Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 239-246.

The Necessity of Parsing for Predicate Argument Recognition

Daniel Gildea and Martha Palmer
University of Pennsylvania
dgildea,mpalmer@cis.upenn.edu

Abstract

Broad-coverage corpora annotated with semantic role, or argument structure, information are becoming available for the first time. Statistical systems have been trained to automatically label semantic roles from the output of statistical parsers on unannotated text. In this paper, we quantify the effect of parser accuracy on these systems' performance, and examine the question of whether a flatter "chunked" representation of the input can be as effective for the purposes of semantic role identification.

1 Introduction

Over the past decade, most work in the field of information extraction has shifted from complex rule-based, systems designed to handle a wide variety of semantic phenomena including quantification, anaphora, aspect and modality (e.g. Alshawi (1992)), to simpler finite-state or statistical systems such as Hobbs et al. (1997) and Miller et al. (1998). Much of the evaluation of these systems has been conducted on extracting relations for specific semantic domains such as corporate acquisitions or terrorist events in the framework of the DARPA Message Understanding Conferences.

Recently, attention has turned to creating corpora annotated for argument structure for a broader range of predicates. The Propbank project at the University of Pennsylvania (Kingsbury and Palmer, 2002) and the FrameNet project at the International Computer Science Institute (Baker et al., 1998) share the goal of documenting the syntactic realization of arguments of the predicates of the general English lexicon by annotating a corpus with semantic roles. Even for a single predicate, semantic arguments often have multiple syntactic realizations, as shown by the following paraphrases:

(1) John will meet with Mary.
John will meet Mary.
John and Mary will meet.

(2) The door opened.
Mary opened the door.

Correctly identifying the semantic roles of the sentence constituents is a crucial part of interpreting text, and in addition to forming an important part of the information extraction problem, can serve as an intermediate step in machine translation or automatic summarization. In this paper, we examine how the information provided by modern statistical parsers such as Collins (1997) and Charniak (1997) contributes to solving this problem. We measure the effect of parser accuracy on semantic role prediction from parse trees, and determine whether a complete tree is indeed necessary for accurate role prediction.

Gildea and Jurafsky (2002) describe a statistical system trained on the data from the FrameNet project to automatically assign semantic roles. The system first passed sentences through an automatic parser, extracted syntactic features from the parses, and estimated probabilities for semantic roles from the syntactic and lexical features. Both training and test sentences were automatically parsed, as no hand-annotated parse trees were available for the corpus. While the errors introduced by the parser no doubt negatively affected the results obtained, there was no direct way of quantifying this effect. Of the systems evaluated for the Message Understanding Conference task, Miller et al. (1998) made use of an integrated syntactic and semantic model producing a full parse tree, and achieved results comparable to other systems that did not make use of a complete parse. As in the FrameNet case, the parser was not trained on the corpus for which semantic annotations were available, and the effect of better, or even perfect, parses could not be measured.

One of the differences between the two semantic annotation projects is that the sentences chosen

for annotation for Propbank are from the same Wall Street Journal corpus chosen for annotation for the original Penn Treebank project, and thus hand-checked syntactic parse trees are available for the entire dataset. In this paper, we compare the performance of a system based on gold-standard parses with one using automatically generated parser output. We also examine whether it is possible that the additional information contained in a full parse tree is negated by the errors present in automatic parser output, by testing a role-labeling system based on a flat or "chunked" representation of the input.

2 The Data

The results in this paper are primarily derived from the Propbank corpus, and will be compared to earlier results from the FrameNet corpus. Before proceeding to the experiments, this section will briefly describe the similarities and differences between the two sets of data.

While the goals of the two projects are similar in many respects, their methodologies are quite different. FrameNet is focused on semantic frames, which are defined as schematic representation of situations involving various participants, props, and other conceptual roles (Fillmore, 1976). The project methodology has proceeded on a frame-by-frame basis, that is by first choosing a semantic frame, defining the frame and its participants or **frame elements**, and listing the various lexical predicates which invoke the frame, and then finding example sentences of each predicate in the corpus (the British National Corpus was used) and annotating each frame element. The example sentences were chosen primarily for coverage of all the syntactic realizations of the frame elements, and simple examples of these realizations were preferred over those involving complex syntactic structure not immediate relevant to the lexical predicate itself. From the perspective of an automatic classification system, the overrepresentation of rare syntactic realizations may cause the system to perform more poorly than it might on more statistically representative data. On the other hand, the exclusion of complex examples may make the task artificially easy. Only sentences where the lexical predicate was used "in frame" were annotated. A word with multiple distinct senses would generally be analyzed as belonging to different frames in each sense, but may only be found in the FrameNet corpus in the sense for which a frame has been defined. It is interesting to note that the semantic frames are a helpful way of generalizing between predicates; words in the same frame have been found frequently to share the same syntactic argument structure. A more complete description of the FrameNet project can be found in (Baker et al., 1998; Johnson et al., 2001), and the ramifications for automatic classification are discussed more thoroughly in (Gildea and Jurafsky, 2002).

The philosophy of the Propbank project can be likened to FrameNet without frames. While the semantic roles of FrameNet are defined at the level of the frame, in Propbank, roles are defined on a per-predicate basis. The core arguments of each predicate are simply numbered, while remaining arguments are given labels such as "temporal" or "locative". While the two types of label names are reminiscent of the traditional argument/adjunct distinction, this is primarily as a convenience in defining roles, and no claims are intended as to optionality or other traditional argument/adjunct tests. To date, Propbank has addressed only verbs, where FrameNet includes nouns and adjectives. Propbank's annotation process has proceeded from the most to least common verbs, and all examples of each verb from the corpus are annotated. Thus, the data for each predicate are statistically representative of the corpus, as are the frequencies of the predicates themselves. Annotation takes place with reference to the Penn Treebank trees — not only are annotators shown the trees when analyzing a sentence, they are constrained to assign the semantic labels to portions of the sentence corresponding to nodes in the tree.

Propbank annotators tag all examples of a given verb, regardless of word sense. The tagging guidelines for a verb may contain many "rolesets", corresponding to word sense at a relatively coarse-grained level. The need for multiple rolesets is determined by the roles themselves, that is, uses of the verb with different arguments are given separate rolesets. However, the preliminary version of the data used in the experiments below are not tagged for word sense, or for the roleset used. Sense tagging is planned for a second pass through the data. In many cases the roleset can be determined from the argument annotations themselves. However, we did not make any attempt to distinguish sense in our experiments, and simply attempted to predict argument labels based on the identity of the lexical predicate.

3 The Experiments

In previous work using the FrameNet corpus, Gildea and Jurafsky (2002) developed a system to predict semantic roles from sentences and their parse trees as determined by the statistical parser of Collins (1997). We will briefly review their

probability model before adapting the system to handle unparsed data.

Probabilities of a parse constituent belonging to a given semantic role were calculated from the following features:

Phrase Type: This feature indicates the syntactic type of the phrase expressing the semantic roles: examples include noun phrase (NP), verb phrase (VP), and clause (S). Phrase types were derived automatically from parse trees generated by the parser, as shown in Figure 1. The parse constituent spanning each set of words annotated as an argument was found, and the constituent's nonterminal label was taken as the phrase type. As an example of how this feature is useful, in communication frames, the SPEAKER is likely to appear as a noun phrase, TOPIC as a prepositional phrase or noun phrase, and MEDIUM as a prepositional phrase, as in: "We talked about the proposal over the phone." When no parse constituent was found with boundaries matching those of an argument during testing, the largest constituent beginning at the argument's left boundary and lying entirely within the element was used to calculate the features.

Parse Tree Path: This feature is designed to capture the syntactic relation of a constituent to the predicate. It is defined as the **path** from the predicate through the parse tree to the constituent in question, represented as a string of parse tree nonterminals linked by symbols indicating upward or downward movement through the tree, as shown in Figure 2. Although the path is composed as a string of symbols, our systems will treat the string as an atomic value. The path includes, as the first element of the string, the part of speech of the predicate, and, as the last element, the phrase type or syntactic category of the sentence constituent marked as an argument.

Position: This feature simply indicates whether the constituent to be labeled occurs before or after the predicate defining the semantic frame. This feature is highly correlated with grammatical function, since subjects will generally appear before a verb, and objects after. This feature may overcome the shortcomings of reading grammatical function from the parse tree, as well as errors in the parser output.

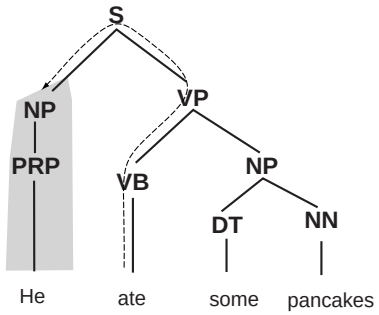

Figure 2: In this example, the **path** from the predicate *ate* to the argument *He* can be represented as VB↑VP↑S↓NP, with ↑ indicating upward movement in the parse tree and ↓ downward movement.

Voice: The distinction between active and passive verbs plays an important role in the connection between semantic role and grammatical function, since direct objects of active verbs correspond to subjects of passive verbs. From the parser output, verbs were classified as active or passive by building a set of 10 passive-identifying patterns. Each of the patterns requires both a passive auxiliary (some form of "to be" or "to get") and a past participle.

Head Word: Lexical dependencies provide important information in labeling semantic roles, as one might expect from their use in statistical models for parsing. Since the parser used assigns each constituent a head word as an integral part of the parsing model, the head words of the constituents can be read from the parser output. For example, in a communication frame, noun phrases headed by "Bill", "brother", or "he" are more likely to be the SPEAKER, while those headed by "proposal", "story", or "question" are more likely to be the TOPIC.

To predict argument roles in new data, we wish to estimate the probability of each role given these five features and the predicate p: $P(r|pt, path, position, voice, hw, p)$. Due to the sparsity of the data, it is not possible to estimate this probability from the counts in the training. Instead, we estimate probabilities from various subsets of the features, and interpolate a linear combination of the resulting distributions. The interpolation is performed over the most specific distributions for which data are available, which can be thought of as choosing the topmost distributions available from a backoff lattice, shown in Figure 3.

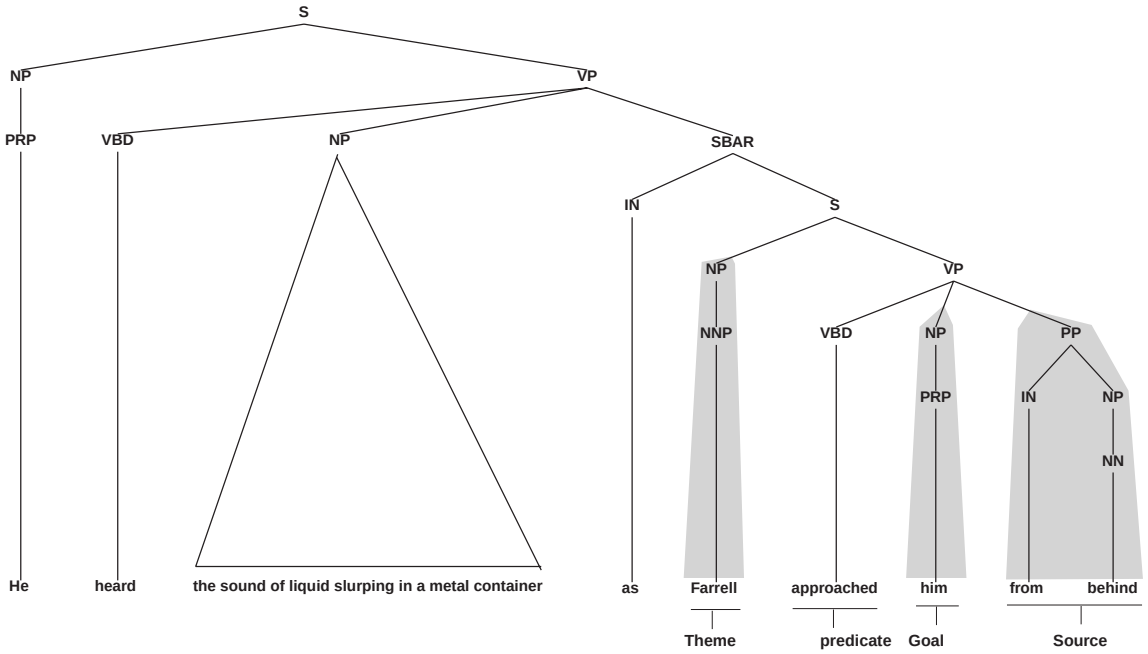

Figure 1: A sample sentence with parser output (above) and argument structure annotation (below). Parse constituents corresponding to frame elements are highlighted.

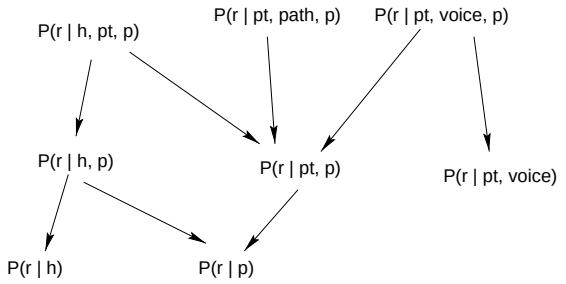

Figure 3: Backoff lattice with more specific distributions towards the top.

We applied the same system, using the same features to a preliminary release of the Propbank data. The dataset used contained annotations for 26,138 predicate-argument structures containing 65,364 individual arguments and containing examples from 1,527 lexical predicates (types). In order to provide results comparable with the statistical parsing literature, annotations from Section 23 of the Treebank were used as the test set; all other sections were included in the training set.

The system was tested under two conditions, one in which it is given the constituents which are arguments to the predicate and merely has to predict the correct role, and one in which it has to both find the arguments in the sentence and label them correctly. Results are shown in Tables 1 and 2.

Although results for Propbank are lower than for FrameNet, this appears to be primarily due to the smaller number of training examples for each predicate, rather than the difference in annotation style between the two corpora. The FrameNet data contained at least ten examples from each predicate, while 17% of the Propbank data had fewer than ten training examples. Removing these examples from the test set gives 84.1% accuracy with gold-standard parses and 80.5% accuracy with automatic parses.

As our path feature is a somewhat unusual way of looking at parse trees, its behavior in the system warrants a closer look. The path feature is most useful as a way of finding arguments in the unknown boundary condition. Removing the path feature from the known-boundary system results in only a small degradation in performance, from 82.3% to 81.7%. One reason for the relatively small impact may be sparseness of the feature — 7% of paths in the test set are unseen in training data. The most common values of the feature are shown in Table 3, where the first two rows correspond to standard subject and object positions. One reason for sparsity is seen in the third row: in the Treebank, the adjunction of an adverbial phrase or modal verb can cause an additional VP node to appear in our path feature. We tried two variations of the path feature to address this problem. The first collapses sequences of nodes with

	Accuracy		
	FrameNet	*Propbank*	*Propbank* > 10 *ex.*
Gold-standard parses		82.8	84.1
Automatic parses	82.0	79.2	80.5

Table 1: Accuracy of semantic role prediction for known boundaries — the system is given the constituents to classify.

	FrameNet		*Propbank*		*Propbank* > 10	
	Precision	*Recall*	*Precision*	*Recall*	*Precision*	*Recall*
Gold-standard parses			71.1	64.4	73.5	71.7
Automatic parses	64.6	61.2	57.7	50.0	59.0	55.4

Table 2: Accuracy of semantic role prediction for unknown boundaries — the system must identify the correct constituents as arguments and give them the correct roles.

Path	*Frequency*
VB↑VP↓NP	17.6%
VB↑VP↑S↓NP	16.4
VB↑VP↑VP↑S↓NP	7.8
VB↑VP↓PP	7.6
VB↑VP↓PP↓NP	7.3
VB↑VP↓SBAR↓S	4.3
VB↑VP↓S	4.3
VB↑VP↓ADVP	2.4
1031 others	76.0

Table 3: Common values for parse tree path in Propbank data, using gold-standard parses.

the same label, for example combining rows 2 and 3 of Table 3. The second variation uses only two values for the feature: NP under S (subject position), and NP under VP (object position). Neither variation improved performance in the known boundary condition. As a gauge of how closely the Propbank argument labels correspond to the path feature overall, we note that by always assigning the most common role for each path, for example always assigning ARG0 to the subject position, and using no other features, we obtain the correct role 69.4% of the time, vs. 82.3% for the complete system.

4 Is Parsing Necessary?

Many recent information extraction systems for limited domains have relied on finite-state systems that do not build a full parse tree for the sentence being analyzed. Among such systems, (Hobbs et al., 1997) built finite-state recognizers for various entities, which were then cascaded to form recognizers for higher-level relations, while (Ray and Craven, 2001) used low-level "chunks" from a general-purpose syntactic analyzer as observations in a trained Hidden Markov Model. Such an approach has a large advantage in speed, as the extensive search of modern statistical parsers is avoided. It is also possible that this approach may be more robust to error than parsers. Although we expect the attachment decisions made by a parser to be relevant to determining whether a constituent of a sentence is an argument of a particular predicate, and what its relation to the predicate is, those decisions may be so frequently incorrect that a much simpler system can do just as well. In this section we test this hypothesis by comparing a system which is given only a flat, "chunked" representation of the input sentence to the parse-tree-based systems described above. In this representation, base-level constituent boundaries and labels are present, but there are no dependencies between constituents, as shown by the following sample sentence:

(3) [$_{NP}$ Big investment banks] [$_{VP}$ refused to step] [$_{ADVP}$ up] [$_{PP}$ to] [$_{NP}$ the plate] [$_{VP}$ to support] [$_{NP}$ the beleaguered floor traders] [$_{PP}$ by] [$_{VP}$ buying] [$_{NP}$ big blocks] [$_{PP}$ of] [$_{NP}$ stock] , [$_{NP}$ traders] [$_{VP}$ say] .

Our chunks were derived from the Treebank trees using the conversion described by Tjong Kim Sang and Buchholz (2000). Thus, the experiments were carried out using "gold-standard" rather than automatically derived chunk boundaries, which we believe will provide an upper bound on the performance of a chunk-based system.

The information provided by the parse tree can be decomposed into three pieces: the constituent boundaries, the grammatical relationship between predicate and argument, expressed by our path feature, and the head word of each candidate constituent. We will examine the contribution of each

of these information sources, beginning with the problem of assigning the correct role in the case where the boundaries of the arguments in the sentence are known, and then turning to the problem of finding arguments in the sentence.

When the argument boundaries are known, the grammatical relationship of the the constituent to the predicate turns out to be of little value. Removing the path feature from the system described above results in only a small degradation in performance, from 82.3% to 81.7%. While the path feature serves to distinguish subjects from objects, the combination of the constituent position before or after the predicate and the active/passive voice feature serves the same purpose. However, this result still makes use of the parser output for finding the constituent's head word. We implemented a simple algorithm to guess the argument's head word from the chunked output: if the argument begins at a chunk boundary, taking the last word of the chunk, and in all other cases, taking the first word of the argument. This heuristic matches the head word read from the parse tree 77% of the the time, as it correctly identifies the final word of simple noun phrases as the head, the preposition as the head of prepositional phrases, and the complementizer as the head of sentential complements. Using this process for determining head words, the system drops to 77.0% accuracy, indicating that identifying the relevant head word from semantic role prediction is in itself an important function of the parser. This chunker-based result is only slightly lower than the 79.2% obtained using automatic parses in the known boundary condition. These results for the known boundary condition are summarized in Table 4.

Path	*Head*	*Accuracy*
gold parse	gold parse	82.3
auto parse	auto parse	79.2
not used	gold parse	81.7
not used	chunks	77.0

Table 4: Summary of results for known boundary condition

We might expect the information provided by the parser to be more important in identifying the arguments in the sentence than in assigning them the correct role. While it is easy to guess whether a noun phrase is a subject or object given only its position relative to the predicate, identifying complex noun phrases and determining whether they are arguments of a verb may be more difficult without the attachment information provided by the parser.

To test this, we implemented a system in which the argument labels were assigned to chunks, with the path feature used by the parse-tree-based system replaced by a number expressing the distance in chunks to the left or right of the predicate.

Of the 3990 arguments in our test set, only 39.8% correspond to a single chunk in the flattened sentence representation, giving an upper bound to the performance of this system. In particular, sentential complements (which comprise 11% of the data) and prepositional phrases (which comprise 10%) always correspond to more than one chunk, and therefore cannot be correctly labeled by our system which assigns roles to single chunks. In fact, this system achieves 27.6% precision and 22.0% recall.

In order to see how much of the performance degradation is caused by the difficulty of finding exact argument boundaries in the chunked representation, we can relax the scoring criteria to count as correct all cases where the system correctly identifies the first chunk belonging to an argument. For example, if the system assigns the correct label to the preposition beginning a prepositional phrase, the argument will be counted as correct, even though the system does not find the argument's righthand boundary. With this scoring regime, the chunk-based system performs at 49.5% precision and 35.1% recall, still significantly lower than the 57.7% precision/50.0% recall for exact matches using automatically generated parses. Results for the unknown boundary condition are summarized in Table 5.

	Precision	*Recall*
gold parse	71.1	64.4
auto parse	57.7	50.0
chunk	27.6	22.0
chunk, relaxed scoring	49.5	35.1

Table 5: Summary of results for unknown boundary condition

As an example for comparing the behavior of the tree-based and chunk-based systems, consider the following sentence, with human annotations showing the arguments of the predicate *support*:

(4) [$_{ARG0}$ Big investment banks] refused to step up to the plate to *support* [$_{ARG1}$ the beleaguered floor traders] [$_{MNR}$ by buying big blocks of stock] , traders say .

Our tree-based system assigned the following analysis:

(5) Big investment banks refused to step up to the plate to *support* [$_{ARG1}$ the beleaguered

floor traders] [$_{MNR}$ by buying big blocks of stock] , traders say .

In this case, the system failed to find the predicate's ARG0 relation, because it is syntactically distant from the verb *support*. The original Treebank syntactic tree contains a trace which would allow one to recover this relation, co-indexing the empty subject position of *support* with the noun phrase "Big investment banks". However, our automatic parser output does not include such traces, nor does our system make use of them. The chunk-based system assigns the following argument labels:

(6) Big investment banks refused to step up to [$_{ARG0}$ the plate] to *support* [$_{ARG1}$ the beleaguered floor traders] by buying big blocks of stock , traders say .

Here, as before, the true ARG0 relation is not found, and it would be difficult to imagine identifying it without building a complete syntactic parse of the sentence. But now, unlike in the tree-based output, the ARG0 label is mistakenly attached to a noun phrase immediately before the predicate. The ARG1 relation in direct object position is fairly easily identifiable in the chunked representation as a noun phrase directly following the verb. The prepositional phrase expressing the Manner relation, however, is not identified by the chunk-based system. The tree-based system's path feature for this constituent is VB↑VP↓PP, which identifies the prepositional phrase as attaching to the verb, and increases its probability of being assigned an argument label. The chunk-based system sees this as a prepositional phrase appearing as the second chunk after the predicate. Although this may be a typical position for the Manner relation, the fact that the preposition attaches to the predicate rather than to its direct object is not represented.

In interpreting these results, it is important to keep in mind the differences between this task and other information extraction datasets. In comparison to the domain-specific relations evaluated by the Message Understanding Conference (MUC) tasks, we have a wider variety of relations but fewer training instances for each. The relations may themselves be less susceptible to finite state methods. For example, a named-entity system which indentifies corporation names can go a long way towards finding the "employment" relation of MUC, and similarly systems for tagging genes and proteins help a great deal for relations in the biomedical domain. Both Propbank and FrameNet tend to include longer arguments with internal syntactic structure, making parsing decisions more important in finding argument boundaries. They also involve abstract relations, with a wide variety of possible fillers for each role.

5 Conclusion

Our chunk-based system takes the last word of the chunk as its head word for the purposes of predicting roles, but does not make use of the identities of the chunk's other words or the intervening words between a chunk and the predicate, unlike Hidden Markov Model-like systems such as Bikel et al. (1997), McCallum et al. (2000) and Lafferty et al. (2001). While a more elaborate finite-state system might do better, it is possible that additional features would not be helpful given the small amount of data for each predicate. By using a gold-standard chunking representation, we have obtained higher performance over what could be expected from an entirely automatic system based on a flat representation of the data.

We feel that our results show that statistical parsers, although computationally expensive, do a good job of providing relevant information for semantic interpretation. Not only the constituent structure but also head word information, produced as a side product, are important features. Parsers, however, still have a long way to go. Our results using hand-annotated parse trees show that improvements in parsing should translate directly into better semantic interpretations.

Acknowledgments This work was undertaken with funding from the Institute for Research in Cognitive Science at the University of Pennsylvania and from the Propbank project, DoD Grant MDA904-00C-2136.

References

Hiyan Alshawi, editor. 1992. *The Core Language Engine*. MIT Press, Cambridge, MA.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of COLING/ACL*, pages 86–90, Montreal, Canada.

D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *AAAI-97*, pages 598–603, Menlo Park, August. AAAI Press.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the ACL*, pages 16–23, Madrid, Spain.

Charles J. Fillmore. 1976. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech*, volume 280, pages 20–32.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, in press.

Jerry R. Hobbs, Douglas Appelt, John Bear, David Israel, Megumi Kameyama, Mark E. Stickel, and Mabry Tyson. 1997. FASTUS: A cascaded finite-state transducer for extracting information from natural-language text. In Emmanuel Roche and Yves Schabes, editors, *Finite-State Language Processing*, pages 383–406. MIT Press, Cambridge, MA.

Christopher R. Johnson, Charles J. Fillmore, Esther J. Wood, Josef Ruppenhofer, Margaret Urban, Miriam R. L. Petruk, and Collin F. Baker. 2001. The FrameNet project: Tools for lexicon building. Version 0.7, http://www.icsi.berkeley.edu/~framenet/book.html.

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, Canary Islands, Spain.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Machine Learning: Proceedings of the Eighteenth International Conference (ICML 2001)*, Stanford, California.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*, pages 591–598, Stanford, California.

Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, Ralph Weischedel, and the Annotation Group. 1998. Algorithms that learn to extract information – BBN: Description of the SIFT system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, April.

Soumya Ray and Mark Craven. 2001. Representing sentence structure in hidden markov model for information extraction. In *Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, Seattle, Washington.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, Lisbon, Portugal.