1, 101-114.
- Gerdes, Kim & Hi-Yon Yoo, 2003, La topologie comme interface entre syntaxe et prosodie, un système de génération appliqué au grec moderne, *TALN 2003*, Batz-sur-Mer, 125-134.
- Hudson Richard, 2000, "Discontinuity", *Dependency Grammars, T.A.L.*, 41:1, Hermès, Paris, 15-56.
- Kahane Sylvain, 2006, Polarized Unification Grammars, *Coling-ACL '06*, Sydney, 8 p.
- Kahane, Sylvain, Alexis Nasr, Owen Rambow, 1998, Pseudo-Projectivity: a Polynomially Parsable Non-Projective Dependency Grammar, *COLING-ACL '98*, Montreal, 646-52.
- Kathol Andreas, 1995, *Linearization-based German Syntax*, PhD thesis, Ohio State University.
- Kiefer, B., H.-U. Krieger, J. Carroll, and R. Malouf, 1999, A bag of useful techniques for efficient and robust parsing. *ACL 1999*, 473-480.
- Alexander Koller and Kristina Striegnitz. 2002. Generation as dependency parsing. *ACL 2002*.
- Nasr Alexis, 1995, A formalism and a parser for Lexicalised Dependency Grammars. *4th Int. Workshop on Parsing Technologies*, SUNY Press.
- Nasr, Alexis, 2003, Factoring surface syntactic structures. *First International Conference on Meaning-Text Theory*, Paris, 249-258.
- Nishida, Kenji, Kentaro Torisawa and Jun'ichi Tsujii, 2001, Compiling an HPSG-based grammar into more than one CFG. *PACLING 2001*, 199-206.
- Nivre, Joakim. Jens Nilsson, 2005, Pseudo-Projective Dependency Parsing, *ACL 2005*, 99-106.
- Pollard, C. and I. Sag, 1994, *Head-Driven Phrase Structure Grammar*, CSLI Publications.
- Yoo, Hiyon & Kim Gerdes, 2004, A dependency account of Korean Word Order, *Linguistic Society of Korea 2004*, Seoul.
- Yoo, Hiyon 2003, *Ordre des mots et prosodie : Essai de description et de formalisation pour le grec moderne*, PhD thesis, Université Paris 7.

Stochastic Language Generation Using WIDL-expressions and its Application in Machine Translation and Summarization

Radu Soricut

Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
radu@isi.edu

Daniel Marcu

Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
marcu@isi.edu

Abstract

We propose WIDL-expressions as a flexible formalism that facilitates the integration of a generic sentence realization system within end-to-end language processing applications. WIDL-expressions represent compactly probability distributions over finite sets of candidate realizations, and have optimal algorithms for realization via interpolation with language model probability distributions. We show the effectiveness of a WIDL-based NLG system in two sentence realization tasks: automatic translation and headline generation.

1 Introduction

The Natural Language Generation (NLG) community has produced over the years a considerable number of generic sentence realization systems: Penman (Matthiessen and Bateman, 1991), FUF (Elhadad, 1991), Nitrogen (Knight and Hatzivassiloglou, 1995), Fergus (Bangalore and Rambow, 2000), HALogen (Langkilde-Geary, 2002), Amalgam (Corston-Oliver et al., 2002), etc. However, when it comes to end-to-end, text-to-text applications – Machine Translation, Summarization, Question Answering – these generic systems either cannot be employed, or, in instances where they can be, the results are significantly below that of state-of-the-art, application-specific systems (Hajic et al., 2002; Habash, 2003). We believe two reasons explain this state of affairs.

First, these generic NLG systems use input representation languages with complex syntax and semantics. These languages involve deep, semantic-based subject-verb or verb-object relations (such as ACTOR, AGENT, PATIENT, etc., for Penman

and FUF), syntactic relations (such as *subject*, *object*, *premod*, etc., for HALogen), or lexical dependencies (Fergus, Amalgam). Such inputs cannot be accurately produced by state-of-the-art analysis components from arbitrary textual input in the context of text-to-text applications.

Second, most of the recent systems (starting with Nitrogen) have adopted a hybrid approach to generation, which has increased their robustness. These hybrid systems use, in a first phase, symbolic knowledge to (over)generate a large set of candidate realizations, and, in a second phase, statistical knowledge about the target language (such as stochastic language models) to rank the candidate realizations and find the best scoring one. The disadvantage of the hybrid approach – from the perspective of integrating these systems within end-to-end applications – is that the two generation phases cannot be tightly coupled. More precisely, input-driven preferences and target language-driven preferences cannot be integrated in a true probabilistic model that can be trained and tuned for maximum performance.

In this paper, we propose WIDL-expressions (WIDL stands for Weighted Interleave, Disjunction, and Lock, after the names of the main operators) as a representation formalism that facilitates the integration of a generic sentence realization system within end-to-end language applications. The WIDL formalism, an extension of the IDL-expressions formalism of Nederhof and Satta (2004), has several crucial properties that differentiate it from previously-proposed NLG representation formalisms. First, it has a simple syntax (expressions are built using four operators) and a simple, formal semantics (probability distributions over finite sets of strings). Second, it is a compact representation that grows linearly

in the number of words available for generation (see Section 2). (In contrast, representations such as word lattices (Knight and Hatzivassiloglou, 1995) or non-recursive CFGs (Langkilde-Geary, 2002) require exponential space in the number of words available for generation (Nederhof and Satta, 2004).) Third, it has good computational properties, such as optimal algorithms for intersection with n -gram language models (Section 3). Fourth, it is flexible with respect to the amount of linguistic processing required to produce WIDL-expressions directly from text (Sections 4 and 5). Fifth, it allows for a tight integration of input-specific preferences and target-language preferences via interpolation of probability distributions using log-linear models. We show the effectiveness of our proposal by directly employing a generic WIDL-based generation system in two end-to-end tasks: machine translation and automatic headline generation.

2 The WIDL Representation Language

2.1 WIDL-expressions

In this section, we introduce WIDL-expressions, a formal language used to *compactly* represent probability distributions over finite sets of strings.

Given a finite alphabet of symbols Σ , atomic WIDL-expressions are of the form a , with $a \in \Sigma$. For a WIDL-expression $\omega = a$, its semantics is a probability distribution $\sigma_{\text{widl}}(\omega) : \text{Dom}_\omega \rightarrow [0, 1]$, where $\text{Dom}_\omega = \{a\}$ and $\sigma_{\text{widl}}(\omega)(a) = 1$. Complex WIDL-expressions are created from other WIDL-expressions, by employing the following four operators, as well as operator distribution functions δ_i from an alphabet Δ .

Weighted Disjunction. If $\omega_1, \dots, \omega_n$ are WIDL-expressions, then $\omega = \vee_{\delta_0}(\omega_1, \dots, \omega_n)$, with $\delta_0 : \{1, \dots, n\} \rightarrow [0, 1]$, specified such that $\sum_{k \in \text{dom}(\delta_0)} \delta_0(k) = 1$, is a WIDL-expression. Its semantics is a probability distribution $\sigma_{\text{widl}}(\omega) : \text{Dom}_\omega \rightarrow [0, 1]$, where $\text{Dom}_\omega = \bigcup_{i=1}^n \text{Dom}_{\omega_i}$, and the probability values are induced by δ_0 and $\sigma_{\text{widl}}(\omega_i)$, $1 \leq i \leq n$. For example, if $\omega = \vee_{\delta_0}(a, b)$, $\delta_0 = \{1 \rightarrow 0.8, 2 \rightarrow 0.2\}$, its semantics is a probability distribution $\sigma_{\text{widl}}(\omega)$ over $\text{Dom}_\omega = \{a, b\}$, defined by $\sigma_{\text{widl}}(\omega)(a) = \delta_0(1) = 0.8$ and $\sigma_{\text{widl}}(\omega)(b) = \delta_0(2) = 0.2$.

Precedence. If ω_1, ω_2 are WIDL-expressions, then $\omega = \omega_1 \cdot \omega_2$ is a WIDL-expression. Its

semantics is a probability distribution $\sigma_{\text{widl}}(\omega) : \text{Dom}_\omega \rightarrow [0, 1]$, where Dom_ω is the set of all strings that obey the precedence imposed over the arguments, and the probability values are induced by $\sigma_{\text{widl}}(\omega_1)$ and $\sigma_{\text{widl}}(\omega_2)$. For example, if $\omega_1 = \vee_{\delta_1}(a, b)$, $\delta_1 = \{1 \rightarrow 0.8, 2 \rightarrow 0.2\}$, and $\omega_2 = \vee_{\delta_2}(c, d)$, $\delta_2 = \{1 \rightarrow 0.6, 2 \rightarrow 0.4\}$, then $\omega = \omega_1 \cdot \omega_2$ represents a probability distribution $\sigma_{\text{widl}}(\omega)$ over the set $\text{Dom}_\omega = \{ac, ad, bc, bd\}$, defined by $\sigma_{\text{widl}}(\omega)(ac) = \delta_1(1)\delta_2(1) = 0.48$, $\sigma_{\text{widl}}(\omega)(ad) = \delta_1(1)\delta_2(2) = 0.32$, etc.

Weighted Interleave. If $\omega_1, \dots, \omega_n$ are WIDL-expressions, then $\omega = \parallel_{\delta_0}(\omega_1, \omega_2, \dots, \omega_n)$, with $\delta_0 : S \cup \{\text{other perms}\} \cup \{\text{shuffles}\} \rightarrow [0, 1]$, $S \subseteq \text{Perm}_n$, specified such that $\sum_{a \in \text{dom}(\delta_0)} \delta_0(a) = 1$, is a WIDL-expression. Its semantics is a probability distribution $\sigma_{\text{widl}}(\omega) : \text{Dom}_\omega \rightarrow [0, 1]$, where Dom_ω consists of all the possible interleavings of strings from Dom_{ω_i} , $1 \leq i \leq n$, and the probability values are induced by δ_0 and $\sigma_{\text{widl}}(\omega_i)$. The distribution function δ_0 is defined either explicitly, over $S \subseteq \text{Perm}_n$ (the set of all permutations of n elements), or implicitly, as $\delta_0(\text{other perms})$. Because the set of argument permutations is a subset of all possible interleavings, δ_0 also needs to specify the probability mass for the strings that are not argument permutations, $\delta_0(\text{shuffles})$. For example, if $\omega = \parallel_{\delta_0}(a \cdot b, c)$, $\delta_0 = \{1 \ 2 \rightarrow 0.80, \text{other perms} \xrightarrow{\text{uniform}} 0.15, \text{shuffles} \xrightarrow{\text{uniform}} 0.05\}$, its semantics is a probability distribution $\sigma_{\text{widl}}(\omega)$, with domain $\text{Dom}_\omega = \{abc, cab, acb\}$, defined by $\sigma_{\text{widl}}(\omega)(abc) = \delta_0(1 \ 2) = 0.80$, $\sigma_{\text{widl}}(\omega)(cab) = \frac{\delta_0(\text{other perms})}{1} = 0.15$, $\sigma_{\text{widl}}(\omega)(acb) = \frac{\delta_0(\text{shuffles})}{1} = 0.05$.

Lock. If ω' is a WIDL-expression, then $\omega = \times(\omega')$ is a WIDL-expression. The semantic mapping $\sigma_{\text{widl}}(\omega)$ is the same as $\sigma_{\text{widl}}(\omega')$, except that Dom_ω contains strings in which no additional symbol can be interleaved. For example, if $\omega = \parallel_{\delta_0}(\times(a \cdot b), c)$, $\delta_0 = \{1 \ 2 \rightarrow 0.80, \text{other perms} \rightarrow 0.20\}$, its semantics is a probability distribution $\sigma_{\text{widl}}(\omega)$, with domain $\text{Dom}_\omega = \{cab, abc\}$, defined by $\sigma_{\text{widl}}(\omega)(abc) = \delta_0(1 \ 2) = 0.80$, $\sigma_{\text{widl}}(\omega)(cab) = \frac{\delta_0(\text{other perms})}{1} = 0.20$.

In Figure 1, we show a more complex WIDL-expression. The probability distribution δ_1 associated with the operator $\parallel_{\delta_1}$ assigns probability 0.2 to the argument order 2 1 3; from a probability mass of 0.7, it assigns uniformly, for each of the remaining $3! - 1 = 5$ argument permutations, a permutation probability value of $\frac{0.7}{5} = 0.14$. The

$$\|_{\delta_1}(\times(\text{turkish} \cdot \text{government}), \vee_{\delta_2}(\times(\text{rebels} \cdot \text{fighting}), \times(\text{attacked} \cdot \text{rebels})), \text{in} \cdot \text{iraq}), \\ \delta_1 = \{2 \ 1 \ 3 \rightarrow 0.2, \text{ other perms } \xrightarrow{\text{uniform}} 0.7, \text{ shuffles } \xrightarrow{\text{uniform}} 0.1\}, \delta_2 = \{1 \rightarrow 0.65, 2 \rightarrow 0.35\}$$

Figure 1: An example of a WIDL-expression.

remaining probability mass of 0.1 is left for the 12 shuffles associated with the unlocked expression $\text{in} \cdot \text{iraq}$, for a shuffle probability of $\frac{0.1}{12} = 0.008$. The list below enumerates some of the $\langle \text{string}, p(\text{string}) \rangle$ pairs that belong to the probability distribution defined by our example:

rebels fighting turkish government in iraq	0.130
in iraq attacked rebels turkish goverment	0.049
in turkish goverment iraq rebels fighting	0.005

The following result characterizes an important representation property for WIDL-expressions.

Theorem 1 *A WIDL-expression ω over Σ and Δ using n atomic expressions has space complexity $O(n)$, if the operator distribution functions of ω have space complexity at most $O(n)$.*

For proofs and more details regarding WIDL-expressions, we refer the interested reader to (Soricut, 2006). Theorem 1 ensures that high-complexity hypothesis spaces can be represented efficiently by WIDL-expressions (Section 5).

2.2 WIDL-graphs and Probabilistic Finite-State Acceptors

WIDL-graphs. Equivalent at the representation level with WIDL-expressions, WIDL-graphs allow for formulations of algorithms that process them. For each WIDL-expression ω , there exists an equivalent WIDL-graph G_ω . As an example, we illustrate in Figure 2(a) the WIDL-graph corresponding to the WIDL-expression in Figure 1. WIDL-graphs have an initial vertex v_s and a final vertex v_e . Vertices v_0, v_6 , and v_{20} with in-going edges labeled $\vdash_{\delta_1}^1, \vdash_{\delta_1}^2$, and $\vdash_{\delta_1}^3$, respectively, and vertices v_5, v_{19} , and v_{23} with out-going edges labeled $\dashv_{\delta_1}^1, \dashv_{\delta_1}^2$, and $\dashv_{\delta_1}^3$, respectively, result from the expansion of the $\|_{\delta_1}$ operator. Vertices v_7 and v_{13} with in-going edges labeled $(\frac{1}{\delta_2}), (\frac{2}{\delta_2})$, respectively, and vertices v_{12} and v_{18} with out-going edges labeled $)_{\delta_2}^1,)_{\delta_2}^2$, respectively, result from the expansion of the $\vee_{\delta_2}$ operator.

With each WIDL-graph G_ω , we associate a probability distribution. The domain of this distribution is the finite collection of strings that can be generated from the paths of a WIDL-specific

traversal of G_ω , starting from v_s and ending in v_e . Each path (and its associated string) has a probability value induced by the probability distribution functions associated with the edge labels of G_ω . A WIDL-expression ω and its corresponding WIDL-graph G_ω are said to be equivalent because they represent the same distribution $\sigma_{\text{widl}}(\omega)$.

WIDL-graphs and Probabilistic FSA. Probabilistic finite-state acceptors (pFSA) are a well-known formalism for representing probability distributions (Mohri et al., 2002). For a WIDL-expression ω , we define a mapping, called UNFOLD, between the WIDL-graph G_ω and a pFSA A_ω . A state s in A_ω is created for each set of WIDL-graph vertices that can be reached simultaneously when traversing the graph. State s records, in what we call a $\|$ -stack (interleave stack), the order in which $\vdash_{\delta}^i, \dashv_{\delta}^i$ -bordered sub-graphs are traversed. Consider Figure 2(b), in which state $[v_0 v_9 v_{23}, \{\langle \delta_1 3 \ 2 \rangle\}]$ (at the bottom) corresponds to reaching vertices v_0, v_9 , and v_{23} (see the WIDL-graph in Figure 2(a)), by first reaching vertex v_{23} (inside the $\vdash_{\delta_1}^3, \dashv_{\delta_1}^3$ -bordered sub-graph), and then reaching vertex v_9 (inside the $\vdash_{\delta_1}^2, \dashv_{\delta_1}^2$ -bordered sub-graph).

A transition labeled a between two A_ω states s_1 and s_2 in A_ω exists if there exists a vertex v_j in the description of s_1 and a vertex v_k in the description of s_2 such that there exists a path in G_ω between v_j and v_k , and a is the only Σ -labeled transitions in this path. For example, transition $[v_0 v_9 v_{23}, \{\langle \delta_1 3 \ 2 \rangle\}] \xrightarrow{\text{rebels}} [v_0 v_{19} v_{23}, \{\langle \delta_1 3 \ 2 \rangle\}]$ (Figure 2(b)) results from unfolding the path $v_9 \xrightarrow{\varepsilon} v_{10} \xrightarrow{\text{rebels}} v_{11} \xrightarrow{\varepsilon} v_{12} \xrightarrow{(\frac{1}{\delta_2})} v_{19}$ (Figure 2(a)). A transition labeled ε between two A_ω states s_1 and s_2 in A_ω exists if there exists a vertex v_j in the description of s_1 and vertices $v_k^1, \dots, v_k^n$ in the description of s_2 , such that $v_j \xrightarrow{\vdash_{\delta}^i} v_k^i \in G_\omega, 1 \leq i \leq n$ (see transition $[v_s,] \xrightarrow{\varepsilon} [v_0 v_6 v_{20}, \{\langle \delta_1 \rangle \delta_1 \}]$), or if there exists vertices $v_j^1, \dots, v_j^n$ in the description of s_1 and vertex v_k in the description of s_2 , such that $v_j^i \xrightarrow{\dashv_{\delta}^i} v_k \in G_\omega, 1 \leq i \leq n$. The ε -transitions

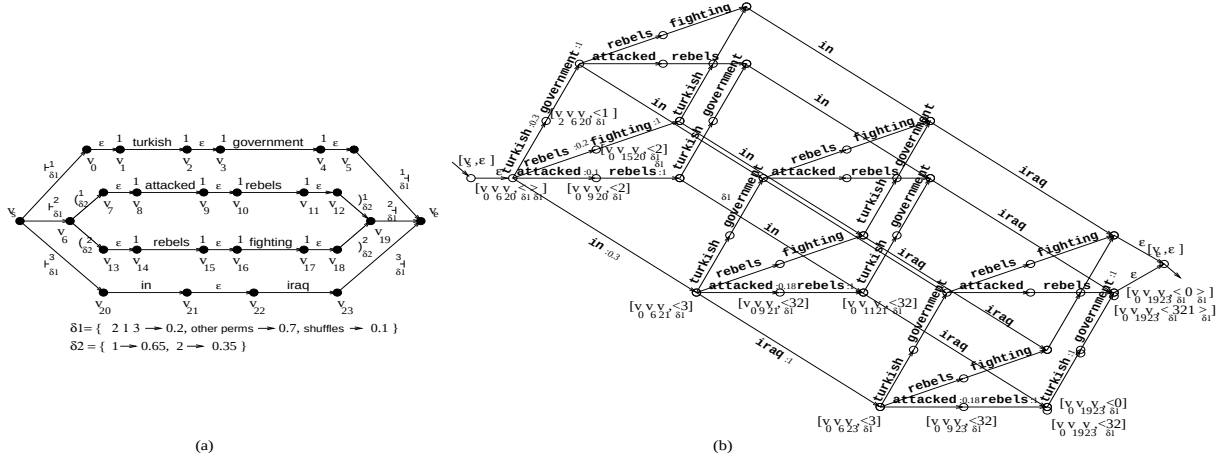


Figure 2: The WIDL-graph corresponding to the WIDL-expression in Figure 1 is shown in (a). The probabilistic finite-state acceptor (pFSA) that corresponds to the WIDL-graph is shown in (b).

are responsible for adding and removing, respectively, the $\langle \delta, \rangle_\delta$ symbols in the $\parallel$ -stack. The probabilities associated with A_ω transitions are computed using the vertex set and the $\parallel$ -stack of each A_ω state, together with the distribution functions of the $\vee$ and $\parallel$ operators. For a detailed presentation of the UNFOLD relation we refer the reader to (Soricut, 2006).

3 Stochastic Language Generation from WIDL-expressions

3.1 Interpolating Probability Distributions in a Log-linear Framework

Let us assume a finite set E of strings over a finite alphabet Σ , representing the set of possible sentence realizations. In a log-linear framework, we have a vector of feature functions $\bar{h} = \langle h_0 h_1 \dots h_M \rangle$, and a vector of parameters $\bar{\lambda} = \langle \lambda_0 \lambda_1 \dots \lambda_M \rangle$. For any $e \in E$, the interpolated probability $P(e)$ can be written under a log-linear model as in Equation 1:

$$P(e) = \frac{\exp[\sum_{m=0}^M \lambda_m h_m(e)]}{\sum_{e' \in E} \exp[\sum_{m=0}^M \lambda_m h_m(e')]} \quad (1)$$

We can formulate the search problem of finding the most probable realization e under this model as shown in Equation 2, and therefore we do not need to be concerned about computing expensive normalization factors.

$$\arg \max_e P(e) = \arg \max_e \exp[\sum_{m=0}^M \lambda_m h_m(e)] \quad (2)$$

For a given WIDL-expression ω over Σ , the set E is defined by $\text{dom}(\sigma_{\text{widl}}(\omega))$, and feature function

h_0 is taken to be $\sigma_{\text{widl}}(\omega)$. Any language model we want to employ may be added in Equation 2 as a feature function $h_i, i \geq 1$.

3.2 Algorithms for Intersecting WIDL-expressions with Language Models

Algorithm WIDL-NGLM-A* (Figure 3) solves the search problem defined by Equation 2 for a WIDL-expression ω (which provides feature function h_0) and M n -gram language models (which provide feature functions $h_1, \dots, h_M$). It does so by incrementally computing UNFOLD for G_ω (i.e., on-demand computation of the corresponding pFSA A_ω), by keeping track of a set of active states, called *active*. The set of newly UNFOLDED states is called *unfold*. Using Equation 1 (unnormalized), we EVALUATE the current $P(e)$ scores for the *unfold* states. Additionally, EVALUATE uses an admissible heuristic function to compute future (admissible) scores for the *unfold* states.

The algorithm PUSHES each state from the current *unfold* into a priority queue Q , which sorts the states according to their total score (current + admissible). In the next iteration, *active* is a singleton set containing the state POPed out from the top of Q . The admissible heuristic function we use is the one defined in (Soricut and Marcu, 2005), using Equation 1 (unnormalized) for computing the event costs. Given the existence of the admissible heuristic and the monotonicity property of the unfolding provided by the priority queue Q , the proof for A* optimality (Russell and Norvig, 1995) guarantees that WIDL-NGLM-A* finds a path in A_ω that provides an optimal solution.

```

WIDL-NGLM-A*( $G_\omega, \bar{h}, \bar{\lambda}$ )
1  active  $\leftarrow \{[v_s, \{\}]\}$ 
2  flag  $\leftarrow 1$ 
3  while flag
4      do unfold  $\leftarrow \text{UNFOLD}(G_\omega, \textit{active})$ 
5           $\text{EVALUATE}(\textit{unfold}, \bar{h}, \bar{\lambda})$ 
6          if active =  $\{[v_e, \{\}]\}$ 
7              then flag  $\leftarrow 0$ 
8              for each state in unfold
                  do  $\text{PUSH}(Q, \textit{state})$ 
                  active  $\leftarrow \text{POP}(Q)$ 
9  return active

```

Figure 3: A* algorithm for interpolating WIDL-expressions with n -gram language models.

An important property of the WIDL-NGLM-A* algorithm is that the UNFOLD relation (and, implicitly, the A_ω acceptor) is computed only partially, for those states for which the total cost is less than the cost of the optimal path. This results in important savings, both in space and time, over simply running a single-source shortest-path algorithm for directed acyclic graphs (Cormen et al., 2001) over the full acceptor A_ω (Soricut and Marcu, 2005).

4 Headline Generation using WIDL-expressions

We employ the WIDL formalism (Section 2) and the WIDL-NGLM-A* algorithm (Section 3) in a summarization application that aims at producing both informative and fluent headlines. Our headlines are generated in an abstractive, bottom-up manner, starting from words and phrases. A more common, extractive approach operates top-down, by starting from an extracted sentence that is compressed (Dorr et al., 2003) and annotated with additional information (Zajic et al., 2004).

Automatic Creation of WIDL-expressions for Headline Generation. We generate WIDL-expressions starting from an input document. First, we extract a weighted list of topic keywords from the input document using the algorithm of Zhou and Hovy (2003). This list is enriched with phrases created from the lexical dependencies the topic keywords have in the input document. We associate probability distributions with these phrases using their frequency (we assume

Keywords {iraq 0.32, syria 0.25, rebels 0.22, kurdish 0.17, turkish 0.14, attack 0.10}

Phrases

iraq {in iraq 0.4, northern iraq 0.5, iraq and iran 0.1},
 syria { into syria 0.6, and syria 0.4 }
 rebels { attacked rebels 0.7, rebels fighting 0.3 }

...

↓ WIDL-expression & trigram interpolation

TURKISH GOVERNMENT ATTACKED REBELS IN IRAQ AND SYRIA

Figure 4: Input and output for our automatic headline generation system.

that higher frequency is indicative of increased importance) and their position in the document (we assume that proximity to the beginning of the document is also indicative of importance). In Figure 4, we present an example of input keywords and lexical-dependency phrases automatically extracted from a document describing incidents at the Turkey-Iraq border.

The algorithm for producing WIDL-expressions combines the lexical-dependency phrases for each keyword using a $\vee$ operator with the associated probability values for each phrase multiplied with the probability value of each topic keyword. It then combines all the $\vee$ -headed expressions into a single WIDL-expression using a $\parallel$ operator with uniform probability. The WIDL-expression in Figure 1 is a (scaled-down) example of the expressions created by this algorithm. On average, a WIDL-expression created by this algorithm, using $m = 6$ keywords and an average of $k = 4$ lexical-dependency phrases per keyword, compactly encodes a candidate set of about 3 million possible realizations. As the specification of the $\parallel_\delta$ operator takes space $O(1)$ for uniform δ , Theorem 1 guarantees that the space complexity of these expressions is $O(mk)$.

Finally, we generate headlines from WIDL-expressions using the WIDL-NGLM-A* algorithm, which interpolates the probability distributions represented by the WIDL-expressions with n -gram language model distributions. The output presented in Figure 4 is the most likely headline realization produced by our system.

Headline Generation Evaluation. To evaluate the accuracy of our headline generation system, we use the documents from the DUC 2003 evaluation competition. Half of these documents are used as development set (283 documents),

ALG	#(uni)	#(bi)	Len.	Rouge ₁	Rouge ₂
Extractive					
Lead10	458	114	9.9	20.8	11.1
HedgeTrimmer [†]	399	104	7.4	18.1	9.9
Topiary [‡]	576	115	9.9	26.2	12.5
Abstractive					
Keywords	585	22	9.9	26.6	5.5
Webcl	311	76	7.3	14.1	7.5
WIDL-A*	562	126	10.0	25.5	12.9

Table 1: Headline generation evaluation. We compare extractive algorithms against abstractive algorithms, including our WIDL-based algorithm.

and the other half is used as test set (273 documents). We automatically measure performance by comparing the produced headlines against one reference headline produced by a human using ROUGE₂ (Lin, 2004).

For each input document, we train two language models, using the SRI Language Model Toolkit (with modified Kneser-Ney smoothing). A general trigram language model, trained on 170M English words from the Wall Street Journal, is used to model fluency. A document-specific trigram language model, trained on-the-fly for each input document, accounts for both fluency and content validity. We also employ a word-count model (which counts the number of words in a proposed realization) and a phrase-count model (which counts the number of phrases in a proposed realization), which allow us to learn to produce headlines that have restrictions in the number of words allowed (10, in our case). The interpolation weights $\bar{\lambda}$ (Equation 2) are trained using discriminative training (Och, 2003) using ROUGE₂ as the objective function, on the development set.

The results are presented in Table 1. We compare the performance of several extractive algorithms (which operate on an extracted sentence to arrive at a headline) against several abstractive algorithms (which create headlines starting from scratch). For the extractive algorithms, **Lead10** is a baseline which simply proposes as headline the lead sentence, cut after the first 10 words. **HedgeTrimmer**[†] is our implementation of the Hedge Trimmer system (Dorr et al., 2003), and **Topiary**[‡] is our implementation of the Topiary system (Zajic et al., 2004). For the abstractive algorithms, **Keywords** is a baseline that proposes as headline the sequence of topic keywords, **Webcl** is the system

THREE GORGES PROJECT IN CHINA HAS WON APPROVAL
WATER IS LINK BETWEEN CLUSTER OF E. COLI CASES
SRI LANKA 'S JOINT VENTURE TO EXPAND EXPORTS
OPPOSITION TO EUROPEAN UNION SINGLE CURRENCY EURO
OF INDIA AND BANGLADESH WATER BARRAGE

Figure 5: Headlines generated automatically using a WIDL-based sentence realization system.

described in (Zhou and Hovy, 2003), and **WIDL-A*** is the algorithm described in this paper.

This evaluation shows that our WIDL-based approach to generation is capable of obtaining headlines that compare favorably, in both content and fluency, with extractive, state-of-the-art results (Zajic et al., 2004), while it outperforms a previously-proposed abstractive system by a wide margin (Zhou and Hovy, 2003). Also note that our evaluation makes these results directly comparable, as they use the same parsing and topic identification algorithms. In Figure 5, we present a sample of headlines produced by our system, which includes both good and not-so-good outputs.

5 Machine Translation using WIDL-expressions

We also employ our WIDL-based realization engine in a machine translation application that uses a two-phase generation approach: in a first phase, WIDL-expressions representing large sets of possible translations are created from input foreign-language sentences. In a second phase, we use our generic, WIDL-based sentence realization engine to intersect WIDL-expressions with an n -gram language model. In the experiments reported here, we translate between Chinese (source language) and English (target language).

Automatic Creation of WIDL-expressions for MT. We generate WIDL-expressions from Chinese strings by exploiting a phrase-based translation table (Koehn et al., 2003). We use an algorithm resembling probabilistic bottom-up parsing to build a WIDL-expression for an input Chinese string: each contiguous span (i, j) over a Chinese string $C_{i,j}$ is considered a possible “constituent”, and the “non-terminals” associated with each constituent are the English phrase translations $E_{i,j}^k$ that correspond in the translation table to the Chinese string $C_{i,j}$. Multiple-word English phrases, such as $w_1 w_2 w_3$, are represented as WIDL-expressions using the precedence $(\cdot)$ and

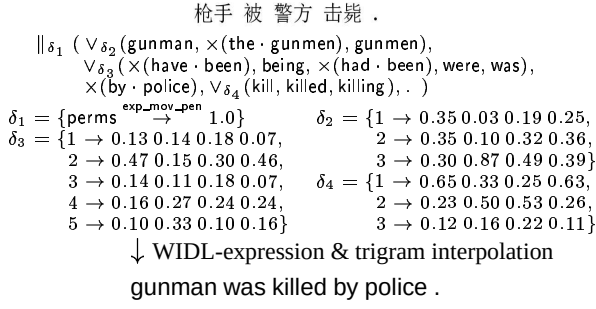


Figure 6: A Chinese string is converted into a WIDL-expression, which provides a translation as the best scoring hypothesis under the interpolation with a trigram language model.

lock ($\times$) operators, as $\times(w_1 \cdot w_2 \cdot w_3)$. To limit the number of possible translations $E_{i,j}^k$ corresponding to a Chinese span $C_{i,j}$, we use a probabilistic beam b and a histogram beam s to beam out low probability translation alternatives. At this point, each $C_{i,j}$ span is “tiled” with likely translations $E_{i,j}^k$ taken from the translation table.

Tiles that are adjacent are joined together in a larger tile by a $\|_{\delta}$ operator, where $\delta = \{\text{perms} \xrightarrow{\text{exp\_mov\_pen}} 1\}$. That is, reordering of the component tiles are permitted by the $\|_{\delta}$ operators (assigned non-zero probability), but the longer the movement from the original order of the tiles, the lower the probability. (This distortion model is similar with the one used in (Koehn, 2004).) When multiple tiles are available for the same span (i, j) , they are joined by a $\vee_{\delta}$ operator, where δ is specified by the probability distributions specified in the translation table. Usually, statistical phrase-based translation tables specify not only one, but multiple distributions that account for context preferences. In our experiments, we consider four probability distributions: $p(\bar{f}|\bar{e})$, $p(\bar{e}|\bar{f})$, $p_{lex}(\bar{f}|\bar{e})$, and $p_{lex}(\bar{e}|\bar{f})$, where $\bar{f}$ and $\bar{e}$ are Chinese-English phrase translations as they appear in the translation table. In Figure 6, we show an example of WIDL-expression created by this algorithm¹.

On average, a WIDL-expression created by this algorithm, using an average of $m = 38$ tiles per sentence (for an average input sentence length of 30 words) and an average of $k = 9$ possible translations per tile, encodes a candidate set of about 10^{50} possible translations. As the specification of the $\|_{\delta}$ operators takes space $O(1)$, Theorem 1

¹English reference: the gunman was shot dead by the police.

guarantees that these WIDL-expressions encode compactly these huge spaces in $O(mk)$.

In the second phase, we employ our WIDL-based realization engine to interpolate the distribution probabilities of WIDL-expressions with a trigram language model. In the notation of Equation 2, we use four feature functions $h_0, \dots, h_3$ for the WIDL-expression distributions (one for each probability distribution encoded); a feature function h_4 for a trigram language model; a feature function h_5 for a word-count model, and a feature function h_6 for a phrase-count model.

As acknowledged in the Machine Translation literature (Germann et al., 2003), full A* search is not usually possible, due to the large size of the search spaces. We therefore use an approximation algorithm, called WIDL-NGLM-A_k^{*}, which considers for unfolding only the nodes extracted from the priority queue Q which already unfolded a path of length greater than or equal to the maximum length already unfolded minus k (we used $k = 2$ in the experiments reported here).

MT Performance Evaluation. When evaluated against the state-of-the-art, phrase-based decoder Pharaoh (Koehn, 2004), using the same experimental conditions – translation table trained on the FBIS corpus (7.2M Chinese words and 9.2M English words of parallel text), trigram language model trained on 155M words of English newswire, interpolation weights $\bar{\lambda}$ (Equation 2) trained using discriminative training (Och, 2003) (on the 2002 NIST MT evaluation set), probabilistic beam b set to 0.01, histogram beam s set to 10 – and BLEU (Papineni et al., 2002) as our metric, the WIDL-NGLM-A₂^{*} algorithm produces translations that have a BLEU score of 0.2570, while Pharaoh translations have a BLEU score of 0.2635. The difference is not statistically significant at 95% confidence level.

These results show that the WIDL-based approach to machine translation is powerful enough to achieve translation accuracy comparable with state-of-the-art systems in machine translation.

6 Conclusions

The approach to sentence realization we advocate in this paper relies on WIDL-expressions, a formal language with convenient theoretical properties that can accommodate a wide range of generation scenarios. In the worst case, one can work with simple bags of words that encode no context

preferences (Soricut and Marcu, 2005). One can also work with bags of words and phrases that encode context preferences, a scenario that applies to current approaches in statistical machine translation (Section 5). And one can also encode context and ordering preferences typically used in summarization (Section 4).

The generation engine we describe enables a tight coupling of content selection with sentence realization preferences. Its algorithm comes with theoretical guarantees about its optimality. Because the requirements for producing WIDL-expressions are minimal, our WIDL-based generation engine can be employed, with state-of-the-art results, in a variety of text-to-text applications.

Acknowledgments This work was partially supported under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022.

References

- Srinivas Bangalore and Owen Rambow. 2000. Using TAG, a tree model, and a language model for generation. In *Proceedings of the Fifth International Workshop on Tree-Adjoining Grammars (TAG+)*.
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2001. *Introduction to Algorithms*. The MIT Press and McGraw-Hill.
- Simon Corston-Oliver, Michael Gamon, Eric K. Ringger, and Robert Moore. 2002. An overview of Amalgam: A machine-learned generation module. In *Proceedings of the INLG*.
- Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: a parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL Text Summarization Workshop*, pages 1–8.
- Michael Elhadad. 1991. FUF User manual — version 5.0. Technical Report CUCS-038-91, Department of Computer Science, Columbia University.
- Ulrich Germann, Mike Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2003. Fast decoding and optimal decoding for machine translation. *Artificial Intelligence*, 154(1–2):127–143.
- Nizar Habash. 2003. Matador: A large-scale Spanish-English GHMT system. In *Proceedings of AMTA*.
- J. Hajic, M. Cmejrek, B. Dorr, Y. Ding, J. Eisner, D. Gildea, T. Koo, K. Parton, G. Penn, D. Radev, and O. Rambow. 2002. Natural language generation in the context of machine translation. Summer workshop final report, Johns Hopkins University.
- K. Knight and V. Hatzivassiloglou. 1995. Two level, many-path generation. In *Proceedings of the ACL*.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase based translation. In *Proceedings of the HLT-NAACL*, pages 127–133.
- Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the AMTA*, pages 115–124.
- I. Langkilde-Geary. 2002. *A foundation for general-purpose natural language generation: sentence realization using probabilistic models of language*. Ph.D. thesis, University of Southern California.
- Chin-Yew Lin. 2004. ROUGE: a package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*.
- Christian Matthiessen and John Bateman. 1991. *Text Generation and Systemic-Functional Linguistic*. Pinter Publishers, London.
- Mehryar Mohri, Fernando Pereira, and Michael Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69–88.
- Mark-Jan Nederhof and Giorgio Satta. 2004. IDL-expressions: a formalism for representing and parsing finite languages in natural language processing. *Journal of Artificial Intelligence Research*, pages 287–317.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the ACL*, pages 160–167.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the ACL*, pages 311–318.
- Stuart Russell and Peter Norvig. 1995. *Artificial Intelligence. A Modern Approach*. Prentice Hall.
- Radu Soricut and Daniel Marcu. 2005. Towards developing generation algorithms for text-to-text applications. In *Proceedings of the ACL*, pages 66–74.
- Radu Soricut. 2006. *Natural Language Generation for Text-to-Text Applications Using an Information-Slim Representation*. Ph.D. thesis, University of Southern California.
- David Zajic, Bonnie J. Dorr, and Richard Schwartz. 2004. BBN/UMD at DUC-2004: Topiary. In *Proceedings of the NAACL Workshop on Document Understanding*, pages 112–119.
- Liang Zhou and Eduard Hovy. 2003. Headline summarization at ISI. In *Proceedings of the NAACL Workshop on Document Understanding*.

Learning to Say It Well: Reranking Realizations by Predicted Synthesis Quality

Crystal Nakatsu and Michael White

Department of Linguistics
The Ohio State University
Columbus, OH 43210 USA
{cnakatsu,mwhite}@ling.ohio-state.edu

Abstract

This paper presents a method for adapting a language generator to the strengths and weaknesses of a synthetic voice, thereby improving the naturalness of synthetic speech in a spoken language dialogue system. The method trains a discriminative reranker to select paraphrases that are predicted to sound natural when synthesized. The ranker is trained on realizer and synthesizer features in supervised fashion, using human judgements of synthetic voice quality on a sample of the paraphrases representative of the generator's capability. Results from a cross-validation study indicate that discriminative paraphrase reranking can achieve substantial improvements in naturalness on average, ameliorating the problem of highly variable synthesis quality typically encountered with today's unit selection synthesizers.

1 Introduction

Unit selection synthesis¹—a technique which concatenates segments of natural speech selected from a database—has been found to be capable of producing high quality synthetic speech, especially for utterances that are similar to the speech in the database in terms of style, delivery, and coverage (Black and Lenzo, 2001). In particular, in the limited domain of a spoken language dialogue system, it is possible to achieve highly natural synthesis with a purpose-built voice (Black and Lenzo, 2000). However, it can be difficult to develop

a synthetic voice for a dialogue system that produces natural speech completely reliably, and thus in practice output quality can be quite variable. Two important factors in this regard are the labeling process for the speech database and the direction of the dialogue system's further development, after the voice has been built: when labels are assigned fully automatically to the recorded speech, label boundaries may be inaccurate, leading to unnatural sounding joins in speech output; and when further system development leads to the generation of utterances that are less like those in the recording script, such utterances must be synthesized using smaller units with more joins between them, which can lead to a considerable dropoff in quality.

As suggested by Bulyko and Ostendorf (2002), one avenue for improving synthesis quality in a dialogue system is to have the system choose what to say in part by taking into account what is likely to sound natural when synthesized. The idea is to take advantage of the generator's periphrastic ability:² given a set of generated paraphrases that suitably express the desired content in the dialogue context, the system can select the specific paraphrase to use as its response according to the predicted quality of the speech synthesized for that paraphrase. In this way, if there are significant differences in the predicted synthesis quality for the various paraphrases—and if these predictions are generally borne out—then, by selecting paraphrases with high predicted synthesis quality, the dialogue system (as a whole) can more reliably produce natural sounding speech.

In this paper, we present an application of dis-

¹See e.g. (Hunt and Black, 1996; Black and Taylor, 1997; Beutnagel et al., 1999).

²See e.g. (Iordanskaja et al., 1991; Langkilde and Knight, 1998; Barzilay and McKeown, 2001; Pang et al., 2003) for discussion of paraphrase in generation.

criminative reranking to the task of adapting a language generator to the strengths and weaknesses of a particular synthetic voice. Our method involves training a reranker to select paraphrases that are predicted to sound natural when synthesized, from the N-best realizations produced by the generator. The reranker is trained in supervised fashion, using human judgements of synthetic voice quality on a representative sample of the paraphrases. In principle, the method can be employed with any speech synthesizer. Additionally, when features derived from the synthesizer's unit selection search can be made available, further quality improvements become possible.

The paper is organized as follows. In Section 2, we review previous work on integrating choice in language generation and speech synthesis, and on learning discriminative rerankers for generation. In Section 3, we present our method. In Section 4, we describe a cross-validation study whose results indicate that discriminative paraphrase reranking can achieve substantial improvements in naturalness on average. Finally, in Section 5, we conclude with a summary and a discussion of future work.

2 Previous Work

Most previous work on integrating language generation and synthesis, e.g. (Davis and Hirschberg, 1988; Prevost and Steedman, 1994; Hitzeman et al., 1998; Pan et al., 2002), has focused on how to use the information present in the language generation component in order to specify contextually appropriate intonation for the speech synthesizer to target. For example, syntactic structure, information structure and dialogue context have all been argued to play a role in improving prosody prediction, compared to unrestricted text-to-speech synthesis. While this topic remains an important area of research, our focus is instead on a different opportunity that arises in a dialogue system, namely, the possibility of choosing the exact wording and prosody of a response according to how natural it is likely to sound when synthesized.

To our knowledge, Bulyko and Ostendorf (2002) were the first to propose allowing the choice of wording and prosody to be jointly determined by the language generator and speech synthesizer. In their approach, a template-based generator passes a prosodically annotated word net-

work to the speech synthesizer, rather than a single text string (or prosodically annotated text string). To perform the unit selection search on this expanded input efficiently, they employ weighted finite-state transducers, where each step of network expansion is then followed by minimization. The weights are determined by concatenation (join) costs, relative frequencies (negative log probabilities) of the word sequences, and prosodic prediction costs, for cases where the prosody is not determined by the templates. In a perception experiment, they demonstrated that by expanding the space of candidate responses, their system achieved higher quality speech output.

Following (Bulyko and Ostendorf, 2002), Stone et al. (2004) developed a method for jointly determining wording, speech and gesture. In their approach, a template-based generator produces a word lattice with intonational phrase breaks. A unit selection algorithm then searches for a low-cost way of realizing a path through this lattice that combines captured motion samples with recorded speech samples to create coherent phrases, blending segments of speech and motion together phrase-by-phrase into extended utterances. Video demonstrations indicate that natural and highly expressive results can be achieved, though no human evaluations are reported.

In an alternative approach, Pan and Weng (2002) proposed integrating instance-based realization and synthesis. In their framework, sentence structure, wording, prosody and speech waveforms from a domain-specific corpus are simultaneously reused. To do so, they add prosodic and acoustic costs to the insertion, deletion and replacement costs used for instance-based surface realization. Their contribution focuses on how to design an appropriate speech corpus to facilitate an integrated approach to instance-based realization and synthesis, and does not report evaluation results.

A drawback of these approaches to integrating choice in language generation and synthesis is that they cannot be used with most existing speech synthesizers, which do not accept (annotated) word lattices as input. In contrast, the approach we introduce here can be employed with any speech synthesizer in principle. All that is required is that the language generator be capable of producing N-best outputs; that is, the generator must be able to construct a set of suitable paraphrases ex-

pressing the desired content, from which the top N realizations can be selected for reranking according to their predicted synthesis quality. Once the realizations have been reranked, the top scoring realization can be sent to the synthesizer as usual. Alternatively, when features derived from the synthesizer's unit selection search can be made available—and if the time demands of the dialogue system permit—several of the top scoring reranked realizations can be sent to the synthesizer, and the resulting utterances can be rescored with the extended feature set.

Our reranking approach has been inspired by previous work on reranking in parsing and generation, especially (Collins, 2000) and (Walker et al., 2002). As in Walker et al.'s (2002) method for training a sentence plan ranker, we use our generator to produce a representative sample of paraphrases and then solicit human judgements of their naturalness to use as data for training the ranker. This method is attractive when there is no suitable corpus of naturally occurring dialogues available for training purposes, as is often the case for systems that engage in human-computer dialogues that differ substantially from human-human ones. The primary difference between Walker et al.'s work and ours is that theirs examines the impact on text quality of sentence planning decisions such as aggregation, whereas ours focuses on the impact of the lexical and syntactic choice at the surface realization level on speech synthesis quality, according to the strengths and weaknesses of a particular synthetic voice.

3 Reranking Realizations by Predicted Synthesis Quality

3.1 Generating Alternatives

Our experiments with integrating language generation and synthesis have been carried out in the context of the COMIC³ multimodal dialogue system (den Os and Boves, 2003). The COMIC system adds a dialogue interface to a CAD-like application used in sales situations to help clients redesign their bathrooms. The input to the system includes speech, handwriting, and pen gestures; the output combines synthesized speech, an animated talking head, deictic gestures at on-screen objects, and direct control of the underlying application.

³Conversational Multimodal Interaction with Computers, <http://www.hcrc.ed.ac.uk/comic/>.

Drawing on the materials used in (Foster and White, 2005) to evaluate adaptive generation in COMIC, we selected a sample of 104 sentences from 38 different output turns across three dialogues. For each sentence in the set, a variant was included that expressed the same content adapted to a different user model or adapted to a different dialogue history. For example, a description of a certain design's colour scheme for one user might be phrased as *As you can see, the tiles have a blue and green colour scheme*, whereas a variant expression of the same content for a different user could be *Although the tiles have a blue colour scheme, the design does also feature green*, if the user disprefers blue.

In COMIC, the sentence planner uses XSLT to generate disjunctive logical forms (LFs), which specify a range of possible paraphrases in a nested free-choice form (Foster and White, 2004). Such disjunctive LFs can be efficiently realized using the OpenCCG realizer (White, 2004; White, 2006b; White, 2006a). Note that for the experiments reported here, we manually augmented the disjunctive LFs for the 104 sentences in our sample to make greater use of the periphrastic capabilities of the COMIC grammar; it remains for future work to augment the COMIC sentence planner produce these more richly disjunctive LFs automatically.

OpenCCG includes an extensible API for integrating language modeling and realization. To select preferred word orders, from among all those allowed by the grammar for the input LF, we used a backoff trigram model trained on approximately 750 example target sentences, where certain words were replaced with their semantic classes (e.g. MANUFACTURER, COLOUR) for better generalization. For each of the 104 sentences in our sample, we performed 25-best realization from the disjunctive LF, and then randomly selected up to 12 different realizations to include in our experiments based on a simulated coin flip for each realization, starting with the top-scoring one. We used this procedure to sample from a larger portion of the N -best realizations, while keeping the sample size manageable.

Figure 1 shows an example of 12 paraphrases for a sentence chosen for inclusion in our sample. Note that the realizations include words with pitch accent annotations as well as boundary tones as separate, punctuation-like words. Generally the

- this_{H*} design_{H*} uses tiles from Villeroy_and_Boch_{H*}'s Funny_Day_{H*} collection LL% .
- this_{H*} design_{H*} is based_on the Funny_Day_{H*} collection by Villeroy_and_Boch_{H*} LL% .
- this_{H*} design_{H*} is based_on Funny_Day_{H*} LL% , by Villeroy_and_Boch_{H*} LL% .
- this_{H*} design_{H*} draws from the Funny_Day_{H*} collection by Villeroy_and_Boch_{H*} LL% .
- this_{H*} one draws from Funny_Day_{H*} LL% , by Villeroy_and_Boch_{H*} LL% .
- here_{L+H*} LH% we have a design that is based_on the Funny_Day_{H*} collection by Villeroy_and_Boch_{H*} LL% .
- this_{H*} design_{H*} draws from Villeroy_and_Boch_{H*}'s Funny_Day_{H*} series LL% .
- here is a design that draws from Funny_Day_{H*} LL% , by Villeroy_and_Boch_{H*} LL% .
- this_{H*} one draws from Villeroy_and_Boch_{H*}'s Funny_Day_{H*} collection LL% .
- this_{H*} draws from the Funny_Day_{H*} collection by Villeroy_and_Boch_{H*} LL% .
- this_{H*} one draws from the Funny_Day_{H*} collection by Villeroy_and_Boch_{H*} LL% .
- here is a design that draws from Villeroy_and_Boch_{H*}'s Funny_Day_{H*} collection LL% .

Figure 1: Example of sampled periphrastic alternatives for a sentence.

quality of the sampled paraphrases is very high, only occasionally including dispreferred word orders such as *We here have a design in the family style*, where *here* is in medial position rather than fronted.⁴

3.2 Synthesizing Utterances

For synthesis, OpenCCG's output realizations are converted to APML,⁵ a markup language which allows pitch accents and boundary tones to be specified, and then passed to the Festival speech synthesis system (Taylor et al., 1998; Clark et al., 2004). Festival uses the prosodic markup in the text analysis phase of synthesis in place of the structures that it would otherwise have to predict from the text. The synthesiser then uses the context provided by the markup to enforce the selec-

tion of suitable units from the database.

A custom synthetic voice for the COMIC system was developed, as follows. First, a domain-specific recording script was prepared by selecting about 150 sentences from the larger set of target sentences used to train the system's n-gram model. The sentences were greedily selected with the goals of ensuring that (i) all words (including proper names) in the target sentences appeared at least once in the record script, and (ii) all bigrams at the level of semantic classes (e.g. MANUFACTURER, COLOUR) were covered as well. For the cross-validation study reported in the next section, we also built a trigram model on the words in the domain-specific recording script, *without* replacing any words with semantic classes, so that we could examine whether the more frequent occurrence of the specific words and phrases in this part of the script is predictive of synthesis quality.

The domain-specific script was augmented with a set of 600 newspaper sentences selected for di-phone coverage. The newspaper sentences make it possible for the voice to synthesize words outside of the domain-specific script, though not necessarily with the same quality. Once these scripts were in place, an amateur voice talent was recorded reading the sentences in the scripts during two recording sessions. Finally, after the speech files were semi-automatically segmented into individual sentences, the speech database was constructed, using fully automatic labeling.

We have found that the utterances synthesized with the COMIC voice vary considerably in their naturalness, due to two main factors. First, the system underwent further development after the voice was built, leading to the addition of a variety of new phrases to the system's repertoire, as well as many extra proper names (and their pronunciations); since these names and phrases usually require going outside of the domain-specific part of the speech database, they often (though not always) exhibit a considerable dropoff in synthesis quality.⁶ And second, the boundaries of the automatically assigned unit labels were not always accurate, leading to problems with unnatural joins and reduced intelligibility. To improve the reliability of the COMIC voice, we could have recorded more speech, or manually corrected label bound-

⁴In other examples medial position is preferred, e.g. *This design here is in the family style*.

⁵Affective Presentation Markup Language; see <http://www.cstr.ed.ac.uk/projects/festival/apml.html>.

⁶Note that in the current version of the system, proper names are always required parts of the output, and thus the discriminative reranker cannot learn to simply choose paraphrases that leave out problematic names.

aries; the goal of this paper is to examine whether the naturalness of a dialogue system’s output can be improved in a less labor-intensive way.

3.3 Rating Synthesis Quality

To obtain data for training our realization reranker, we solicited judgements of the naturalness of the synthesized speech produced by Festival for the utterances in our sample COMIC corpus. Two judges (the first two authors) provided judgements on a 1–7 point scale, with higher scores representing more natural synthesis. Ratings were gathered using WebExp2,⁷ with the periphrastic alternatives for each sentence presented as a group in a randomized order. Note that for practical reasons, the utterances were presented out of the dialogue context, though both judges were familiar with the kinds of dialogues that the COMIC system is capable of.

Though the numbers on the seven point scale were not assigned labels, they were roughly taken to be “horrible,” “poor,” “fair,” “ok,” “good,” “very good” and “perfect.” The average assigned rating across all utterances was 4.05 (“ok”), with a standard deviation of 1.56. The correlation between the two judges’ ratings was 0.45, with one judge’s ratings consistently higher than the other’s.

Some common problems noted by the judges included slurred words, especially *the* sometimes sounding like *ther* or even *their*; clipped words, such as *has* shortened at times to the point of sounding like *is*, or *though* clipped to unintelligibility; unnatural phrasing or emphasis, e.g. occasional pauses before a possessive ’s, or words such as *style* sounding emphasized when they should be deaccented; unnatural rate changes; “choppy” speech from poor joins; and some unintelligible proper names.

3.4 Ranking

While Collins (2000) and Walker et al. (2002) develop their rankers using the RankBoost algorithm (Freund et al., 1998), we have instead chosen to use Joachims’ (2002) method of formulating ranking tasks as Support Vector Machine (SVM) constraint optimization problems.⁸ This choice has been motivated primarily by convenience, as Joachims’ SVM^{light} package is easy to

use; we leave it for future work to compare the performance of RankBoost and SVM^{light} on our ranking task.

The ranker takes as input a set of paraphrases that express the desired content of each sentence, optionally together with synthesized utterances for each paraphrase. The output is a ranking of the paraphrases according to the predicted naturalness of their corresponding synthesized utterances. Ranking is more appropriate than classification for our purposes, as naturalness is a graded assessment rather than a categorical one.

To encode the ranking task as an SVM constraint optimization problem, each paraphrase j of a sentence i is represented by a feature vector $\Phi(s_{ij}) = \langle f_1(s_{ij}), \dots, f_m(s_{ij}) \rangle$, where m is the number of features. In the training data, the feature vectors are paired with the average value of their corresponding human judgements of naturalness. From this data, ordered pairs of paraphrases (s_{ij}, s_{ik}) are derived, where s_{ij} has a higher naturalness rating than s_{ik} . The constraint optimization problem is then to derive a parameter vector $\vec{w}$ that yields a ranking score function $\vec{w} \cdot \Phi(s_{ij})$ which minimizes the number of pairwise ranking violations. Ideally, for every ordered pair (s_{ij}, s_{ik}) , we would have $\vec{w} \cdot \Phi(s_{ij}) > \vec{w} \cdot \Phi(s_{ik})$; in practice, it is often impossible or intractable to find such a parameter vector, and thus slack variables are introduced that allow for training errors. A parameter to the algorithm controls the trade-off between ranking margin and training error.

In testing, the ranker’s accuracy can be determined by comparing the ranking scores for every ordered pair (s_{ij}, s_{ik}) in the test data, and determining whether the actual preferences are borne out by the predicted preference, i.e. whether $\vec{w} \cdot \Phi(s_{ij}) > \vec{w} \cdot \Phi(s_{ik})$ as desired. Note that the ranking scores, unlike the original ratings, do not have any meaning in the absolute sense; their import is only to order alternative paraphrases by their predicted naturalness.

In our ranking experiments, we have used SVM^{light} with all parameters set to their default values.

3.5 Features

Table 1 shows the feature sets we have investigated for reranking, distinguished by the availability of the features and the need for discriminative training. The first row shows the feature sets that are

⁷http://www.hcrc.ed.ac.uk/web_exp/

⁸See (Barzilay and Lapata, 2005) for another application of SVM ranking in generation, namely to the task of ranking alternative text orderings for local coherence.

Table 1: Feature sets for reranking.

Availability	Discriminative	
	<i>no</i>	<i>yes</i>
Realizer	NGRAMS	WORDS
Synthesizer	COSTS	ALL

available to the realizer. There are two n-gram models that can be used to directly rank alternative realizations: NGRAM-1, the language model used in COMIC, and NGRAM-2, the language model derived from the domain-specific recording script; for feature values, the negative logarithms are used. There are also two WORDS feature sets (shown in the second column): WORDS-BI, which includes NGRAMS plus a feature for every possible unigram and bigram, where the value of the feature is the count of the unigram or bigram in a given realization; and WORDS-TRI, which includes all the features in WORDS-BI, plus a feature for every possible trigram. The second row shows the feature sets that require information from the synthesizer. The COSTS feature set includes NGRAMS plus the total join and target costs from the unit selection search. Note that a weighted sum of these costs could be used to directly rerank realizations, in much the same way as relative frequencies and concatenation costs are used in (Bulyko and Ostendorf, 2002); in our experiments, we let SVM^{light} determine how to weight these costs. Finally, there are two ALL feature sets: ALL-BI includes NGRAMS, WORDS-BI and COSTS, plus features for every possible phone and diphone, and features for every specific unit in the database; ALL-TRI includes NGRAMS, WORDS-TRI, COSTS, and a feature for every phone, diphone and triphone, as well as specific units in the database. As with WORDS, the value of a feature is the count of that feature in a given synthesized utterance.

4 Cross-Validation Study

To train and test our ranker on our feature sets, we partitioned the corpus into 10 folds and performed 10-fold cross-validation. For each fold, 90% of the examples were used for training the ranker and the remaining unseen 10% were used for testing. The folds were created by randomly choosing from among the sentence groups, resulting in all of the paraphrases for a given sentence occurring in the same fold, and each occurring ex-

Table 2: Comparison of results for differing feature sets, topline and baseline.

Features	Mean Score	SD	Accuracy (%)
BEST	5.38	1.11	100.0
WORDS-TRI	4.95	1.24	77.3
ALL-BI	4.95	1.24	77.9
ALL-TRI	4.90	1.25	78.0
WORDS-BI	4.86	1.28	76.8
COSTS	4.69	1.27	68.2
NGRAM-2	4.34	1.38	56.2
NGRAM-1	4.30	1.29	53.3
RANDOM	4.11	1.22	50.0

actly once in the testing set as a whole.

We evaluated the performance of our ranker by determining the average score of the best ranked paraphrase for each sentence, under each of the following feature combinations: NGRAM-1, NGRAM-2, COSTS, WORDS-BI, WORDS-TRI, ALL-BI, and ALL-TRI. Note that since we used the human ratings to calculate the score of the highest ranked utterance, the score of the highest ranked utterance cannot be higher than that of the highest human-rated utterance. Therefore, we effectively set the human ratings as the topline (BEST). For the baseline, we randomly chose an utterance from among the alternatives, and used its associated score. In 15 tests generating the random scores, our average scores ranged from 3.88–4.18. We report the median score of 4.11 as the average for the baseline, along with the mean of the topline and each of the feature subsets, in Table 2.

We also report the ordering accuracy of each feature set used by the ranker in Table 2. As mentioned in Section 3.4, the ordering accuracy of the ranker using a given feature set is determined by c/N , where c is the number of correctly ordered pairs (of each paraphrase, not just the top ranked one) produced by the ranker, and N is the total number of human-ranked ordered pairs.

As Table 2 indicates, the mean of BEST is 5.38, whereas our ranker using WORDS-TRI features achieves a mean score of 4.95. This is a difference of 0.42 on a seven point scale, or only a 6% difference. The ordering accuracy of WORDS-TRI is 77.3%.

We also measured the improvement of our ranker with each feature set over the random baseline as a percentage of the maximum possible gain (which would be to reproduce the human topline). The results appear in Figure 2. As the

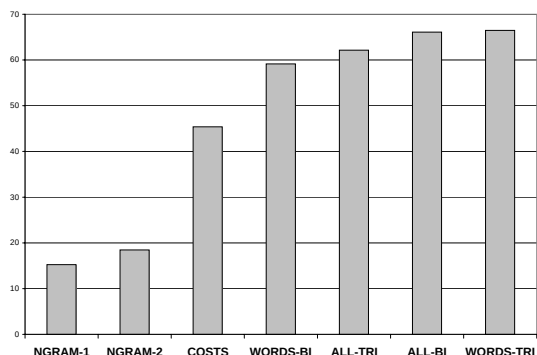


Figure 2: Improvement as a percentage of the maximum possible gain over the random baseline.

figure indicates, the maximum possible gain our ranker achieves over the baseline is 66% (using the WORDS-TRI or ALL-BI feature set). By comparison, NGRAM-1 and NGRAM-2 achieve less than 20% of the possible gain.

To verify our main hypothesis that our ranker would significantly outperform the baselines, we computed paired one-tailed t -tests between WORDS-TRI and RANDOM ($t = 2.4$, $p < 8.9 \times 10^{-13}$), and WORDS-TRI and NGRAM-1 ($t = 1.4$, $p < 4.5 \times 10^{-8}$). Both differences were highly significant. We also performed seven post-hoc comparisons using two-tailed t -tests, as we did not have an *a priori* expectation as to which feature set would work better. Using the Bonferroni adjustment for multiple comparisons, the p -value required to achieve an overall level of significance of 0.05 is 0.007. In the first post-hoc test, we found a significant difference between BEST and WORDS-TRI ($t = 8.0$, $p < 1.86 \times 10^{-12}$), indicating that there is room for improvement of our ranker. However, in considering the top scoring feature sets, we did not find a significant difference between WORDS-TRI and WORDS-BI ($t = 2.3$, $p < 0.022$), from which we infer that the difference among all of WORDS-TRI, ALL-BI, ALL-TRI and WORDS-BI is not significant also. This suggests that the synthesizer features have no substantial impact on our ranker, as we would expect ALL-TRI to be significantly higher than WORDS-TRI if so. However, since COSTS does significantly improve upon NGRAM2 ($t = 3.5$, $p < 0.001$), there is some value to the use of synthesizer features in the absence of WORDS. We also looked at the comparison for the WORDS models and COSTS. While WORDS-BI did not perform significantly better than COSTS ($t =$

2.3, $p < 0.025$), the added trigrams in WORDS-TRI did improve ranker performance significantly over COSTS ($t = 3.7$, $p < 3.29 \times 10^{-4}$). Since COSTS ranks realizations in the much the same way as (Bulyko and Ostendorf, 2002), the fact that WORDS-TRI outperforms COSTS indicates that our discriminative reranking method can significantly improve upon their non-discriminative approach.

5 Conclusions

In this paper, we have presented a method for adapting a language generator to the strengths and weaknesses of a particular synthetic voice by training a discriminative reranker to select phrases that are predicted to sound natural when synthesized. In contrast to previous work on this topic, our method can be employed with any speech synthesizer in principle, so long as features derived from the synthesizer’s unit selection search can be made available. In a case study with the COMIC dialogue system, we have demonstrated substantial improvements in the naturalness of the resulting synthetic speech, achieving two-thirds of the maximum possible gain, and raising the average rating from “ok” to “good.” We have also shown that in this study, our discriminative method significantly outperforms an approach that performs selection based solely on corpus frequencies together with target and join costs.

In future work, we intend to verify the results of our cross-validation study in a perception experiment with naïve subjects. We also plan to investigate whether additional features derived from the synthesizer can better detect unnatural pauses or changes in speech rate, as well as F0 contours that fail to exhibit the targeting accenting pattern. Finally, we plan to examine whether gains in quality can be achieved with an off-the-shelf, general purpose voice that are similar to those we have observed using COMIC’s limited domain voice.

Acknowledgements

We thank Mary Ellen Foster, Eric Fosler-Lussier and the anonymous reviewers for helpful comments and discussion.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Pro-*

- ceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor.
- Regina Barzilay and Kathleen McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proc. ACL/EACL*.
- M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, and A. Syrdal. 1999. The AT&T Next-Gen TTS system. In *Joint Meeting of ASA, EAA, and DAGA*.
- Alan Black and Kevin Lenzo. 2000. Limited domain synthesis. In *Proceedings of ICSLP2000*, Beijing, China.
- Alan Black and Kevin Lenzo. 2001. Optimal data selection for unit selection synthesis. In *4th ISCA Speech Synthesis Workshop*, Pitlochry, Scotland.
- Alan Black and Paul Taylor. 1997. Automatically clustering similar units for unit selection in speech synthesis. In *Eurospeech '97*.
- Ivan Bulyko and Mari Ostendorf. 2002. Efficient integrated response generation from multiple targets using weighted finite state transducers. *Computer Speech and Language*, 16:533–550.
- Robert A.J. Clark, Korin Richmond, and Simon King. 2004. Festival 2 – build your own general purpose unit selection speech synthesiser. In *5th ISCA Speech Synthesis Workshop*, pages 173–178, Pittsburgh, PA.
- Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proc. ICML*.
- James Raymond Davis and Julia Hirschberg. 1988. Assigning intonational features in synthesized spoken directions. In *Proc. ACL*.
- Els den Os and Lou Boves. 2003. Towards ambient intelligence: Multimodal computers that understand our intentions. In *Proc. eChallenges-03*.
- Mary Ellen Foster and Michael White. 2004. Techniques for Text Planning with XSLT. In *Proc. 4th NLPXML Workshop*.
- Mary Ellen Foster and Michael White. 2005. Assessing the impact of adaptive generation in the COMIC multimodal dialogue system. In *Proc. IJCAI-05 Workshop on Knowledge and Representation in Practical Dialogue Systems*.
- Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. 1998. An efficient boosting algorithm for combining preferences. In *Machine Learning: Proc. of the Fifteenth International Conference*.
- Janet Hitzeman, Alan W. Black, Chris Mellish, Jon Oberlander, and Paul Taylor. 1998. On the use of automatically generated discourse-level information in a concept-to-speech synthesis system. In *Proc. ICSLP-98*.
- A. Hunt and A. Black. 1996. Unit selection in a concatenative speech synthesis system using a large speech database. In *Proc. ICASSP-96*, Atlanta, Georgia.
- Lidija Iordanskaja, Richard Kittredge, and Alain Polguère. 1991. Lexical selection and paraphrase in a meaning-text generation model. In Cécile L. Paris, William R. Swartout, and William C. Mann, editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, pages 293–312. Kluwer.
- Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proc. KDD*.
- Irene Langkilde and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proc. COLING-ACL*.
- Shimei Pan and Wubin Weng. 2002. Designing a speech corpus for instance-based spoken language generation. In *Proc. of the International Natural Language Generation Conference (INLG-02)*.
- Shimei Pan, Kathleen McKeown, and Julia Hirschberg. 2002. Exploring features from natural language generation for prosody modeling. *Computer Speech and Language*, 16:457–490.
- Bo Pang, Kevin Knight, and Daniel Marcu. 2003. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proc. HLT/NAACL*.
- Scott Prevost and Mark Steedman. 1994. Specifying intonation from context for speech synthesis. *Speech Communication*, 15:139–153.
- Matthew Stone, Doug DeCarlo, Insuk Oh, Christian Rodriguez, Adrian Stere, Alyssa Lees, and Chris Bregler. 2004. Speaking with hands: Creating animated conversational characters from recordings of human performance. *ACM Transactions on Graphics (SIGGRAPH)*, 23(3).
- P. Taylor, A. Black, and R. Caley. 1998. The architecture of the the Festival speech synthesis system. In *Third International Workshop on Speech Synthesis*, Sydney, Australia.
- Marilyn A. Walker, Owen C. Rambow, and Monica Rogati. 2002. Training a sentence planner for spoken dialogue using boosting. *Computer Speech and Language*, 16:409–433.
- Michael White. 2004. Reining in CCG Chart Realization. In *Proc. INLG-04*.
- Michael White. 2006a. CCG chart realization from disjunctive logical forms. In *Proc. INLG-06*. To appear.
- Michael White. 2006b. Efficient Realization of Coordinate Structures in Combinatory Categorical Grammar. *Research on Language & Computation*, online first, March.

An Effective Two-Stage Model for Exploiting Non-Local Dependencies in Named Entity Recognition

Vijay Krishnan

Computer Science Department
Stanford University
Stanford, CA 94305
vijayk@cs.stanford.edu

Christopher D. Manning

Computer Science Department
Stanford University
Stanford, CA 94305
manning@cs.stanford.edu

Abstract

This paper shows that a simple two-stage approach to handle non-local dependencies in Named Entity Recognition (NER) can outperform existing approaches that handle non-local dependencies, while being much more computationally efficient. NER systems typically use sequence models for tractable inference, but this makes them unable to capture the long distance structure present in text. We use a Conditional Random Field (CRF) based NER system using local features to make predictions and then train another CRF which uses both local information and features extracted from the output of the first CRF. Using features capturing non-local dependencies from the same document, our approach yields a 12.6% relative error reduction on the F1 score, over state-of-the-art NER systems using local-information alone, when compared to the 9.3% relative error reduction offered by the best systems that exploit non-local information. Our approach also makes it easy to incorporate non-local information from other documents in the test corpus, and this gives us a 13.3% error reduction over NER systems using local-information alone. Additionally, our running time for inference is just the inference time of two sequential CRFs, which is much less than that of other more complicated approaches that directly model the dependencies and do approximate inference.

1 Introduction

Named entity recognition (NER) seeks to locate and classify atomic elements in unstructured

text into predefined entities such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. A particular problem for Named Entity Recognition (NER) systems is to exploit the presence of useful information regarding labels assigned at a long distance from a given entity. An example is the label-consistency constraint that if our text has two occurrences of *New York* separated by other tokens, we would want our learner to encourage both these entities to get the same label.

Most statistical models currently used for Named Entity Recognition, use sequence models and thereby capture local structure. Hidden Markov Models (HMMs) (Leek, 1997; Freitag and McCallum, 1999), Conditional Markov Models (CMMs) (Borthwick, 1999; McCallum et al., 2000), and Conditional Random Fields (CRFs) (Lafferty et al., 2001) have been successfully employed in NER and other information extraction tasks. All these models encode the Markov property i.e. labels directly depend only on the labels assigned to a small window around them. These models exploit this property for tractable computation as this allows the Forward-Backward, Viterbi and Clique Calibration algorithms to become tractable. Although this constraint is essential to make exact inference tractable, it makes us unable to exploit the non-local structure present in natural language.

Label consistency is an example of a non-local dependency important in NER. Apart from label consistency between the same token sequences, we would also like to exploit richer sources of dependencies between similar token sequences. For example, as shown in Figure 1, we would want it to encourage *Einstein* to be labeled “Person” if there is strong evidence that *Albert Einstein* should be labeled “Person”. Sequence models unfortu-

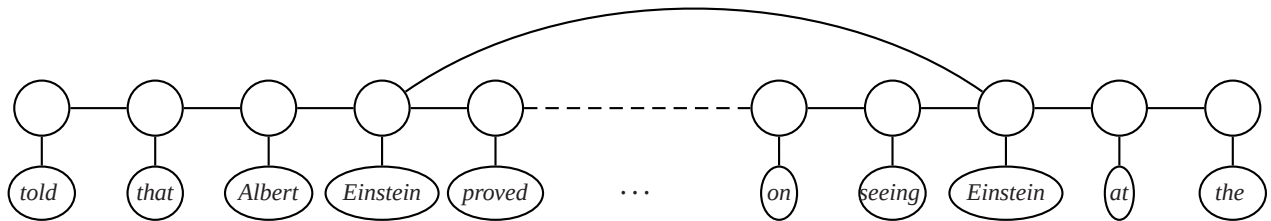


Figure 1: An example of the label consistency problem. Here we would like our model to encourage entities *Albert Einstein* and *Einstein* to get the same label, so as to improve the chance that both are labeled *PERSON*.

nately cannot model this due to their Markovian assumption.

Recent approaches attempting to capture non-local dependencies model the non-local dependencies directly, and use approximate inference algorithms, since exact inference is in general, not tractable for graphs with non-local structure.

Bunescu and Mooney (2004) define a *Relational Markov Network* (RMN) which explicitly models long-distance dependencies, and use it to represent relations between entities. Sutton and McCallum (2004) augment a sequential CRF with *skip-edges* i.e. edges between different occurrences of a token, in a document. Both these approaches use loopy belief propagation (Pearl, 1988; Yedidia et al., 2000) for approximate inference.

Finkel et al. (2005) hand-set penalties for inconsistency in entity labeling at different occurrences in the text, based on some statistics from training data. They then employ Gibbs sampling (Geman and Geman, 1984) for dealing with their local feature weights and their non-local penalties to do approximate inference.

We present a simple two-stage approach where our second CRF uses features derived from the output of the first CRF. This gives us the advantage of defining a rich set of features to model non-local dependencies, and also eliminates the need to do approximate inference, since we do not explicitly capture the non-local dependencies in a single model, like the more complex existing approaches. This also enables us to do inference efficiently since our inference time is merely the inference time of two sequential CRF's; in contrast Finkel et al. (2005) reported an increase in running time by a factor of 30 over the sequential CRF, with their Gibbs sampling approximate inference.

In all, our approach is simpler, yields higher F1 scores, and is also much more computationally efficient than existing approaches modeling non-local dependencies.

2 Conditional Random Fields

We use a Conditional Random Field (Lafferty et al., 2001; Sha and Pereira, 2003) since it represents the state of the art in sequence modeling and has also been very effective at Named Entity Recognition. It allows us both discriminative training that CMMs offer as well and the bi-directional flow of probabilistic information across the sequence that HMMs allow, thereby giving us the best of both worlds. Due to the bi-directional flow of information, CRFs guard against the myopic locally attractive decisions that CMMs make. It is customary to use the Viterbi algorithm, to find the most probably state sequence during inference. A large number of possibly redundant and correlated features can be supplied without fear of further reducing the accuracy of a high-dimensional distribution. These are well-documented benefits (Lafferty et al., 2001).

2.1 Our Baseline CRF for Named Entity Recognition

Our baseline CRF is a sequence model in which labels for tokens directly depend only on the labels corresponding to the previous and next tokens. We use features that have been shown to be effective in NER, namely the current, previous and next words, character n-grams of the current word, Part of Speech tag of the current word and surrounding words, the shallow parse chunk of the current word, shape of the current word, the surrounding word shape sequence, the presence of a word in a left window of size 5 around the current word and the presence of a word in a left window of size 5 around the current word. This gives us a competitive baseline CRF using local information alone, whose performance is close to the best published local CRF models, for Named Entity Recognition

3 Label Consistency

The intuition for modeling label consistency is that within a particular document, different occur-

	Document Level Statistics				Corpus Level Statistics			
	PER	LOC	ORG	MISC	PER	LOC	ORG	MISC
PER	3141	4	5	0	33830	113	153	0
LOC		6436	188	3		346966	6749	60
ORG			2975	0			43892	223
MISC				2030				66286

Table 1: Table showing the number of pairs of different occurrences of the same token sequence, where one occurrence is given a certain label and the other occurrence is given a certain label. We show these counts both within documents, as well as over the whole corpus. As we would expect, most pairs of the same entity sequence are labeled the same(i.e. the diagonal has most of the density) at both the document and corpus levels. These statistics are from the CoNLL 2003 English training set.

	Document Level Statistics				Corpus Level Statistics			
	PER	LOC	ORG	MISC	PER	LOC	ORG	MISC
PER	1941	5	2	3	9111	401	261	38
LOC	0	167	6	63	68	4560	580	1543
ORG	22	328	819	191	221	19683	5131	4752
MISC	14	224	7	365	50	12713	329	8768

Table 2: Table showing the number of (token sequence, token subsequence) pairs where the token sequence is assigned a certain entity label, and the token subsequence is assigned a certain entity label. We show these counts both within documents, as well as over the whole corpus. Rows correspond to sequences, and columns to subsequences. These statistics are from the CoNLL 2003 English training set.

rences of a particular token sequence (or similar token sequences) are unlikely to have different entity labels. While this constraint holds strongly at the level of a document, there exists additional value to be derived by enforcing this constraint less strongly across different documents. We want to model label consistency as a *soft* and not a *hard* constraint; while we want to encourage different occurrences of similar token sequences to get labeled as the same entity, we do not want to force this to always hold, since there do exist exceptions, as can be seen from the off-diagonal entries of tables 1 and 2.

A named entity recognition system modeling this structure would encourage all the occurrences of the token sequence to the same entity type, thereby sharing evidence among them. Thus, if the system has strong evidence about the label of a given token sequence, but is relatively unsure about the label to be assigned to another occurrence of a similar token sequence, the system can gain significantly by using the information about the label assigned to the former occurrence, to label the relatively ambiguous token sequence, leading to accuracy improvements.

The strength of the label consistency constraint, can be seen from statistics extracted from the CoNLL 2003 English training data. Table 1 shows the counts of entity labels pairs assigned for each pair of identical token sequences both within a document and across the whole corpus. As we would expect, inconsistent labelings are relatively rare and most pairs of the same entity sequence

are labeled the same(i.e. the diagonal has most of the density) at both the document and corpus levels. A notable exception to this is the labeling of the same text as both organization and location within the same document and across documents. This is a due to the large amount of sports news in the CoNLL dataset due to which city and country names are often also team names. We will see that our approach is capable of exploiting this as well, i.e. we can learn a model which would not penalize an Organization-Location inconsistency as strongly as it penalizes other inconsistencies.

In addition, we also want to model subsequence constraints: having seen *Albert Einstein* earlier in a document as a person is a good indicator that a subsequent occurrence of *Einstein* should also be labeled as a person. Here, we would expect that a subsequence would gain much more by knowing the label of a supersequence, than the other way around.

However, as can be seen from table 2, we find that the consistency constraint does not hold nearly so strictly in this case. A very common case of this in the CoNLL dataset is that of documents containing references to both *The China Daily*, a newspaper, and *China*, the country (Finkel et al., 2005). The first should be labeled as an organization, and second as a location. The counts of subsequence labelings within a document and across documents listed in Table 2, show that there are many off-diagonal entries: the *China Daily* case is among the most common, occurring 328 times in the dataset. Just as we can model off-diagonal pat-

terns with exact token sequence matches, we can also model off-diagonal patterns for the token sub-sequence case.

In addition, we could also derive some value by enforcing some label consistency at the level of an individual token. Obviously, our model would learn much lower weights for these constraints, when compared to label consistency at the level of token sequences.

4 Our Approach to Handling non-local Dependencies

To handle the non-local dependencies between same and similar token sequences, we define three sets of feature pairs where one member of the feature pair corresponds to a function of aggregate statistics of the output of the first CRF at the document level, and the other member corresponds to a function of aggregate statistics of the output of the first CRF over the whole test corpus. Thus this gives us six additional feature types for the second round CRF, namely Document-level Token-majority features, Document-level Entity-majority features, Document-level Superentity-majority features, Corpus-level Token-majority features, Corpus-level Entity-majority features and Corpus-level Superentity-majority features. These feature types are described in detail below.

All these features are a function of the output labels of the first CRF, where predictions on the test set are obtained by training on all the data, and predictions on the train data are obtained by 10 fold cross-validation (details in the next section). Our features fired based on document and corpus level statistics are:

- **Token-majority features:** These refer to the majority label assigned to the particular token in the document/corpus. Eg: Suppose we have three occurrences of the token *Australia*, such that two are labeled *Location* and one is labeled *Organization*, our token-majority feature would take value *Location* for all three occurrences of the token. This feature can enable us to capture some dependence between token sequences corresponding to a single entity and having common tokens.
- **Entity-majority features:** These refer to the majority label assigned to the particular entity in the document/corpus. Eg: Suppose we

have three occurrences of the *entity sequence* (we define it as a token sequence labeled as a single entity by the first stage CRF) *Bank of Australia*, such that two are labeled *Organization* and one is labeled *Location*, our entity-majority feature would take value *Organization* for all tokens in all three occurrences of the entity sequence. This feature enables us to capture the dependence between identical entity sequences. For token labeled as not a Named Entity by the first CRF, this feature returns the majority label assigned to that token when it occurs as a single token named entity.

- **Superentity-majority features:** These refer to the majority label assigned to supersequences of the particular entity in the document/corpus. By entity supersequences, we refer to entity sequences, that strictly contain within their span, another entity sequence. For example, if we have two occurrences of *Bank of Australia* labeled *Organization* and one occurrence of *Australia Cup* labeled *Miscellaneous*, then for all occurrences of the entity *Australia*, the superentity-majority feature would take value *Organization*. This feature enables us to take into account labels assigned to supersequences of a particular entity, while labeling it. For token labeled as not a Named Entity by the first CRF, this feature returns the majority label assigned to all entities containing the token within their span.

The last feature enables entity sequences to benefit from labels assigned to entities which are entity supersequences of it. We attempted to add subentity-majority features, analogous to the superentity-majority features to model dependence on entity subsequences, but got no benefit from it. This is intuitive, since the basic sequence model would usually be much more certain about labels assigned to the entity supersequences, since they are longer and have more contextual information. As a result of this, while there would be several cases in which the basic sequence model would be uncertain about labels of entity subsequences but relatively certain about labels of token supersequences, the converse is very unlikely. Thus, it is difficult to profit from labels of entity subsequences while labeling entity sequences. We also attempted using more fine

grained features corresponding to the majority label of supersequences that takes into account the position of the entity sequence in the entity supersequence (whether the entity sequence occurs in the start, middle or end of the supersequence), but could obtain no additional gains from this.

It is to be noted that while deciding if token sequences are equal or hold a subsequence-supersequence relation, we ignore case, which clearly performs better than being sensitive to case. This is because our dataset contains several entities in allCaps such as *AUSTRALIA*, especially in news headlines. Ignoring case enables us to model dependences with other occurrences with a different case such as *Australia*.

It may appear at first glance, that our framework can only learn to encourage entities to switch to the most popular label assigned to other occurrences of the entity sequence and similar entity sequences. However this framework is capable of learning interesting off-diagonal patterns as well. To understand this, let us consider the example of different occurrences of token sequences being labeled *Location* and *Organization*. Suppose, the majority label of the token sequence is *Location*. While this majority label would encourage the second CRF to switch the labels of all occurrences of the token sequence to *Location*, it would not strongly discourage the CRF from labeling these as *Organization*, since there would be several occurrences of token sequences in the training data labeled *Organization*, with the majority label of the token sequence being *Location*. However it would discourage the other labels strongly. The reasoning is analogous when the majority label is *Organization*.

In case of a tie (when computing the majority label), if the label assigned to a particular token sequence is one of the majority labels, we fire the feature corresponding to that particular label being the majority label, instead of breaking ties arbitrarily. This is done to encourage the second stage CRF to make its decision based on local information, in the absence of compelling non-local information to choose a different label.

5 Advantages of our approach

With our two-stage approach, we manage to get improvements on the F1 measure over existing approaches that model non-local dependencies. At the same time, the simplicity of our two-stage ap-

proach keeps inference time down to just the inference time of two sequential CRFs, when compared to approaches such as those of Finkel et al. (2005) who report that their inference time with Gibbs sampling goes up by a factor of about 30, compared to the Viterbi algorithm for the sequential CRF.

Below, we give some intuition about areas for improvement in existing work and explain how our approach incorporates the improvements.

- Most existing work to capture label-consistency, has attempted to create all $\binom{n}{2}$ pairwise dependencies between the different occurrences of an entity, (Finkel et al., 2005; Sutton and McCallum, 2004), where n is the number of occurrences of the given entity. This complicates the dependency graph making inference harder. It also leads to the penalty for deviation in labeling to grow linearly with n , since each entity would be connected to $\Theta(n)$ entities. When an entity occurs several times, these models would force all occurrences to take the same value. This is not what we want, since there exist several instances in real-life data where different entities like persons and organizations share the same name. Thus, our approach makes a certain entity's label depend on certain *aggregate information* of other labels assigned to the same entity, and does not enforce pairwise dependencies.
- We also exploit the fact that the predictions of a learner that takes non-local dependencies into account would have a good amount of overlap with a sequential CRF, since the sequence model is already quite competitive. We use this intuition to approximate the aggregate information about labels assigned to other occurrences of the entity by the non-local model, with the aggregate information about labels assigned to other occurrences of the entity by the sequence model. This intuition enables us to learn weights for non-local dependencies in two stages; we first get predictions from a regular sequential CRF and in turn use aggregate information about predictions made by the CRF as extra features to train a second CRF.
- Most work has looked to model non-local dependencies only within a document (Finkel

et al., 2005; Chieu and Ng, 2002; Sutton and McCallum, 2004; Bunescu and Mooney, 2004). Our model can capture the weaker but still important consistency constraints across the whole document collection, whereas previous work has not, for reasons of tractability. Capturing label-consistency at the level of the whole test corpus is particularly helpful for token sequences that appear only once in their documents, but occur a few times over the corpus, since they do not have strong non-local information from within the document.

- For training our second-stage CRF, we need to get predictions on our train data as well as test data. Suppose we were to use the same train data to train the first CRF, we would get unrealistically good predictions on our train data, which would not be reflective of its performance on the test data. One option is to partition the train data. This however, can lead to a drop in performance, since the second CRF would be trained on less data. To overcome this problem, we make predictions on our train data by doing a 10-fold cross validation on the train data. For predictions on the test data, we use all the training data to train the CRF. Intuitively, we would expect that the quality of predictions with 90% of the train data would be similar to the quality of predictions with all the training data. It turns out that this is indeed the case, as can be seen from our improved performance.

6 Experiments

6.1 Dataset and Evaluation

We test the effectiveness of our technique on the CoNLL 2003 English named entity recognition dataset downloadable from <http://cnts.uia.ac.be/conll2003/ner/>. The data comprises Reuters newswire articles annotated with four entity types: *person* (PER), *location* (LOC), *organization* (ORG), and *miscellaneous* (MISC). The data is separated into a training set, a development set (testa), and a test set (testb). The training set contains 945 documents, and approximately 203,000 tokens and the test set has 231 documents and approximately 46,000 tokens. Performance on this task is evaluated by measuring the precision and recall of annotated entities (and not tokens), combined into an F1 score. There is no partial credit for labeling part

of an entity sequence correctly; an incorrect entity boundary is penalized as both a false positive and as a false negative.

6.2 Results and Discussion

It can be seen from table 3, that we achieve a 12.6% relative error reduction, by restricting ourselves to features approximating non-local dependency within a document, which is higher than other approaches modeling non-local dependencies within a document. Additionally, by incorporating non-local dependencies across documents in the test corpus, we manage a 13.3% relative error reduction, over an already competitive baseline. We can see that all three features approximating non-local dependencies within a document yield reasonable gains. As we would expect the additional gains from features approximating non-local dependencies across the whole test corpus are relatively small.

We use the approximate randomization test (Yeh, 2000) for statistical significance of the difference between the basic sequential CRF and our second round CRF, which has additional features derived from the output of the first CRF. With a 1000 iterations, our improvements were statistically significant with a p-value of 0.001. Since this value is less than the cutoff threshold of 0.05, we reject the null hypothesis.

The simplicity of our approach makes it easy to incorporate dependencies across the whole corpus, which would be relatively much harder to incorporate in approaches like (Bunescu and Mooney, 2004) and (Finkel et al., 2005). Additionally, our approach makes it possible to do inference in just about twice the inference time with a single sequential CRF; in contrast, approaches like Gibbs Sampling that model the dependencies directly can increase inference time by a factor of 30 (Finkel et al., 2005).

An analysis of errors by the first stage CRF revealed that most errors are that of single token entities being mislabeled or missed altogether followed by a much smaller percentage of multiple token entities mislabelled completely. All our features directly encode information that is useful to reducing these errors. The widely prevalent boundary detection error is that of missing a single-token entity (i.e. labeling it as *Other(O)*). Our approach helps correct many such errors based on occurrences of the token in other

F1 scores on the CoNLL Dataset						
Approach	LOC	ORG	MISC	PER	ALL	Relative Error reduction
Bunescu and Mooney (2004) (Relational Markov Networks)						
Only Local Templates	-	-	-	-	80.09	11.1%
Global and Local Templates	-	-	-	-	82.30	
Finkel et al. (2005)(Gibbs Sampling)						
Local+Viterbi	88.16	80.83	78.51	90.36	85.51	9.3%
Non Local+Gibbs	88.51	81.72	80.43	92.29	86.86	
Our Approach with the 2-stage CRF						
Baseline CRF	88.09	80.88	78.26	89.76	85.29	12.6%
+ Document token-majority features	89.17	80.15	78.73	91.60	86.50	
+ Document entity-majority features	89.50	81.98	79.38	91.74	86.75	
+ Document superentity-majority features	89.52	82.27	79.76	92.71	87.15	
+ Corpus token-majority features	89.48	82.36	79.59	92.65	87.13	
+ Corpus entity-majority features	89.72	82.40	79.71	92.65	87.23	
+ Corpus superentity-majority features (All features)	89.80	82.39	79.76	92.57	87.24	13.3%

Table 3: Table showing improvements obtained with our additional features, over the baseline CRF. We also compare our performance against (Bunescu and Mooney, 2004) and (Finkel et al., 2005) and find that we manage higher relative improvement than existing work despite starting from a very competitive baseline CRF.

named entities. Other kinds of boundary detection errors involving multiple tokens are very rare. Our approach can also handle these errors by encouraging certain tokens to take different labels. This together with the clique features encoding the markovian dependency among neighbours can correct some multiple-token boundary detection errors.

7 Related Work

Recent work looking to directly model non-local dependencies and do approximate inference are that of Bunescu and Mooney (2004), who use a Relational Markov Network (RMN) (Taskar et al., 2002) to explicitly model long-distance dependencies, Sutton and McCallum (2004), who introduce skip-chain CRFs, which add additional non-local edges to the underlying CRF sequence model (which Bunescu and Mooney (2004) lack) and Finkel et al. (2005) who hand-set penalties for inconsistency in labels based on the training data and then use Gibbs Sampling for doing approximate inference where the goal is to obtain the label sequence that maximizes the product of the CRF objective function and their penalty. Unfortunately, in the RMN model, the dependencies must be defined in the model structure before doing any inference, and so the authors use heuristic part-of-speech patterns, and then add dependencies between these text spans using clique templates. This generates an extremely large number of overlapping candidate entities, which renders necessary additional templates to enforce the constraint that text subsequences cannot both be

different entities, something that is more naturally modeled by a CRF. Another disadvantage of this approach is that it uses loopy belief propagation and a voted perceptron for approximate learning and inference, which are inherently unstable algorithms leading to convergence problems, as noted by the authors. In the skip-chain CRFs model, the decision of which nodes to connect is also made heuristically, and because the authors focus on named entity recognition, they chose to connect all pairs of identical capitalized words. They also utilize loopy belief propagation for approximate learning and inference. It is hard to directly extend their approach to model dependencies richer than those at the token level.

The approach of Finkel et al. (2005) makes it possible to model a broader class of long-distance dependencies than Sutton and McCallum (2004), because they do not need to make any initial assumptions about which nodes should be connected and they too model dependencies between whole token sequences representing entities and between entity token sequences and their token supersequences that are entities. The disadvantage of their approach is the relatively ad-hoc selection of penalties and the high computational cost of running Gibbs sampling.

Early work in discriminative NER employed two stage approaches that are broadly similar to ours, but the effectiveness of this approach appears to have been overlooked in more recent work. Mikheev et al. (1999) exploit label consistency information within a document using relatively ad hoc multi-stage labeling procedures. Borth-

wick (1999) used a two-stage approach similar to ours with CMM's where *Reference Resolution* features which encoded the frequency of occurrences of other entities similar to the current token sequence, were derived from the output of the first stage. Malouf (2002) and Curran and Clark (2003) condition the label of a token at a particular position on the label of the most recent previous instance of that same token in a previous sentence of the same document. This violates the Markov property and therefore instead of finding the maximum likelihood sequence over the entire document (exact inference), they label one sentence at a time, which allows them to condition on the maximum likelihood sequence of previous sentences. While this approach is quite effective for enforcing label consistency in many NLP tasks, it permits a forward flow of information only, which can result in loss of valuable information. Chieu and Ng (2002) propose a solution to this problem: for each token, they define additional features based on known information, taken from other occurrences of the same token in the document. This approach has the advantage of allowing the training procedure to automatically learn good weights for these "global" features relative to the local ones. However, it is hard to extend this to incorporate other types of non-local structure.

8 Conclusion

We presented a two stage approach to model non-local dependencies and saw that it outperformed existing approaches to modeling non-local dependencies. Our approach also made it easy to exploit various dependencies across documents in the test corpus, whereas incorporating this information in most existing approaches would make them intractable due to the complexity of the resultant graphical model. Our simple approach is also very computationally efficient since the inference time is just twice the inference time of the basic sequential CRF, while for approaches doing approximate inference, the inference time is often well over an order of magnitude over the basic sequential CRF. The simplicity of our approach makes it easy to understand, implement, and adapt to new applications.

Acknowledgments

We wish to Jenny R. Finkel for discussions on NER and her CRF code. Also, thanks to Trond

Grenager for NER discussions and to William Morgan for help with statistical significance tests. Also, thanks to Vignesh Ganapathy for helpful discussions and Rohini Rajaraman for comments on the writeup.

This work was supported in part by a Scottish Enterprise Edinburgh-Stanford Link grant (R37588), as part of the EASIE project.

References

- A. Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. thesis, New York University.
- R. Bunescu and R. J. Mooney. 2004. Collective information extraction with relational Markov networks. In *Proceedings of the 42nd ACL*, pages 439–446.
- H. L. Chieu and H. T. Ng. 2002. Named entity recognition: a maximum entropy approach using global information. In *Proceedings of the 19th Coling*, pages 190–196.
- J. R. Curran and S. Clark. 2003. Language independent NER using a maximum entropy tagger. In *Proceedings of the 7th CoNLL*, pages 164–167.
- J. Finkel, T. Grenager, and C. D. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 42nd ACL*.
- D. Freitag and A. McCallum. 1999. Information extraction with HMMs and shrinkage. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*.
- S. Geman and D. Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.
- J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th ICML*, pages 282–289. Morgan Kaufmann, San Francisco, CA.
- T. R. Leek. 1997. Information extraction using hidden Markov models. Master's thesis, U.C. San Diego.
- R. Malouf. 2002. Markov models for language-independent named entity recognition. In *Proceedings of the 6th CoNLL*, pages 187–190.
- A. McCallum, D. Freitag, and F. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the 17th ICML*, pages 591–598. Morgan Kaufmann, San Francisco, CA.
- A. Mikheev, M. Moens, and C. Grover. 1999. Named entity recognition without gazetteers. In *Proceedings of the 9th EACL*, pages 1–8.
- J. Pearl. 1988. Probabilistic reasoning in intelligent systems: Networks of plausible inference. In *Morgan Kaufmann*.
- F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of NAACL-2003*, pages 134–141.
- C. Sutton and A. McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *ICML Workshop on Statistical Relational Learning and Its connections to Other Fields*.
- B. Taskar, P. Abbeel, and D. Koller. 2002. Discriminative probabilistic models for relational data. In *Proceedings of UAI-02*.
- J. S. Yedidia, W. T. Freeman, and Y. Weiss. 2000. Generalized belief propagation. In *Proceedings of NIPS-2000*, pages 689–695.
- Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of COLING 2000*.

Learning Transliteration Lexicons from the Web

Jin-Shea Kuo^{1,2}

¹Chung-Hwa Telecom.
Laboratories, Taiwan

jskuo@cht.com.tw

Haizhou Li

Institute for Infocomm
Research, Singapore

hzli@ieee.org

Ying-Kuei Yang²

²National Taiwan University of
Science and Technology, Taiwan

ykyang@mouse.ee.
ntust.edu.tw

Abstract

This paper presents an adaptive learning framework for Phonetic Similarity Modeling (PSM) that supports the automatic construction of transliteration lexicons. The learning algorithm starts with minimum prior knowledge about machine transliteration, and acquires knowledge iteratively from the Web. We study the active learning and the unsupervised learning strategies that minimize human supervision in terms of data labeling. The learning process refines the PSM and constructs a transliteration lexicon at the same time. We evaluate the proposed PSM and its learning algorithm through a series of systematic experiments, which show that the proposed framework is reliably effective on two independent databases.

1 Introduction

In applications such as cross-lingual information retrieval (CLIR) and machine translation (MT), there is an increasing need to translate out-of-vocabulary (OOV) words, for example from an alphabetical language to Chinese. Foreign proper names constitute a good portion of OOV words, which are translated into Chinese through transliteration. Transliteration is a process of translating a foreign word into a native language by preserving its pronunciation in the original language, otherwise known as *translation-by-sound*.

MT and CLIR systems rely heavily on bilingual lexicons, which are typically compiled manually. However, in view of the current information explosion, it is labor intensive, if not impossible, to compile a complete proper nouns lexicon. The Web is growing at a fast pace and is providing a live information source that is rich in transliterations. This paper presents a novel

solution for automatically constructing an English-Chinese transliteration lexicon from the Web.

Research on automatic transliteration has reported promising results for *regular* transliteration (Wan and Verspoor, 1998; Li et al, 2004), where transliterations follow rigid guidelines. However, in Web publishing, translators in different countries and regions may not observe common guidelines. They often skew the transliterations in different ways to create special meanings to the sound equivalents, resulting in *casual* transliterations. In this case, the common generative models (Li et al, 2004) fail to predict the transliteration most of the time. For example, “Coca Cola” is transliterated into “可口可樂 /Ke-Kou-Ke-Le/” as a sound equivalent in Chinese, which literally means “happiness in the mouth”. In this paper, we are interested in constructing lexicons that cover both *regular* and *casual* transliterations.

When a new English word is first introduced, many transliterations are invented. Most of them are *casual* transliterations because a regular transliteration typically does not have many variations. After a while, the transliterations converge into one or two popular ones. For example, “Taxi” becomes “的士 /Di-Shi/” in China and “德士 /De-Shi/” in Singapore. Therefore, the adequacy of a transliteration entry could be judged by its popularity and its conformity with the *translation-by-sound* principle. In any case, the phonetic similarity should serve as the primary basis of judgment.

This paper is organized as follows. In Section 2, we briefly introduce prior works pertaining to machine transliteration. In Section 3, we propose a phonetic similarity model (PSM) for confidence scoring of transliteration. In Section 4, we propose an adaptive learning process for PSM modeling and lexicon construction. In Section 5, we conduct experiments to evaluate different adaptive learning strategies. Finally, we conclude in Section 6.

2 Related Work

In general, studies of transliteration fall into two categories: transliteration modeling (TM) and extraction of transliteration pairs (EX) from corpora.

The TM approach models phoneme-based or grapheme-based mapping rules using a generative model that is trained from a large bilingual lexicon, with the objective of translating unknown words on the fly. The efforts are centered on establishing the phonetic relationship between transliteration pairs. Most of these works are devoted to phoneme¹-based transliteration modeling (Wan and Verspoor 1998, Knight and Graehl, 1998). Suppose that *EW* is an English word and *CW* is its prospective Chinese transliteration. The phoneme-based approach first converts *EW* into an intermediate phonemic representation *P*, and then converts the phonemic representation into its Chinese counterpart *CW*. In this way, *EW* and *CW* form an *E-C* transliteration pair.

In this approach, we model the transliteration using two conditional probabilities, $P(CW|P)$ and $P(P|EW)$, in a generative model $P(CW|EW) = P(CW|P)P(P|EW)$. Meng (2001) proposed a rule-based mapping approach. Virga and Khudanpur (2003) and Kuo et al (2005) adopted the noisy-channel modeling framework. Li et al (2004) took a different approach by introducing a joint source-channel model for direct orthography mapping (DOM), which treats transliteration as a statistical machine translation problem under monotonic constraints. The DOM approach, which is a grapheme-based approach, significantly outperforms the phoneme-based approaches in *regular* transliterations. It is noted that the state-of-the-art accuracy reported by Li et al (2004) for *regular* transliterations of the Xinhua database is about 70.1%, which leaves much room for improvement if one expects to use a generative model to construct a lexicon for *casual* transliterations.

EX research is motivated by information retrieval techniques, where people attempt to extract transliteration pairs from corpora. The EX approach aims to construct a large and up-to-date transliteration lexicon from live corpora. Towards this objective, some have proposed extracting translation pairs from parallel or comparable bitext using co-occurrence analysis

or a context-vector approach (Fung and Yee, 1998; Nie et al, 1999). These methods compare the semantic similarities between words without taking their phonetic similarities into accounts. Lee and Chang (2003) proposed using a probabilistic model to identify *E-C* pairs from aligned sentences using phonetic clues. Lam et al (2004) proposed using semantic and phonetic clues to extract *E-C* pairs from comparable corpora. However, these approaches are subject to the availability of parallel or comparable bitext. A method that explores non-aligned text was proposed by harvesting *katakana*-English pairs from query logs (Brill et al, 2001). It was discovered that the unsupervised learning of such a transliteration model could be overwhelmed by noisy data, resulting in a decrease in model accuracy.

Many efforts have been made in using Web-based resources for harvesting transliteration/translation pairs. These include exploring query logs (Brill et al, 2001), unrelated corpus (Rapp, 1999), and parallel or comparable corpus (Fung and Yee, 1998; Nie et al, 1999; Huang et al 2005). To establish correspondence, these algorithms usually rely on one or more statistical clues, such as the correlation between word frequencies, cognates of similar spelling or pronunciations. They include two aspects. First, a robust mechanism that establishes statistical relationships between bilingual words, such as a phonetic similarity model which is motivated by the TM research; and second, an effective learning framework that is able to adaptively discover new events from the Web. In the prior work, most of the phonetic similarity models were trained on a static lexicon. In this paper, we address the EX problem by exploiting a novel Web-based resource. We also propose a phonetic similarity model that generates confidence scores for the validation of *E-C* pairs.

In Chinese webpages, translated or transliterated terms are frequently accompanied by their original Latin words. The latter serve as the appositives of the former. A sample search result for the query submission “Kuro” is the bilingual snippet² “...經營 Kuro 庫洛 P2P 音樂交換軟體的飛行網, 3 日發表 P2P 與版權爭議的解決方案— C2C (Content to Community)...”. The co-occurrence statistics in such a snippet was shown to be useful in constructing a transitive translation model (Lu et al, 2002). In the

¹ Both phoneme and syllable based approaches are referred to as phoneme-based here.

² A bilingual snippet refers to a Chinese predominant text with embedded English appositives.

example above, “Content to Community” is not a transliteration of C2C, but rather an acronym expansion, while “庫洛 /Ku-Luo/”, as underlined, presents a transliteration for “Kuro”. What is important is that the *E-C* pairs are always closely collocated. Inspired by this observation, we propose an algorithm that searches over the close context of an English word in a bilingual snippet for the word’s transliteration candidates.

The contributions of this paper include: (i) an approach to harvesting real life *E-C* transliteration pairs from the Web; (ii) a phonetic similarity model that evaluates the confidence of so extracted *E-C* pair candidates; (iii) a comparative study of several machine learning strategies.

3 Phonetic Similarity Model

English and Chinese have different syllable structures. Chinese is a syllabic language where each Chinese character is a syllable in either consonant-vowel (CV) or consonant-vowel-nasal (CVN) structure. A Chinese word consists of a sequence of characters, phonetically a sequence of syllables. Thus, in first *E-C* transliteration, it is a natural choice to syllabify an English word by converting its phoneme sequence into a sequence of Chinese-like syllables, and then convert it into a sequence of Chinese characters.

There have been several effective algorithms for the syllabification of English words for transliteration. Typical syllabification algorithms first convert English graphemes to phonemes, referred to as the letter-to-sound transformation, then syllabify the phoneme sequence into a syllable sequence. For this method, a letter-to-sound conversion is needed (Pagel, 1998; Jurafsky, 2000). The phoneme-based syllabification algorithm is referred to as PSA. Another syllabification technique attempts to map the grapheme of an English word to syllables directly (Kuo and Yang, 2004). The grapheme-based syllabification algorithm is referred to as GSA. In general, the size of a phoneme inventory is smaller than that of a grapheme inventory. The PSA therefore requires less training data for statistical modeling (Knight, 1998); on the other hand, the grapheme-based method gets rid of the letter-to-sound conversion, which is one of the main causes of transliteration errors (Li et al, 2004).

Assuming that Chinese transliterations always co-occur in proximity to their original English words, we propose a phonetic similarity

modeling (PSM) that measures the phonetic similarity between candidate transliteration pairs. In a bilingual snippet, when an English word *EW* is spotted, the method searches for the word’s possible Chinese transliteration *CW* in its neighborhood. *EW* can be a single word or a phrase of multiple English words. Next, we formulate the PSM and the estimation of its parameters.

3.1 Generative Model

Let $ES = \{es_1, \dots, es_m, \dots, es_M\}$ be a sequence of English syllables derived from *EW*, using the PSA or GSA approach, and $CS = \{cs_1, \dots, cs_n, \dots, cs_N\}$ be the sequence of Chinese syllables derived from *CW*, represented by a Chinese character string $CW \rightarrow c_1, \dots, c_n, \dots, c_N$. *EW* and *CW* is a transliteration pair. The *E-C* transliteration can be considered a generative process formulated by the noisy channel model, with *EW* as the input and *CW* as the output. $P(EW/CW)$ is estimated to characterize the noisy channel, known as the transliteration probability. $P(CW)$ is a language model to characterize the source language. Applying Bayes’ rule, we have

$$P(CW/EW) = P(EW/CW)P(CW)/P(EW) \quad (1)$$

Following the *translation-by-sound* principle, the transliteration probability $P(EW/CW)$ can be approximated by the phonetic confusion probability $P(ES/CS)$, which is given as

$$P(ES/CS) = \max_{\Delta \in \Phi} P(ES, \Delta/CS), \quad (2)$$

where Φ is the set of all possible alignment paths between *ES* and *CS*. It is not trivial to find the best alignment path Δ . One can resort to a dynamic programming algorithm. Assuming conditional independence of syllables in *ES* and *CS*, we have $P(ES/CS) = \prod_{m=1}^M p(es_m/cs_m)$ in a special case where $M = N$. Note that, typically, we have $N \leq M$ due to syllable elision. We introduce a null syllable φ and a dynamic warping strategy to evaluate $P(ES/CS)$ when $M \neq N$ (Kuo et al, 2005). With the phonetic approximation, Eq.(1) can be rewritten as

$$P(CW/EW) \approx P(ES/CS)P(CW)/P(EW) \quad (3)$$

The language model in Eq.(3) can be represented by Chinese characters *n*-gram statistics.

$$P(CW) = \prod_{n=1}^N p(c_n/c_{n-1}, c_{n-2}, \dots, c_1) \quad (4)$$

In adopting bigram, Eq.(4) is rewritten as $P(CW) \approx p(c_1) \prod_{n=2}^N p(c_n / c_{n-1})$. Note that the context of EW usually has a number of competing Chinese transliteration candidates in a set, denoted as Ω . We rank the candidates by Eq.(1) to find the most likely CW for a given EW . In this process, $P(EW)$ can be ignored because it is the same for all CW candidates. The CW candidate that gives the highest posterior probability is considered the most probable candidate CW' .

$$\begin{aligned} CW' &= \arg \max_{CW \in \Omega} P(CW / EW) \\ &\approx \arg \max_{CW \in \Omega} P(ES / CS) P(CW) \end{aligned} \quad (5)$$

However, the most probable CW' isn't necessarily the desired transliteration. The next step is to examine if CW' and EW indeed form a genuine $E-C$ pair. We define the confidence of the $E-C$ pair as the posterior odds similar to that in a hypothesis test under the Bayesian interpretation. We have H_0 , which hypothesizes that CW' and EW form an $E-C$ pair, and H_1 , which hypothesizes otherwise. The posterior odds is given as follows,

$$\sigma = \frac{P(H_0 / EW)}{P(H_1 / EW)} \approx \frac{P(ES / CS') P(CW')}{\sum_{\substack{CW \in \Omega \\ CW \neq CW'}} P(ES / CS) P(CW)} \quad (6)$$

where CS' is the syllable sequence of CW' , $p(H_1 / EW)$ is approximated by the probability mass of the competing candidates of CW' , or $\sum_{\substack{CW \in \Omega \\ CW \neq CW'}} P(ES / CS) P(CW)$. The higher the σ

is, the more probable that hypothesis H_0 overtakes H_1 . The PSM formulation can be seen as an extension to prior work (Brill et al, 2001) in transliteration modeling. We introduce the posterior odds σ as the confidence score so that $E-C$ pairs that are extracted from different contexts can be directly compared. In practice, we set a threshold for σ to decide a cutoff point for $E-C$ pairs short-listing.

3.2 PSM Estimation

The PSM parameters are estimated from the statistics of a given transliteration lexicon, which is a collection of manually selected $E-C$ pairs in supervised learning, or a collection of high confidence $E-C$ pairs in unsupervised learning. An initial PSM is bootstrapped using prior knowledge such as rule-based syllable mapping. Then we align the $E-C$ pairs with the PSM and

derive syllable mapping statistics for PSA and GSA syllabifications. A final PSM is a linear combination of the PSA-based PSM (PSA-PSM) and the GSA-based PSM (GSA-PSM). The PSM parameter $p(es_m / cs_n)$ can be estimated by an *Expectation-Maximization* (EM) process (Dempster, 1977). In the *Expectation* step, we compute the counts of events such as $\# < es_m, cs_n >$ and $\# < cs_n >$ by force-aligning the $E-C$ pairs in the training lexicon Ψ . In the *Maximization* step, we estimate the PSM parameters $p(es_m / cs_n)$ by

$$p(es_m / cs_n) = \# < es_m, cs_n > / \# < cs_n >. \quad (7)$$

As the EM process guarantees non-decreasing likelihood probability $\prod_{\Psi} P(ES / CS)$, we let the EM process iterate until $\prod_{\Psi} P(ES / CS)$ converges. The EM process can be thought of as a refining process to obtain the best alignment between the $E-C$ syllables and at the same time a re-estimating process for PSM parameters. It is summarized as follows.

Start: Bootstrap PSM parameters $p(es_m / cs_n)$ using prior phonetic mapping knowledge

E-Step: Force-align corpus Ψ using existing $p(es_m / cs_n)$ and compute the counts of $\# < es_m, cs_n >$ and $\# < cs_n >$;

M-Step: Re-estimate $p(es_m / cs_n)$ using the counts from E-Step.

Iterate: Repeat E-Step and M-Step until $\prod_{\Psi} P(ES / CS)$ converges.

4 Adaptive Learning Framework

We propose an adaptive learning framework under which we learn PSM and harvest $E-C$ pairs from the Web at the same time. Conceptually, the adaptive learning is carried out as follows.

We obtain bilingual snippets from the Web by iteratively submitting queries to the Web search engines (Brin and Page, 1998). For each batch of querying, the query results are all normalized to plain text, from which we further extract qualified sentences. A qualified sentence has at least one English word. Under this criterion, a collection of qualified sentences can be extracted automatically. To label the $E-C$ pairs, each qualified sentence is manually checked based on the following transliteration criteria: (i) if an EW is partly translated phonetically and partly translated semantically, only the phonetic transliteration constituent is extracted to form a

transliteration pair; (ii) elision of English sound is accepted; (iii) multiple *E-C* pairs can appear in one sentence; (iv) an *EW* can have multiple valid Chinese transliterations and vice versa. The validation process results in a collection of qualified *E-C* pairs, also referred to as Distinct Qualified Transliteration Pairs (DQTPs).

As formulated in Section 3, the PSM is trained using a training lexicon in a data driven manner. It is therefore very important to ensure that in the learning process we have prepared a quality training lexicon. We establish a baseline system using supervised learning. In this approach, we use human labeled data to train a model. The advantage is that it is able to establish a model quickly as long as labeled data are available. However, this method also suffers from some practical issues. First, the derived model can only be as good as the data that it sees. An adaptive mechanism is therefore needed for the model to acquire new knowledge from the dynamically growing Web. Second, a massive annotation of database is labor intensive, if not entirely impossible.

To reduce the annotation needed, we discuss three adaptive strategies cast in the machine learning framework, namely active learning, unsupervised learning and active-unsupervised learning. The learning strategies can be depicted in Figure 1 with their difference being discussed next. We also train a baseline system using supervised learning approach as a reference point for benchmarking purpose.

4.1 Active Learning

Active learning is based on the assumption that a small number of labeled samples, which are DQTPs here, and a large number of unlabeled

samples are available. This assumption is valid in most NLP tasks. In contrast to supervised learning, where the entire corpus is labeled manually, active learning selects the most useful samples for labeling and adds the labeled examples to the training set to retrain the model. This procedure is repeated until the model achieves a certain level of performance. Practically, a batch of samples is selected each time. This is called batch-based sample selection (Lewis and Catlett, 1994), as shown in the search and ranking block in Figure 1.

For an active learning to be effective, we propose using three measures to select candidates for human labeling. First, we would like to select the most uncertain samples that are potentially highly informative for the PSM model. The informativeness of a sample can be quantified by its confidence score σ as in the PSM formulation. Ranking the *E-C* pairs by σ is referred to as C-rank. The samples of low C-rank are the interesting samples to be labeled. Second, we would like to select candidates that are of low frequency. Ranking by frequency is called F-rank. During Web crawling, most of the search engines use various strategies to prevent spamming and one of fundamental tasks is to remove the duplicated Web pages. Therefore, we assume that the bilingual snippets are all unique. Intuitively, *E-C* pairs of low frequency indicate uncommon events which are of higher interest to the model. Third, we would like to select samples upon which the PSA-PSM and GSA-PSM disagree the most. The disagreed upon samples represent new knowledge to the PSM. In short, we select low C-rank, low F-rank and PSM-disagreed samples for labeling because the high C-rank, high F-rank and PSM-agreed samples are already well known to the model.

4.2 Unsupervised Learning

Unsupervised learning skips the human labeling step. It minimizes human supervision by automatically labeling the data. This can be effective if prior knowledge about a task is available, for example, if an initial PSM can be built based on human crafted phonetic mapping rules. This is entirely possible. Kuo et al (2005) proposed using a cross-lingual phonetic confusion matrix resulting from automatic speech recognition to bootstrap an initial PSM model. The task of labeling samples is basically to distinguish the qualified transliteration pairs from the rest. Unlike the sample selection method in active learning, here we would like to

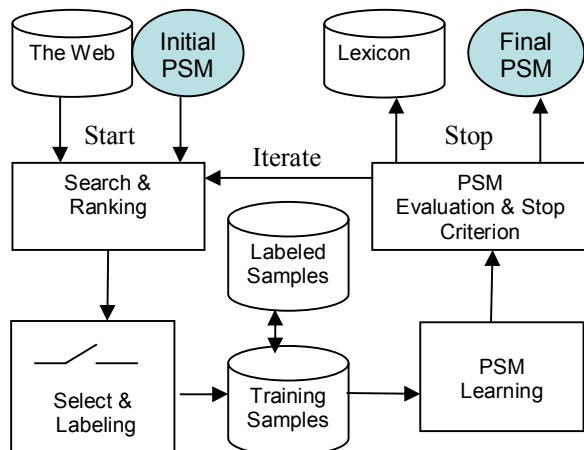


Figure 1. An adaptive learning framework for automatic construction of transliteration lexicon.

select the samples that are of high C-rank and high F-rank because they are more likely to be the desired transliteration pairs.

The difference between the active learning and the unsupervised learning strategies lies in that the former selects samples for human labeling, such as in the select & labeling block in Figure 1 before passing on for PSM learning, while the latter selects the samples automatically and assumes they are all correct DQTPs. The disadvantage of unsupervised learning is that it tends to reinforce its existing knowledge rather than to discover new events.

4.3 Active-Unsupervised Learning

The active learning and the unsupervised learning strategies can be complementary. Active learning minimizes the labeling effort by intelligently short-listing informative and representative samples for labeling. It makes sure that the PSM learns new and informative knowledge over iterations. Unsupervised learning effectively exploits the unlabelled data. It reinforces the knowledge that PSM has acquired and allows PSM to adapt to changes at no cost. However, we do not expect unsupervised learning to acquire new knowledge like active learning does. Intuitively, a better solution is to integrate the two strategies into one, referred to as the active-unsupervised learning strategy. In this strategy, we use active learning to select a small amount of informative and representative samples for labeling. At the same time, we select samples of high confidence score from the rest and consider them correct *E-C* pairs. We then merge the labeled set with the high-confidence set in the PSM re-training.

5 Experiments

We first construct a development corpus by crawling of webpages. This corpus consists of about 500 MB of webpages, called SET1 (Kuo et al, 2005). Out of 80,094 qualified sentences, 8,898 DQTPs are manually extracted from SET1, which serve as the gold standard in testing. To establish a baseline system, we first train a PSM using all 8,898 DQTPs in supervised manner and conduct a closed test on SET1 as in Table 1. We further implement three PSM learning strategies and conduct a systematic series of experiments.

	Precision	Recall	F-measure
closed-test	0.79	0.69	0.74

Table 1. Supervised learning test on SET1

5.1 Unsupervised Learning

We follow the formulation described in Section 4.2. First, we derive an initial PSM using randomly selected 100 seed DQTPs and simulate the Web-based learning process with the SET1: (i) select high F-rank and high C-rank *E-C* pairs using PSM, (ii) add the selected *E-C* pairs to the DQTP pool as if they are true DQTPs, and (iii) reestimate PSM by using the updated DQTP pool.

In Figure 2, we report the F-measure over iterations. The U_HF curve reflects the learning progress of using *E-C* pairs that occur more than once in the SET1 corpus (high F-rank). The U_HF_HR curve reflects the learning progress using a subset of *E-C* pairs from U_HF which has high posterior odds as defined in Eq.(6). Both selection strategies aim to select *E-C* pairs, which are as genuine as possible.

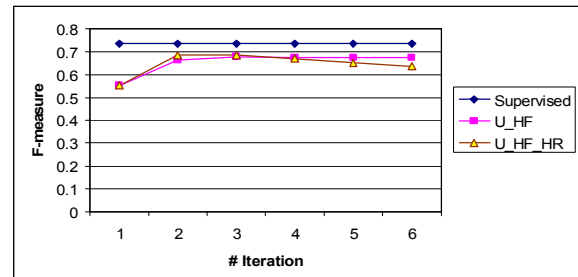


Figure 2. F-measure over iterations for unsupervised learning on SET1.

We found that both U_HF and U_HF_HR give similar results in terms of F-measure. Without surprise, more iterations don't always lead to better performance because unsupervised learning doesn't aim to acquiring new knowledge over iterations. Nevertheless, unsupervised learning improves the initial PSM in the first iteration substantially. It can serve as an effective PSM adaptation method.

5.2 Active Learning

The objective of active learning is to minimize human supervision by automatically selecting the most informative samples to be labeled. The effect of active learning is that it maximizes performance improvement with minimum annotation effort. Like in unsupervised learning, we start with the same 100 seed DQTPs and an initial PSM model and carry out experiments on SET1: (i) select low F-rank, low C-rank and GSA-PSM and PSA-PSM disagreed *E-C* pairs; (ii) label the selected pairs by removing the non-*E-C* pairs and add the labeled *E-C* pairs to the DQTP pool, and (iii) reestimate the PSM by using the updated DQTP pool.

To select the samples, we employ 3 different strategies: A_LF_LR, where we only select low F-rank and low C-rank candidates for labeling. A_DIFF, where we only select those that GSA-PSM and PSA-PSM disagreed upon; and A_DIFF_LF_LR, the union of A_LF_LR and A_DIFF selections. As shown in Figure 3, the F-measure of A_DIFF (0.729) and A_DIFF_LF_LR (0.731) approximate to that of supervised learning 0.735) after four iterations.

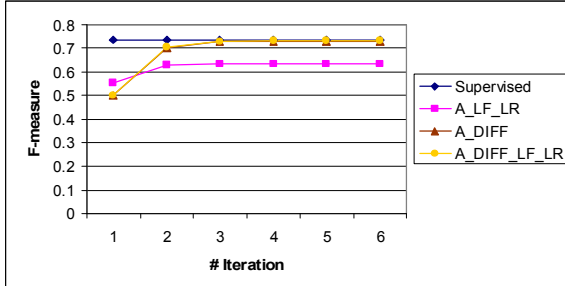


Figure 3. F-measure over iterations for active learning on SET1.

With almost identical performance as supervised learning, the active learning approach has greatly reduced the number of samples for manual labeling as reported in Table 2. It is found that for active learning to reach the performance of supervised learning, A_DIFF is the most effective strategy. It reduces the labeling effort by 89.0%, from 80,094 samples to 8,750.

	Sample selection	#samples labeled
Active learning	A_LF_LR	1,671
	A_DIFF	8,750
	A_DIFF_LF_LR	9,683
Supervised learning		80,094

Table 2. Number of total samples for manual labeling in 6 iterations of Figure 3.

5.3 Active Unsupervised Learning

It would be interesting to study the performance of combining unsupervised learning and active learning. The experiment is similar to that of active learning except that, in step (iii) of active learning, we take the unlabeled *high confidence* candidates (high F-rank and high C-rank as in U_HF_HR of Section 5.1) as the true labeled samples and add into the DQTP pool. The result is shown in Figure 4. Although active unsupervised learning was reported having promising results (Riccardi and Hakkani-Tur, 2003) in some NLP tasks, it has not been as effective as active learning alone in this

experiment probably due to the fact the unlabeled *high confidence* candidates are still too noisy to be informative.

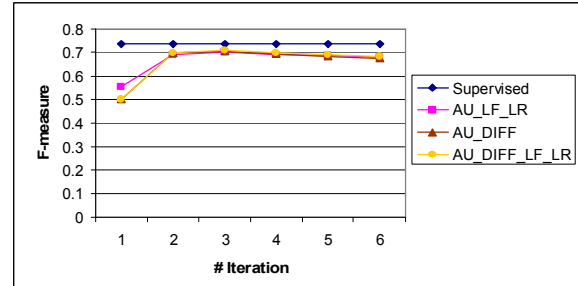


Figure 4. F-measure over iterations for active unsupervised learning on SET1.

5.4 Learning Transliteration Lexicons

The ultimate objective of building a PSM is to extract a transliteration lexicon from the Web by iteratively submitting queries and harvesting new transliteration pairs from the return results until no more new pairs. For example, by submitting “Robert” to search engines, we may get “Robert-羅伯特”, “Richard-理查” and “Charles-查爾斯” in return. In this way, new queries can be generated iteratively, thus new pairs are discovered. We pick the best performing SET1-derived PSM trained using A_DIFF_LF_LR active learning strategy and test it on a new database SET2 which is obtained in the same way as SET1.

	Before adaptation	After adaptation
#distinct E-C pairs	137,711	130,456
Precision	0.777	0.846
#expected DQTPs	107,001	110,365

Table 3. SET1-derived PSM adapted towards SET2.

SET2 contains 67,944 Web pages amounting to 3.17 GB. We extracted 2,122,026 qualified sentences from SET2. Using the PSM, we extract 137,711 distinct E-C pairs. As the gold standard for SET2 is unavailable, we randomly select 1,000 pairs for manual checking. A precision of 0.777 is reported. In this way, 107,001 DQTPs can be expected. We further carry out one iteration of unsupervised learning using U_HF_HR to adapt the SET1-derived PSM towards SET2. The results before and after adaptation are reported in Table 3. Like the experiment in Section 5.1, the unsupervised learning improves the PSM in terms of precision significantly.

6 Conclusions

We have proposed a framework for harvesting *E-C* transliteration lexicons from the Web using bilingual snippets. In this framework, we formulate the PSM learning and *E-C* pair evaluation methods. We have studied three strategies for PSM learning aiming at reducing the human supervision.

The experiments show that unsupervised learning is an effective way for rapid PSM adaptation while active learning is the most effective in achieving high performance. We find that the Web is a resourceful live corpus for real life *E-C* transliteration lexicon learning, especially for casual transliterations. In this paper, we use two Web databases SET1 and SET2 for simplicity. The proposed framework can be easily extended to an incremental learning framework for live databases. This paper has focused solely on use of phonetic clues for lexicon and PSM learning. We have good reason to expect the combining semantic and phonetic clues to improve the performance further.

References

- E. Brill, G. Kacmarcik, C. Brockett. 2001. Automatically Harvesting Katakana-English Term Pairs from Search Engine Query Logs, In *Proc. of NLPERS*, pp. 393-399.
- S. Brin and L. Page. 1998. The Anatomy of a Large-scale Hypertextual Web Search Engine, In *Proc. of 7th WWW*, pp. 107-117.
- 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, Ser. B. Vol. 39*, pp. 1-38.
- P. Fung and L.-Y. Yee. 1998. An IR Approach for Translating New Words from Nonparallel, Comparable Texts. In *Proc. of 17th COLING and 36th ACL*, pp. 414-420.
- F. Huang, Y. Zhang and Stephan Vogel. 2005. Mining Key Phrase Translations from Web Corpora. In *Proc. of HLT-EMNLP*, pp. 483-490.
- D. Jurafsky and J. H. Martin. 2000. *Speech and Language Processing*, pp. 102-120, Prentice-Hall, New Jersey.
- K. Knight and J. Graehl. 1998. Machine Transliteration, *Computational Linguistics*, Vol. 24, No. 4, pp. 599-612.
- J.-S. Kuo and Y.-K. Yang. 2004. Constructing Transliterations Lexicons from Web Corpora, In *the Companion Volume, 42nd ACL*, pp. 102-105.
- J.-S. Kuo and Y.-K. Yang. 2005. Incorporating Pronunciation Variation into Extraction of Transliterated-term Pairs from Web Corpora, In *Proc. of ICCV*, pp. 131-138.
- C.-J. Lee and J.-S. Chang. 2003. Acquisition of English-Chinese Transliterated Word Pairs from Parallel-Aligned Texts Using a Statistical Machine Transliteration Model, In *Proc. of HLT-NAACL Workshop Data Driven MT and Beyond*, pp. 96-103.
- D. D. Lewis and J. Catlett. 1994. Heterogeneous Uncertainty Sampling for Supervised Learning, In *Proc. of ICML 1994*, pp. 148-156.
- H. Li, M. Zhang and J. Su. 2004. A Joint Source Channel Model for Machine Transliteration, In *Proc. of 42nd ACL*, pp. 159-166.
- W. Lam, R.-Z. Huang and P.-S. Cheung. 2004. Learning Phonetic Similarity for Matching Named Entity Translations and Mining New Translations, In *Proc. of 27th ACM SIGIR*, pp. 289-296.
- W.-H. Lu, L.-F. Chien and H.-J. Lee. 2002. Translation of Web Queries Using Anchor Text Mining, *TALIP*, Vol. 1, Issue 2, pp. 159- 172.
- H. M. Meng, W.-K. Lo, B. Chen and T. Tang. 2001. Generate Phonetic Cognates to Handle Name Entities in English-Chinese Cross-Language Spoken Document Retrieval, In *Proc. of ASRU*, pp. 311-314.
- J.-Y. Nie, P. Isabelle, M. Simard, and R. Durand. 1999. Cross-language Information Retrieval based on Parallel Texts and Automatic Mining of Parallel Text from the Web”, In *Proc. of 22nd ACM SIGIR*, pp 74-81.
- V. Pagel, K. Lenzo and A. Black. 1998. Letter to Sound Rules for Accented Lexicon Compression, In *Proc. of ICSLP*, pp. 2015-2020.
- R. Rapp. 1999. Automatic Identification of Word Translations from Unrelated English and German Corpora, In *Proc. of 37th ACL*, pp. 519-526.
- G. Riccardi and D. Hakkani-Tür. 2003. Active and Unsupervised Learning for Automatic Speech Recognition. In *Proc. of 8th Eurospeech*.
- P. Virga and S. Khudanpur. 2003. Transliteration of Proper Names in Cross-Lingual Information Retrieval, In *Proc. of 41st ACL Workshop on Multilingual and Mixed Language Named Entity Recognition*, pp. 57-64.
- S. Wan and C. M. Verspoor. 1998. Automatic English-Chinese Name Transliteration for Development of Multilingual Resources, In *Proc. of 17th COLING and 36th ACL*, pp.1352-1356.

Punjabi Machine Transliteration

M. G. Abbas Malik

Department of Linguistics
Denis Diderot, University of Paris 7
Paris, France
abbas.malik@gmail.com

Abstract

Machine Transliteration is to transcribe a word written in a script with approximate phonetic equivalence in another language. It is useful for machine translation, cross-lingual information retrieval, multilingual text and speech processing. Punjabi Machine Transliteration (PMT) is a special case of machine transliteration and is a process of converting a word from Shahmukhi (based on Arabic script) to Gurmukhi (derivation of Landa, Sharda and Takri, old scripts of Indian subcontinent), two scripts of Punjabi, irrespective of the type of word.

The Punjabi Machine Transliteration System uses transliteration rules (character mappings and dependency rules) for transliteration of Shahmukhi words into Gurmukhi. The PMT system can transliterate every word written in Shahmukhi.

1 Introduction

Punjabi is the mother tongue of more than 110 million people of Pakistan (66 million), India (44 million) and many millions in America, Canada and Europe. It has been written in two mutually incomprehensible scripts Shahmukhi and Gurmukhi for centuries. Punjabis from Pakistan are unable to comprehend Punjabi written in Gurmukhi and Punjabis from India are unable to comprehend Punjabi written in Shahmukhi. In contrast, they do not have any problem to understand the verbal expression of each other. Punjabi Machine Transliteration (PMT) system is an effort to bridge the written communication gap between the two scripts for the benefit of the millions of Punjabis around the globe.

Transliteration refers to phonetic translation across two languages with different writing systems (Knight & Graehl, 1998), such as Arabic to English (Nasreen & Leah, 2003). Most prior work has been done for Machine Translation (MT) (Knight & Leah, 97; Paola & Sanjeev, 2003; Knight & Stall, 1998) from English to other major languages of the world like Arabic, Chinese, etc. for cross-lingual information retrieval (Pirkola et al, 2003), for the development of multilingual resources (Yan et al, 2003; Kang & Kim, 2000) and for the development of cross-lingual applications.

PMT is a special kind of machine transliteration. It converts a Shahmukhi word into a Gurmukhi word irrespective of the type constraints of the word. It not only preserves the phonetics of the transliterated word but in contrast to usual transliteration, also preserves the meaning.

Two scripts are discussed and compared. Based on this comparison and analysis, character mappings between Shahmukhi and Gurmukhi are drawn and transliteration rules are discussed. Finally, architecture and process of the PMT system are discussed. When it is applied to Punjabi Unicode encoded text especially designed for testing, the results were compiled and analyzed. PMT system will provide basis for *Cross-Scriptural Information Retrieval* (CSIR) and *Cross-Scriptural Application Development* (CSAD).

2 Punjabi Machine Transliteration

According to Paola (2003), "When writing a foreign name in one's native language, one tries to preserve the way it sounds, i.e. one uses an orthographic representation which, when read aloud by the native speaker of the language, sounds as it would when spoken by a speaker of the foreign language – a process referred to as Transliteration". Usually, transliteration is referred to phonetic translation of a word of some

specific type (proper nouns, technical terms, etc) across languages with different writing systems. Native speakers may not understand the meaning of transliterated word.

PMT is a special type of Machine Transliteration in which a word is transliterated across two different writing systems used for the same language. It is independent of the type constraint of the word. It preserves both the phonetics as well as the meaning of transliterated word.

3 Scripts of Punjabi

3.1 Shahmukhi

Shahmukhi derives its character set from the Arabic alphabet. It is a right-to-left script and the shape assumed by a character in a word is context sensitive, i.e. the shape of a character is different depending whether the position of the character is at the beginning, in the middle or at the end of the word. Normally, it is written in Nastalique, a highly complex writing system that is cursive and context-sensitive. A sentence illustrating Shahmukhi is given below:

پنجابی میری ماں جوگی ماں بولی اے۔

It has 49 consonants, 16 diacritical marks and 16 vowels, etc. (Malik 2005)

3.2 Gurmukhi

Gurmukhi derives its character set from old scripts of the Indian Sub-continent i.e. Landa (script of North West), Sharda (script of Kashmir) and Takri (script of western Himalaya). It is a left-to-right syllabic script. A sentence illustrating Gurmukhi is given below:

ਪੰਜਾਬੀ ਮੇਰੀ ਮਾਣ ਜੇਗੀ ਮਾਂ ਬੋਲੀ ਏ.

It has 38 consonants, 10 vowels characters, 9 vowel symbols, 2 symbols for nasal sounds and 1 symbol that duplicates the sound of a consonant. (Bhatia 2003, Malik 2005)

4 Analysis and PMT Rules

Punjabi is written in two completely different scripts. One script is right-to-left and the other is left-to-right. One is Arabic based cursive and the other is syllabic. But both of them represent the phonetic repository of Punjabi. These phonetic sounds are used to determine the relation between the characters of two scripts. On the basis of this idea, character mappings are determined.

For the analysis and comparison, both scripts are subdivided into different group on the basis

of types of characters e.g. consonants, vowels, diacritical marks, etc.

4.1 Consonant Mapping

Consonants can be further subdivided into two groups:

Aspirated Consonants: There are sixteen aspirated consonants in Punjabi (Malik, 2005). Ten of these aspirated consonants ($\text{[b}^{\text{h}}]$, $\text{[p}^{\text{h}}]$, $\text{[t}^{\text{h}}]$, $\text{[t}^{\text{h}}]$, $\text{[d}^{\text{h}}]$, $\text{[t}^{\text{h}}]$, $\text{[d}^{\text{h}}]$, $\text{[d}^{\text{h}}]$, $\text{[k}^{\text{h}}]$, $\text{[g}^{\text{h}}]$) are very frequently used in Punjabi as compared to the remaining six aspirates ($\text{[r}^{\text{h}}]$, $\text{[t}^{\text{h}}]$, $\text{[l}^{\text{h}}]$, $\text{[m}^{\text{h}}]$, $\text{[n}^{\text{h}}]$, $\text{[v}^{\text{h}}]$). In Shahmukhi, aspirated consonants are represented by the combination of a consonant (to be aspirated) and HEH-DOACHASHMEE ([h]). For example $\text{[b]} + \text{[h]} = \text{[b}^{\text{h}}]$ and $\text{[d]} + \text{[h]} = \text{[d}^{\text{h}}]$.

In Gurmukhi, each frequently used aspirated-consonant is represented by a unique character. But, less frequent aspirated consonants are represented by the combination of a consonant (to be aspirated) and sub-joined PAIREEN HAAHAA e.g. $\text{[l]} + \text{[h]} = \text{[l}^{\text{h}}]$ and $\text{[v]} + \text{[h]} = \text{[v}^{\text{h}}]$, where [h] is the sub-joiner.

The sub-joiner character ([h]) tells that the following [h] is going to change the shape of PAIREEN HAAHHA.

The mapping of ten frequently used aspirated consonants is given in Table 1.

Sr.	Shahmukhi	Gurmukhi	Sr.	Shahmukhi	Gurmukhi
1	$\text{[b}^{\text{h}}]$	ਭ	6	$\text{[t}^{\text{h}}]$	ਛ
2	$\text{[p}^{\text{h}}]$	ਫ	7	$\text{[d}^{\text{h}}]$	ਧ
3	$\text{[t}^{\text{h}}]$	ਥ	8	$\text{[d}^{\text{h}}]$	ਢ
4	$\text{[t}^{\text{h}}]$	ਠ	9	$\text{[k}^{\text{h}}]$	ਖ
5	$\text{[d}^{\text{h}}]$	ਡ	10	$\text{[g}^{\text{h}}]$	ਘ

Table 1: Aspirated Consonants Mapping

The mapping for the remaining six aspirates is covered under non-aspirated consonants.

Non-Aspirated Consonants: In case of non-aspirated consonants, Shahmukhi has more consonants than Gurmukhi, which follows the one symbol for one sound principle. On the other hand there are more than one characters for a single sound in Shahmukhi. For example, Seh

(ث), Seen (س) and Sad (س) represent [s] and [s] has one equivalent in Gurmukhi i.e. Sassaa (ਸ). Similarly other characters like ਅ [a], ਤ [t], ਚ [h] and ਜ [z] have multiple equivalents in Shahmukhi. Non-aspirated consonants mapping is given in Table 2.

Sr.	Shahmukhi	Gurmukhi	Sr.	Shahmukhi	Gurmukhi
1	ب [b]	ਬ	21	ط [t]	ਤ
2	پ [p]	ਪ	22	ظ [z]	ਜ
3	ت [t]	ਤ	23	ع [ʔ]	ਅ
4	ث [t]	ਟ	24	غ [x]	ਗ
5	ث [s]	ਸ	25	ف [f]	ਫ
6	ج [d]	ਜ	26	ق [q]	ਕ
7	ح [h]	ਚ	27	ك [k]	ਕ
8	ح [h]	ਚ	28	گ [g]	ਗ
9	خ [x]	ਖ	29	ل [l]	ਲ
10	د [d]	ਦ	30	و [u]	ਲ
11	ذ [d]	ਡ	31	م [m]	ਮ
12	ز [z]	ਜ	32	ن [n]	ਨ
13	ر [r]	ਰ	33	ه [h]	ਣ
14	ر [r]	ੜ	35	و [u]	ਂ
15	ز [z]	ਜ	35	و [v]	ਵ
16	ز [z]	ਜ	36	ه [h]	ਹ
17	س [s]	ਸ	37	ه [h]	੍ਹ
18	ش [ʃ]	ਸ਼	38	ي [j]	ਯ
19	س [s]	ਸ	39	ي [j]	ਯ
20	ز [z]	ਜ			

Table 2: Non-Aspirated Consonants Mapping

4.2 Vowel Mapping

Punjabi contains ten vowels. In Shahmukhi, these vowels are represented with help of four long vowels (Alef Madda (ا), Alef (ا), Vav (و) and Choti Yeh (ي)) and three short vowels (Arabic Fatha – Zabar (َ), Arabic Damma – Pesh (ُ) and Arabic Kasra – Zer (ِ)). Note that the last two long vowels are also used as consonants.

Hamza (ء) is a special character and always comes between two vowel sounds as a place holder. For example, in آسائش [asaiʃ] (comfort), Hamza (ء) is separating two vowel sounds Alef (ا) and Zer (ِ), in آو [ao] (come), Hamza (ء) is separating two vowel sounds Alef Madda (ا) [a] and Vav (و) [o], etc. In the first example آسائش [asaiʃ] (comfort), Hamza (ء) is separating two vowel sounds Alef (ا) and Zer (ِ), but normally Zer (ِ) is dropped by common people. So Hamza (ء) is mapped on ਏ [ɪ] when it is followed by a consonant.

In Gurmukhi, vowels are represented by ten independent vowel characters (ਅ, ਆ, ਇ, ਈ, ਉ, ਊ, ਏ, ਐ, ਓ, ਔ) and nine dependent vowel signs (ਾ, ਿ, ੀ, ੁ, ੂ, ੇ, ੈ, ੌ, ੐). When a vowel sound comes at the start of a word or is independent of some consonant in the middle or end of a word, independent vowels are used; otherwise dependent vowel signs are used. The analysis of vowels is shown in Table 4 and the vowel mapping is given in Table 3.

Sr.	Shahmukhi	Gurmukhi	Sr.	Shahmukhi	Gurmukhi
1	ا [ə]	ਅ	11	ا [ə]	ਅ, ਾ
2	ا [a]	ਆ	12	ِ [ɪ]	ਿ
3	ا [ɪ]	ਇ	13	ي [i]	ੀ
4	ا [i]	ਈ	14	ِ [u]	ੁ
5	ا [u]	ਉ	15	و [u]	ੂ
6	ا [u]	ਊ	16	َ [e]	ੇ
7	َ [e]	ਏ	17	َ [æ]	ੈ
8	َ [æ]	ਐ	18	و [o]	ੋ
9	و [o]	ਓ	19	ِ [ɔ]	ੌ
10	ِ [ɔ]	ਔ	20	ء [ɪ]	ਇ

Table 3: Vowels Mapping

Vowel	Shahmukhi	Gurmukhi	Example
ɑ	Represented by Alef Madda (ا) in the beginning of a word and by Alef (ا) in the middle or at the end of a word.	Represented by ਆ and ਾ	ادى → ਆਦਮੀ [admi] (man) جاوئا → ਜਾਵਣਾ [ḡavṇa] (go)
ə	Represented by Alef (ا) in the beginning of a word and with Zabar (◌ِ) elsewhere.	Represented by ਅ in the beginning.	اڄ → ਅੱਜ [adḡḡ] (today)
e	Represented by the combinations of Alef (ا) and Choti Yeh (ي) in the beginning; a consonant and Choti Yeh (ي) in the middle and a consonant and Baree Yeh (ـ) at the end of a word.	Represented by ऐ and े	هه → ਏਧਰ [edḡar] (here), مير → ਮੇਰਾ [mera] (mine), سار → ਸਾਰੇ [sare] (all)
æ	Represented by the combination of Alef (ا), Zabar (◌ِ) and Choti Yeh (ي) in the beginning; a consonant, Zabar (◌ِ) and Choti Yeh (ي) in the middle and a consonant, Zabar (◌ِ) and Baree Yeh (ـ) at the end of a word.	Represented by ਐ and ੈ	هه → ਐਹ [æh] (this), میل → ਮੈਲ [mæ] (dirt), هه → ਹੈ [hæ] (is)
ɪ	Represented by the combination of Alef (ا) and Zer (◌َ) in the beginning and a consonant and Zer (◌َ) in the middle of a word. It never appears at the end of a word.	Represented by ਇ and ਿ	اكو → ਇੱਕੋ [ɪkko] (one), بارش → ਬਾਰਿਸ਼ [barish] (rain)
i	Represented by the combination of Alef (ا), Zer (◌َ) and Choti Yeh (ي) in the beginning; a consonant, Zer (◌َ) and Choti Yeh (ي) in the middle and a consonant and Choti Yeh (ي) at the end of a word	Represented by ਈ and ੀ	اير → ਈਤਰ [it̪ar] (mean) اميرى → ਅਮੀਰੀ [amiri] (richness), پنجابی → ਪੰਜਾਬੀ [pandḡabi] (Punjabi)
u	Represented by the combination of Alef (ا) and Pesh (◌ُ) in the beginning; a consonant and Pesh (◌ُ) in the middle of a word. It never appears at the end of a word.	Represented by ਉ and ੂ	اودھر → ਉੱਧਰ [udḡdhr] (there) مूल → ਮੁੱਲ [mull] (price)
u	Represented by the combination of Alef (ا), Pesh (◌ُ) and Vav (و) in the beginning, a consonant, Pesh (◌ُ) and Vav (و) in the middle and at the end of a word.	Represented by ਊ and ੂ	اوردو → ਉਰਦੂ [urdu] صورت → ਸੂਰਤ [surt] (face)
o	Represented by the combination of Alef (ا) and Vav (و) in the beginning; a consonant and Vav (و) in the middle and at the end of a word.	Represented by ਓ and ੋ	اوچھاڑ → ਓਛਾੜ [otḡhar] (cover), پڑھولا → ਪੜ੍ਹੋਲਾ [prhola] (a big pot in which wheat is stored)
ɔ	Represented by the combination of Alef (ا), Zabar (◌ِ) and Vav (و) in the beginning; a consonant, Zabar (◌ِ) and Vav (و) in the middle and at the end of a word.	Represented by ਔ and ੈ	اوڑا → ਔੜਾ [ɔɽa] (hindrance), موت → ਮੌਤ [mɔt] (death)
Note: Where → means ‘its equivalent in Gurmukhi is’.			

Table 4: Vowels Analysis of Punjabi for PMT

4.3 Sub-Joins (PAIREEN) of Gurmukhi

There are three PAIREEN (sub-joins) in Gurmukhi, “Haahaa”, “Vaavaa” and “Raaraa” shown in Table 5. For PMT, if HEH-DOACHASHMEE (ੜ) does come after the less frequently used aspirated consonants then it is transliterated into PAIREEN Haahaa. Other PAIREENS are very rare in their usage and are used only in Sanskrit loan words. In present day writings, PAIREEN Vaavaa and Raaraa are being replaced by normal Vaavaa (ਵ) and Raaraa (ਰ) respectively.

Sr.	PAIREEN	Shahmukhi	Gurmukhi	English
1	ੜ	لپس	ਲਿੱਪ	Lips
2	ੜ	چندرا	ਚੰਦ੍ਰਮਾ	Moon
3	ੜ	سؤیماں	ਸ਼੍ਰੋਮਾਨ	Self-respect

Table 5: Sub-joins (PAIREEN) of Gurmukhi

4.4 Diacritical Marks

Both in Shahmukhi and Gurmukhi, diacritical marks (dependent vowel signs in Gurmukhi) are the back bone of the vowel system and are very important for the correct pronunciation and understanding the meaning of a word. There are sixteen diacritical marks in Shahmukhi and nine dependent vowel signs in Gurmukhi (Malik, 2005). The mapping of diacritical marks is given in Table 6.

Sr.	Shahmukhi	Gurmukhi	Sr.	Shahmukhi	Gurmukhi
1	◌[ə]	---	9	◌[ɪ]	ਿਨ
2	◌[ɪ]	ਿ	10	◌[ɪ]	ੌ
3	◌[u]	ੁ	11	◌[ɪ]	---
4	◌[ɪ]	---	12	◌[ɪ]	---
5	◌[ə]	ਨ	13	◌[ɪ]	---
6	◌[u]	ਨੁ	14	◌[ɪ]	---
7	◌[ɪ]	---	15	◌[ɪ]	---
8	◌[ɪ]	---	16	◌[a]	ੜ

Table 6: Diacritical Mapping

Diacritical marks in Shahmukhi are very important for the correct pronunciation and understanding the meaning of a word. But they are sparingly used in writing by common people. In the normal text of Shahmukhi books, newspapers, and magazines etc. one will not find the diacritical marks. The pronunciation of a word and its meaning would be comprehended with the help of the context in which it is used.

For example,

ایکہ سڑک بڑی چوڑی ہے۔
میری چوڑی لال ہے۔

In the first sentence, the word چوڑی is pronounced as [tʃʊɾi] and it conveys the meaning of ‘wide’. In the second sentence, the word چوڑی is pronounced as [tʃuɾi] and it conveys the meaning of ‘bangle’. There should be Zabar (◌) after Cheh (چ) and Pesh (◌) after Cheh (چ) in the first and second words respectively, to remove the ambiguities.

It is clear from the above example that diacritical marks are essential for removing ambiguities, natural language processing and speech synthesis.

4.5 Other Symbols

Punctuation marks in Gurmukhi are the same as in English, except the full stop. DANDA (।) and double DANDA (॥) of Devanagari script are used for the full stop instead. In case of Shahmukhi, these are same as in Arabic. The mapping of digits and punctuation marks is given in Table 7.

Sr.	Shahmukhi	Gurmukhi	Sr.	Shahmukhi	Gurmukhi
1	•	◌	8	◌	?
2	◌	◌	9	◌	◌
3	◌	◌	10	◌	◌
4	◌	◌	11	◌	◌
5	◌	◌	12	◌	◌
6	◌	◌	13	◌	◌
7	◌	◌	14	◌	◌

Table 7: Other Symbols Mapping

4.6 Dependency Rules

Character mappings alone are not sufficient for PMT. They require certain dependency or contextual rules for producing correct transliteration. The basic idea behind these rules is the same as that of the character mappings. These rules include rules for aspirated consonants, non-aspirated consonants, Alef (ا), Alef Madda (آ), Vav (و), Choti Yeh (ي) etc. Only some of these rules are discussed here due to space limitations.

Rules for Consonants: Shahmukhi consonants are transliterated into their equivalent

This rule is applicable to all consonants in Table 1 and 2 except Ain (ع), Noon (ن), Noonghunna (و), Vav (و), Heh Gol (ه), Dochashmee Heh (ح), Choti Yeh (ي) and Baree Yeh (ا). These characters are treated separately.

- If Hamza (ء) is followed by Choti Yeh (ي), then Hamza (ء) and Choti Yeh (ي) will be transliterated into **ਈ** [i].

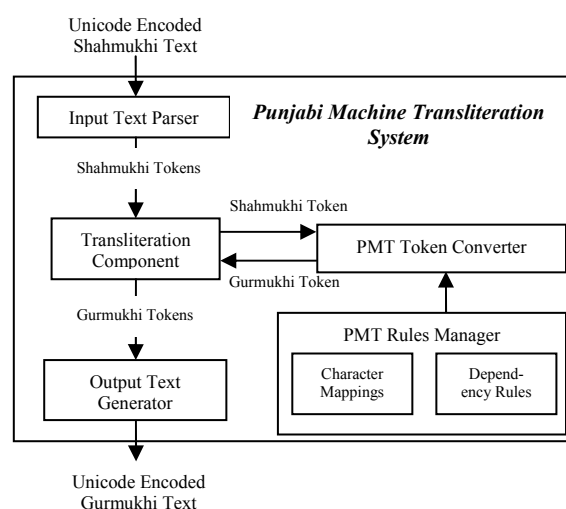
- If Hamza (ء) is followed by Zer (◌◌), then Hamza (ء) and Zer (◌◌) will be transliterated into **ئ** [I].

In all other cases, Hamza (ء) will be transliterated into **ʾ** [I].

5.1 System Architecture

Unicode encoded Shahmukhi text input is received by the Input Text Parser that parses it into Shahmukhi words by using simple

parsing techniques. These words are called Shahmukhi Tokens. Then these tokens are given to the Transliteration Component. This component gives each token to the PMT Token Converter that converts a Shahmukhi Token into a Gurmukhi Token by using the PMT Rules Manager, which consists of character mappings and dependency rules. The PMT Token Converter then gives the Gurmukhi Token back to the Transliteration Component. When all Shahmukhi Tokens are converted into Gurmukhi Tokens, then all Gurmukhi Tokens are passed to the Output Text Generator that generates the output Unicode encoded Gurmukhi text. The main PMT process is done by the PMT Token Converter and the PMT Rules Manager.



PMT system is a rule based transliteration system and is very robust. It is fast and accurate in its working. It can be used in domains involving Information Communication Technology (web, WAP, instant messaging, etc.).

5.2 PMT Process

The PMT Process is implemented in the PMT Token Converter and the PMT Rules Manager. For PMT, each Shahmukhi Token is parsed into its constituent characters and the character dependencies are determined on the basis of the occurrence and the contextual placement of the character in the token. In each Shahmukhi Token, there are some characters that bear dependencies and some characters are independent of such contextual dependencies for transliteration. If the character under consideration bears a dependency, then it is resolved and transliterated with the help of dependency rules.

If the character under consideration does not bear a dependency, then its transliteration is achieved by character mapping. This is done through mapping a character of the Shahmukhi token to its equivalent Gurmukhi character with the help of character mapping tables 1, 2, 3, 6 and 7, whichever is applicable. In this way, a Shahmukhi Token is transliterated into its equivalent Gurmukhi Token.

Consider some input Shahmukhi text S . First it is parsed into Shahmukhi Tokens ($S_1, S_2 \dots S_N$). Suppose that $S_i = \text{“ﻮﻟﯩﺠﺎ”}$ [valejā] is the i^{th} Shahmukhi Token. S_i is parsed into characters Vav (ﻮ) [v], Alef (ﻮ) [a], Lam (ﻞ) [l], Choti Yeh (ﺞ) [j], Alef (ﻮ) [a] and Noon Ghunna (ﻦ) [ŋ]. Then PMT mappings and dependency rules are applied to transliterate the Shahmukhi Token into a Gurmukhi Token. The Gurmukhi Token $G_i = \text{“ਵਾਲਿਆਂ”}$ is generated from S_i . The step by step process is clearly shown in Table 8.

Sr.	Character(s) Parsed	Gurmukhi Token	Mapping or Rule Applied
1	ﻮ $\rightarrow$ ਵ [v]	ਵ	Mapping Table 4
2	ﻮ $\rightarrow$ ਾ [a]	ਵਾ	Rule for ALEF
3	ﻞ $\rightarrow$ ਲ [l]	ਵਾਲ	Mapping Table 4
4	ﺞ $\rightarrow$ ਿਆ [ia]	ਵਾਲਿਆ	Rule for YEH
5	ﻦ $\rightarrow$ ਂ [ŋ]	ਵਾਲਿਆਂ	Rule for NOONGHUNNA

Note: $\rightarrow$ is read as ‘is transliterated into’.

Table 8: Methodology of PMTS

In this way, all Shahmukhi Tokens are transliterated into Gurmukhi Tokens ($G_1, G_2 \dots G_n$). From these Gurmukhi Tokens, Gurmukhi text G is generated.

The important point to be noted here is that input Shahmukhi text must contain all necessary diacritical marks, which are necessary for the correct pronunciation and understanding the meaning of the transliterated word.

6 Evaluation Experiments

6.1 Input Selection

The first task for evaluation of the PMT system is the selection of input texts. To consider the historical aspects, two manuscripts, poetry by *Maqbal* (Maqbal) and *Heer by Waris Shah* (Waris, 1766) were selected. Geographically

Punjab is divided into four parts eastern Punjab (Indian Punjab), central Punjab, southern Punjab and northern Punjab. All these geographical regions represent the major dialects of Punjabi. Hayms of Baba Nanak (eastern Punjab), Heer by Waris Shah (central Punjab), Hayms by Khawaja Farid (southern Punjab) and Saif-ul-Malooq by Mian Muhammad Bakhsh (northern Punjab) were selected for the evaluation of PMT system. All the above selected texts are categorized as classical literature of Punjabi. In modern literature, poetry and short stories of different poets and writers were selected from some issues of *Puncham* (monthly Punjabi magazine since 1985) and other published books. All of these selected texts were then compiled into Unicode encoded text as none of them were available in this form before.

The main task after the compilation of all the selected texts into Unicode encoded texts is to put all necessary diacritical marks in the text. This is done with help of dictionaries. The accuracy of the PMT system depends upon the necessary diacritical marks. Absence of the necessary diacritical marks affects the accuracy greatly.

6.2 Results

After the compilation of selected input texts, they are transliterated into Gurmukhi texts by using the PMT system. Then the transliterated Gurmukhi texts are tested for errors and accuracy. Testing is done manually with help of dictionaries of Shahmukhi and Gurmukhi by persons who know both scripts. The results are given in Table 9.

Source	Total Words	Accuracy
Manuscripts	1,007	98.21
Baba Nanak	3,918	98.47
Khawaja Farid	2,289	98.25
Waris Shah	14,225	98.95
Mian Muhammad Bakhsh	7,245	98.52
Modern literature	16,736	99.39
Total	45,420	98.95

Table 9: Results of PMT System

If we look at the results, it is clear that the PMT system gives more than 98% accuracy on classical literature and more than 99% accuracy on the modern literature. So PMT system fulfills the requirement of transliteration across two scripts of Punjabi. The only constraint to achieve this accuracy is that input text must contain all necessary diacritical marks for removing ambiguities.

7 Conclusion

Shahmukhi and Gurmukhi being the only two prevailing scripts for Punjabi expressions encompass a population of almost 110 million around the globe. PMT is an endeavor to bridge the ethnical, cultural and geographical divisions between the Punjabi speaking communities. By implementing this system of transliteration, new horizons for thought, idea and belief will be shared and the world will gain an impetus on the efforts harmonizing relationships between nations. The large repository of historical, literary and religious work done by generations will now be available for easy transformation and critique for all. The research has future milestone enabling PMT system for back machine transliteration from Gurmukhi to Shahmukhi.

Reference

- Ari Pirkola, Jarmo Toivonen, Heikki Keskustalo, Kari Visala, and Kalervo Järvelin. 2003. *Fuzzy Translation of Cross-Lingual Spelling Variants*. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. pp: 345 – 352
- Baba Guru Nanak, arranged by Muhammad Asif Khan. 1998. *آکھیا بیا نانک نے* (*Sayings of Baba Nanak* in Punjabi Shahmukhi). Pakistan Punjabi Adbi Board, Lahore
- Bhatia, Tej K. 2003. *The Gurmukhi Script and Other Writing Systems of Punjab: History, Structure and Identity*. International Symposium on Indic Script: Past and future organized by Research Institute for the Languages and Cultures of Asia and Africa and Tokyo University of Foreign Studies, December 17 – 19. pp: 181 – 213
- In-Ho Kang and GilChang Kim. 2000. *English-to-Korean transliteration using multiple unbounded overlapping phoneme chunks*. In Proceedings of the 17th conference on Computational Linguistics. 1: 418 – 424
- Khawaja Farid (arranged by Muhammad Asif Khan). *آکھیا خواجہ فرید نے* (*Sayings of Khawaja Farid* in Punjabi Shahmukhi). Pakistan Punjabi Adbi Board, Lahore
- Knight, K. and Stalls, B. G. 1998. *Translating Names and Technical Terms in Arabic Tex*. Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages
- Knight, Kevin and Graehl, Jonathan. 1997. *Machine Transliteration*. In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics. pp. 128-135
- Knight, Kevin; Morgan Kaufmann and Graehl, Jonathan. 1998. *Machine Transliteration*. In Computational Linguistics. 24(4): 599 – 612
- Malik, M. G. Abbas. 2005. *Towards Unicode Compatible Punjabi Character Set*. In proceedings of 27th Internationalization and Unicode Conference, 6 – 8 April, Berlin, Germany
- Maqbal. *مدح غوث*. Punjabi Manuscript in Oriental Section, Main Library University of the Punjab, Quaid-e-Azam Campus, Lahore Pakistan; 7 pages; Access # 8773
- Mian Muhammad Bakhsh (Edited by Fareer Muhammad Faqeer). 2000. *Saif-ul-Malooq*. Al-Faisal Pub. Urdu Bazar, Lahore
- Nasreen AbdulJaleel, Leah S. Larkey. 2003. *Statistical transliteration for English-Arabic cross language information retrieval*. In Proceedings of the 12th international conference on information and knowledge management. pp: 139 – 146
- Paola Virga and Sanjeev Khudanpur. 2003. *Transliteration of proper names in cross-language applications*. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. pp: 365 – 366
- Rahman Tariq. 2004. *Language Policy and Localization in Pakistan: Proposal for a Paradigmatic Shift*. Crossing the Digital Divide, SCALLA Conference on Computational Linguistics, 5 – 7 January 2004
- Sung Young Jung, SungLim Hong and Eunok Peak. 2000. *An English to Korean transliteration model of extended markov window*. In Proceedings of the 17th conference on Computational Linguistics. 1:383 – 389
- Tanveer Bukhari. 2000. *لغت پنجابی اردو*. Urdu Science Board, 299 Uper Mall, Lahore
- Waris Shah. 1766. *ہمیر رانجھا*. Punjabi Manuscript in Oriental Section, Main Library University of the Punjab, Quaid-e-Azam Campus, Lahore Pakistan; 48 pages; Access # [Ui VI 135/]1443
- Waris Shah (arranged by Naseem Ijaz). 1977. *ہمیر رانجھا*. Lehran, Punjabi Journal, Lahore
- Yan Qu, Gregory Grefenstette, David A. Evans. 2003. *Automatic transliteration for Japanese-to-English text retrieval*. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. pp: 353 – 360

Multilingual Document Clustering: an Heuristic Approach Based on Cognate Named Entities

Soto Montalvo

GAVAB Group

URJC

soto.montalvo@urjc.es

Raquel Martínez

NLP&IR Group

UNED

raquel@lsi.uned.es

Arantza Casillas

Dpt. EE

UPV-EHU

arantza.casillas@ehu.es

Víctor Fresno

GAVAB Group

URJC

victor.fresno@urjc.es

Abstract

This paper presents an approach for Multilingual Document Clustering in comparable corpora. The algorithm is of heuristic nature and it uses as unique evidence for clustering the identification of cognate named entities between both sides of the comparable corpora. One of the main advantages of this approach is that it does not depend on bilingual or multilingual resources. However, it depends on the possibility of identifying cognate named entities between the languages used in the corpus. An additional advantage of the approach is that it does not need any information about the right number of clusters; the algorithm calculates it. We have tested this approach with a comparable corpus of news written in English and Spanish. In addition, we have compared the results with a system which translates selected document features. The obtained results are encouraging.

1 Introduction

Multilingual Document Clustering (MDC) involves dividing a set of n documents, written in different languages, into a specified number k of clusters, so the documents that are similar to other documents are in the same cluster. Meanwhile a multilingual cluster is composed of documents written in different languages, a monolingual cluster is composed of documents written in one language.

MDC has many applications. The increasing amount of documents written in different languages that are available electronically, leads to

develop applications to manage that amount of information for filtering, retrieving and grouping multilingual documents. MDC tools can make easier tasks such as Cross-Lingual Information Retrieval, the training of parameters in statistics based machine translation, or the alignment of parallel and non parallel corpora, among others.

MDC systems have developed different solutions to group related documents. The strategies employed can be classified in two main groups: the ones which use translation technologies, and the ones that transform the document into a language-independent representation.

One of the crucial issues regarding the methods based on document or features translation is the correctness of the proper translation. Bilingual resources usually suggest more than one sense for a source word and it is not a trivial task to select the appropriate one. Although word-sense disambiguation methods can be applied, these are not free of errors. On the other hand, methods based on language-independent representation also have limitations. For instance, those based on thesaurus depend on the thesaurus scope. Numbers or dates identification can be appropriate for some types of clustering and documents; however, for other types of documents or clustering it could not be so relevant and even it could be a source of noise.

In this work we dealt with MDC and we proposed an approach based only on cognate Named Entities (NE) identification. We have tested this approach with a comparable corpus of news written in English and Spanish, obtaining encouraging results. One of the main advantages of this approach is that it does not depend on multilingual resources such as dictionaries, machine translation systems, thesaurus or gazetteers. In addition, no information about the right number of clusters has

to be provided to the algorithm. It only depends on the possibility of identifying cognate named entities between the languages involved in the corpus. It could be particularly appropriate for news corpus, where named entities play an important role.

In order to compare the results of our approach with other based on features translation, we also dealt with this one, as baseline approach. The system uses EuroWordNet (Vossen, 1998) to translate the features. We tried different features categories and combinations of them in order to determine which ones lead to improve MDC results in this approach.

In the following section we relate previous work in the field. In Section 3 we present our approach for MDC. Section 4 describes the system we compare our approach with, as well as the experiments and the results. Finally, Section 5 summarizes the conclusions and the future work.

2 Related Work

MDC is normally applied with parallel (Silva et al., 2004) or comparable corpus (Chen and Lin, 2000), (Rauber et al., 2001), (Lawrence, 2003), (Steinberger et al., 2002), (Mathieu et al., 2004), (Pouliquen et al., 2004). In the case of the comparable corpora, the documents usually are news articles.

Considering the approaches based on translation technology, two different strategies are employed: (1) translate the whole document to an anchor language, and (2) translate some features of the document to an anchor language.

With regard to the first approach, some authors use machine translation systems, whereas others translate the document word by word consulting a bilingual dictionary. In (Lawrence, 2003), the author presents several experiments for clustering a Russian-English multilingual corpus; several of these experiments are based on using a machine translation system. Columbia's Newsblaster system (Kirk et al., 2004) clusters news into events, it categorizes events into broad topic and summarizes multiple articles on each event. In the clustering process non-English documents are translated using simple dictionary lookup techniques for translating Japanese and Russian documents, and the Systran translation system for the other languages used in the system.

When the solution involves translating only some features, first it is necessary to select these

features (usually entities, verbs, nouns) and then translate them with a bilingual dictionary or/and consulting a parallel corpus.

In (Mathieu et al., 2004) before the clustering process, the authors perform a linguistic analysis which extracts lemmas and recognizes named entities (location, organization, person, time expression, numeric expression, product or event); then, the documents are represented by a set of terms (*keywords* or named entity types). In addition, they use document frequency to select relevant features among the extracted terms. Finally, the solution uses bilingual dictionaries to translate the selected features. In (Rauber et al., 2001) the authors present a methodology in which documents are parsed to extract features: all the words which appear in n documents except the stop-words. Then, standard machine translation techniques are used to create a monolingual corpus. After the translation process the documents are automatically organized into separate clusters using an un-supervised neural network.

Some approaches first carry out an independent clustering in each language, that is a monolingual clustering, and then they find relations among the obtained clusters generating the multilingual clusters. Others solutions start with a multilingual clustering to look for relations between the documents of all the involved languages. This is the case of (Chen and Lin, 2000), where the authors propose an architecture of multilingual news summarizer which includes monolingual and multilingual clustering; the multilingual clustering takes input from the monolingual clusters. The authors select different type of features depending on the clustering: for the monolingual clustering they use only named entities, for the multilingual clustering they extract verbs besides named entities.

The strategies that use language-independent representation try to normalize or standardize the document contents in a language-neutral way; for example: (1) by mapping text contents to an independent knowledge representation, or (2) by recognizing language independent text features inside the documents. Both approaches can be employed isolated or combined.

The first approach involves the use of existing multilingual linguistic resources, such as thesaurus, to create a text representation consisting of a set of thesaurus items. Normally, in a multilingual thesaurus, elements in different languages are

related via language-independent items. So, two documents written in different languages can be considered similar if they have similar representation according to the thesaurus. In some cases, it is necessary to use the thesaurus in combination with a machine learning method for mapping correctly documents onto thesaurus. In (Steinberger et. al., 2002) the authors present an approach to calculate the semantic similarity by representing the document contents in a language independent way, using the descriptor terms of the multilingual thesaurus *Eurovoc*.

The second approach, recognition of language independent text features, involves the recognition of elements such as: dates, numbers, and named entities. In others works, for instance (Silva et. al., 2004), the authors present a method based on Relevant Expressions (RE). The RE are multilingual lexical units of any length automatically extracted from the documents using the LiPXtractor extractor, a language independent statistics-based tool. The RE are used as base features to obtain a reduced set of new features for the multilingual clustering, but the clusters obtained are monolingual.

Others works combine recognition of independent text features (numbers, dates, names, cognates) with mapping text contents to a thesaurus. In (Pouliquen et. al., 2004) the cross-lingual news cluster similarity is based on a linear combination of three types of input: (a) cognates, (b) automatically detected references of geographical place names, and (c) the results of a mapping process onto a multilingual classification system which maps documents onto the multilingual thesaurus *Eurovoc*. In (Steinberger et. al., 2004) it is proposed to extract language-independent text features using gazetteers and regular expressions besides thesaurus and classification systems.

None of the revised works use as unique evidence for multilingual clustering the identification of cognate named entities between both sides of the comparable corpora.

3 MDC by Cognate NE Identification

We propose an approach for MDC based only on cognate NE identification. The NEs categories that we take into account are: PERSON, ORGANIZATION, LOCATION, and MISCELLANY. Other numerical categories such as DATE,

TIME or NUMBER are not considered because we think they are less relevant regarding the content of the document. In addition, they can lead to group documents with few content in common.

The process has two main phases: (1) cognate NE identification and (2) clustering. Both phases are described in detail in the following sections.

3.1 Cognate NE identification

This phase consists of three steps:

1. Detection and classification of the NEs in each side of the corpus.
2. Identification of cognates between the NEs of both sides of the comparable corpus.
3. To work out a statistic of the number of documents that share cognates of the different NE categories.

Regarding the first step, it is carried out in each side of the corpus separately. In our case we used a corpus with morphosyntactical annotations and the NEs identified and classified with the FreeLing tool (Carreras et al., 2004).

In order to identify the cognates between NEs 4 steps are carried out:

- Obtaining two list of NEs, one for each language.
- Identification of entity mentions in each language. For instance, “Ernesto Zedillo”, “Zedillo”, “Sr. Zedillo” will be considered as the same entity after this step since they refer to the same person. This step is only applied to entities of PERSON category. The identification of NE mentions, as well as cognate NE, is based on the use of the Levenshtein edit-distance function (LD). This measure is obtained by finding the cheapest way to transform one string into another. Transformations are the one-step operations of insertion, deletion and substitution. The result is an integer value that is normalized by the length of the longest string. In addition, constraints regarding the number of words that the NEs are made up, as well as the order of the words are applied.
- Identification of cognates between the NEs of both sides of the comparable corpus. It is also based on the LD. In addition, also

constraints regarding the number and the order of the words are applied. First, we tried cognate identification only between NEs of the same category (PERSON with PERSON, ...) or between any category and MISCELLANY (PERSON with MISCELLANY, ...). Next, with the rest of NEs that have not been considered as cognate, a next step is applied without the constraint of being to the same category or MISCELLANY. As result of this step a list of corresponding bilingual cognates is obtained.

- The same procedure carried out for obtaining bilingual cognates is used to obtain two more lists of cognates, one per language, between the NEs of the same language.

Finally, a statistic of the number of documents that share cognates of the different NE categories is worked out. This information can be used by the algorithm (or the user) to select the NE category used as constraint in the clustering steps 1(a) and 2(b).

3.2 Clustering

The algorithm for clustering multilingual documents based on cognate NEs is of heuristic nature. It consists of 3 main phases: (1) first clusters creation, (2) addition of remaining documents to existing clusters, and (3) final cluster adjustment.

1. First clusters creation. This phase consists of 2 steps.
 - (a) First, documents in different languages that have more cognates in common than a threshold are grouped into the same cluster. In addition, at least one of the cognates has to be of a specific category (PERSON, LOCATION or ORGANIZATION), and the number of mentions has to be similar; a threshold determines the similarity degree. After this step some documents are assigned to clusters while the others are free (with no cluster assigned).
 - (b) Next, it is tried to assign each free document to an existing cluster. This is possible if there is a document in the cluster that has more cognates in common with the free document than a threshold, with

no constraints regarding the NE category. If it is not possible, a new cluster is created. This step can also have as result free documents.

At this point the number of clusters created is fixed for the next phase.

2. Addition of the rest of the documents to existing clusters. This phase is carried out in 2 steps.
 - (a) A document is added to a cluster that contains a document which has more cognates in common than a threshold.
 - (b) Until now, the cognate NEs have been compared between both sides of the corpus, that is a bilingual comparison. In this step, the NEs of a language are compared with those of the same language. This can be described like a monolingual comparison step. The aim is to group similar documents of the same language if the bilingual comparison steps have not been successful. As in the other cases, a document is added to a cluster with at least a document of the same language which has more cognates in common than a threshold. In addition, at least one of the cognates have to be of a specific category (PERSON, LOCATION or ORGANIZATION).
3. Final cluster adjustment. Finally, if there are still free documents, each one is assigned to the cluster with more cognates in common, without constraints or threshold. Nonetheless, if free documents are left because they do not have any cognates in common with those assigned to the existing clusters, new clusters can be created.

Most of the thresholds can be customized in order to permit and make the experiments easier. In addition, the parameters customization allows the adaptation to different type of corpus or content. For example, in steps 1(a) and 2(b) we enforce at least on match in a specific NE category. This parameter can be customized in order to guide the grouping towards some type of NE. In Section 4.5 the exact values we used are described.

Our approach is an heuristic method that following an agglomerative approach and in an iterative way, decides the number of clusters and

locates each document in a cluster; everything is based in cognate NEs identification. The final number of clusters depends on the threshold values.

4 Evaluation

We wanted not only determine whether our approach was successful for MDC or not, but we also wanted to compare its results with other approach based on feature translation. That is why we try MDC by selecting and translating the features of the documents.

In this Section, first the MCD by feature translation is described; next, the corpus, the experiments and the results are presented.

4.1 MDC by Feature Translation

In this approach we emphasize the feature selection based on NEs identification and the grammatical category of the words. The selection of features we applied is based on previous work (Casillas et. al, 2004), in which several document representations are tested in order to study which of them lead to better monolingual clustering results. We used this MDC approach as baseline method.

The approach we implemented consists of the following steps:

1. Selection of features (NE, noun, verb, adjective, ...) and its context (the whole document or the first paragraph). Normally, the journalist style includes the heart of the news in the first paragraph; taking this into account we have experimented with the whole document and only with the first paragraph.
2. Translation of the features by using EuroWordNet 1.0. We translate English into Spanish. When more than one sense for a single word is provided, we disambiguate by selecting one sense if it appears in the Spanish corpus. Since we work with a comparable corpus, we expect that the correct translation of a word appears in it.
3. In order to generate the document representation we use the TF-IDF function to weight the features.
4. Use of an clustering algorithm. Particularly, we used a partitioning algorithm of the CLUTO (Karypis, 2002) library for clustering.

4.2 Corpus

A Comparable Corpus is a collection of similar texts in different languages or in different varieties of a language. In this work we compiled a collection of news written in Spanish and English belonging to the same period of time. The news are categorized and come from the news agency EFE compiled by HERMES project (<http://nlp.uned.es/hermes/index.html>). That collection can be considered like a comparable corpus. We have used three subset of that collection. The first subset, call $S1$, consists on 65 news, 32 in Spanish and 33 in English; we used it in order to train the threshold values. The second one, $S2$, is composed of 79 Spanish news and 70 English news, that is 149 news. The third subset, $S3$, contains 179 news: 93 in Spanish and 86 in English.

In order to test the MDC results we carried out a manual clustering with each subset. Three persons read every document and grouped them considering the content of each one. They judged independently and only the identical resultant clusters were selected. The human clustering solution is composed of 12 clusters for subset $S1$, 26 clusters for subset $S2$, and 33 clusters for $S3$. All the clusters are multilingual in the three subsets.

In the experimentation process of our approach the first subset, $S1$, was used to train the parameters and threshold values; with the second one and the third one the best parameters values were applied.

4.3 Evaluation metric

The quality of the experimentation results are determined by means of an external evaluation measure, the F-measure (van Rijsbergen, 1974). This measure compares the human solution with the system one. The F-measure combines the precision and recall measures:

$$F(i, j) = \frac{2 \times Recall(i, j) \times Precision(i, j)}{(Precision(i, j) + Recall(i, j))}, \quad (1)$$

where $Recall(i, j) = \frac{n_{ij}}{n_i}$, $Precision(i, j) = \frac{n_{ij}}{n_j}$, n_{ij} is the number of members of cluster human solution i in cluster j , n_j is the number of members of cluster j and n_i is the number of members of cluster human solution i . For all the clusters:

$$F = \sum_i \frac{n_i}{n} max\{F(i)\} \quad (2)$$

The closer to 1 the F-measure value the better.

4.4 Experiments and Results with MDC by Feature Translation

After trying with features of different grammatical categories and combinations of them, Table 1 and Table 2 only show the best results of the experiments.

The first column of both tables indicates the features used in clustering: NOM (nouns), VER (verbs), ADJ (adjectives), ALL (all the lemmas), NE (named entities), and 1^{rst} PAR (those of the first paragraph of the previous categories). The second column is the F-measure, and the third one indicates the number of multilingual clusters obtained. Note that the number of total clusters of each subset is provided to the clustering algorithm. As can be seen in the tables, the results depend on the features selected.

4.5 Experiments and Results with MDC by Cognate NE

The threshold for the LD in order to determine whether two NEs are cognate or not is 0.2, except for entities of ORGANIZATION and LOCATION categories which is 0.3 when they have more than one word.

Regarding the thresholds of the clustering phase (Section 3.2), after training the thresholds with the collection *S1* of 65 news articles we have concluded:

- The first step in the clustering phase, 1(a), performs a good first grouping with threshold relatively high; in this case 6 or 7. That is, documents in different languages that have more cognates in common than 6 or 7 are grouped into the same cluster. In addition, at least one of the cognates have to be of an specific category, and the difference between the number of mentions have to be equal or less than 2. Of course, these threshold are applied after checking that there are documents that meet the requirements. If they do not, thresholds are reduced. This first step creates multilingual clusters with high cohesiveness.
- Steps 1(b) and 2(a) lead to good results with small threshold values: 1 or 2. They are designed to give priority to the addition of documents to existing clusters. In fact, only step 1(b) can create new clusters.
- Step 2(b) tries to group similar documents of the same language when the bilingual com-

parison steps could not be able to deal with them. This step leads to good results with a threshold value similar to 1(a) step, and with the same NE category.

On the other hand, regarding the NE category enforce on match in steps 1(a) and 2(b), we tried with the two NE categories of cognates shared by the most number of documents. Particularly, with *S2* and *S3* corpus the NE categories of the cognates shared by the most number of documents was LOCATION followed by PERSON. We experimented with both categories.

Table 3 and Table 4 show the results of the application of the cognate NE approach to subsets *S2* and *S3* respectively. The first column of both tables indicates the thresholds for each step of the algorithm. Second and third columns show the results by selecting PERSON category as NE category to be shared by at least a cognate in steps 1(a) and 2(b); whereas fourth and fifth columns are calculated with LOCATION NE category. The results are quite similar but slightly better with LOCATION category, that is the cognate NE category shared by the most number of documents. Although none of the results got the exact number of clusters, it is remarkable that the resulting values are close to the right ones. In fact, no information about the right number of cluster is provided to the algorithm.

If we compare the performance of the two approaches (Table 3 with Table 1 and Table 4 with Table 2) our approach obtains better results. With the subset *S3* the results of the F-measure of both approaches are more similar than with the subset *S2*, but the F-measure values of our approach are still slightly better.

To sum up, our approach obtains slightly better results than the one based on feature translation with the same corpora. In addition, the number of multilingual clusters is closer to the reference solution. We think that it is remarkable that our approach reaches results that can be comparable with those obtained by means of features translation. We will have to test the algorithm with different corpora (with some monolingual clusters, different languages) in order to confirm its performance.

5 Conclusions and Future Work

We have presented a novel approach for Multilingual Document Clustering based only on cognate

Selected Features	F-measure	Multilin. Clus./Total
NOM, VER	0.8533	21/26
NOM, ADJ	0.8405	21/26
ALL	0.8209	21/26
NE	0.8117	19/26
NOM, VER, ADJ	0.7984	20/26
NOM, VER, ADJ, 1 ^{rst} PAR	0.7570	21/26
NOM, ADJ, 1 ^{rst} PAR	0.7515	22/26
ALL, 1 ^{rst} PAR	0.7473	19/26
NOM, VER, 1 ^{rst} PAR	0.7371	20/26

Table 1: MDC results with the feature translation approach and subset S_2

Selected Features	F-measure	Multilin. Clus. /Total
NOM, ADJ	0.8291	26/33
ALL	0.8126	27/33
NOM, VER	0.8028	26/33
NE	0.8015	23/33
NOM, VER, ADJ	0.7917	25/33
NOM, ADJ, 1 ^{rst} PAR	0.7520	28/33
NOM, VER, ADJ, 1 ^{rst} PAR	0.7484	26/33
ALL, 1 ^{rst} PAR	0.7288	26/33
NOM, VER, 1 ^{rst} PAR	0.7200	24/33

Table 2: MDC results with the feature translation approach and subset S_3

Thresholds				1(a), 2(b) match on PERSON		1(a), 2(b) match on LOCATION	
Steps				Results	Clusters	Results	Clusters
1(a)	1(b)	2(a)	2(b)	F-measure	Multil./Calc./Total	F-measure	Multil./Calc./Total
6	2	1	5	0.9097	24/24/26	0.9097	24/24/26
6	2	1	6	0.8961	24/24/26	0.8961	24/24/26
6	2	1	7	0.8955	24/24/26	0.8955	24/24/26
6	2	2	5	0.8861	24/24/26	0.8913	24/24/26
7	2	1	5	0.8859	24/24/26	0.8913	24/24/26
6	2	2	4	0.8785	24/24/26	0.8899	24/24/26
6	2	2	6	0.8773	24/24/26	0.8833	24/24/26
6	2	2	7	0.8773	24/24/26	0.8708	24/24/26

Table 3: MDC results with the cognate NE approach and S_2 subset

Thresholds				1(a), 2(b) match on PERSON		1(a), 2(b) match on LOCATION	
Steps				Results	Clusters	Results	Clusters
1(a)	1(b)	2(a)	2(b)	F-measure	Multil./Calc./Total	F-measure	Multil./Calc./Total
7	2	1	5	0.8587	30/30/33	0.8621	30/30/33
6	2	1	5	0.8552	30/30/33	0.8552	30/30/33
6	2	1	6	0.8482	30/30/33	0.8483	30/30/33
6	2	1	7	0.8471	30/30/33	0.8470	30/30/33
6	2	2	5	0.8354	30/30/33	0.8393	30/30/33
6	2	2	6	0.8353	30/30/33	0.8474	30/30/33
6	2	2	4	0.8323	30/30/33	0.8474	30/30/33
6	2	2	7	0.8213	30/30/33	0.8134	30/30/33

Table 4: MDC results with the cognate NE approach and S_3 subset

named entities identification. One of the main advantages of this approach is that it does not depend on multilingual resources such as dictionaries, machine translation systems, thesaurus or gazetteers. The only requirement to fulfill is that the languages involved in the corpus have to permit the possibility of identifying cognate named entities. Another advantage of the approach is that it does not need any information about the right number of clusters. In fact, the algorithm calculates it by using the threshold values of the algorithm.

We have tested this approach with a comparable corpus of news written in English and Spanish, obtaining encouraging results. We think that this approach could be particularly appropriate for news articles corpus, where named entities play an important role. Even more, when there is no previous evidence of the right number of clusters. In addition, we have compared our approach with other based on feature translation, resulting that our approach presents a slightly better performance.

Future work will include the compilation of more corpora, the incorporation of machine learning techniques in order to obtain the thresholds more appropriate for different type of corpus. In addition, we will study if changing the order of the bilingual and monolingual comparison steps the performance varies significantly for different type of corpus.

Acknowledgements

We wish to thank the anonymous reviewers for their helpful and instructive comments. This work has been partially supported by MCyT TIN2005-08943-C02-02.

References

Benoit Mathieu, Romanic Besancon and Christian Fluhr. 2004. "Multilingual document clusters discovery". RIAO'2004, p. 1-10.

Arantza Casillas, M. Teresa González de Lena and Raquel Martínez. 2004. "Sampling and Feature Selection in a Genetic Algorithm for Document Clustering". *Computational Linguistics and Intelligent Text Processing*, CICLing'04. Lecture Notes in Computer Science, Springer-Verlag, p. 601-612.

Hsin-Hsi Chen and Chuan-Jie Lin. 2000. "A Multilingual News Summarizer". *Proceedings of 18th International Conference on Computational Linguistics*, p. 159-165.

Xavier Carreras, I. Chao, Lluís Padró and M. Padró. 2004. "An Open-Source Suite of Language Analyzers". *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*. Lisbon, Portugal. <http://garraf.epsevg.upc.es/freeling/>.

Karypis G. 2002. "CLUTO: A Clustering Toolkit". *Technical Report: 02-017*. University of Minnesota, Department of Computer Science, Minneapolis, MN 55455.

David Kirk Evans, Judith L. Klavans and Kathleen McKeown. 2004. "Columbian Newsblaster: Multilingual News Summarization on the Web". *Proceedings of the Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics Annual Meeting*, HLT-NAACL'2004.

Lawrence J. Leftin. 2003. "Newsblaster Russian-English Clustering Performance Analysis". *Columbia computer science Technical Reports*.

Bruno Pouliquen, Ralf Steinberger, Camelia Ignat, Emilia Ksper and Irina Temikova. 2004. "Multilingual and cross-lingual news topic tracking". *Proceedings of the 20th International Conference on computational Linguistics*, p. 23-27.

Andreas Rauber, Michael Dittenbach and Dieter Merkl. 2001. "Towards Automatic Content-Based Organization of Multilingual Digital Libraries: An English, French, and German View of the Russian Information Agency Novosti News". *Third All-Russian Conference Digital Libraries: Advanced Methods and Technologies*, Digital Collections Petrozavodsk, RCDI'2001.

van Rijsbergen, C.J. 1974. "Foundations of evaluation". *Journal of Documentation*, 30 (1974), p. 365-373.

Joaquin Silva, J. Mexia, Carlos Coelho and Gabriel Lopes. 2004. "A Statistical Approach for Multilingual Document Clustering and Topic Extraction from Clusters". *Pliska Studia Mathematica Bulgarica*, v.16,p. 207-228.

Ralf Steinberger, Bruno Pouliquen, and Johan Scheer. 2002. "Cross-Lingual Document Similarity Calculation Using the Multilingual Thesaurus EU-ROVOC". *Computational Linguistics and Intelligent Text Processing*, CICLing'02. Lecture Notes in Computer Science, Springer-Verlag, p. 415-424.

Ralf Steinberger, Bruno Pouliquen, and Camelia Ignat. 2004. "Exploiting multilingual nomenclatures and language-independent text features as an interlingua for cross-lingual text analysis applications". *Slovenian Language Technology Conference. Information Society, SLTC* 2004.

Vossen, P. 1998. "Introduction to EuroWordNet". *Computers and the Humanities Special Issue on EuroWordNet*.

Time Period Identification of Events in Text

Taichi Noro[†] Takashi Inui^{††} Hiroya Takamura[‡] Manabu Okumura[‡]

[†]Interdisciplinary Graduate School of Science and Engineering

Tokyo Institute of Technology

4259 Nagatsuta-cho, Midori-ku, Yokohama, Kanagawa, Japan

^{††}Japan Society for the Promotion of Science

[‡]Precision and Intelligence Laboratory, Tokyo Institute of Technology

{norot, tinui}@lr.pi.titech.ac.jp, {takamura, oku}@pi.titech.ac.jp

Abstract

This study aims at identifying when an event written in text occurs. In particular, we classify a sentence for an event into four time-slots; morning, daytime, evening, and night. To realize our goal, we focus on expressions associated with time-slot (time-associated words). However, listing up all the time-associated words is impractical, because there are numerous time-associated expressions. We therefore use a semi-supervised learning method, the Naïve Bayes classifier backed up with the Expectation Maximization algorithm, in order to iteratively extract time-associated words while improving the classifier. We also propose to use Support Vector Machines to filter out noisy instances that indicates no specific time period. As a result of experiments, the proposed method achieved 0.864 of accuracy and outperformed other methods.

1 Introduction

In recent years, the spread of the internet has accelerated. The documents on the internet have increased their importance as targets of business marketing. Such circumstances have evoked many studies on information extraction from text especially on the internet, such as sentiment analysis and extraction of location information. In this paper, we focus on the extraction of temporal information. Many authors of documents on the web often write about events in their daily life. Identifying when the events occur provides us valuable information. For example, we can

use temporal information as a new axis in the information retrieval. From time-annotated text, companies can figure out when customers use their products. We can explore activities of users for marketing researches, such as “What do people eat in the morning?”, “What do people spend money for in daytime?”

Most of previous work on temporal processing of events in text dealt with only newswire text. In those researches, it is assumed that temporal expressions indicating the time-period of events are often explicitly written in text. Some examples of explicit temporal expressions are as follows: “on March 23”, “at 7 p.m.”.

However, other types of text including web diaries and blogs contain few explicit temporal expressions. Therefore one cannot acquire sufficient temporal information using existing methods. Although dealing with such text as web diaries and blogs is a hard problem, those types of text are excellent information sources due to their overwhelmingly huge amount.

In this paper, we propose a method for estimating occurrence time of events expressed in informal text. In particular, we classify sentences in text into one of four time-slots; morning, daytime, evening, and night. To realize our goal, we focus on expressions associated with time-slot (hereafter, called time-associated words), such as “commute (morning)”, “nap (daytime)” and “cocktail (night)”. Explicit temporal expressions have more certain information than the time-associated words. However, these expressions are rare in usual text. On the other hand, although the time-associated words provide us only indirect information for estimating occurrence time of events, these words frequently appear in usual text. Actually, Figure 2 (we will discuss the graph in Section 5.2, again) shows the number of sentences including explicit tem-

poral expressions and time-associated words respectively in text. The numbers are obtained from a corpus we used in this paper. We can figure out that there are much more time-associated words than explicit temporal expressions in blog text. In other words, we can deal with wide coverage of sentences in informal text by our method with time-associated words.

However, listing up all the time-associated words is impractical, because there are numerous time-associated expressions. Therefore, we use a semi-supervised method with a small amount of labeled data and a large amount of unlabeled data, because to prepare a large quantity of labeled data is costly, while unlabeled data is easy to obtain. Specifically, we adopt the Naïve Bayes classifier backed up with the Expectation Maximization (EM) algorithm (Dempster et al., 1977) for semi-supervised learning. In addition, we propose to use Support Vector Machines to filter out noisy sentences that degrade the performance of the semi-supervised method.

In our experiments using blog data, we obtained 0.864 of accuracy, and we have shown effectiveness of the proposed method.

This paper is organized as follows. In Section 2 we briefly describe related work. In Section 3 we describe the details of our corpus. The proposed method is presented in Section 4. In Section 5, we describe experimental results and discussions. We conclude the paper in Section 6.

2 Related Work

The task of time period identification is new and has not been explored much to date.

Setzer et al. (2001) and Mani et al. (2000) aimed at annotating newswire text for analyzing temporal information. However, these previous work are different from ours, because these work only dealt with newswire text including a lot of explicit temporal expressions.

Tsuchiya et al. (2005) pursued a similar goal as ours. They manually prepared a dictionary with temporal information. They use the hand-crafted dictionary and some inference rules to determine the time periods of events. In contrast, we do not resort to such a hand-crafted material, which requires much labor and cost. Our method automatically acquires temporal information from actual data of people's activities (blog). Henceforth, we can get temporal information associated with your daily life that would be not existed in a dictionary.

3 Corpus

In this section, we describe a corpus made from blog entries. The corpus is used for training and test data of machine learning methods mentioned in Section 4.

The blog entries we used are collected by the method of Nanno et al. (2004). All the entries are written in Japanese. All the entries are split into sentences automatically by some heuristic rules. In the next section, we are going to explain “time-slot” tag added at every sentence.

3.1 Time-Slot Tag

The “time-slot” tag represents when an event occurs in five classes; “morning”, “daytime”, “evening”, “night”, and “time-unknown”. “Time-unknown” means that there is no temporal information. We set the criteria of time-slot tags as follows.

Morning: 04:00--10:59

from early morning till before noon, breakfast

Daytime: 11:00--15:59

from noon till before dusk, lunch

Evening: 16:00--17:59

from dusk till before sunset

Night: 18:00--03:59

from sunset till dawn, dinner

Note that above criteria are just interpreted as rough standards. We think time-slot recognized by authors is more important. For example, in a case of “about 3 o'clock this morning” we judge the case as “morning” (not “night”) with the expression written by the author “this morning”.

To annotate sentences in text, we used two different clues. One is the explicit temporal expressions or time-associated words included in the sentence to be judged. The other is contextual information around the sentences to be judged. The examples corresponding to the former case are as follows:

Example 1

- a. I went to post office by bicycle in the morning.
- b. I had spaghetti at restaurant at noon.
- c. I cooked stew as dinner on that day.

Suppose that the two sentences in Example 2 appear successively in a document. In this case, we first judge the first sentence as morning. Next, we judge the second sentence as morning by contextual information (i.e., the preceding sentence is judged as morning), although we cannot know the time period just from the content of the second sentence itself.

Example 2

1. I went to X by bicycle in the morning.
2. I went to a shop on the way back from X.

3.2 Corpus Statistics

We manually annotated the corpus. The number of the blog entries is 7,413. The number of sentences is 70,775. Of 70,775, the number of sentences representing any events¹ is 14,220. The frequency distribution of time-slot tags is shown in Table 1. We can figure out that the number of time-unknown sentences is much larger than the other sentences from this table. This bias would affect our classification process. Therefore, we propose a method for tackling the problem.

morning	711
daytime	599
evening	207
night	1,035
time-unknown	11,668
Total	14,220

Table 1: The numbers of time-slot tags.

4 Proposed Method

4.1 Basic Idea

Suppose, for example, “breakfast” is a strong clue for the morning class, i.e. the word is a time-associated word of morning. Thereby we can classify the sentence “I have cereal for breakfast.” into the morning class. Then “cereal” will be a time-associated word of morning. Therefore we can use “cereal” as a clue of time-slot classification. By iterating this process, we can obtain a lot of time-associated words with bootstrapping method, improving sentence classification performance at the same time.

To realize the bootstrapping method, we use the EM algorithm. This algorithm has a theoretical base of likelihood maximization of incomplete data and can enhance supervised learning methods. We specifically adopted the combination of the Naïve Bayes classifier and the EM algorithm. This combination has been proven to be effective in the text classification (Nigam et al., 2000).

¹ The aim of this study is time-slot classification of events. Therefore we treat only sentences expressing an event.

4.2 Naïve Bayes Classifier

In this section, we describe multinomial model that is a kind of Naïve Bayes classifiers.

A generative probability of example x given a category c has the form:

$$P(x | c, \theta) = P(|x|) |x|! \prod_w \frac{P(w | c)^{N(w, x)}}{N(w, x)} \quad (1)$$

where $P(|x|)$ denotes the probability that a sentence of length $|x|$ occurs, $N(w, x)$ denotes the number of occurrences of w in text x . The occurrence of a sentence is modeled as a set of trials, in which a word is drawn from the whole vocabulary.

In time-slot classification, the x is correspond to each sentence, the c is correspond to one of time-slots in {morning, daytime, evening, night}. Features are words in the sentence. A detailed description of features will be described in Section 4.5.

4.3 Incorporation of Unlabeled Data with the EM Algorithm

The EM algorithm (Dempster et al., 1977) is a method to estimate a model that has the maximal likelihood of the data when some variables cannot be observed (these variables are called latent variables). Nigam et al. (2000) proposed a combination of the Naïve Bayes classifiers and the EM algorithm.

Ignoring the unrelated factors of Eq. (1), we obtain

$$P(x | c, \theta) \propto \prod_w P(w | c)^{N(w, x)}, \quad (2)$$

$$P(x | \theta) \propto \sum_c P(c) \prod_w P(w | c)^{N(w, x)}. \quad (3)$$

We express model parameters as θ .

If we regard c as a latent variable and introduce a Dirichlet distribution as the prior distribution for the parameters, the Q-function (i.e., the expected log-likelihood) of this model is defined as:

$$Q(\theta | \bar{\theta}) = \log(P(\theta)) + \sum_{x \in D} \sum_c P(x | c, \bar{\theta}) \times \log \left(P(c) \prod_w P(w | c)^{N(w, x)} \right), \quad (4)$$

where $P(\theta) \propto \prod_c (P(c)^{\alpha-1} \prod_w (P(w | c)^{\alpha-1}))$. α is a user given parameter and D is the set of examples used for model estimation.

We obtain the next EM equation from this Q-function:

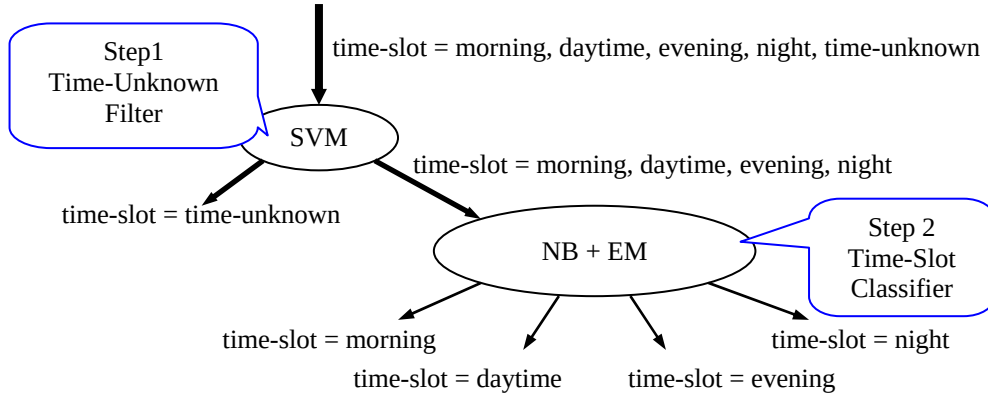


Figure 1: The flow of 2-step classification.

E-step:

$$P(c | x, \bar{\theta}) = \frac{P(c | \bar{\theta})P(x | c, \bar{\theta})}{\sum_c P(c | \bar{\theta})P(x | c, \bar{\theta})}, \quad (5)$$

M-step:

$$P(c) = \frac{(\alpha - 1) + \sum_{x \in D} P(c | x, \bar{\theta})}{(\alpha - 1)|C| + |D|}, \quad (6)$$

$$P(w | c) = \frac{(\alpha - 1) + \sum_{x \in D} P(c | x, \bar{\theta})N(w, x)}{(\alpha - 1)|W| + \sum_w \sum_{x \in D} P(c | x, \bar{\theta})N(w, x)}, \quad (7)$$

where $|C|$ denotes the number of categories, $|W|$ denotes the number of features variety. For labeled example x , Eq. (5) is not used. Instead, $P(c | x, \bar{\theta})$ is set as 1.0 if c is the category of x , otherwise 0.

Instead of the usual EM algorithm, we use the tempered EM algorithm (Hofmann, 2001). This algorithm allows coordinating complexity of the model. We can realize this algorithm by substituting the next equation for Eq. (5) at E-step:

$$P(c | x, \bar{\theta}) = \frac{\{P(c | \bar{\theta})P(x | c, \bar{\theta})\}^\beta}{\sum_c \{P(c | \bar{\theta})P(x | c, \bar{\theta})\}^\beta}, \quad (8)$$

where β denotes a hyper parameter for coordinating complexity of the model, and it is positive value. By decreasing this hyper-parameter β , we can reduce the influence of intermediate classification results if those results are unreliable.

Too much influence by unlabeled data sometimes deteriorates the model estimation. Therefore, we introduce a new hyper-parameter $\lambda (0 \leq \lambda \leq 1)$ which acts as weight on unlabeled data. We exchange the second term in the right-hand-side of Eq. (4) for the next equation:

$$\sum_{x \in D^l} \sum_c P(c | x, \bar{\theta}) \log \left(P(c) \prod_w P(w | c)^{N(w, x)} \right) + \lambda \sum_{x \in D^u} \sum_c P(c | x, \bar{\theta}) \log \left(P(c) \prod_w P(w | c)^{N(w, x)} \right),$$

where D^l denotes labeled data, D^u denotes unlabeled data. We can reduce the influence of unlabeled data by decreasing the value of λ .

We derived new update rules from this new Q-function. The EM computation stops when the difference in values of the Q-function is smaller than a threshold.

4.4 Class Imbalance Problem

We have two problems with respect to “time-unknown” tag.

The first problem is the *class imbalance* problem (Japkowicz 2000). The number of time-unknown time-slot sentences is much larger than that of the other sentences as shown in Table 1. There are more than ten times as many time-unknown time-slot sentences as the other sentences.

Second, there are no time-associated words in the sentences categorized into “time-unknown”. Thus the feature distribution of time-unknown time-slot sentences is remarkably different from the others. It would be expected that they adversely affect proposed method.

There have been some methodologies in order to solve the class imbalance problem, such as Zhang and Mani (2003), Fan et al. (1999) and Abe et al. (2004). However, in our case, we have to resolve the latter problem in addition to the class imbalance problem. To deal with two problems above simultaneously and precisely, we develop a cascaded classification procedure.

4.5 Time-Slot Classification Method

It's desirable to treat only "time-known" sentences at NB+EM process to avoid the above-mentioned problems. We prepare another classifier for filtering time-unknown sentences before NB+EM process for that purpose. Thus, we propose a classification method in 2 steps (Method A). The flow of the 2-step classification is shown in Figure 1. In this figure, ovals represent classifiers, and arrows represent flow of data.

The first classifier (hereafter, "time-unknown" filter) classifies sentences into two classes; "time-unknown" and "time-known". The "time-known" class is a coarse class consisting of four time-slots (morning, daytime, evening, and night). We use Support Vector Machines as a classifier. The features we used are all words included in the sentence to be classified.

The second classifier (time-slot classifier) classifies "time-known" sentences into four classes. We use Naïve Bayes classifier backed up with the Expectation Maximization (EM) algorithm mentioned in Section 4.3.

The features for the time-slot classifier are words, whose part of speech is noun or verb. The set of these features are called NORMAL in the rest of this paper. In addition, we use information from the previous and the following sentences in the blog entry. The words included in such sentences are also used as features. The set of these features are called CONTEXT. The features in CONTEXT would be effective for estimating time-slot of the sentences as mentioned in Example2 in Section 3.1.

We also use a simple classifier (Method B) for comparison. The Method B classifies all time-slots (morning ~ night, time-unknown) sentences at just one step. We use Naïve Bayes classifier backed up with the Expectation Maximization (EM) algorithm at this learning. The features are words (whose part-of-speech is noun or verb) included in the sentence to be classified.

5 Experimental Results and Discussion

5.1 Time-Slot Classifier with Time-Associated Words

5.1.1 Time-Unknown Filter

We used 11,668 positive (time-unknown) samples and 2,552 negative (morning ~ night) samples. We conducted a classification experiment by Support Vector Machines with 10-fold cross

validation. We used TinySVM² software package for implementation. The soft margin parameter is automatically estimated by 10-fold cross validation with training data. The result is shown in Table 2.

Table 2 clarified that the "time-unknown" filter achieved good performance; F-measure of 0.899. In addition, since we obtained a high recall (0.969), many of the noisy sentences will be filtered out at this step and the classifier of the second step is likely to perform well.

Accuracy	0.878
Precision	0.838
Recall	0.969
F-measure	0.899

Table 2: Classification result of the time-unknown filter.

5.1.2 Time-Slot Classification

In step 2, we used "time-known" sentences classified by the unknown filter as test data. We conducted a classification experiment by Naïve Bayes classifier + the EM algorithm with 10-fold cross validation. For unlabeled data, we used 64,782 sentences, which have no intersection with the labeled data. The parameters, λ and β , are automatically estimated by 10-fold cross validation with training data. The result is shown in Table 3.

Method	Accuracy	
	NORMAL	CONTEXT
Explicit	0.109	
Baseline	0.406	
NB	0.567	0.464
NB + EM	0.673	0.670

Table 3: The result of time-slot classifier.

² <http://www.chasen.org/~taku/software/TinySVM>

		output of time-slot classifier					sum
		morning	daytime	evening	night	time-unknown	
time-slot tag	morning	332	14	1	37	327	711
	daytime	30	212	1	44	312	599
	evening	4	5	70	18	110	207
	night	21	19	4	382	609	1035
	time-unknown	85	66	13	203	11301	11668
sum		472	316	89	684	12659	14220

Table 4: Confusion matrix of output.

rank	morning		daytime		evening		night	
	word	$p(c w)$	word	$p(c w)$	word	$p(c w)$	word	$p(c w)$
1	this morning	0.729	noon	0.728	evening	0.750	last night	0.702
2	morning	0.673	early after noon	0.674	sunset	0.557	night	0.689
3	breakfast	0.659	afternoon	0.667	academy	0.448	fireworks	0.688
4	early morning	0.656	daytime	0.655	dusk	0.430	dinner	0.684
5	before noon	0.617	lunch	0.653	Hills	0.429	go to bed	0.664
6	compacted snow	0.603	lunch	0.636	run on	0.429	night	0.641
7	commute	0.561	lunch break	0.629	directions	0.429	bow	0.634
8	---	0.541	lunch	0.607	pinecone	0.429	overtime	0.606
9	parade	0.540	noon	0.567	priest	0.428	year-end party	0.603
10	wake up	0.520	butterfly	0.558	sand beach	0.428	dinner	0.574
11	leave harbor	0.504	Chinese food	0.554	---	0.413	beach	0.572
12	rise late	0.504	forenoon	0.541	Omori	0.413	cocktail	0.570
13	cargo work	0.504	breast-feeding	0.536	fan	0.413	me	0.562
14	alarm clock	0.497	nap	0.521	Haneda	0.412	Tomoyuki	0.560
15	---	0.494	diaper	0.511	preview	0.402	return home	0.557
16	sunglow	0.490	Japanese food	0.502	cloud	0.396	close	0.555
17	wheel	0.479	star festival	0.502	Dominus	0.392	stay up late	0.551
18	wake up	0.477	hot noodle	0.502	slip	0.392	tonight	0.549
19	perm	0.474	pharmacy	0.477	tasting	0.391	night	0.534
20	morning paper	0.470	noodle	0.476	nest	0.386	every night	0.521

Table 5: Time-associated words examples.

In Table 3, “Explicit” indicates the result by a simple classifier based on regular expressions³ including explicit temporal expressions. The baseline method classifies all sentences into night because the number of night sentences is the largest. The “CONTEXT” column shows the results obtained by classifiers learned with the features in CONTEXT in addition to the features

³ For example, we classify sentences matching following regular expressions into morning class:

[(gozen)(gozen-no)(asa) (asa-no)(am)(AM)(am-no)(AM-no)][456789(10)] j i, [(04)(05)(06)(07)(08)(09)]j i, [(04)(05)(06)(07)(08)(09)]:[0-9]{2,2}, [456789(10)][(am)(AM)].

(“gozen”, “gozen - no” means before noon. “asa”, “asa-no” means morning. “j i” means o’clock.)

in NORMAL. The accuracy of the Explicit method is lower than the baseline. This means existing methods based on explicit temporal expressions cannot work well in blog text. The accuracy of the method ‘NB’ exceeds that of the baseline by 16%. Furthermore, the accuracy of the proposed method ‘NB+EM’ exceeds that of the ‘NB’ by 11%. Thus, we figure out that using unlabeled data improves the performance of our time-slot classification.

In this experiment, unfortunately, CONTEXT only deteriorated the accuracy. The time-slot tags of the sentences preceding or following the target sentence may still provide information to improve the accuracy. Thus, we tried a sequential tagging method for sentences, in which tags are

predicted in the order of their occurrence. The predicted tags are used as features in the prediction of the next tag. This type of sequential tagging method regard as a chunking procedure (Kudo and Matsumoto, 2000) at sentence level. We conducted time-slot (five classes) classification experiment, and tried forward tagging and backward tagging, with several window sizes. We used YamCha⁴, the multi-purpose text chunker using Support Vector Machines, as an experimental tool. However, any tagging direction and window sizes did not improve the performance of classification. Although a chunking method has possibility of correctly classifying a sequence of text units, it can be adversely biased by the preceding or the following tag. The sentences in blog used in our experiments would not have a very clear tendency in order of tags. This is why the chunking-method failed to improve the performance in this task. We would like to try other bias-free methods such as Conditional Random Fields (Lafferty et al., 2001) for future work.

5.1.3 2-step Classification

Finally, we show an accuracy of the 2-step classifier (Method A) and compare it with those of other classifiers in Table 6. The accuracies are calculated with the equation:

$$\frac{\left(\begin{array}{l} \# \text{ time-unknown sentences correctly classified by the time-unknown filter} \\ + \left(\begin{array}{l} \# \text{ known sentences correctly classified by the time-slot classifier} \end{array} \right) \end{array} \right)}{\# \text{ sentences with a time-slot tag value}}$$

In Table 6, the baseline method classifies all sentences into time-unknown because the number of time-unknown sentences is the largest. Accuracy of Method A (proposed method) is higher than that of Method B (4.1% over). These results show that time-unknown sentences adversely affect the classifier learning, and 2-step classification is an effective method.

Table 4 shows the confusion matrix corresponding to the Method A (NORMAL). From this table, we can see Method A works well for classification of morning, daytime, evening, and night, but has some difficulty in

Method	Conclusive accuracy
Explicit	0.833
Baseline	0.821
Method A (NORMAL)	0.864
Method A (CONTEXT)	0.862
Method B	0.823

Table 6: Comparison of the methods for five class classification

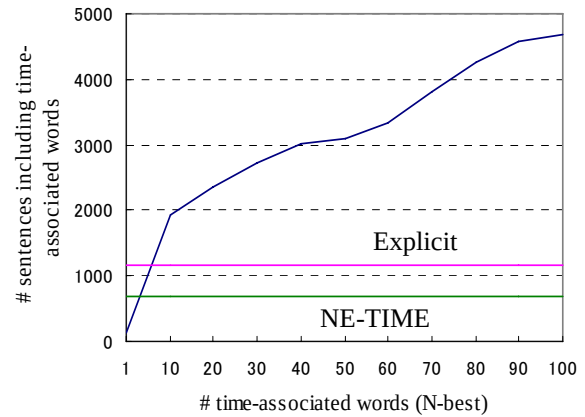


Figure 2: Change of # sentences that have time-associated words: “Explicit” indicates the number of sentences including explicit temporal expressions, “NE-TIME” indicates the number of sentences including NE-TIME tag.

classification of time-unknown. The 11.7% of samples were wrongly classified into “night” or “unknown”.

We briefly describe an error analysis. We found that our classifier tends to wrongly classify samples in which two or more events are written in a sentence. The followings are examples:

Example 3

- a. I attended a party last night, and I got back on the first train in this morning because the party was running over.
- b. I bought a cake this morning, and ate it after the dinner.

5.2 Examples of Time-Associated Words

Table 5 shows some time-associated words obtained by the proposed method. The words are sorted in the descending order of the value of $P(c|w)$. Although some consist of two or three words, their original forms in Japanese consist of one word. There are some expressions appearing more than once, such as “dinner”. Actually these expressions have different forms in Japanese. Meaningless (non-word) strings caused by mor-

⁴ <http://www.chasen.org/~taku/software/YamCha>

phological analysis error are presented as the symbol “---”. We obtained a lot of interesting time-associated words, such as “commute (morning)”, “fireworks (night)”, and “cocktail (night)”. Most words obtained are significantly different from explicit temporal expressions and NE-TIME expressions.

Figure 2 shows the number of sentences including time-associated words in blog text. The horizontal axis represents the number of time-associated words. We sort the words in the descending order of $P(c|w)$ and selected the top N words. The vertical axis represents the number of sentences including any N-best time-associated words. We also show the number of sentences including explicit temporal expressions, and the number of sentences including NE-TIME tag (Sekine and Isahara, 1999) for comparison. The set of explicit temporal expressions was extracted by the method described in Section 5.1.2. We used a Japanese linguistic analyzer “Cabocha⁵” to obtain NE-TIME information. From this graph, we can confirm that the number of target sentences of our proposed method is larger than that of existing methods.

6 Conclusion

In our study, we proposed a method for identifying when an event in text occurs. We succeeded in using a semi-supervised method, the Naïve Bayes Classifier enhanced by the EM algorithm, with a small amount of labeled data and a large amount of unlabeled data. In order to avoid the class imbalance problem, we used a 2-step classifier, which first filters out time-unknown sentences and then classifies the remaining sentences into one of 4 classes. The proposed method outperformed the simple 1-step method. We obtained 86.4% of accuracy that exceeds the existing method and the baseline method.

References

- Naoki Abe, Bianca Zadrozny, John Langford. 2004. An Iterative Method for Multi-class Cost-sensitive Learning. In *Proc. of the 10th. ACM SIGKDD*, pp.3–11.
- Arthur P. Dempster, Nan M. laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, Vol. 39, No. 1, pp.1–38.
- Wei Fan, Salvatore J. Stolfo, Junxin Zhang, Philip K. Chan. 1999. AdaCost: Misclassification Cost-sensitive Boosting. In *Proc. of ICML*, pp.97–105.
- Thomas Hofmann. 2001. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42:177–196.
- Nathalie Japkowicz. 2000. Learning from Imbalanced Data Sets: A Comparison of Various Strategies. In *Proc. of the AAAI Workshop on Learning from Imbalanced Data Sets*, pp.10–15.
- Taku Kudo, Yuji Matsumoto. 2000. Use of Support Vector Learning for Chunking Identification, In *Proc of the 4th CoNLL*, pp.142–144.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data, In *Proc. of ICML*, pp.282–289.
- Inderjeet Mani, George Wilson 2000. Robust Temporal Processing of News. In *Proc. of the 38th ACL*, pp.69–76.
- Tomoyuki Nanno, Yasuhiro Suzuki, Toshiaki Fujiki, Manabu Okumura. 2004. Automatically Collecting and Monitoring Japanese Weblogs. *Journal for Japanese Society for Artificial Intelligence*, Vol.19, No.6, pp.511–520. (in Japanese)
- Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, Vol. 39, No.2/3, pp.103–134.
- Satoshi Sekine, Hitoshi Isahara. 1999. IREX project overview. *Proceedings of the IREX Workshop*.
- Andrea Setzer, Robert Gaizauskas. 2001. A Pilot Study on Annotating Temporal Relations in Text. In *Proc. of the ACL-2001 Workshop on Temporal and Spatial Information Processing*, Toulouse, France, July, pp.88–95.
- Seiji Tsuchiya, Hirokazu Watabe, Tsukasa Kawaoka. 2005. Evaluation of a Time Judgement Technique Based on an Association Mechanism. *IPSG SIG Technical Reports*, 2005-NL-168, pp.113–118. (in Japanese)
- Jianping Zhang, Inderjeet Mani. 2003. kNN Approach to Unbalanced Data Distributions: A Case Study involving Information Extraction. In *Proc. of ICML Workshop on Learning from Imbalanced Datasets II.*, pp.42–48.

⁵ <http://chasen.org/~taku/software/cabocha/>

Optimal Constituent Alignment with Edge Covers for Semantic Projection

Sebastian Padó
Computational Linguistics
Saarland University
Saarbrücken, Germany
pado@coli.uni-sb.de

Mirella Lapata
School of Informatics
University of Edinburgh
Edinburgh, UK
mlap@inf.ed.ac.uk

Abstract

Given a parallel corpus, semantic projection attempts to transfer semantic role annotations from one language to another, typically by exploiting word alignments. In this paper, we present an improved method for obtaining *constituent alignments* between parallel sentences to guide the role projection task. Our extensions are twofold: (a) we model constituent alignment as *minimum weight edge covers* in a bipartite graph, which allows us to find a globally optimal solution efficiently; (b) we propose *tree pruning* as a promising strategy for reducing alignment noise. Experimental results on an English-German parallel corpus demonstrate improvements over state-of-the-art models.

1 Introduction

Recent years have witnessed increased interest in data-driven methods for many natural language processing (NLP) tasks, ranging from part-of-speech tagging, to parsing, and semantic role labelling. The success of these methods is due partly to the availability of large amounts of training data annotated with rich linguistic information. Unfortunately, such resources are largely absent for almost all languages except English. Given the data requirements for supervised learning, and the current paucity of suitable data for many languages, methods for generating annotations (semi-)automatically are becoming increasingly popular.

Annotation projection tackles this problem by leveraging parallel corpora and the high-accuracy tools (e.g., parsers, taggers) available for a few languages. Specifically, through the use of word alignments, annotations are transferred from resource-rich languages onto low density ones. The projection process can be decomposed into three steps: (a) determining the units of projection;

these are typically words but can also be chunks or syntactic constituents; (b) inducing alignments between the projection units and projecting annotations along these alignments; (c) reducing the amount of noise in the projected annotations, often due to errors and omissions in the word alignment.

The degree to which analyses are parallel across languages is crucial for the success of projection approaches. A number of recent studies rely on this notion of parallelism and demonstrate that annotations can be adequately projected for parts of speech (Yarowsky and Ngai, 2001; Hi and Hwa, 2005), chunks (Yarowsky and Ngai, 2001), and dependencies (Hwa et al., 2002).

In previous work (Padó and Lapata, 2005) we considered the annotation projection of semantic roles conveyed by sentential constituents such as AGENT, PATIENT, or INSTRUMENT. Semantic roles exhibit a high degree of parallelism across languages (Boas, 2005) and thus appear amenable to projection. Furthermore, corpora labelled with semantic role information can be used to train shallow semantic parsers (Gildea and Jurafsky, 2002), which could in turn benefit applications in need of broad-coverage semantic analysis. Examples include question answering, information extraction, and notably machine translation.

Our experiments concentrated primarily on the first projection step, i.e., establishing the right level of linguistic analysis for effecting projection. We showed that projection schemes based on constituent alignments significantly outperform schemes that rely exclusively on word alignments. A *local* optimisation strategy was used to find constituent alignments, while relying on a simple filtering technique to handle noise.

The study described here generalises our earlier semantic role projection framework in two important ways. First, we formalise constituent projection as the search for a *minimum weight edge cover* in a weighted bipartite graph. This formalisation

efficiently yields constituent alignments that are globally optimal. Second, we propose *tree pruning* as a general noise reduction strategy, which exploits both structural and linguistic information to enable projection. Furthermore, we quantitatively assess the impact of noise on the task by evaluating both on automatic and manual word alignments.

In Section 2, we describe the task of role-semantic projection and the syntax-based framework introduced in Padó and Lapata (2005). Section 3 explains how semantic role projection can be modelled with minimum weight edge covers in bipartite graphs. Section 4 presents our tree pruning strategy. We present our evaluation framework and results in Section 5. A discussion of related and future work concludes the paper.

2 Cross-lingual Semantic Role projection

Semantic role projection is illustrated in Figure 1 using English and German as the source-target language pair. We assume a FrameNet-style semantic analysis (Fillmore et al., 2003). In this paradigm, the semantics of predicates and their arguments are described in terms of *frames*, conceptual structures which model prototypical situations. The English sentence *Kim promised to be on time* in Figure 1 is an instance of the COMMITMENT frame. In this particular example, the frame introduces two roles, i.e., SPEAKER (*Kim*) and MESSAGE (*to be on time*). Other possible, though unrealised, roles are ADDRESSEE, MESSAGE, and TOPIC. The COMMITMENT frame can be introduced by *promise* and several other verbs and nouns such as *consent* or *threat*.

We also assume that frame-semantic annotations can be obtained reliably through shallow semantic parsing.¹ Following the assignment of semantic roles on the English side, (imperfect) word alignments are used to infer *semantic alignments* between constituents (e.g., *to be on time* is aligned with *pünktlich zu kommen*), and the role labels are transferred from one language to the other. Note that role projection can only take place if the source predicate (here *promised*) is word-aligned to a target predicate (here *versprach*) evoking the same frame; if this is not the case (e.g., in metaphors), projected roles will not be generally appropriate.

We represent the source and target sentences as sets of linguistic units, U_s and U_t , respectively.

¹See Carreras and Màrquez (2005) for an overview of recent approaches to semantic parsing.

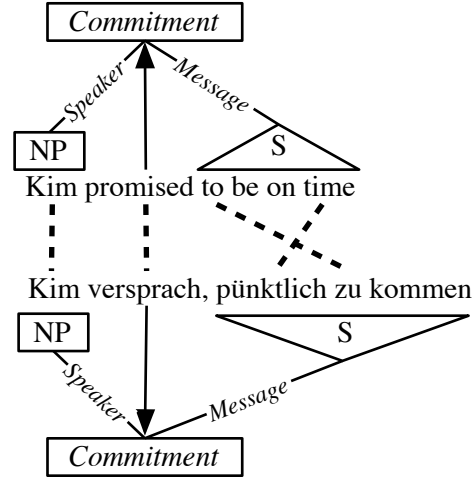


Figure 1: Projection of semantic roles from English to German (word alignments as dotted lines)

The assignment of semantic roles on the source side is a function $role_s : R \rightarrow 2^{U_s}$ from roles to sets of source units. Constituent alignments are obtained in two steps. First, a real-valued function $sim : U_s \times U_t \rightarrow \mathbb{R}$ estimates pairwise similarities between source and target units. To make our model robust to alignment noise, we use only content words to compute the similarity function. Next, a *decision procedure* uses the similarity function to determine the set of semantically equivalent, i.e., aligned units $A \subseteq U_s \times U_t$. Once A is known, semantic projection reduces to transferring the semantic roles from the source units onto their aligned target counterparts:

$$role_t(r) = \{u_t \mid \exists u_s \in role_s(r) : (u_s, u_t) \in A\}$$

In Padó and Lapata (2005), we evaluated two main parameters within this framework: (a) the choice of linguistic units and (b) methods for computing semantic alignments. Our results revealed that constituent-based models outperformed word-based ones by a wide margin (0.65 Fscore vs. 0.46), thus demonstrating the importance of bracketing in amending errors and omissions in the automatic word alignment. We also compared two simplistic alignment schemes, *backward alignment* and *forward alignment*. The first scheme aligns each target constituent to its most similar source constituent, whereas the second (A_f) aligns each source constituent to its most similar target constituent:

$$A_f = \{(u_s, u_t) \mid u_t = \operatorname{argmax}_{u'_t \in U_t} sim(u_s, u'_t)\}$$

An example constituent alignment obtained from the forward scheme is shown in Figure 2 (left side). The nodes represent constituents in the source and target language and the edges indicate the resulting alignment. Forward alignment generally outperformed backward alignment (0.65 F-score vs. 0.45). Both procedures have a time complexity quadratic in the maximal number of sentence nodes: $O(|U_s||U_t|) = O(\max(|U_s|, |U_t|)^2)$.

A shortcoming common to both decision procedures is that they are *local*, i.e., they optimise the alignment for each node independently of all other nodes. Consider again Figure 2. Here, the forward procedure creates alignments for all source nodes, but leaves constituents from the target set unaligned (see target node (1)). Moreover, local alignment methods constitute a rather weak model of semantic equivalence since they allow one target node to correspond to any number of source nodes (see target node (3) in Figure 2, which is aligned to three source nodes). In fact, by allowing any alignment between constituents, the local models can disregard important linguistic information, thus potentially leading to suboptimal results. We investigate this possibility by proposing well-understood global optimisation models which suitably constrain the resulting alignments.

Besides matching constituents reliably, poor word alignments are a major stumbling block for achieving accurate projections. Previous research addresses this problem in a post-processing step, by reestimating parameter values (Yarowsky and Ngai, 2001), by applying transformation rules (Hwa et al., 2002), by using manually labelled data (Hi and Hwa, 2005), or by relying on linguistic criteria (Padó and Lapata, 2005). In this paper, we present a novel filtering technique based on tree pruning which removes extraneous constituents in a preprocessing stage, thereby disassociating filtering from the alignment computation.

In the remainder of this paper, we present the details of our global optimisation and filtering techniques. We only consider constituent-based models, since these obtained the best performance in our previous study (Padó and Lapata, 2005).

3 Globally optimal constituent alignment

We model constituent alignment as a *minimum weight bipartite edge cover* problem. A bipartite graph is a graph $G = (V, E)$ whose node set V is partitioned into two nonempty sets V_1 and V_2 in such a way that every edge E joins a node in V_1

to a node in V_2 . In a weighted bipartite graph a weight is assigned to each edge. An edge cover is a subgraph of a bipartite graph so that each node is linked to at least one node of the other partition. A minimum weight edge cover is an edge cover with the least possible sum of edge weights.

In our projection application, the two partitions are the sets of source and target sentence constituents, U_s and U_t , respectively. Each source node is connected to all target nodes and each target node to all source nodes; these edges can be thought of as potential constituent alignments. The edge weights, which represent the (dis)similarity between nodes u_s and u_t are set to $1 - \text{sim}(u_s, u_t)$.² The minimum weight edge cover then represents the alignment with the maximal similarity between source and target constituents. Below, we present details on graph *edge covers* and a more restricted kind, *minimum weight perfect bipartite matchings*. We also discuss their computation.

Edge covers Given a bipartite graph G , a minimum weight edge cover A_e can be defined as:

$$A_e = \underset{\text{Edge cover } E}{\operatorname{argmin}} \sum_{(u_s, u_t) \in E} 1 - \text{sim}(u_s, u_t)$$

An example edge cover is illustrated in Figure 2 (middle). Edge covers are somewhat more constrained compared to the local model described above: all source and target nodes have to take part in some alignment. We argue that this is desirable in modelling constituent alignment, since important linguistic units will not be ignored. As can be seen, edge covers allow one-to-many alignments which are common when translating from one language to another. For example, an English constituent might be *split* into several German constituents or alternatively two English constituents might be *merged* into a single German constituent. In Figure 2, the source nodes (3) and (4) correspond to target node (4). Since each node of either side has to participate in at least one alignment, edge covers cannot account for *insertions* arising when constituents in the source language have no counterpart in their target language, or vice versa, as is the case for *deletions*.

Weighted perfect bipartite matchings *Perfect bipartite matchings* are a more constrained version of edge covers, in which each node has *exactly* one adjacent edge. This restricts constituent

²The choice of similarity function is discussed in Section 5.

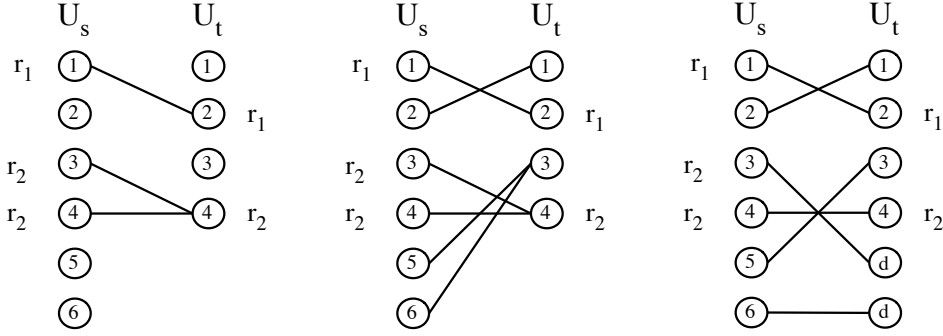


Figure 2: Constituent alignments and role projections resulting from different decision procedures (U_s, U_t : sets of source and target constituents; r_1, r_2 : two semantic roles). Left: local forward alignment; middle: edge cover; right: perfect matching with dummy nodes

alignment to a bijective function: each source constituent is linked to exactly one target constituent, and vice versa. Analogously, a *minimum weight perfect bipartite matching* A_m is a minimum weight edge cover obeying the one-to-one constraint:

$$A_m = \operatorname{argmin}_{\text{Matching } M} \sum_{(u_s, u_t) \in M} 1 - \text{sim}(u_s, u_t)$$

An example of a perfect bipartite matching is given in Figure 2 (right), where each node has exactly one adjacent edge. Note that the target side contains two nodes labelled (d), a shorthand for “dummy” node. Since sentence pairs will often differ in length, the resulting graph partitions will have different sizes as well. In such cases, dummy nodes are introduced in the smaller partition to enable perfect matching. Dummy nodes are assigned a similarity of zero with all other nodes. Alignments to dummy nodes (such as for source nodes (3) and (6)) are ignored during projection.

Perfect matchings are more restrictive models of constituent alignment than edge covers. Being bijective, the resulting alignments cannot model splitting or merging operations at all. Insertions and deletions can be modelled only indirectly by aligning nodes in the larger partition to dummy nodes on the other side (see the source side in Figure 2 where nodes (3) and (6) are aligned to (d)). Section 5 assesses if these modelling limitations impact the quality of the resulting alignments.

Algorithms Minimum weight perfect matchings in bipartite graphs can be computed efficiently in cubic time using algorithms for network optimisation (Fredman and Tarjan, 1987; time $O(|U_s|^2 \log |U_s| + |U_s|^2 |U_t|)$) or algorithms for

the equivalent linear assignment problem (Jonker and Volgenant, 1987; time $O(\max(|U_s|, |U_t|)^3)$). Their complexity is a linear factor slower than the quadratic runtime of the local optimisation methods presented in Section 2.

The computation of (general) edge covers has been investigated by Eiter and Mannila (1997) in the context of distance metrics for point sets. They show that edge covers can be reduced to minimum weight perfect matchings of an auxiliary bipartite graph with two partitions of size $|U_s| + |U_t|$. This allows the computation of general minimum weight edge covers in time $O((|U_s| + |U_t|)^3)$.

4 Filtering via Tree Pruning

We introduce two filtering techniques which effectively remove constituents from source and target trees before alignment takes place. Tree pruning as a preprocessing step is more general and more efficient than our original post-processing filter (Padó and Lapata, 2005) which was embedded into the similarity function. Not only does tree pruning not interfere with the similarity function but also reduces the size of the graph, thus speeding up the algorithms discussed in the previous section.

We present two instantiations of tree pruning: word-based filtering, which subsumes our earlier method, and argument-based filtering, which eliminates unlikely argument candidates.

Word-based filtering This technique removes terminal nodes from parse trees according to certain linguistic or alignment-based criteria. We apply two word-based filters in our experiments. The first removes non-content words, i.e., all words which are not adjectives, adverbs, verbs, or nouns, from the source and target sen-

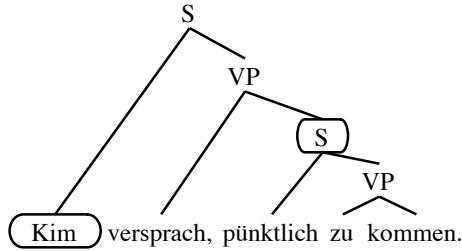


Figure 3: Filtering of unlikely arguments (predicate in boldface, potential arguments in boxes).

tences (Padó and Lapata, 2005). We also use a novel filter which removes all words which remain unaligned in the automatic word alignment. Non-terminal nodes whose terminals are removed by these filters, are also pruned.

Argument filtering Previous work in shallow semantic parsing has demonstrated that not all nodes in a tree are equally probable as semantic roles for a given predicate (Xue and Palmer, 2004). In fact, assuming a perfect parse, there is a “set of likely arguments”, to which almost all semantic roles should be assigned to. This set of likely arguments consists of all constituents which are a child of some ancestor of the predicate, provided that (a) they do not dominate the predicate themselves and (b) there is no sentence boundary between a constituent and its predicate. This definition covers long-distance dependencies such as control constructions for verbs, or support constructions for nouns and adjectives, and can be extended slightly to accommodate coordination.

This argument-based filter reduces *target* trees to a set of likely arguments. In the example in Figure 3, all tree nodes are removed except *Kim* and *pünktlich zu kommen*.

5 Evaluation Set-up

Data For evaluation, we used the parallel corpus³ from our earlier work (Padó and Lapata, 2005). It consists of 1,000 English-German sentence pairs from the Europarl corpus (Koehn, 2005). The sentences were automatically parsed (using Collin’s 1997 parser for English and Dubey’s 2005 parser for German), and manually annotated with FrameNet-like semantic roles (see Padó and Lapata 2005 for details.)

Word alignments were computed with the GIZA++ toolkit (Och and Ney, 2003), using the

entire English-German Europarl bitext as training data (20M words). We used the GIZA++ default settings to induce alignments for both directions (source-target, target-source). Following common practise in MT (Koehn et al., 2003), we considered only their intersection (bidirectional alignments are known to exhibit high precision). We also produced manual word alignments for all sentences in our corpus, using the GIZA++ alignments as a starting point and following the Blinker annotation guidelines (Melamed, 1998).

Method and parameter choice The constituent alignment models we present are *unsupervised* in that they do not require labelled data for inferring correct alignments. Nevertheless, our models have three parameters: (a) the similarity measure for identifying semantically equivalent constituents; (b) the filtering procedure for removing noise in the data (e.g., wrong alignments); and (c) the decision procedure for projection.

We retained the similarity measure introduced in Padó and Lapata (2005) which computes the overlap between a source constituent and its candidate projection, in both directions. Let $y(c_s)$ and $y(c_t)$ denote the yield of a source and target constituent, respectively, and $al(T)$ the union of all word alignments for a token set T :

$$sim(c_s, c_t) = \frac{|y(c_t) \cap al(y(c_s))|}{|y(c_s)|} \frac{|y(c_s) \cap al(y(c_t))|}{|y(c_t)|}$$

We examined three filtering procedures (see Section 4): removing non-aligned words (NA), removing non-content words (NC), and removing unlikely arguments (Arg). These were combined with three decision procedures: local forward alignment (Forward), perfect matching (Perf-Match), and edge cover matching (EdgeCover) (see Section 3). We used Jonker and Volgenant’s (1987) solver⁴ to compute weighted perfect matchings.

In order to find optimal parameter settings for our models, we split our corpus randomly into a development and test set (both 50% of the data) and examined the parameter space exhaustively on the development set. The performance of the best models was then assessed on the test data. The models had to predict semantic roles for German, using English gold standard roles as input, and were evaluated against German gold standard

³The corpus can be downloaded from <http://www.coli.uni-saarland.de/~pado/projection/>.

⁴The software is available from <http://www.magiclogic.com/assignment.html>.

	Model	Prec	Rec	F-score
No Filter	WordBL	45.6	44.8	45.1
	Forward	66.0	56.5	60.9
	PerfMatch	71.7	54.7	62.1
	EdgeCover	65.6	57.3	61.2
	UpperBnd	85.0	84.0	84.0

	Model	Prec	Rec	F-score
NA Filter	WordBL	45.6	44.8	45.1
	Forward	74.1	56.1	63.9
	PerfMatch	73.3	62.1	67.2
	EdgeCover	70.5	62.9	66.5
	UpperBnd	85.0	84.0	84.0

	Model	Prec	Rec	F-score
NC Filter	WordBL	45.6	44.8	45.1
	Forward	64.3	47.8	54.8
	PerfMatch	73.1	56.9	64.0
	EdgeCover	67.5	57.0	61.8
	UpperBnd	85.0	84.0	84.0

	Model	Prec	Rec	F-score
Arg Filter	WordBL	45.6	44.8	45.1
	Forward	69.9	60.7	65.0
	PerfMatch	80.4	48.1	60.2
	EdgeCover	69.6	60.6	64.8
	UpperBnd	85.0	84.0	84.0

Table 1: Model comparison using intersective alignments (development set)

roles. To gauge the extent to which alignment errors are harmful, we present results both on intersective and manual alignments.

Upper bound and baseline In Padó and Lapata (2005), we assessed the feasibility of semantic role projection by measuring how well annotators agreed on identifying roles and their spans. We obtained an inter-annotator agreement of 0.84 (F-score), which can serve as an upper bound for the projection task. As a baseline, we use a simple word-based model (WordBL) from the same study. The units of this model are words, and the span of a projected role is the union of all target terminals aligned to a terminal of the source role.

6 Results

Development set Our results on the development set are summarised in Table 1. We show how performance varies for each model according to different filtering procedures when automatically produced word alignments are used. No filtering is applied to the baseline model (WordBL).

Without filtering, local and global models yield comparable performance. Models based on perfect bipartite matchings (PerfMatch) and edge covers (EdgeCover) obtain slight F-score improvements over the forward alignment model (Forward). It is worth noticing that PerfMatch yields a significantly higher precision (using a χ^2 test, $p < 0.01$) than Forward and EdgeCover. This indicates that, even without filtering, PerfMatch delivers rather accurate projections, however with low recall.

Model performance seems to increase with tree pruning. When non-aligned words are removed (Table 1, NA Filter), PerfMatch and EdgeCover

reach an F-score of 67.2 and 66.5, respectively. This is an increase of approximately 3% over the local Forward model. Although the latter model yields high precision (74.1%), its recall is significantly lower than PerfMatch and EdgeCover ($p < 0.01$). This demonstrates the usefulness of filtering for the more constrained global models which as discussed in Section 3 can only represent a limited set of alignment possibilities.

The non-content words filter (NC filter) yields smaller improvements. In fact, for the Forward model, results are worse than applying no filtering at all. We conjecture that NC is an overly aggressive filter which removes projection-critical words. This is supported by the relatively low recall values. In comparison to NA, recall drops by 8.3% for Forward and by almost 6% for PerfMatch and EdgeCover. Nevertheless, both PerfMatch and EdgeCover outperform the local Forward model. PerfMatch is the best performing model reaching an F-score of 64.0%.

We now consider how the models behave when the argument-based filter is applied (Arg, Table 1, bottom). As can be seen, the local model benefits most from this filter, whereas PerfMatch is worst affected; it obtains its highest precision (80.4%) as well as its lowest recall (48.1%). This is somewhat expected since the filter removes the majority of nodes in the target partition causing a proliferation of dummy nodes. The resulting edge covers are relatively “unnatural”, thus counterbalancing the advantages of global optimisation.

To summarise, we find on the development set that PerfMatch in the NA Filter condition obtains the best performance (F-score 67.2%), followed closely by EdgeCover (F-score 66.5%) in the same

	Model	Prec	Rec	F-score
Intersective	WordBL	45.7	45.0	43.3
	Forward (Arg)	72.4	63.2	67.5
	PerfMatch (NA)	75.7	63.7	69.2
	EdgeCover (NA)	73.0	64.9	68.7
	UpperBnd	85.0	84.0	84.0

	Model	Prec	Rec	F-score
Manual	WordBL	62.1	60.7	61.4
	Forward (Arg)	72.2	68.6	70.4
	PerfMatch (NA)	75.7	67.5	71.4
	EdgeCover (NA)	71.9	69.3	70.6
	UpperBnd	85.0	84.0	84.0

Table 2: Model comparison using intersective and manual alignments (test set)

condition. In general, PerfMatch seems less sensitive to the type of filtering used; it yields best results in three out of four filtering conditions (see boldface figures in Table 1). Our results further indicate that Arg boosts the performance of the local model by guiding it towards linguistically appropriate alignments.⁵

A comparative analysis of the output of PerfMatch and EdgeCover revealed that the two models make similar errors (85% overlap). Disagreements, however, arise with regard to misparses. Consider as an example the sentence pair:

The Charter is [_{NP} an opportunity to bring the EU closer to the people.]

Die Charta ist [_{NP} eine Chance], [_S die EU den Bürgern näherzubringen.]

An ideal algorithm would align the English NP to both the German NP and S. EdgeCover, which can model one-to-many-relationships, acts “confidently” and aligns the NP to the German S to maximise the overlap similarity, incurring both a precision and a recall error. PerfMatch, on the other hand, cannot handle one-to-many relationships, acts “cautiously” and aligns the English NP to a dummy node, leading to a recall error. Thus, even though EdgeCover’s analysis is partly right, it will come out worse than PerfMatch, given the current dataset and evaluation method.

Test set We now examine whether our results carry over to the test data. Table 2 shows the

⁵Experiments using different filter combinations did not lead to performance gains over individual filters and are not reported here due to lack of space.

performance of the best models (Forward (Arg), PerfMatch (NA), and EdgeCover (NA)) on automatic (Intersective) and manual (Manual) alignments.⁶ All models perform significantly better than the baseline but significantly worse than the upper bound (both in terms of precision and recall, $p < 0.01$). PerfMatch and EdgeCover yield better F-scores than the Forward model. In fact, PerfMatch yields a significantly better precision than Forward ($p < 0.01$).

Relatively small performance gains are observed when manual alignments are used. The F-score increases by 2.9% for Forward, 2.2% for PerfMatch, and 1.9% for EdgeCover. Also note that this better performance is primarily due to a significant increase in recall ($p < 0.01$), but not precision. This is an encouraging result indicating that our filters and graph-based algorithms eliminate alignment noise to a large extent. Analysis of the models’ output revealed that the remaining errors are mostly due to incorrect parses (none of the parsers employed in this work were trained on the Europarl corpus) but also to modelling deficiencies. Recall from Section 3 that our global models cannot currently capture one-to-zero correspondences, i.e., deletions and insertions.

7 Related work

Previous work has primarily focused on the projection of grammatical (Yarowsky and Ngai, 2001) and syntactic information (Hwa et al., 2002). An exception is Fung and Chen (2004), who also attempt to induce FrameNet-style annotations in Chinese. Their method maps English FrameNet entries to concepts listed in HowNet⁷, an on-line ontology for Chinese, without using parallel texts.

The present work extends our earlier projection framework (Padó and Lapata, 2005) by proposing global methods for automatic constituent alignment. Although our models are evaluated on the semantic role projection task, we believe they also show promise in the context of statistical machine translation. Especially for systems that use syntactic information to enhance translation quality. For example, Xia and McCord (2004) exploit constituent alignment for rearranging sentences in the source language so as to make their word or-

⁶Our results on the test set are slightly higher in comparison to the development set. The fluctuation reflects natural randomness in the partitioning of our corpus.

⁷See http://www.keenage.com/zhiwang/e_zhiwang.html.

der similar to that of the target language. They learn tree reordering rules by aligning constituents heuristically using a naive local optimisation procedure analogous to forward alignment. A similar approach is described in Collins et al. (2005); however, the rules are manually specified and the constituent alignment step reduces to inspection of the source-target sentence pairs. The global optimisation models presented in this paper could be easily employed for the reordering task common to both approaches.

Other approaches treat rewrite rules not as a preprocessing step (e.g., for reordering source strings), but as a part of the translation model itself (Gildea, 2003; Gildea, 2004). Constituent alignments are learnt by estimating the probability of tree transformations, such as node deletions, insertions, and reorderings. These models have a greater expressive power than our edge cover models; however, this implies that approximations are often used to make computation feasible.

8 Conclusions

In this paper, we have proposed a novel method for obtaining constituent alignments between parallel sentences and have shown that it is useful for semantic role projection. A key aspect of our approach is the formalisation of constituent alignment as the search for a minimum weight edge cover in a bipartite graph. This formalisation provides efficient mechanisms for aligning constituents and yields results superior to heuristic approaches. Furthermore, we have shown that tree-based noise filtering techniques are essential for good performance.

Our approach rests on the assumption that constituent alignment can be determined solely from the lexical similarity between constituents. Although this allows us to model constituent alignments efficiently as edge covers, it falls short of modelling translational divergences such as substitutions or insertions/deletions. In future work, we will investigate minimal tree edit distance (Bille, 2005) and related formalisms which are defined on tree structures and can therefore model divergences explicitly. However, it is an open question whether cross-linguistic syntactic analyses are similar enough to allow for structure-driven computation of alignments.

Acknowledgments The authors acknowledge the support of DFG (Padó; grant Pi-154/9-2) and EPSRC (Lapata; grant GR/T04540/01).

References

- P. Bille. 2005. A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1-3):217–239.
- H. C. Boas. 2005. Semantic frames as interlingual representations for multilingual lexical databases. *International Journal of Lexicography*, 18(4):445–478.
- X. Carreras, L. Màrquez, eds. 2005. *Proceedings of the CoNLL shared task: Semantic role labelling*, Boston, MA, 2005.
- M. Collins, P. Koehn, I. Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd ACL*, 531–540, Ann Arbor, MI.
- M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the ACL/EACL*, 16–23, Madrid, Spain.
- A. Dubey. 2005. What to do when lexicalization fails: parsing German with suffix analysis and smoothing. In *Proceedings of the 43rd ACL*, 314–321, Ann Arbor, MI.
- T. Eiter, H. Mannila. 1997. Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133.
- C. J. Fillmore, C. R. Johnson, M. R. Petruck. 2003. Background to FrameNet. *International Journal of Lexicography*, 16:235–250.
- M. L. Fredman, R. E. Tarjan. 1987. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615.
- P. Fung, B. Chen. 2004. BiFrameNet: Bilingual frame semantics resources construction by cross-lingual induction. In *Proceedings of the 20th COLING*, 931–935, Geneva, Switzerland.
- D. Gildea, D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.
- D. Gildea. 2003. Loosely tree-based alignment for machine translation. In *Proceedings of the 41st ACL*, 80–87, Sapporo, Japan.
- D. Gildea. 2004. Dependencies vs. constituents for tree-based alignment. In *Proceedings of the EMNLP*, 214–221, Barcelona, Spain.
- C. Hi, R. Hwa. 2005. A backoff model for bootstrapping resources for non-english languages. In *Proceedings of the HLT/EMNLP*, 851–858, Vancouver, BC.
- R. Hwa, P. Resnik, A. Weinberg, O. Kolak. 2002. Evaluation of translational correspondence using annotation projection. In *Proceedings of the 40th ACL*, 392–399, Philadelphia, PA.
- R. Jonker, T. Volgenant. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38:325–340.
- P. Koehn, F. J. Och, D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the HLT/NAACL*, 127–133, Edmonton, AL.
- P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the MT Summit X*, Phuket, Thailand.
- I. D. Melamed. 1998. Manual annotation of translational equivalence: The Blinker project. Technical Report IRCS TR #98-07, IRCS, University of Pennsylvania, 1998.
- F. J. Och, H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–52.
- S. Padó, M. Lapata. 2005. Cross-lingual projection of role-semantic information. In *Proceedings of the HLT/EMNLP*, 859–866, Vancouver, BC.
- F. Xia, M. McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the 20th COLING*, 508–514, Geneva, Switzerland.
- N. Xue, M. Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of the EMNLP*, 88–94, Barcelona, Spain.
- D. Yarowsky, G. Ngai. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the HLT*, 161–168, San Diego, CA.

Utilizing Co-Occurrence of Answers in Question Answering

Min Wu¹ and Tomek Strzalkowski^{1,2}

¹ ILS Institute, University at Albany, State University of New York
1400 Washington Ave SS261, Albany NY, 12222

² Institute of Computer Science, Polish Academy of Sciences
minwu@cs.albany.edu, tomek@csc.albany.edu

Abstract

In this paper, we discuss how to utilize the co-occurrence of answers in building an automatic question answering system that answers a series of questions on a specific topic in a batch mode. Experiments show that the answers to the many of the questions in the series usually have a high degree of co-occurrence in relevant document passages. This feature sometimes can't be easily utilized in an automatic QA system which processes questions independently. However it can be utilized in a QA system that processes questions in a batch mode. We have used our previous TREC QA system as baseline and augmented it with new answer clustering and co-occurrence maximization components to build the batch QA system. The experiment results show that the QA system running under the batch mode get significant performance improvement over our baseline TREC QA system.

1 Introduction

Question answering of a series of questions on one topic has gained more and more research interest in the recent years. The current TREC QA test set contains factoid and list questions grouped into different series, where each series has the target of a definition associated with it (Overview of the TREC 2004 Question Answering Track, Voorhees 2005). Usually, the target is also called "topic" by QA researchers. One of the restrictions of TREC QA is that "questions within a series must be processed in order, without looking ahead." That is, systems are allowed to use answers to earlier questions to help answer

later questions in the same series, but can not use later questions to help answer earlier questions. This requirement models the dialogue discourse between the user and the QA system. However our experiments on interactive QA system show that some impatient QA users will throw a bunch of questions to the system and waiting for the answers returned in all. This prompted us to consider building a QA system which can accept as many questions as possible from users once in all and utilizing the relations between these questions to help find answers. We would also like to know the performance difference between the QA system processing the question series in an order and the QA system processing the question series as a whole. We call the second type of QA system as batch QA system to avoid the ambiguity in the following description in this paper.

What kind of relations between questions could be utilized is a key problem in building the batch QA system. By observing the test questions of TREC QA, we found that the questions given under the same topic are not independent at all. Figure-1 shows a series of three questions proposed under the topic "Russian submarine Kursk Sinks" and some relevant passages to this topic found in the TREC data set. These passages contain answers not to just one but to two or three of the questions. This indicates that the answers to these questions have high co-occurrence.

In an automatic QA system which processes the questions independently, the answers to the questions may or may not always be extracted due to algorithmic limitations or noisy information around the correct answer. However in building a batch QA system, the interdependence between the answers could be utilized to help to filter out the noisy information and pinpoint the correct answer for each question in the series.

Topic Russian submarine Kursk sinks

1. When did the submarine sink? August 12
2. How many crewmen were lost in the disaster? 118
3. In what sea did the submarine sink? Barents Sea

Some Related Passages

Russian officials have speculated that the Kursk collided with another vessel in the Barents Sea, and usually blame an unspecified foreign submarine. All 118 officers and sailors aboard were killed.

The Russian governmental commission on the accident of the submarine Kursk sinking in the Barents Sea on August 12 has rejected 11 original explanations for the disaster.

.... as the same one carried aboard the nuclear submarine Kursk, which sank in the Barents Sea on Aug. 12, killing all 118 crewmen aboard.

The navy said Saturday that most of the 118-man crew died Aug. 12 when a huge explosion

Chief of Staff of the Russian Northern Fleet Mikhail Motsak Monday officially confirmed the deaths of 118 crewmen on board the Kursk nuclear submarine that went to the bottom of the Barents Sea on August 12.

Figure-1 Questions and Related Passages

We will discuss later in this paper how to utilize the co-occurrence of answers to a series of questions in building a batch QA system. The remainder of this paper is organized as follows. In the next section, we review the current techniques used in building an automatic QA system. Section 3 introduces the answers co-occurrence and how to cluster questions by the co-occurrence of their answers. Section 4.1 describes our TREC QA system and section 4.2 describes how to build a batch QA system by augmenting the TREC QA system with question clustering and answer co-occurrence maximization. Section 4.3 describes the experiments and explains the experimental results. Finally we conclude with the discussion of future work.

2 Related Work

During recent years, many automatic QA systems have been developed and the techniques used in these systems cover logic inference, syntactic relation analysis, information extraction and proximity search, some systems also utilize

pre-compiled knowledge base and external online knowledge resource.

The LCC system (Moldovan & Rus, 2001; Harabagiu et al. 2004) uses a logic prover to select answer from related passages. With the aid of extended WordNet and knowledge base, the text terms are converted to logical forms that can be proved to match the question logical forms. The IBM's PIQUANT system (Chu-Carroll et al, 2003; Prager et al, 2004) adopts a QA-by-Dossier-with-Constraints approach, which utilizes the natural constraints between the answer to the main question and the answers to the auxiliary questions. Syntactic dependency matching has also been applied in many QA systems (Cui et al, 2005; Katz and Lin 2003). The syntactic dependency relations of a candidate sentence are matched against the syntactic dependency relations in the question in order to decide if the candidate sentence contains the answer. Although surface text pattern matching is a comparatively simple method, it is very efficient for simple factoid questions and is used by many QA systems (Hovy et al 2001; Soubbotin, M. and S. Soubbotin 2003). As a powerful web search engine and external online knowledge resource, Google has been widely adopted in QA systems (Hovy et al 2001; Cui 2005) as a tool to help passage retrieval and answer validation.

Current QA systems mentioned above and represented at TREC have been developed to answer one question at the time. This may partially be an artifact of the earlier TREC QA evaluations which used large sets of independent questions. It may also partially reflect the intention of the current TREC QA Track that the question series introduced in TREC QA 2004 (Voorhees 2005) simulate an interaction with a human, thus expected to arrive one at a time.

The co-occurrence of answers of a series of highly related questions has not yet been fully utilized in current automatic QA systems participating TREC. In this situation, we think it worthwhile to find out whether a series of highly related questions on a specific topic such as the TREC QA test questions can be answered together in a batch mode by utilizing the co-occurrences of the answers and how much it will help improve the QA system performance.

3 Answer Co-Occurrence and Question Clustering

Many QA systems utilize the co-occurrence of question terms in passage retrieval (Cui 2005).

Some QA systems utilize the co-occurrence of question terms and answer terms in answer validation. These methods are based on the assumption that the co-occurrences of question terms and answer terms are relatively higher than the co-occurrences of other terms. Usually the co-occurrence are measured by pointwise mutual information between terms.

During the development of our TREC QA system, we found the answers of some questions in a series have higher co-occurrence. For example, in a series of questions on a topic of disaster event, the answers to questions such as “when the event occurred”, “where the event occurred” and “how many were injured in the event” have high co-occurrence in relatively short passages. Also, in a series of questions on a topic of some person, the answers to questions such as “when did he die”, “where did he die” and “how did he die” have high co-occurrence. To utilize this answers co-occurrence effectively in a batch QA system, we need to know which questions are expected to have higher answers co-occurrence and cluster these questions to maximize the answers co-occurrence among the questions in the cluster.

Currently, the topics used in TREC QA test questions fall into four categories: “Person”, “Organization”, “Event” and “Things”. The topic can be viewed as an object and the series of questions can be viewed as asking for the attributes of the object. In this point of view, to find out which questions have higher answers co-occurrence is to find out which attributes of the object (topic) have high co-occurrence.

We started with three categories of TREC QA topics: “Event”, “Person” and “Organization”. For “Event” topic category, we divided it into two sub-categories: “Disaster Event” and “Sport Event”. From the 2004 & 2005 TREC QA test questions, we manually collected frequently asked questions on each topic category and mapped these questions to the corresponding attributes of the topic. We focused on frequently asked questions because these questions are easier to be classified and thus served as a good starting point for our work. However for this technique to scale in the future, we are expecting to integrate automatic topic model detection into the system. For topic category “Person”, the attributes and corresponding named entity (NE) tags list as follows.

Attribute	Attribute’s NE tag
Birth Date	Date
Birth Place	Location
Death Date	Date
Death Place	Location
Death Reason	Disease, Accident
Death Age	Number
Nationality	Nationality
Occupation	Occupation
Father	Person
Mother	Person
Wife	Person
Children	Person
Number of Children	Number
Real Name	Person, Other
Nick Name	Person, Other
Affiliation	Organization
Education	Organization

For each topic category, we collected 20 sample topics as well as the corresponding attributes information about these topics. The sample topic “Rocky Marciano” and the attributes are listed as follows:

Attribute	Attribute Value
Birth Date	September 1, 1923
Birth Place	Brockton, MA
Death Date	August 31, 1969
Death Place	Iowa
Death Reason	airplane crash
Death Age	45
Buried Place	Fort Lauderdale, FL
Nationality	American
Occupation	heavyweight champion boxer
Father	Pierino Marchegiano
Mother	Pasqualena Marchegiano
Wife	Barbara Cousins
Children	Mary Ann, Rocco Kevin
No. of Children	two
Real Name	Rocco Francis Marchegiano
Nick Name	none
Affiliation	none
Education	none

From each attribute of the sample topic, an appropriate question can be formulated and relevant passages about this question were retrieved from TREC data (AQUAINT Data) and the web. A topic-related passages collection was formed by the relevant passages of questions on all attributes under the topic. Among the topic-related passages, the pointwise mutual information (PMI) of attribute values were calculated which consequently formed a symmetric mutual information matrix. The PMI of two attribute values x and y was calculated by the following equation.

$$PMI(x, y) = \log \frac{p(x, y)}{p(x)p(y)}$$

All the mutual information matrixes under the topic category were added up and averaged in order to get one mutual information matrix which reflects the general co-occurrence rela-

tions between attributes under the topic category. We clustered the attributes by their mutual information value. Our clustering strategy was to cluster attributes whose pointwise mutual information is greater than a threshold λ . We choose λ as equal to 60% of the maximum value in the matrix.

The operations described above were automatically carried out by our carefully designed training system. The clusters learned for each

“Person” Topic

Cluster1: Birth Date; Birth Place
Cluster2a: Death Date; Death Place;
Death Reason; Death Age
Cluster2b: Death Date; Birth Date
Cluster3: Father; Mother
Cluster4: Wife; Children; Number of Children
Cluster5: Nationality; Occupation

“Disaster Event” Topic

Cluster1: Event Date; Event Location; Event Casualty;
Cluster2: Organization Involved; Person Involved

“Sport Event” Topic

Cluster1: Winner; Winning Score
Cluster2: Location, Date

“Organization” Topic

Cluster1: Founded Date; Founded Location; Founder
Cluster2: Headquarters; Number of Members

topic category is listed as follows.

The reason for the clustering of attributes of topic category is for the convenience of building a batch QA system. When a batch QA system is processing a series of questions under a topic, some of the questions in the series are mapped to the attributes of the topic and thus grouped together according to the attribute clusters. Then questions in the same group are processed together to obtain a maximum of answers co-occurrence. More details are given in section 4.2.

4 Experiment Setup and Evaluation

4.1 Baseline System

The baseline system is an automatic IE-driven (Information Extraction) QA system. We call it IE-driven because the main techniques used in the baseline system: surface pattern matching and N-gram proximity search need to be applied to NE-tagged (Named Entity) passages. The system architecture is illustrated in Figure-2. The

components indicated by dash lines are not included in the baseline system and they are added to the baseline system to build a batch QA system. As shown in the figure with light color, the two components are question classification and co-occurrence maximization. Both our baseline system and batch QA system didn’t utilize any pre-compiled knowledge base.

In the question analysis component, questions are classified by their syntactic structure and answer target. The answer targets are classified as named entity types. The retrieved documents are segmented into passages and filtered by topic keywords, question keywords and answer target.

The answer selection methods we used are surface text pattern matching and n-gram proximity search. We build a pattern learning system to automatically extract answer patterns from the TREC data and the web. These answer patterns are scored by their frequency, sorted by question type and represented as regular expressions with terms of “NP”, “VP”, “VPN”, “ADVP”, “be”, “in”, “of”, “on”, “by”, “at”, “which”, “when”, “where”, “who”, “,”, “-”, “(”. Some sample answer patterns of question type “when_be_np_vp” are listed as follows.

```
ADVP1 VP in <Date>([^\>]+?)</Date>
NP1.{1,15}VP.{1,30} in <Date>([^\>]+?)</Date>
NP1.{1,30} be VP in <Date>([^\>]+?)</Date>
NP1, which be VP in <Date>([^\>]+?)</Date>
VP NP1.{1,15} at .{1,15}<Date>([^\>]+?)</Date>
ADVP1.{1,80}NP1.{1,80}<Date>([^\>]+?)</Date>
NP1, VP in <Date>([^\>]+?)</Date>
NP1 of <Date>([^\>]+?)</Date>
NP1 be VP in <Date>([^\>]+?)</Date>
```

When applying these answer patterns to extract answer from candidate passages, the terms such as “NP”, “VP”, “VPN”, “ADVP” and “be” are replaced with the corresponding question terms. The replaced patterns can be matched directly to the candidate passages and answer candidate be extracted.

Some similar proximity search methods have been applied in document and passage retrieval in the previous research. We applied n-gram proximity search to answer questions whose answers can’t be extracted by surface text pattern matching. Around every named entity in the filtered candidate passages, question terms as well as topic terms are matched as n-grams. A question term is tokenized by word. We matched the longest possible sequence of tokenized word within the 100 word sliding window around the named entity. Once a sequence is matched, the corresponding word tokens are removed from the

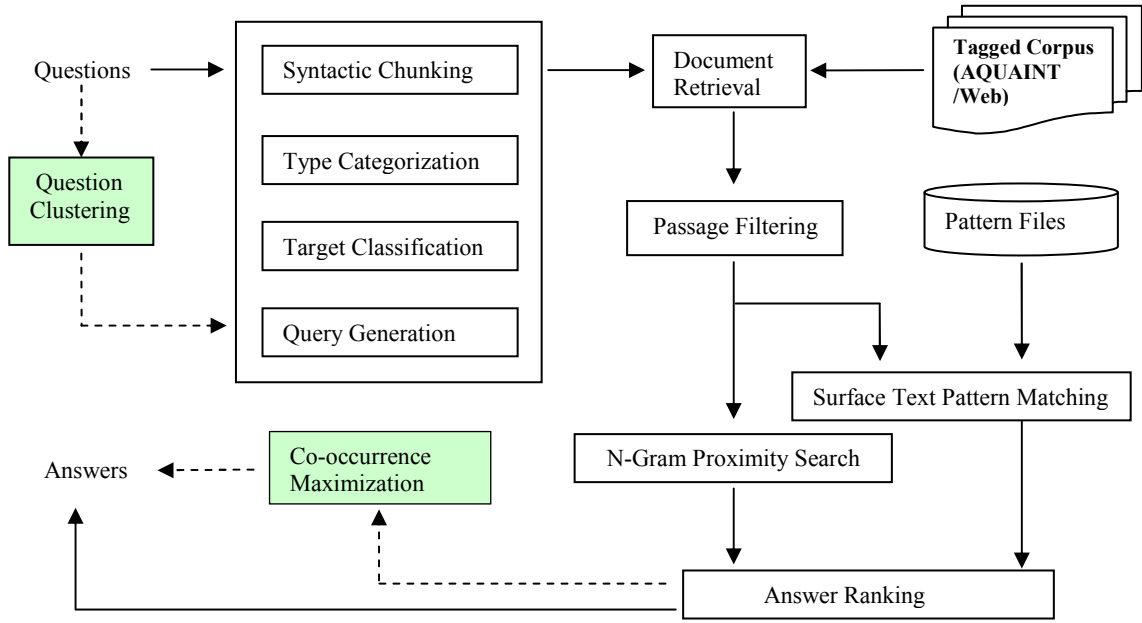


Figure-2 Baseline QA System & Batch QA System (dashed lines and light colored component)

token list and the same searching and matching is repeated until the token list is empty or no sequence of tokenized word can be matched. The named entity is scored by the average weighted distance score of question terms and topic terms.

Let $Num(t_i...t_j)$ denotes the number of all matched n-grams, $d(E, t_i...t_j)$ denotes the word distance between the named entity and the matched n-gram, $W1(t_i...t_j)$ denotes the topic weight of the matched n-gram, $W2(t_i...t_j)$ denotes the length weight of the matched n-gram. If $t_i...t_j$ contains topic terms or question verb phrase, 0.5 is assigned to $W1$, otherwise 1.0 is assigned. The value assigned to length weight $W2$ is determined by λ , the ratio value of matched n-gram length to question term length. How to assign $W2$ is illustrated as follows.

$$\begin{aligned} W2(t_i...t_j) &= 0.4 & \text{if } \lambda < 0.4; \\ W2(t_i...t_j) &= 0.6 & \text{if } 0.4 \leq \lambda \leq 0.6; \\ W2(t_i...t_j) &= 0.8 & \text{if } \lambda > 0.6; \\ W2(t_i...t_j) &= 0.9 & \text{if } \lambda > 0.75. \end{aligned}$$

The weighted distance score $D(E, QTerm)$ of the question term and the final score $S(E)$ of the named entity are calculated by the following equations.

$$D(E, QTerm) = \frac{\sum_{t_i...t_j} \frac{d(E, t_i...t_j) \times W1(t_i...t_j)}{W2(t_i...t_j)}}{Num(t_i...t_j)}$$

$$S(E) = \frac{\sum_i^N D(E, QTerm_i)}{N}$$

4.2 Batch QA System

The batch QA system is built from the baseline system and two added components: question classification and co-occurrence maximization. In a batch QA system, questions are classified before they are syntactically and semantically analyzed. The classification process consists of two steps: topic categorization and question mapping. Firstly the topic of the series questions is classified into appropriate topic category and then the questions can be mapped to the corresponding attribute and clustered according to the mapped attributes. Since the attributes of topic category is collected from frequently asked questions, there are some questions in the question series which can't be mapped to any attribute. These unmapped questions are processed individually.

The topic categorization is done by a Naïve Bayes classifier which employs features such as stemmed question terms and named entities in the question. The training data is a collection of 85 question series labeled as one of four topic categories: "Person", "Disaster Event", "Sport Event" and "Organization". The mapping of question to topic attribute is an example-based syntactic pattern matching and keywords matching.

The questions grouped together are processed as a question cluster. After the processing of answer selection and ranking, each question in the cluster gets top 10 scored candidate answers which forms an answer vector $A(a_1, ..., a_{10})$.

Suppose there are n questions in the cluster, the task of answer co-occurrence maximization is to retrieve a combination of n answers which has maximum pointwise mutual information (PMI). This combination is assumed to be the answers to the questions in the cluster.

There are a total of 10^n possible combinations among all the candidate answers. If the PMI of every combination should be calculated, it is computationally inefficient. Also, some combinations containing noisy information may have higher co-occurrence than the correct answer combination. For example, the correct answers combination to questions showed in figure-1 is “August 12; 118; Barents Sea”. However, there is also a combination of “Aug. 12, two; U.S.” which has higher pointwise mutual information due to the frequently occurred noisy information of “two U.S. submarines” and “two explosions in the area Aug. 12 at the time”.

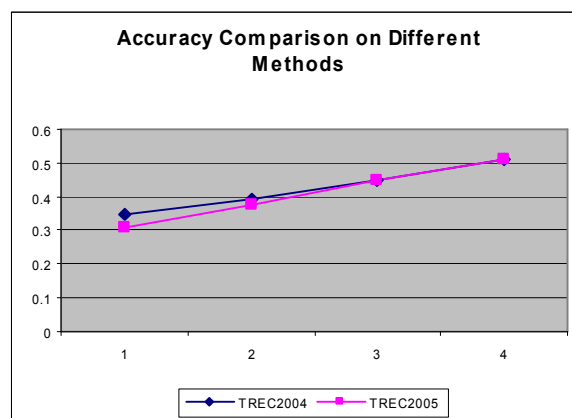
To reduce this negative effect brought by the noisy information, we started from the highest scored answer and put it in the final answer list. Then we added the answers one by one to the final answer list. The added answer has the highest PMI with the answers in the final answer list. It is important here to choose the first answer added to the final answer list correctly. Otherwise, the following added answers will be negatively affected. So in our batch QA system, a correct answer should be scored highest among all the answer candidates of the questions in the cluster. Although this can’t be always achieved, it can be approximated by setting higher threshold both in passage scoring and answer ranking. However, in the baseline system, passages are not scored. They are equally processed because we wanted to retrieve as many answer candidates as possible and answer candidates are ranked by their matching score and redundancy score.

4.3 Performance Evaluation

The data corpus we used is TREC QA data (AQUAINT Corpus). The test questions are TREC QA 2004 and TREC QA 2005 questions. Each topic is followed with a series of factoid questions. The number of questions selected from TREC 2004 collection is 230 and the number of question series is 65. The number of questions selected from TREC 2005 collection is 362 and the number of question series is 75.

We performed 4 different experiments: (1). Baseline system. (2). Batch QA system (Baseline system with co-occurrence maximization). (3). Baseline system with web supporting. (4). Batch

QA with web supporting. We introduced web supporting into the experiments because usually the information on the web tends to share more co-occurrence and redundancy which is also proved by our results.



Compared between the baseline system and batch system, the experiment results show that the overall accuracy score has been improved from 0.34 to 0.39 on TREC 2004 test questions and from 0.31 to 0.37 on TREC 2005 test questions. Compared between the baseline system and batch system with web supporting, the accuracy score can be improved up to 0.498. We also noticed that the average number of questions under each topic in TREC 2004 test questions is 3.538, which is significantly lower than the 4.8267 average in TREC 2005 questions series. This may explain why the improvement we obtained on TREC2004 data is not as significant as the improvement obtained on TREC 2005 questions.

The accuracy score of each TREC2005 question series is also calculated. Figure3-4 shows the comparisons between 4 different experiment methods. We also calculate the number of question series with accuracy increased, unchanged and decreased. It is also shown in the following table. (“+” means number of question series with accuracy increased, “=” unchanged and “-” decreased.)

TREC2005 Question Series (75 question series)	+	-	=
Baseline + Co-occurrence	25	5	45
Baseline + Web	40	2	33
Baseline + Co-occurrence + Web	49	2	24

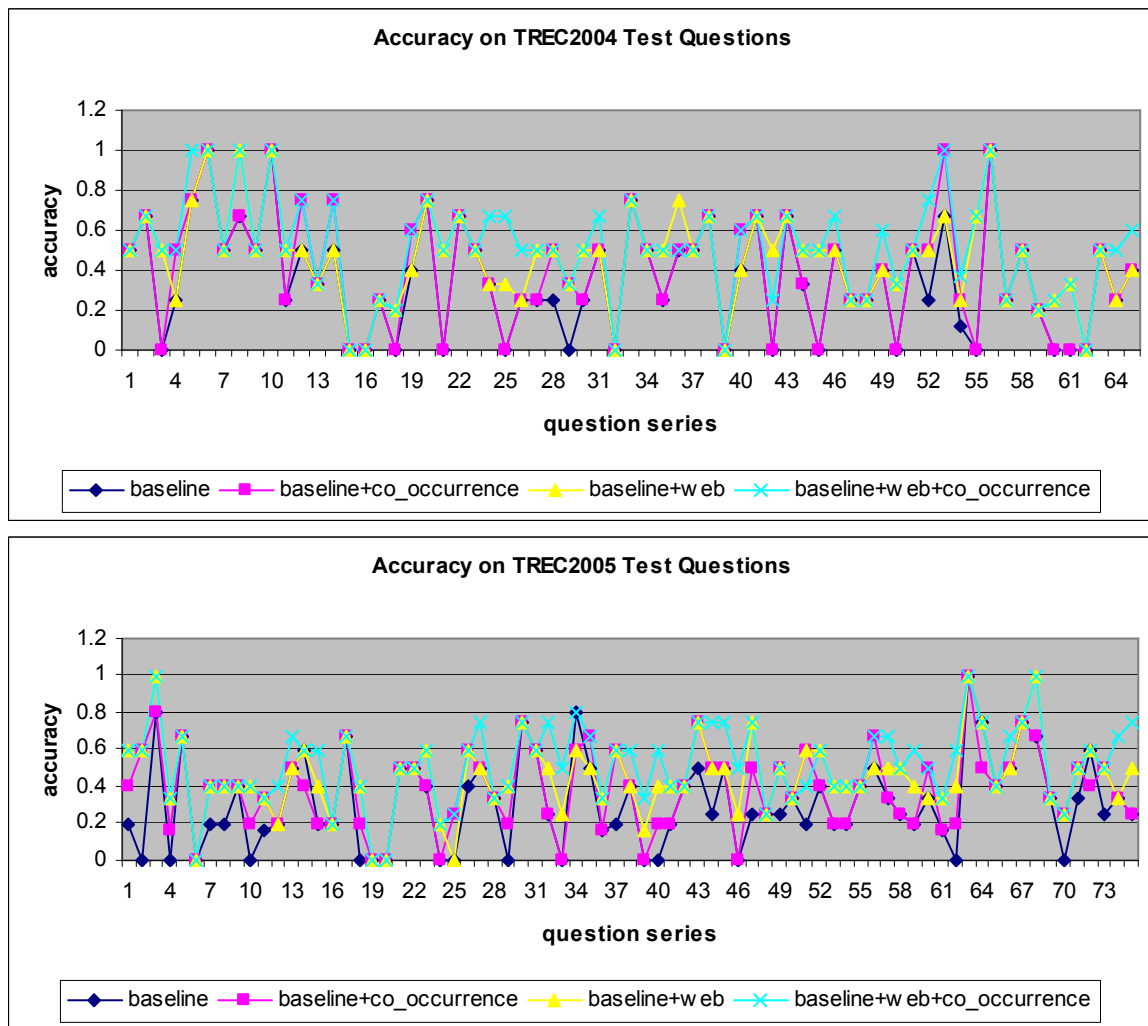


Figure 3-4 Comparison of TREC2004/2005 Question Series Accuracy

Some question series get unchanged accuracy because the questions can't be clustered according to our clustering template so that it can't utilize the co-occurrence of answers in the cluster. Some question series get decreased accuracy because the questions because the noisy information had even higher co-occurrence, the error occurred during the question clustering and the answers didn't show any co-relations in the retrieved passages at all. A deep and further error analysis is necessary for this answer co-occurrence maximization technique to be applied topic independently.

5 Discussion and Future Work

We have demonstrated that in a QA system, answering a series of inter-related questions can be improved by grouping the questions by expected co-occurrence of answers in text. The improvement can be made without exploiting the pre-compiled knowledge base.

Although our system can cluster frequently asked questions on topics of "Events", "Persons" and "Organizations", there are still some highly related questions which can't be clustered by our method. Here are some examples.

Topic Carlos the Jackal

1. When was he captured?
2. Where was he captured?

Topic boxer Floyd Patterson

1. When did he win the title?
2. How old was he when he won the title?
3. Who did he beat to win the title?

To cluster these questions, we plan to utilize event detection techniques and set up an event topic "Carlos the Jackal captured" during the answering process, which will make it easier to cluster "When was the Carlos the Jackal captured?" and "Where was the Carlos the Jackal captured?"

Can this answers co-occurrence maximization approach be applied to improve QA performance

on single questions (i.e. 1-series)? As suggested in the reference paper (Chu-Carroll and Prager), we may be able to add related (unasked) questions to form a cluster around the single question. Another open issue is what kind of effect will this technique bring to answering series of “list” questions, i.e., where each question expects a list of items as answer. As we know that the answers of some “list” questions have pretty high co-occurrence while others don’t have co-occurrence at all. Future work involves experiments conducted on these aspects.

Acknowledgement

The Authors wish to thank BBN for the use of NE tagging software *IdentiFinder*, CIIR at University of Massachusetts for the use of *Inquery* search engine, Stanford University NLP group for the use of Stanford parser. Thanks also to the anonymous reviewers for their helpful comments.

References

- Chu-Carroll, J., J. Prager, C. Welty, K. Czuba and D. Ferrucci. “A Multi-Strategy and Multi-Source Approach to Question Answering”, In *Proceedings of the 11th TREC*, 2003.
- Cui, H., K. Li, R. Sun, T.-S. Chua and M.-Y. Kan. “National University of Singapore at the TREC 13 Question Answering Main Task”. In *Proceedings of the 13th TREC*, 2005.
- Han, K.-S., H. Chung, S.-B. Kim, Y.-I. Song, J.-Y. Lee, and H.-C. Rim. “Korea University Question Answering System at TREC 2004”. In *Proceedings of the 13th TREC*, 2005.
- Harabagiu, S., D. Moldovan, C. Clark, M. Bowden, J. Williams and J. Bensley. “Answer Mining by Combining Extraction Techniques with Abductive Reasoning”. In *Proceedings of 12th TREC*, 2004.
- Hovy, E. L. Gerber, U. Hermjakob, M. Junk and C.-Y. Lin. “Question Answering in Webclopedia”. In *Proceedings of the 9th TREC*, 2001.
- Lin, J., D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz and D. R. Karger. “The Role of Context in Question Answering Systems”. In *CHI* 2003.
- Katz, B. and J. Lin. “Selectively Using Relations to Improve Precision in Question Answering”. In *Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering*. 2003.
- Moldovan, D. and V. Rus. “Logical Form Transformation of WordNet and its Applicability to Question Answering”. In *Proceedings of the ACL*, 2001.
- Monz, C. “Minimal Span Weighting Retrieval for Question Answering” In *Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering*. 2004.
- Prager, J., E. Brown, A. Coden and D. Radev. “Question-Answering by Predictive Annotation”. In *Proceedings of SIGIR 2000*, pp. 184-191. 2000.
- Prager, J., J. Chu-Carroll and K. Czuba. “Question Answering Using Constraint Satisfaction: QA-By-Dossier-With-Constraints”. In *Proceedings of the 42nd ACL*. 2004.
- Ravichandran, D. and E. Hovy. “Learning Surface Text Patterns for a Question Answering System”. In *Proceedings of 40th ACL*. 2002.
- Soubbotin, M. and S. Soubbotin. “Patterns of Potential Answer Expressions as Clues to the Right Answers”. In *Proceedings of 11th TREC*. 2003.
- Voorhees, E. “Using Question Series to Evaluate Question Answering System Effectiveness”. In *Proceedings of HLT 2005*. 2005.

Author Index

- Abdalla, Rashid M., 921
Abekawa, Takeshi, 833
Adler, Meni, 665, 689
Agrawal, Saumya, 649
Al-Onaizan, Yaser, 529
Ayan, Necip Fazil, 9
- Bangalore, Srinivas, 201
Barrett, Leon, 433
Barzilay, Regina, 25
Bergsma, Shane, 33
Bigham, Jeffrey, 809
Blache, Philippe, 57
Blake, Catherine, 601
Blunsom, Phil, 65
Bod, Rens, 865
Bode, Andreas, 513
Bollegala, Danushka, 385
Bouchard-Côté, Alexandre, 761
Brew, Chris, 49
Brockett, Chris, 249
Brody, Samuel, 97
Brunet-Manquat, Francis, 937
By, Tomas, 289
- Cahill, Aoife, 497, 1033
Casillas, Arantza, 1145
Cerccone, Nick, 481
Çetinoğlu, Özlem, 153
Chan, Yee Seng, 89
Charniak, Eugene, 337
Chatterjee, Niladri, 649
Chen, Hsin-Hsi, 1009
Chen, Jinxiu, 129
Chen, Wan, 585
Chen, Yi, 1081
Chiocchetti, Elena, 937
Chu-Carroll, Jennifer, 1073
Clark, Stephen, 697
Clarke, James, 377
Cohen-Sygal, Yael, 145
Cohn, Trevor, 65
Collier, Nigel, 345
Costello, Fintan J., 745
- Curran, James R., 361, 697
- Dagan, Ido, 449
Daumé III, Hal, 305
Davidov, Dmitry, 297
de La Clergerie, Éric, 329
Demner-Fushman, Dina, 841
DeNeefe, Steve, 961
Di Fabbri, Giuseppe, 201
Dolan, William B., 249
Dorr, Bonnie J., 9, 161, 945
Dubey, Amit, 417
Duboue, Pablo, 1073
- Eisner, Jason, 569
Elhadad, Michael, 665, 689
Ellison, T. Mark, 273
- Feinstein, Daniel, 1089
Feng, Zhe, 561
Ferret, Olivier, 281
Florian, Radu, 473
Foth, Kilian A., 289, 321
Frampton, Matthew, 185
Fraser, Alexander, 769
Fresno, Víctor, 1145
Frunza, Oana, 441
Fujii, Atsushi, 657
- Galley, Michel, 961
Gamon, Michael, 249
Gao, Jianfeng, 225, 489
Gerdes, Kim, 1097
Giuglea, Ana-Maria, 929
Glickman, Oren, 449
Gliozzo, Alfio, 449, 553
Goldberg, Yoav, 689
Goldwater, Sharon, 673
Gorman, James, 361
Graehl, Jonathan, 961
Greiner, Russell, 209
Griffiths, Thomas L., 17, 673
Gu, Zhenmei, 481
- Habash, Nizar, 1, 681

Haghighi, Aria, 881
 Hagiwara, Masato, 353
 Hahn, Udo, 785
 Hajič, Jan, 945
 Hale, John, 161
 Hall, Johan, 257
 Halpin, Harry, 857
 Harabagiu, Sanda, 905
 Harper, Mary, 161
 Hauptmann, Alexander, 1057
 Hemforth, Barbara, 57
 Hickl, Andrew, 905
 Higashinaka, Ryuichiro, 265
 Hobbs, Jerry R., 393
 Hockenmaier, Julia, 505
 Hulth, Anette, 537

 Iida, Ryu, 625
 Inagaki, Yasuyoshi, 169
 Inkpen, Diana, 441
 Inui, Kentaro, 625
 Inui, Takashi, 1153
 Ishikawa, Tetsuya, 657
 Ishioka, Tsunenori, 233
 Ishizuka, Mitsuru, 385
 Isozaki, Hideki, 217, 617, 777
 Isu, Naoki, 241

 Jain, Alpa, 809
 Ji, Donghong, 129
 Jiao, Feng, 209
 Jin, Yun, 641
 Jing, Hongyan, 473
 Johnson, Mark, 337, 673
 Judge, John, 497
 Jurafsky, Daniel, 801

 Kahane, Sylvain, 137, 1097
 Kambhatla, Nanda, 473
 Kameda, Masayuki, 233
 Kang, Bo-yeong, 641
 Kashioka, Hideki, 169
 Kate, Rohit J., 913
 Kawai, Atsuo, 241
 Kelleher, John D., 745, 1041
 Keller, Frank, 417
 Khaltar, Badam-Osor, 657
 Kirby, Simon, 273
 Klakow, Dietrich, 889
 Klein, Dan, 433, 761, 881
 Klementiev, Alexandre, 817
 Knight, Kevin, 961

 Koller, Alexander, 409
 Körding, Konrad P., 17
 Korhonen, Anna, 345
 Krasnyanskaya, Anna, 161
 Krishnan, Vijay, 1121
 Kruijff, Geert-Jan M., 745, 1041
 Krymolowski, Yuval, 345
 Kuo, Jin-Shea, 1129
 Kurata, Gakuto, 729

 Lapata, Mirella, 97, 377, 1161
 Lease, Matthew, 161
 Lee, Chi-Hoon, 209
 Lee, Chong Min, 753
 Lemon, Oliver, 185
 Li, Guohua, 1001
 Li, Haizhou, 1129
 Li, Jianfeng, 1001
 Li, Jingyang, 545
 Li, Mu, 1025
 Li, Qing, 641
 Li, Sheng, 457
 Li, Wenjie, 369, 993
 Liang, Percy, 761
 Lifchits, Andrei, 809
 Lin, Dekang, 33, 809
 Lin, Jimmy, 841, 945
 Lin, Ming-Shun, 1009
 Lin, Shouxun, 521, 609
 Lin, Wei-Hao, 1057
 Litman, Diane J., 193
 Liu, Qun, 521, 609
 Liu, Ting, 457
 Liu, Yang, 161
 Liu, Yang, 609
 Lu, Qin, 369
 Lu, Zhimao, 457
 Lv, Yuanhua, 585

 Malik, M.G. Abbas, 1137
 Malioutov, Igor, 25
 Mani, Inderjeet, 753
 Manning, Christopher D., 1121
 Marcu, Daniel, 81, 305, 769, 961, 1105
 Marmorshtein, Efrat, 449
 Martínez, Raquel, 1145
 Maruyama, Takehiko, 169
 Masuda, Katsuya, 1017
 Matsubara, Shigeki, 169
 Matsumoto, Yuji, 625, 705
 McClosky, David, 337
 McDermott, Erik, 217

Megyesi, Beáta B., 537
Melamed, I. Dan, 873, 977
Menzel, Wolfgang, 289, 321
Mihalcea, Rada, 1065
Mitamura, Teruko, 425
Miyao, Yusuke, 465, 1017
Moldovan, Dan, 897
Montalvo, Soto, 1145
Mooney, Raymond J., 913
Moore, Johanna D., 857
Moore, Robert C., 513
Mori, Shinsuke, 729
Morihiro, Koichiro, 241
Moschitti, Alessandro, 401, 929
Mulkar, Rutu, 393
Munteanu, Dragos Stefan, 81
Murray, G. Craig, 945
Myaeng, Sung-Hyon, 641

Nagata, Masaaki, 713
Nagata, Ryo, 241
Nakagawa, Tetsuji, 705
Nakatsu, Crystal, 1113
Navigli, Roberto, 97, 105
Ng, Andrew Y., 801
Ng, Hwee Tou, 89
Nie, Jian-Yun, 585
Nilsson, Jens, 257
Ninomiya, Takashi, 1017
Niu, Cheng, 489
Niu, Zhengyu, 129
Nivre, Joakim, 257
Noro, Taichi, 1153
Novischi, Adrian, 897

Oflazer, Kemal, 153
Ogawa, Yasuhiro, 353
Ohashi, Kazuteru, 713
Ohno, Tomohiro, 169
Ohta, Tomoko, 1017
Okanohara, Daisuke, 465
Okazaki, Naoaki, 385
Okumura, Manabu, 833, 1153
Osborne, Miles, 969

Padó, Sebastian, 1161
Pan, Feng, 393
Pantel, Patrick, 113, 793
Papineni, Kishore, 529
Park, Jihyun, 49
Park, Se-Young, 633
Park, Seong-Bae, 633

Paşca, Marius, 809
Pazienza, Maria Teresa, 849
Pecina, Pavel, 945
Pennacchiotti, Marco, 113, 793, 849
Petrov, Slav, 433
Prager, John, 1073
Prasad, Rashmi, 265
Purver, Matthew, 17
Pustejovsky, James, 753

Rambow, Owen, 681
Rappoport, Ari, 297
Rauzy, Stéphane, 57
Ren, Dengjun, 1001
Roark, Brian, 161
Rotaru, Mihai, 193
Roth, Dan, 817
Roy, Shourya, 737

Sadat, Fatiha, 1
Sagae, Kenji, 425
Sagot, Benoît, 329
Saito, Kuniko, 713
Sarikaya, Ruhi, 577
Schmid, Helmut, 177
Schoorlans, Dale, 209
Sérasset, Gilles, 937
Seretan, Violeta, 953
Shafran, Izhak, 161
Shen, Dan, 889
Shi, Lei, 489
Smith, Noah A., 569
Snover, Matthew, 161
Snow, Rion, 801
Sorensen, Jeffrey S., 577
Soricut, Radu, 1105
Sproat, Richard, 73
Stent, Amanda, 201
Stewart, Robin, 161
Strapparava, Carlo, 449, 553
Strzalkowski, Tomek, 1169
Sturt, Patrick, 417
Su, Jian, 41, 121, 825
Subramaniam, L Venkata, 737
Sudoh, Katsuhito, 617
Sun, Le, 585
Sun, Maosong, 545
Suzuki, Hisami, 225, 1049
Suzuki, Jun, 217

Tae, Yoon-Shik, 633
Takamura, Hiroya, 1153

Takuma, Daisuke, 729
Talbot, David, 969
Tan, Chew Lim, 41, 129
Tao, Tao, 73
Taskar, Ben, 761
Teh, Yee Whye, 985
Tenenbaum, Joshua B., 17
Teufel, Simone, 921
Thater, Stefan, 409
Thayer, Ignacio, 961
Thibaux, Romain, 433
Tillmann, Christoph, 721
Toutanova, Kristina, 1049
Toyama, Katsuhiko, 353
Tsujii, Jun'ichi, 465, 1017
Tsukada, Hajime, 617, 777
Tsuruoka, Yoshimasa, 465, 1017
Turian, Joseph, 873
Turney, Peter D., 313

Vadas, David, 697
van Genabith, Josef, 497, 1033
Verhagen, Marc, 753

Walker, Marilyn A., 265
Wang, Haifeng, 457, 593, 1001
Wang, Mengqiu, 425
Wang, Shaojun, 209
Wang, Shilong, 1081
Wang, Wei, 961
Watanabe, Taro, 777
Waxmonsky, Sonjia, 977
Wehrli, Eric, 953
Wei, Yu-Chuan, 1009
Wellington, Benjamin, 977
Wellner, Ben, 753
Weng, Fuliang, 561
Wermter, Joachim, 785
White, Michael, 1113
Wiebe, Janyce, 1065
Wintner, Shuly, 145, 1089
Wong, Kam-Fai, 993
Wu, Min, 1169
Wu, Mingli, 369

Xia, Yunqing, 993
Xiong, Deyi, 521
Xu, Wei, 369

Yamamoto, Kazuhide, 713
Yang, Xiaofeng, 41
Yang, Ying-Kuei, 1129
Yao, Jianmin, 457

Yih, Wen-tau, 513
Yoshida, Kazuhiro, 1017
Yu, Bin, 225
Yuan, Chunfa, 369
Yung, Lisa, 161

Zanzotto, Fabio Massimo, 401, 849
Zhai, ChengXiang, 73
Zhang, Jie, 825
Zhang, Junlin, 585
Zhang, Min, 121, 825
Zhang, Qi, 561
Zhang, Tong, 721
Zhang, Wei, 585
Zhang, Xian, 545
Zhang, Yang, 1025
Zhou, GuoDong, 121, 825
Zhou, Ming, 489, 1025, 1081
Zhu, Jiang, 593
Zhu, Muhua, 1025
Zitouni, Imed, 473, 577
Zock, Michael, 281