Parsing and Subcategorization Data

Jianguo Li and Chris Brew
Department of Linguistics
The Ohio State University
Columbus, OH, USA
{jianguo|cbrew}@ling.ohio-state.edu

Abstract

In this paper, we compare the performance of a state-of-the-art statistical parser (Bikel, 2004) in parsing written and spoken language and in generating subcategorization cues from written and spoken language. Although Bikel's parser achieves a higher accuracy for parsing written language, it achieves a higher accuracy when extracting subcategorization cues from spoken language. Our experiments also show that current technology for extracting subcategorization frames initially designed for written texts works equally well for spoken language. Additionally, we explore the utility of punctuation in helping parsing and extraction of subcategorization cues. Our experiments show that punctuation is of little help in parsing spoken language and extracting subcategorization cues from spoken language. This indicates that there is no need to add punctuation in transcribing spoken corpora simply in order to help parsers.

1 Introduction

Robust statistical syntactic parsers, made possible by new statistical techniques (Collins, 1999; Charniak, 2000; Bikel, 2004) and by the availability of large, hand-annotated training corpora such as WSJ (Marcus et al., 1993) and Switchboard (Godefrey et al., 1992), have had a major impact on the field of natural language processing. There are many ways to make use of parsers' output. One particular form of data that can be extracted from parses is information about subcategorization. Subcategorization data comes in two forms: subcategorization frame (SCF) and subcategorization cue (SCC). SCFs differ from SCCs in that SCFs contain only arguments while SCCs contain both arguments and adjuncts. Both SCFs and SCCs have been crucial to NLP tasks. For example, SCFs have been used for verb disambiguation and classification (Schulte im Walde, 2000; Merlo and Stevenson, 2001; Lapata and Brew, 2004; Merlo et al., 2005) and SCCs for semantic role labeling (Xue and Palmer, 2004; Punyakanok et al., 2005).

Current technology for automatically acquiring subcategorization data from corpora usually relies on statistical parsers to generate SCCs. While great efforts have been made in parsing written texts and extracting subcategorization data from written texts, spoken corpora have received little attention. This is understandable given that spoken language poses several challenges that are absent in written texts, including disfluency, uncertainty about utterance segmentation and lack of punctuation. Roland and Jurafsky (1998) have suggested that there are substantial subcategorization differences between written corpora and spoken corpora. For example, while written corpora show a much higher percentage of passive structures, spoken corpora usually have a higher percentage of zero-anaphora constructions. We believe that subcategorization data derived from spoken language, if of acceptable quality, would be of more value to NLP tasks involving a syntactic analysis of spoken language. We do not show this here.

The goals of this study are as follows:

1. Test the performance of Bikel's parser in parsing written and spoken language.

2. Compare the accuracy level of SCCs generated from parsed written and spoken lan-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 515–522,
Sydney, July 2006. ©2006 Association for Computational Linguistics

guage. We hope that such a comparison will shed some light on the feasibility of acquiring subcategorization data from spoken language using the current SCF acquisition technology initially designed for written language.

3. Apply our SCF extraction system (Li and Brew, 2005) to spoken and written language separately and compare the accuracy achieved for the acquired SCFs from spoken and written language.

4. Explore the utility of punctuation[1] in parsing and extraction of SCCs. It is generally recognized that punctuation helps in parsing written texts. For example, Roark (2001) finds that removing punctuation from both training and test data (WSJ) decreases his parser's accuracy from 86.4%/86.8% (LR/LP) to 83.4%/84.1%. However, spoken language does not come with punctuation. Even when punctuation is added in the process of transcription, its utility in helping parsing is slight. Both Roark (2001) and Engel et al. (2002) report that removing punctuation from both training and test data (Switchboard) results in only 1% decrease in their parser's accuracy.

2 Experiment Design

Three models will be investigated for parsing and extracting SCCs from the parser's output:

1. **punc**: leaving punctuation in both training and test data.

2. **no-punc**: removing punctuation from both training and test data.

3. **punc-no-punc**: removing punctuation from only the test data.

Following the convention in the parsing community, for written language, we selected sections 02-21 of WSJ as training data and section 23 as test data (Collins, 1999). For spoken language, we designated section 2 and 3 of Switchboard as training data and files of sw4004 to sw4135 of section 4 as test data (Roark, 2001). Since we are also interested in extracting SCCs from the parser's output,

[1]We use punctuation to refer to sentence-internal punctuation unless otherwise specified.

label	clause type	desired SCCs
S	gerundive	(NP)-GERUND
	small clause	NP-NP, (NP)-ADJP
	control	(NP)-INF-*to*
SBAR	control	(NP)-INF-*wh-to*
	with a complementizer	(NP)-S-*wh*, (NP)-S-*that*
	without a complementizer	(NP)-S-*that*

Table 1: SCCs for different clauses

we eliminated from the two test corpora all sentences that do not contain verbs. Our experiments proceed in the following three steps:

1. Tag test data using the POS-tagger described in Ratnaparkhi (1996).

2. Parse the POS-tagged data using Bikel's parser.

3. Extract SCCs from the parser's output. The extractor we built first locates each verb in the parser's output and then identifies the syntactic categories of all its sisters and combines them into an SCC. However, there are cases where the extractor has more work to do.

   - Finite and Infinite Clauses: In the Penn Treebank, **S** and **SBAR** are used to label different types of clauses, obscuring too much detail about the internal structure of each clause. Our extractor is designed to identify the internal structure of different types of clause, as shown in Table 1.
   - Passive Structures: As noted above, Roland and Jurafsky (Roland and Jurafsky, 1998) have noticed that written language tends to have a much higher percentage of passive structures than spoken language. Our extractor is also designed to identify passive structures from the parser's output.

3 Experiment Results

3.1 Parsing and SCCs

We used EVALB measures Labeled Recall (LR) and Labeled Precision (LP) to compare the parsing performance of different models. To compare the accuracy of SCCs proposed from the parser's output, we calculated SCC Recall (SR) and SCC Precision (SP). SR and SP are defined as follows:

$$SR = \frac{\text{number of correct cues from the parser's output}}{\text{number of cues from treebank parse}} \quad (1)$$

WSJ		
model	LR/LP	SR/SP
punc	87.92%/88.29%	76.93%/77.70%
no-punc	86.25%/86.91%	76.96%/76.47%
punc-no-punc	82.31%/83.70%	74.62%/74.88%
Switchboard		
model	LR/LP	SR/SP
punc	83.14%/83.80%	79.04%/78.62%
no-punc	82.42%/83.74%	78.81%/78.37%
punc-no-punc	78.62%/80.68%	75.51%/75.02%

Table 2: Results of parsing and extraction of SCCs

$$SP = \frac{\text{number of correct cues from the parser's output}}{\text{number of cues from the parser's output}} \quad (2)$$

$$\text{SCC Balanced F-measure} = \frac{2 * SR * SP}{SR + SP} \quad (3)$$

The results for parsing WSJ and Switchboard and extracting SCCs are summarized in Table 2.

The LR/LP figures show the following trends:

1. Roark (2001) showed LR/LP of 86.4%/86.8% for punctuated written language, 83.4%/84.1% for unpunctuated written language. We achieve a higher accuracy in both punctuated and unpunctuated written language, and the decrease if punctuation is removed is less

2. For spoken language, Roark (2001) showed LR/LP of 85.2%/85.6% for punctuated spoken language, 84.0%/84.6% for unpunctuated spoken language. We achieve a lower accuracy in both punctuated and unpunctuated spoken language, and the decrease if punctuation is removed is less. The trends in (1) and (2) may be due to parser differences, or to the removal of sentences lacking verbs.

3. Unsurprisingly, if the test data is unpunctuated, but the models have been trained on punctuated language, performance decreases sharply.

In terms of the accuracy of extraction of SCCs, the results follow a similar pattern. However, the utility of punctuation turns out to be even smaller. Removing punctuation from both the training and test data results in a 0.8% drop in the accuracy of SCC extraction for written language and a 0.3% drop for spoken language.

Figure 1 exhibits the relation between the accuracy of parsing and that of extracting SCCs. If we consider WSJ and Switchboard individually, there seems to exist a positive correlation between the accuracy of parsing and that of extracting SCCs. In other words, higher LR/LP indicates

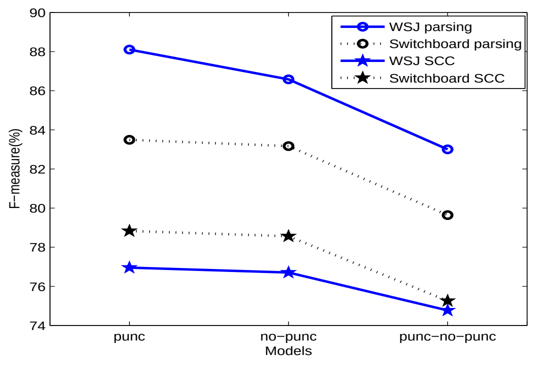

Figure 1: F-measure for parsing and extraction of SCCs

higher SR/SP. However, Figure 1 also shows that although the parser achieves a higher F-measure value for paring WSJ, it achieves a higher F-measure value for generating SCCs from Switchboard.

The fact that the parser achieves a higher accuracy of extracting SCCs from Switchboard than WSJ merits further discussion. Intuitively, it seems to be true that the shorter an SCC is, the more likely that the parser is to get it right. This intuition is confirmed by the data shown in Figure 2. Figure 2 plots the accuracy level of extracting SCCs by SCC's length. It is clear from Figure 2 that as SCCs get longer, the F-measure value drops progressively for both WSJ and Switchboard. Again, Roland and Jurafsky (1998) have suggested that one major subcategorization difference between written and spoken corpora is that spoken corpora have a much higher percentage of the zero-anaphora construction. We then examined the distribution of SCCs of different length in WSJ and Switchboard. Figure 3 shows that SCCs of length 0[2] account for a much higher percentage in Switchboard than WSJ, but it is always the other way around for SCCs of non-zero length. This observation led us to believe that the better performance that Bikel's parser achieves in extracting SCCs from Switchboard may be attributed to the following two factors:

1. Switchboard has a much higher percentage of SCCs of length 0.

2. The parser is very accurate in extracting shorter SCCs.

[2] Verbs have a length-0 SCC if they are intransitive and have no modifiers.

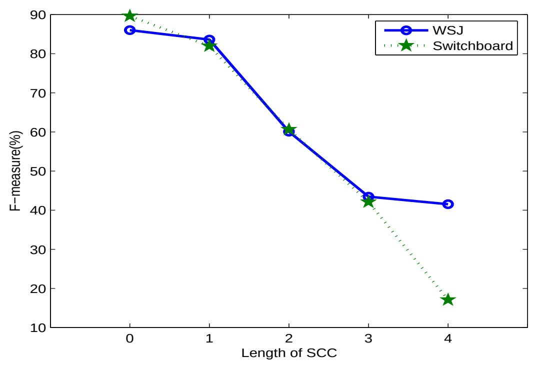

Figure 2: F-measure for SCCs of different length

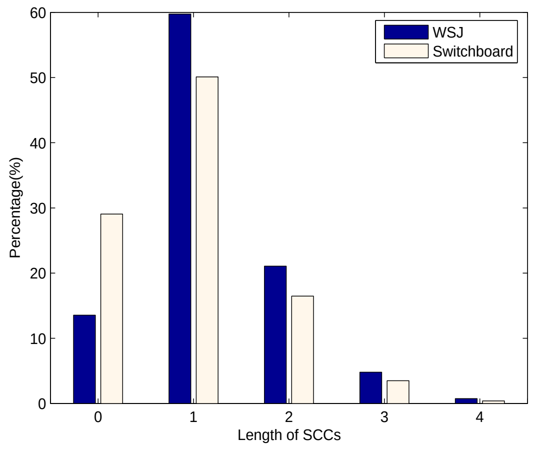

Figure 3: Distribution of SCCs by length

3.2 Extraction of Dependents

In order to estimate the effects of SCCs of length 0, we examined the parser's performance in retrieving dependents of verbs. Every constituent (whether an argument or adjunct) in an SCC generated by the parser is considered a dependent of that verb. SCCs of length 0 will be discounted because verbs that do not take any arguments or adjuncts have no dependents[3]. In addition, this way of evaluating the extraction of SCCs also matches the practice in some NLP tasks such as semantic role labeling (Xue and Palmer, 2004). For the task of semantic role labeling, the total number of dependents correctly retrieved from the parser's output affects the accuracy level of the task.

To do this, we calculated the number of dependents shared by between each SCC proposed from the parser's output and its corresponding SCC proposed from Penn Treebank. We based our calculation on a modified version of Minimum Edit Distance Algorithm. Our algorithm works by creating a shared-dependents matrix with one column for each constituent in the target sequence (SCCs proposed from Penn Treebank) and one row for each constituent in the source sequence (SCCs proposed from the parser's output). Each cell shared-dependent[*i*,*j*] contains the number of constituents shared between the first *i* constituents of the target sequence and the first *j* constituents of the source sequence. Each cell can then be computed as a simple function of the three possible paths through the matrix that arrive there. The algorithm is illustrated in Table 3.

[3]We are aware that subjects are typically also considered dependents, but we did not include subjects in our experiments

```
shared-dependents[i.j] = MAX(
shared-dependents[i-1,j],
shared-dependents[i-1,j-1]+1 if target[i] = source[j],
shared-dependents[i-1,j-1] if target[i] != source[j],
shared-dependents[i,j-1])
```

Table 3: The algorithm for computing shared dependents

INF	#5	1	1	2	**3**
ADVP	#4	1	1	2	2
PP-*in*	#3	1	1	2	2
NP	#2	1	1	1	1
NP	#1	1	1	1	1
	#0	#1	#2	#3	#4
		NP	S-*that*	PP-*in*	INF

Table 4: An example of computing the number of shared dependents

Table 4 shows an example of how the algorithm works with NP-S-*that*-PP-*in*-INF as the target sequence and NP-NP-PP-*in*-ADVP-INF as the source sequence. The algorithm returns 3 as the number of dependents shared by two SCCs.

We compared the performance of Bikel's parser in retrieving dependents from written and spoken language over all three models using Dependency Recall (DR) and Dependency Precision (DP). These metrics are defined as follows:

$$DR = \frac{\text{number of correct dependents from parser's output}}{\text{number of dependents from treebank parse}} \tag{4}$$

$$DP = \frac{\text{number of correct dependents from parser's output}}{\text{number of dependents from parser's output}} \tag{5}$$

$$\text{Dependency F-measure} = \frac{2 * DR * DP}{DR + DP} \tag{6}$$

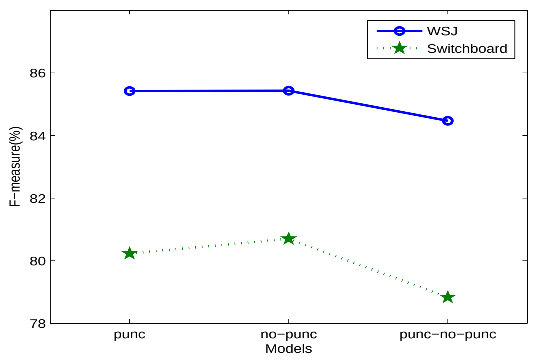

Figure 4: F-measure for extracting dependents

The results of Bikel's parser in retrieving dependents are summarized in Figure 4. Overall, the parser achieves a better performance for WSJ over all three models, just the opposite of what have been observed for SCC extraction. Interestingly, removing punctuation from both the training and test data actually slightly improves the F-measure. This holds true for both WSJ and Switchboard. This Dependency F-measure differs in detail from similar measures in Xue and Palmer (2004). For present purposes all that matters is the relative value for WSJ and Switchboard.

4 Extraction of SCFs from Spoken Language

Our experiments indicate that the SCCs generated by the parser from spoken language are as accurate as those generated from written texts. Hence, we would expect that the current technology for extracting SCFs, initially designed for written texts, should work equally well for spoken language. We previously built a system for automatically extracting SCFs from spoken BNC, and reported accuracy comparable to previous systems that work with only written texts (Li and Brew, 2005). However, Korhonen (2002) has shown that a direct comparison of different systems is very difficult to interpret because of the variations in the number of targeted SCFs, test verbs, gold standards and in the size of the test data. For this reason, we apply our SCF acquisition system separately to a written and spoken corpus of similar size from BNC and compare the accuracy of acquired SCF sets.

4.1 Overview

As noted above, previous studies on automatic extraction of SCFs from corpora usually proceed in two steps and we adopt this approach.

1. Hypothesis Generation: Identify all SCCs from the corpus data.

2. Hypothesis Selection: Determine which SCC is a valid SCF for a particular verb.

4.2 SCF Extraction System

We briefly outline our SCF extraction system for automatically extracting SCFs from corpora, which was based on the design proposed in Briscoe and Carroll (1997).

1. **A Statistical Parser**: Bikel's parser is used to parse input sentences.

2. **An SCF Extractor**: An extractor is use to extract SCCs from the parser's output.

3. **An English Lemmatizer**: MORPHA (Minnen et al., 2000) is used to lemmatize each verb.

4. **An SCF Evaluator**: An evaluator is used to filter out false SCCs based on their likelihood.

An SCC generated by the parser and extractor may be a correct SCC, or it may contain an adjunct, or it may simply be wrong due to tagging or parsing errors. We therefore need an SCF evaluator capable of filtering out false cues. Our evaluator has two parts: the Binomial Hypothesis Test (Brent, 1993) and a back-off algorithm (Sarkar and Zeman, 2000).

1. **The Binomial Hypothesis Test** (BHT): Let *p* be the probability that an scf_i occurs with $verb_j$ that is not supposed to take scf_i. If a verb occurs *n* times and *m* of those times it co-occurs with scf_i, then the scf_i cues are false cues is estimated by the summation of the binomial distribution for $m \le k \le n$:

$$P(m^+, n, p) = \sum_{k=m}^{n} \frac{n!}{k!(n-k)!} p^k (1-p)^{(n-k)} \quad (7)$$

If the value of $P(m^+, n, p)$ is less than or equal to a small threshold value, then the null hypothesis that $verb_j$ does not take scf_i is extremely unlikely to be true. Hence, scf_i is very likely to be a valid SCF for $verb_j$. The

SCCs	SCFs
NP-PP-*before*	
NP-S-*when*	NP
NP-PP-*at*-S-*before*	
NP-PP-*to*-S-*when*	
NP-PP-*to*-PP-*at*	NP-PP-*to*
NP-PP-*to*-S-*because*-ADVP	

Table 5: SCCs and correct SCFs for *introduce*

corpus	WC	SC
number of verb tokens	115,524	109,678
number of verb types	5,234	4,789
verb types seen more than 10 times	1,102	998
number of acquired SCFs	2,688	1,984
average number of SCFs per verb	2.43	1.99

Table 6: Training data for WC and SC

value of *m* and *n* can be directly computed from the extractor's output, but the value of *p* is not easy to obtain. Following Manning (1993), we empirically determined the value of *p*. It was between 0.005 to 0.4 depending on the likelihood of an SCC being a valid SCF.

2. **Back-off Algorithm**: Many SCCs generated by the parser and extractor tend to contain some adjuncts. However, for many SCCs, one of its subsets is likely to be the correct SCF. Table 5 shows some SCCs generated by the extractor and the corresponding SCFs.

   The Back-off Algorithm always starts with the longest SCC for each verb. Assume that this SCC fails the BHT. The evaluator then eliminates the last constituent from the rejected cue, transfers its frequency to its successor and submits the successor to the BHT again. In this way, frequency can accumulate and more valid frames survive the BHT.

4.3 Results and Discussion

We evaluated our SCF extraction system on written and spoken BNC. We chose one million word written corpus (WC) and a comparable spoken corpus (SC) from BNC. Table 6 provides relevant information on the two corpora. We only keep the verbs that occur at least 10 times in our training data.

To compare the performance of our system on WC and SC, we calculated the type precision, type recall and F-measure. Type precision is the percentage of SCF types that our system proposes which are correct according some gold standard and type recall is the percentage of correct SCF types proposed by our system that are listed in the gold standard. We used the 14 verbs [4] selected by Briscoe and Carroll (1997) and evaluated our results of these verbs against the SCF entries in two gold standards: COMLEX (Grishman et al., 1994) and a manually constructed SCF set from the training data. It makes sense to use a manually constructed SCF set while calculating type precision and recall because some of the SCFs in a syntax dictionary such as COMLEX might not occur in the training data at all. We constructed separate SCF sets for the written and spoken BNC.

gold standard	COMLEX		Manually Constructed	
corpus	WC	SC	WC	SC
type precision	93.1%	92.9%	93.1%	92.9%
type recall	49.2%	47.7%	56.5%	57.6%
F-measure	64.4%	63.1%	70.3%	71.1%

Table 7: Type precision and recall and F-measure

The results are summarized in Table 7. As shown in Table 7, the accuracy achieved for WC and SC are very comparable: Our system achieves a slightly better result for WC when using COMLEX as the gold standard and for SC when using manually constructed SCF set as gold standard, suggesting that it is feasible to apply the current technology for automatically extracting SCFs to spoken language.

5 Conclusions and Future Work

5.1 Use of Parser's Output

In this paper, we have shown that it is not necessarily true that statistical parsers always perform worse when dealing with spoken language. The conventional accuracy metrics for parsing (LR/LP) should not be taken as the only metrics in determining the feasibility of applying statistical parsers to spoken language. It is necessary to consider what information we want to extract out of parsers' output and make use of.

1. Extraction of SCFs from Corpora: This task takes SCCs generated by the parser and extractor as input. Our experiments show that

[4]The 14 verbs used in Briscoe and Carroll (1997) are *ask, begin, believe, cause, expect, find, give, help, like, move, produce, provide, seem* and *sway*. We replaced *sway* with *show* because *sway* occurs less than 10 times in our training data.

the SCCs generated for spoken language are as accurate as those generated for written language. We have also shown that it is feasible to apply the current SCF extraction technology to spoken language.

2. Semantic Role Labeling: This task usually operates on parsers' output and the number of dependents of each verb that are correctly retrieved by the parser clearly affects the accuracy of the task. Our experiments show that the parser achieves a much lower accuracy in retrieving dependents from the spoken language than written language. This seems to suggest that a lower accuracy is likely to be achieved for a semantic role labeling task performed on spoken language. We are not aware that this has yet been tried.

5.2 Punctuation and Speech Transcription Practice

Both our experiments and Roark's experiments show that parsing accuracy measured by LR/LP experiences a sharper decrease for WSJ than Switchboard after we removed punctuation from training and test data. In spoken language, commas are largely used to delimit disfluency elements. As noted in Engel et al. (2002), statistical parsers usually condition the probability of a constituent on the types of its neighboring constituents. The way that commas are used in speech transcription seems to have the effect of increasing the range of neighboring constituents, thus fragmenting the data and making it less reliable. On the other hand, in written texts, commas serve as more reliable cues for parsers to identify phrasal and clausal boundaries.

In addition, our experiment demonstrates that punctuation does not help much with extraction of SCCs from spoken language. Removing punctuation from both the training and test data results in rougly a 0.3% decrease in SR/SP. Furthermore, removing punctuation from both training and test data actually slightly improves the performance of Bikel's parser in retrieving dependents from spoken language. All these results seem to suggest that adding punctuation in speech transcription is of little help to statistical parsers including at least three state-of-the-art statistical parsers (Collins, 1999; Charniak, 2000; Bikel, 2004). As a result, there may be other good reasons why someone who wants to build a Switchboard-like corpus should choose to provide punctuation, but there is no need to do so simply in order to help parsers.

However, segmenting utterances into individual units is necessary because statistical parsers require sentence boundaries to be clearly delimited. Current statistical parsers are unable to handle an input string consisting of two sentences. For example, when presented with an input string as in (1) and (2), if the two sentences are separated by a period (1), Bikel's parser wrongly treats the second sentence as a sentential complement of the main verb *like* in the first sentence. As a result, the extractor generates an SCC NP-S for *like*, which is incorrect. The parser returns the same parse after we removed the period (2) and let the parser parse it again.

(1) I like the long hair. It was back in high school.

(2) I like the long hair It was back in high school.

Hence, while adding punctuation in transcribing a Switchboard-like corpus is not of much help to statistical parsers, segmenting utterances into individual units is crucial for statistical parsers. In future work, we plan to develop a system capable of automatically segmenting speech utterances into individual units.

6 Acknowledgments

This study was supported by NSF grant 0347799. Our thanks go to Eric Fosler-Lussier, Mike White and three anonymous reviewers for their valuable comments.

References

D. Bikel. 2004. Intricacies of Collin's parsing models. *Computational Linguistics*, 30(2):479–511.

M. Brent. 1993. From grammar to lexicon: Unsupervised learning of lexical syntax. *Computational Linguistics*, 19(3):243–262.

T. Briscoe and J. Carroll. 1997. Automatic extraction of subcategorization from corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing*, pages 356–363.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 2000 Conference of the North American Chapter of the Association for Computation Linguistics*, pages 132–139.

M. Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania.

D. Engel, E. Charniak, and M. Johnson. 2002. Parsing and disfluency placement. In *Proceedings of 2002 Conference on Empirical Methods of Natural Language Processing*, pages 49–54.

J. Godefrey, E. Holliman, and J. McDaniel. 1992. SWITCHBOARD: Telephone speech corpus for research and development. In *Proceedings of ICASSP-92*, pages 517–520.

R. Grishman, C. Macleod, and A. Meryers. 1994. Comlex syntax: Building a computational lexicon. In *Proceedings of the 1994 International Conference of Computational Linguistics*, pages 268–272.

A. Korhonen. 2002. *Subcategorization Acquisition*. Ph.D. thesis, Cambridge University.

M. Lapata and C. Brew. 2004. Verb class disambiguation using informative priors. *Computational Linguistics*, 30(1):45–73.

J. Li and C. Brew. 2005. Automatic extraction of subcategorization frames from spoken corpora. In *Proceedings of the Interdisciplinary Workshop on the Identification and Representation of Verb Features and Verb Classes*, Saarbracken, Germany.

C. Manning. 1993. Automatic extraction of a large subcategorization dictionary from corpora. In *Proceedings of 31st Annual Meeting of the Association for Computational Linguistics*, pages 235–242.

M. Marcus, G. Kim, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

P. Merlo and S. Stevenson. 2001. Automatic verb classification based on statistical distribution of argument structure. *Computational Linguistics*, 27(3):373–408.

P. Merlo, E. Joanis, and J. Henderson. 2005. Unsupervised verb class disambiguation based on diathesis alternations. manuscripts.

G. Minnen, J. Carroll, and D. Pearce. 2000. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–223.

V. Punyakanok, D. Roth, and W. Yih. 2005. The necessity of syntactic parsing for semantic role labeling. In *Proceedings of the 2nd Midwest Computational Linguistics Colloquium*, pages 15–22.

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods of Natural Language Processing*, pages 133–142.

B. Roark. 2001. *Robust Probabilistic Predictive Processing: Motivation, Models, and Applications*. Ph.D. thesis, Brown University.

D. Roland and D. Jurafsky. 1998. How verb subcategorization frequency is affected by the corpus choice. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 1122–1128.

A. Sarkar and D. Zeman. 2000. Automatic extraction of subcategorization frames for Czech. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 691–697.

S. Schulte im Walde. 2000. Clustering verbs semantically according to alternation behavior. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 747–753.

N. Xue and M. Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of 2004 Conference on Empirical Methods in Natural Language Processing*, pages 88–94.