Strictly Lexical Dependency Parsing

Qin Iris Wang and **Dale Schuurmans**
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada, T6G 2E8
{wqin,dale}@cs.ualberta.ca

Dekang Lin
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, California, USA, 94043
lindek@google.com

Abstract

We present a strictly lexical parsing model where all the parameters are based on the words. This model does not rely on part-of-speech tags or grammatical categories. It maximizes the conditional probability of the parse tree given the sentence. This is in contrast with most previous models that compute the joint probability of the parse tree and the sentence. Although the maximization of joint and conditional probabilities are theoretically equivalent, the conditional model allows us to use distributional word similarity to generalize the observed frequency counts in the training corpus. Our experiments with the Chinese Treebank show that the accuracy of the conditional model is 13.6% higher than the joint model and that the strictly lexicalized conditional model outperforms the corresponding unlexicalized model based on part-of-speech tags.

1 Introduction

There has been a great deal of progress in statistical parsing in the past decade (Collins, 1996; Collins, 1997; Chaniak, 2000). A common characteristic of these parsers is their use of lexicalized statistics. However, it was discovered recently that bi-lexical statistics (parameters that involve two words) actually played much smaller role than previously believed. It was found in (Gildea, 2001) that the removal of bi-lexical statistics from a state-of-the-art PCFG parser resulted very small change in the output. Bikel (2004) observed that the bi-lexical statistics accounted for only 1.49% of the bigram statistics used by the parser. When considering only bigram statistics involved in the highest probability parse, this percentage becomes 28.8%. However, even when the bi-lexical statistics do get used, they are remarkably similar to their back-off values using part-of-speech tags. Therefore, the utility of bi-lexical statistics becomes rather questionable. Klein and Manning (2003) presented an unlexicalized parser that eliminated all lexicalized parameters. Its performance was close to the state-of-the-art lexicalized parsers.

We present a statistical dependency parser that represents the other end of spectrum where all statistical parameters are lexical and the parser does not require part-of-speech tags or grammatical categories. We call this strictly lexicalized parsing.

A part-of-speech lexicon has always been considered to be a necessary component in any natural language parser. This is true in early rule-based as well as modern statistical parsers and in dependency parsers as well as constituency parsers. The need for part-of-speech tags arises from the sparseness of natural language data. They provide generalizations of words that are critical for parsers to deal with the sparseness. Words belonging to the same part-of-speech are expected to have the same syntactic behavior.

Instead of part-of-speech tags, we rely on distributional word similarities computed automatically from a large unannotated text corpus. One of the benefits of strictly lexicalized parsing is that

Proceedings of the Ninth International Workshop on Parsing Technologies (IWPT), pages 152–159,
Vancouver, October 2005.

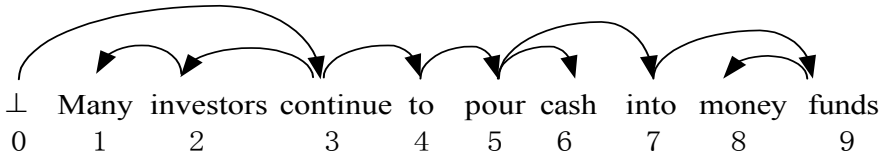

Figure 1. An Example Dependency Tree.

the parser can be trained with a treebank that only contains the dependency relationships between words. The annotators do not need to annotate parts-of-speech or non-terminal symbols (they don't even have to know about them), making the construction of the treebank easier.

Strictly lexicalized parsing is especially beneficial for languages such as Chinese, where parts-of-speech are not as clearly defined as English. In Chinese, clear indicators of a word's part-of-speech such as suffixes -ment, -ous or function words such as the, are largely absent. In fact, monolingual Chinese dictionaries that are mainly intended for native speakers almost never contain part-of-speech information.

In the next section, we present a method for modeling the probabilities of dependency trees. Section 3 applies similarity-based smoothing to the probability model to deal with data sparseness. We then present experimental results with the Chinese Treebank in Section 4 and discuss related work in Section 5.

2 A Probabilistic Dependency Model

Let S be a sentence. The dependency structure T of S is a directed tree connecting the words in S. Each link in the tree represents a dependency relationship between two words, known as the head and the modifier. The direction of the link is from the head to the modifier. We add an artificial root node ($\perp$) at the beginning of each sentence and a dependency link from $\perp$ to the head of the sentence so that the head of the sentence can be treated in the same way as other words. Figure 1 shows an example dependency tree.

We denote a dependency link l by a triple (u, v, d), where u and v are the indices ($u < v$) of the words connected by l, and d specifies the direction of the link l. The value of d is either L or R. If $d = L$, v is the index of the head word; otherwise, u is the index of the head word.

Dependency trees are typically assumed to be projective (without crossing arcs), which means that if there is an arc from h to m, h is an ancestor of all the words between h and m. Let $F(S)$ be the set of possible directed, projective trees spanning on S. The parsing problem is to find

$$\arg\max_{T \in F(S)} P(T \mid S)$$

Generative parsing models are usually defined recursively from top down, even though the decoders (parsers) for such models almost always take a bottom-up approach. The model proposed here is a bottom-up one. Like previous approaches, we decompose the generation of a parse tree into a sequence of steps and define the probability of each step. The probability of the tree is simply the product of the probabilities of the steps involved in the generation process. This scheme requires that different sequences of steps must not lead to the same tree. We achieve this by defining a canonical ordering of the links in a dependency tree. Each generation step corresponds to the construction of a dependency link in the canonical order.

Given two dependency links l and l' with the heads being h and h' and the modifiers being m and m', respectively, the order between l and l' are determined as follows:

- If $h \neq h'$ and there is a directed path from one (say h) to the other (say h'), then l' precedes l.
- If $h \neq h'$ and there does not exist a directed path between h and h', the order between l and l' is determined by the order of h and h' in the sentence (h precedes $h' \Rightarrow l$ precedes l').
- If $h = h'$ and the modifiers m and m' are on different sides of h, the link with modifier on the right precedes the other.
- If $h = h'$ and the modifiers m and m' are on the same side of the head h, the link with its modifier closer to h precedes the other one.

For example, the canonical order of the links in the dependency tree in Figure 1 is: (1, 2, L), (5, 6, R), (8, 9, L), (7, 9, R), (5, 7, R), (4, 5, R), (3, 4, R), (2, 3, L), (0, 3, L).

The generation process according to the canonical order is similar to the head outward generation process in (Collins, 1999), except that it is bottom-up whereas Collins' models are top-down.

Suppose the dependency tree T is constructed in steps G_1, …, G_N in the canonical order of the dependency links, where N is the number of words in the sentence. We can compute the probability of T as follows:

$$\begin{aligned} & P(T \mid S) \\ &= P(G_1, G_2, ..., G_N \mid S) \\ &= \prod_{i=1}^{N} P(G_i \mid S, G_1, ..., G_{i-1}) \end{aligned}$$

Following (Klein and Manning, 2004), we require that the creation of a dependency link from head h to modifier m be preceded by placing a left STOP and a right STOP around the modifier m and ¬STOP between h and m.

Let E_w^L (and E_w^R) denote the event that there are no more modifiers on the left (and right) of a word w. Suppose the dependency link created in the step i is (u, v, d). If $d = L$, G_i is the conjunction of the four events: E_u^R, E_u^L, $\neg E_v^L$ and $link_L(u, v)$. If $d = R$, G_i consists of four events: E_v^L, E_v^R, $\neg E_u^R$ and $link_R(u, v)$.

The event G_i is conditioned on $S, G_1, ..., G_{i-1}$, which are the words in the sentence and a forest of trees constructed up to step i-1. Let C_w^L (and C_w^R) be the number of modifiers of w on its left (and right). We make the following independence assumptions:

- Whether there is any more modifier of w on the d side depends only on the number of modifiers already found on the d side of w. That is, E_w^d depends only on w and C_w^d.
- Whether there is a dependency link from a word h to another word m depends only on the words h and m and the number of modifiers of h between m and h. That is,
  - $link_R(u,v)$ depends only on u, v, and C_u^R.
  - $link_L(u,v)$ depends only on u, v, and C_v^L.

Suppose G_i corresponds to a dependency link (u, v, L). The probability $P(G_i \mid S, G_1, ..., G_{i-1})$ can be computed as:

$$\begin{aligned} & P(G_i \mid S, G_1, ..., G_{i-1}) \\ &= P(E_u^L, E_u^R, \neg E_v^L, link_L(u,v) \mid S, G_1, ..., G_{i-1}) \\ &= P(E_u^L \mid u, C_u^L) \times P(E_u^R \mid u, C_u^R) \times \\ &\quad (1 - P(E_v^L \mid v, C_v^L)) \times P(link_L(u,v) \mid u, v, C_v^L) \end{aligned}$$

The events E_w^R and E_w^L correspond to the STOP events in (Collins, 1999) and (Klein and Manning, 2004). They are crucial for modeling the number of dependents. Without them, the parse trees often contain some 'obvious' errors, such as determiners taking arguments, or prepositions having arguments on their left (instead of right).

Our model requires three types of parameters:

- $P(E_w^d \mid w, C_w^d)$, where w is a word, d is a direction (left or right). This is the probability of a STOP after taking C_w^d modifiers on the d side.
- $P(link_R(u,v) \mid u, v, C_u^R)$ is the probability of v being the $(C_u^R + 1)$'th modifier of u on the right.
- $P(link_L(u,v) \mid u, v, C_v^L)$ is the probability of u being the $(C_v^L + 1)$'th modifier of v on the left.

The Maximum Likelihood estimations of these parameters can be obtained from the frequency counts in the training corpus:

- C(w, c, d): the frequency count of w with c modifiers on the d side.
- C(u, v, c, d): If $d = L$, this is the frequency count words u and v co-occurring in a sentence and v has c modifiers between itself and u. If $d = R$, this is the frequency count words u and v co-occurring in a sentence and u has c modifiers between itself and v.
- K(u, v, c, d): similar to C(u, v, c, d) with an additional constraint that $link_d(u, v)$ is true.

$$P\left(E_w^d \mid w, C_w^d\right) = \frac{C(w,c,d)}{\sum_{c' \geq c} C(w,c',d)}$$, where $c = C_w^d$;

$$P\left(link_R(u,v) \mid u, v, C_u^R\right) = \frac{K(u,v,c,R)}{C(u,v,c,R)},$$

where $c = C_u^R$;

$$P\left(link_L(u,v) \mid u, v, C_v^L\right) = \frac{K(u,v,c,L)}{C(u,v,c,L)},$$

where $c = C_v^L$.

We compute the probability of the tree conditioned on the words. All parameters in our model are conditional probabilities where the left sides of the conditioning bar are binary variables. In contrast, most previous approaches compute joint probability of the tree and the words in the tree. Many of their model parameters consist of the probability of a word in a given context.

We use a dynamic programming algorithm similar to chart parsing as the decoder for this model. The algorithm builds a packed parse forest from bottom up in the canonical order of the parser trees. It attaches all the right children before attaching the left ones to maintain the canonical order as required by our model.

3 Similarity-based Smoothing

3.1 Distributional Word Similarity

Words that tend to appear in the same contexts tend to have similar meanings. This is known as the Distributional Hypothesis in linguistics (Harris, 1968). For example, the words test and exam are similar because both of them follow verbs such as administer, cancel, cheat on, conduct, ... and both of them can be preceded by adjectives such as academic, comprehensive, diagnostic, difficult, ...

Many methods have been proposed to compute distributional similarity between words (Hindle, 1990; Pereira et al., 1993; Grefenstette, 1994; 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.

We define the features of a word *w* to be the set of words that occurred within a small context window of *w* in a large corpus. The context window of an instance of *w* consists of the closest non-stop-word on each side of *w* and the stop-words in between. In our experiments, the set of stop-words are defined as the top 100 most frequent words in the corpus. The value of a feature *w′* is defined as the point-wise mutual information between the *w′* and *w*:

$$PMI(w,w') = -\log\left(\frac{P(w,w')}{P(w)P(w')}\right)$$

where *P*(*w*, *w*') is the probability of *w* and *w*' co-occur in a context window.

The similarity between two vectors is computed as the cosine of the angle between the vectors. The following are the top similar words for the word *keystone* obtained from the English Gigaword Corpus:

centrepiece 0.28, figment 0.27, fulcrum 0.21, culmination 0.20, albatross 0.19, bane 0.19, pariahs 0.18, lifeblood 0.18, crux 0.18, redoubling 0.17, apotheosis 0.17, cornerstones 0.17, perpetuation 0.16, forerunners 0.16, shirking 0.16, cornerstone 0.16, birthright 0.15, hallmark 0.15, centerpiece 0.15, evidenced 0.15, germane 0.15, gist 0.14, reassessing 0.14, engrossed 0.14, Thorn 0.14, biding 0.14, narrowness 0.14, linchpin 0.14, enamored 0.14, formalised 0.14, tenths 0.13, testament 0.13, certainties 0.13, forerunner 0.13, re-evaluating 0.13, antithetical 0.12, extinct 0.12, rarest 0.12, imperiled 0.12, remiss 0.12, hindrance 0.12, detriment 0.12, prouder 0.12, upshot 0.12, cosponsor 0.12, hiccups 0.12, premised 0.12, perversion 0.12, destabilisation 0.12, prefaced 0.11, ……

3.2 Similarity-based Smoothing

The parameters in our model consist of conditional probabilities P(*E*|*C*) where *E* is the binary variable $link_d(u, v)$ or E_w^d and the context *C* is either $\left[w, C_w^d\right]$ or $\left[u, v, C_w^d\right]$, which involves one or two words in the input sentence. Due to the sparseness of natural language data, the contexts observed in the training data only covers a tiny fraction of the contexts whose probability distribution are needed during parsing. The standard approach is to back off the probability to word classes (such as part-of-speech tags). We have taken a different approach. We search in the train-

ing data to find a set of similar contexts to C and estimate the probability of E based on its probabilities in the similar contexts that are observed in the training corpus.

Similarity-based smoothing was used in (Dagan et al., 1999) to estimate word co-occurrence probabilities. Their method performed almost 40% better than the more commonly used back-off method. Unfortunately, similarity-based smoothing has not been successfully applied to statistical parsing up to now.

In (Dagan et al., 1999), the bigram probability $P(w_2|w_1)$ is computed as the weighted average of the conditional probability of w_2 given similar words of w_1.

$$P_{SIM}(w_2 \mid w_1) = \sum_{w'_1 \in S(w_1)} \frac{sim(w_1, w'_1)}{norm(w_1)} P_{MLE}(w_2 \mid w'_1)$$

where $sim(w_1, w'_1)$ denotes the similarity (or an increasing function of the similarity) between w_1 and w'_1, $S(w_1)$ denote the set of words that are most similar to w_1 and $norm(w_1)$ is the normalization factor $norm(w_1) = \sum_{w'_1 \in S(w_1)} sim(w_1, w'_1)$.

The underlying assumption of this smoothing scheme is that a word is more likely to occur after w_1 if it tends to occur after similar words of w_1.

We make a similar assumption: the probability $P(E|C)$ of event E given the context C is computed as the weight average of $P(E|C')$ where C' is a similar context of C and is attested in the training corpus:

$$P_{SIM}(E \mid C) = \sum_{C' \in S(C) \cap O} \frac{sim(C, C')}{norm(C)} P_{MLE}(E \mid C')$$

where $S(C)$ is the set of top-K most similar contexts of C (in the experiments reported in this paper, K = 50); O is the set of contexts observed in the training corpus, $sim(C,C')$ is the similarity between two contexts and $norm(C)$ is the normalization factor.

In our model, a context is either $\left[w, C_w^d\right]$ or $\left[u, v, C_w^d\right]$. Their similar contexts are defined as:

$$S\left(\left[w, C_w^d\right]\right) = \left\{\left[w', C_{w'}^d\right] \middle| w' \in S(w)\right\}$$
$$S\left(\left[u, v, C_w^d\right]\right) = \left\{\left[u', v', C_w^d\right] \middle| u' \in S(u), v' \in S(v)\right\}$$

where $S(w)$ is the set of top-K similar words of w (K = 50).

Since all contexts used in our model contain at least one word, we compute the similarity between two contexts, sim(C, C'), as the geometric average of the similarities between corresponding words:

$$sim\left(\left[w, C_w^d\right], \left[w', C_{w'}^d\right]\right) = sim(w, w')$$
$$sim\left(\left[u, v, C_w^d\right], \left[u', v', C_{w'}^d\right]\right) = \sqrt{sim(u, u') \times sim(v, v')}$$

Similarity-smoothed probability is only necessary when the frequency count of the context C in the training corpus is low. We therefore compute

$$P(E \mid C) = \alpha\, P_{MLE}(E \mid C) + (1 - \alpha)\, P_{SIM}(E \mid C)$$

where the smoothing factor $\alpha = \frac{|C| + 1}{|C| + 5}$ and $|C|$ is the frequency count of the context C in the training data.

A difference between similarity-based smoothing in (Dagan et al., 1999) and our approach is that our model only computes probability distributions of binary variables. Words only appear as parts of contexts on the right side of the conditioning bar. This has two important implications. Firstly, when a context contains two words, we are able to use the cross product of the similar words, whereas (Dagan et al., 1999) can only use the similar words of one of the words. This turns out to have significant impact on the performance (see Section 4).

Secondly, in (Dagan et al., 1999), the distribution $P(\bullet|w'_1)$ may itself be sparsely observed. When $P_{MLE}(w_2 \mid w'_1)$ is 0, it is often due to data sparseness. Their smoothing scheme therefore tends to under-estimate the probability values. This problem is avoided in our approach. If a context did not occur in the training data, we do not include it in the average. If it did occur, the Maximum Likelihood estimation is reasonably accurate even if the context only occurred a few times, since the entropy of the probability distribution is upper-bounded by log 2.

4 Experimental Results

We experimented with our parser on the Chinese Treebank (CTB) 3.0. We used the same data split as (Bikel, 2004): Sections 1-270 and 400-931 as

the training set, Sections 271-300 as testing and Sections 301-325 as the development set. The CTB contains constituency trees. We converted them to dependency trees using the same method and the head table as (Bikel, 2004). Parsing Chinese generally involve segmentation as a preprocessing step. We used the gold standard segmentation in the CTB.

The distributional similarities between the Chinese words are computed using the Chinese Gigaword corpus. We did not segment the Chinese corpus when computing the word similarity.

We measure the quality of the parser by the undirected accuracy, which is defined as the number of correct undirected dependency links divided by the total number of dependency links in the corpus (the treebank parse and the parser output always have the same number of links). The results are summarized in Table 1. It can be seen that the performance of the parser is highly correlated with the length of the sentences.

Max Sentence Length	10	15	20	40
Undirected Accuracy	90.8	85.6	84.0	79.9

Table 1. Evaluation Results on CTB 3.0

We also experimented with several alternative models for dependency parsing. Table 2 summerizes the results of these models on the test corpus with sentences up to 40 words long.

One of the characteristics of our parser is that it uses the similar words of both the head and the modifier for smoothing. The similarity-based smoothing method in (Dagan et al., 1999) uses the similar words of one of the words in a bigram. We can change the definition of similar context as follows so that only one word in a similar context of C may be different from a word in C (see Model (b) in Table 2):

$$S\left(\left[u, v, C_w^d\right]\right) = \left\{\left[u', v, C_w^d\right] \middle| u' \in S(u)\right\} \cup \left\{\left[u, v', C_w^d\right] \middle| v' \in S(v)\right\}$$

where w is either v or u depending on whether d is L or R. This change led to a 2.2% drop in accuracy (compared with Model (a) in Table 2), which we attribute to the fact that many contexts do not have similar contexts in the training corpus.

Since most previous parsing models maximize the joint probability of the parse tree and the sentence P(T, S) instead of P(T | S), we also implemented a joint model (see Model (c) in Table 2):

$$P(T,S) = \prod_{i=1}^{N} P\left(E_{m_i}^L \mid m_i, C_{m_i}^L\right) \times P\left(E_{m_i}^R \mid m_i, C_{m_i}^R\right) \times \left(1 - P\left(E_{h_i}^d \mid h_i, C_{h_i}^d\right)\right) \times P\left(m_i \mid h_i, C_{h_i}^{d_i}\right)$$

where h_i and m_i are the head and the modifier of the i'th dependency link. The probability $P\left(m_i \mid h_i, C_{h_i}^{d_i}\right)$ is smoothed by averaging the probabilities $P\left(m_i \mid h'_i, C_{h_i}^{d_i}\right)$, where h'_i is a similar word of h_i, as in (Dagan et al., 1999). The result was a dramatic decrease in accuracy from the conditional model's 79.9%. to 66.3%.

Our use of distributional word similarity can be viewed as assigning soft clusters to words. In contrast, parts-of-speech can be viewed as hard clusters of words. We can modify both the conditional and joint models to use part-of-speech tags, instead of words. Since there are only a small number of tags, the modified models used MLE without any smoothing except using a small constant as the probability of unseen events. Without smoothing, maximizing the conditional model is equivalent to maximizing the joint model. The accuracy of the unlexicalized models (see Model (d) and Model (e) in Table 2) is 71.1% which is considerably lower than the strictly lexicalized conditional model, but higher than the strictly lexicalized joint model. This demonstrated that soft clusters obtained through distributional word similarity perform better than the part-of-speech tags when used appropriately.

Models		**Accuracy**
(a)	Strictly lexicalized conditional model	**79.9**
(b)	At most one word is different in a similar context	77.7
(c)	Strictly lexicalized joint model	66.3
(d)	Unlexicalized conditional models	71.1
(e)	Unlexicalized joint models	71.1

Table 2. Performance of Alternative Models

5 Related Work

Previous parsing models (e.g., Collins, 1997; Charniak, 2000) maximize the joint probability P(*S*, *T*) of a sentence *S* and its parse tree *T*. We maximize the conditional probability P(*T* | *S*). Although they are theoretically equivalent, the use of conditional model allows us to take advantage of similarity-based smoothing.

Clark et al. (2002) also computes a conditional probability of dependency structures. While the probability space in our model consists of all possible non-projective dependency trees, their probability space is constrained to all the dependency structures that are allowed by a Combinatorial Category Grammar (CCG) and a category dictionary (lexicon). They therefore do not need the STOP markers in their model. Another major difference between our model and (Clark et al., 2002) is that the parameters in our model consist exclusively of conditional probabilities of binary variables.

Ratnaparkhi's maximum entropy model (Ratnaparkhi, 1999) is also a conditional model. However, his model maximizes the probability of the action during each step of the parsing process, instead of overall quality of the parse tree.

Yamada and Matsumoto (2002) presented a dependency parsing model using support vector machines. Their model is a discriminative model that maximizes the differences between scores of the correct parse and the scores of the top competing incorrect parses.

In many dependency parsing models such as (Eisner, 1996) and (MacDonald et al., 2005), the score of a dependency tree is the sum of the scores of the dependency links, which are computed independently of other links. An undesirable consequence of this is that the parser often creates multiple dependency links that are separately likely but jointly improbable (or even impossible). For example, there is nothing in such models to prevent the parser from assigning two subjects to a verb. In the DMV model (Klein and Manning, 2004), the probability of a dependency link is partly conditioned on whether or not there is a head word of the link already has a modifier. Our model is quite similar to the DMV model, except that we compute the conditional probability of the parse tree given the sentence, instead of the joint probability of the parse tree and the sentence.

There have been several previous approaches to parsing Chinese with the Penn Chinese Treebank (e.g., Bikel and Chiang, 2000; Levy and Manning, 2003). Both of these approaches employed phrase-structure joint models and used part-of-speech tags in back-off smoothing. Their results were evaluated with the precision and recall of the bracketings implied in the phrase structure parse trees. In contrast, the accuracy of our model is measured in terms of the dependency relationships. A dependency tree may correspond to more than one constituency trees. Our results are therefore not directly comparable with the precision and recall values in previous research. Moreover, it was argued in (Lin 1995) that dependency based evaluation is much more meaningful for the applications that use parse trees, since the semantic relationships are generally embedded in the dependency relationships.

6 Conclusion

To the best of our knowledge, all previous natural language parsers have to rely on part-of-speech tags. We presented a strictly lexicalized model for dependency parsing that only relies on word statistics. We compared our parser with an unlexicalized parser that employs the same probabilistic model except that the parameters are estimated using gold standard tags in the Chinese Treebank. Our experiments show that the strictly lexicalized parser significantly outperformed its unlexicalized counter-part.

An important distinction between our statistical model from previous parsing models is that all the parameters in our model are conditional probability of binary variables. This allows us to take advantage of similarity-based smoothing, which has not been successfully applied to parsing before.

Acknowledgements

The authors would like to thank Mark Steedman for suggesting the comparison with unlexicalized parsing in Section 4 and the anonymous reviewers for their comments. This work was supported in part by NSERC, the Alberta Ingenuity Centre for Machine Learning and the Canada Research

Chairs program. Qin Iris Wang was also supported by iCORE Scholarship.

References

Daniel M. Bikel. 2004. Intricacies of Collins' Parsing Model. *Computational Linguistics*, 30(4), pp. 479-511.

Daniel M. Bikel and David Chiang. 2000. Two Statistical Parsing Models applied to the Chinese Treebank. In *Proceedings of the second Chinese Language Processing Workshop*, pp. 1-6.

Eugene Charniak. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of the Second Meeting of North American Chapter of Association for Computational Linguistics (NAACL-2000)*, pp. 132-139.

Stephen Clark, Julia Hockenmaier and Mark Steedman. 2002. Building Deep Dependency Structures with a Wide-Coverage CCG Parser. In *Proceedings of the 40th Annual Meeting of the ACL*, pp. 327-334.

Michael Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. In *Proceedings of the 34th Annual Meeting of the ACL*, pp. 184-191. Santa Cruz.

Michael Collins. 1997. Three Generative, Lexicalised Models for Statistical Parsing. In *Proceedings of the 35th Annual Meeting of the ACL (jointly with the 8th Conference of the EACL)*, pp. 16-23. Madrid.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. PhD Dissertation, University of Pennsylvania.

Ido Dagan, Lillian Lee and Fernando Pereira. 1999. Similarity-based models of cooccurrence probabilities. *Machine Learning, Vol. 34(1-3) special issue on Natural Language Learning*, pp. 43-69.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING-96*, pp. 340-345, Copenhagen.

Daniel Gildea. 2001. Corpus Variation and Parser Performance. In Proceedings of *EMNLP-2001*, pp. 167-202. Pittsburgh, PA.

Gregory Grefenstette. 1994. *Explorations in Automatic Thesaurus Discovery.* Kluwer Academic Press, Boston, MA.

Zelig S. Harris. 1968. *Mathematical Structures of Language*. Wiley, New York.

Donald Hindle. 1990. Noun Classification from Predicate-Argument Structures. In *Proceedings of ACL-90*, pp. 268-275. Pittsburg, Pennsylvania.

Dan Klein and Chris Manning. 2002. Fast exact inference with a factored model for natural language parsing. In *Proceedings of Neural Information Processing Systems*.

Dan Klein and Chris Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the ACL,* pp. 423-430.

Dan Klein and Chris Manning. 2004. Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency. In *Proceedings of the 42nd Annual Meeting of the ACL,* pp. 479-486.

Roger Levy and Chris Manning. 2003. Is it harder to parse Chinese, or the Chinese Treebank? In *Proceedings of the 41st Annual Meeting of the ACL,* pp. 439-446.

Dekang Lin. 1995. A dependency-based method for evaluating broad-coverage parsers. In *Proceedings of IJCAI-95*, pp.1420-1425.

Dekang Lin. 1998. Automatic Retrieval and Clustering of Similar Words. In *Proceeding of COLING-ACL98*, pp. 768-774. Montreal, Canada.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL-2005*, pp. 91-98.

Fernando Pereira, Naftali Z. Tishby, and Lillian Lee. 1993. Distributional clustering of English words. In *Proceedings of ACL-1993*, pp. 183-190, Columbus, Ohio.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of the 8th International Workshop on Parsing Technologies*, pp.195-206.