Deep Linguistic Analysis for the Accurate Identification of Predicate-Argument Relations

Yusuke Miyao
Department of Computer Science
University of Tokyo
yusuke@is.s.u-tokyo.ac.jp

Jun'ichi Tsujii
Department of Computer Science
University of Tokyo
CREST, JST
tsujii@is.s.u-tokyo.ac.jp

Abstract

This paper evaluates the accuracy of HPSG parsing in terms of the identification of predicate-argument relations. We could directly compare the output of HPSG parsing with PropBank annotations, by assuming a unique mapping from HPSG semantic representation into PropBank annotation. Even though PropBank was not used for the training of a disambiguation model, an HPSG parser achieved the accuracy competitive with existing studies on the task of identifying PropBank annotations.

1 Introduction

Recently, deep linguistic analysis has successfully been applied to real-world texts. Several parsers have been implemented in various grammar formalisms and empirical evaluation has been reported: LFG (Riezler et al., 2002; Cahill et al., 2002; Burke et al., 2004), LTAG (Chiang, 2000), CCG (Hockenmaier and Steedman, 2002b; Clark et al., 2002; Hockenmaier, 2003), and HPSG (Miyao et al., 2003; Malouf and van Noord, 2004). However, their accuracy was still below the state-of-the-art PCFG parsers (Collins, 1999; Charniak, 2000) in terms of the PARSEVAL score. Since deep parsers can output deeper representation of the structure of a sentence, such as predicate argument structures, several studies reported the accuracy of predicate-argument relations using a treebank developed for each formalism. However, resources used for the evaluation were not available for other formalisms, and the results cannot be compared with each other.

In this paper, we employ PropBank (Kingsbury and Palmer, 2002) for the evaluation of the accuracy of HPSG parsing. In the PropBank, semantic arguments of a predicate and their semantic roles are manually annotated. Since the PropBank has been developed independently of any grammar formalisms, the results are comparable with other published results using the same test data.

Interestingly, several studies suggested that the identification of PropBank annotations would require linguistically-motivated features that can be obtained by deep linguistic analysis (Gildea and Hockenmaier, 2003; Chen and Rambow, 2003). They employed a CCG (Steedman, 2000) or LTAG (Schabes et al., 1988) parser to acquire syntactic/semantic structures, which would be passed to statistical classifier as features. That is, they used deep analysis as a preprocessor to obtain useful features for training a probabilistic model or statistical classifier of a semantic argument identifier. These results imply the superiority of deep linguistic analysis for this task.

Although the statistical approach seems a reasonable way for developing an accurate identifier of PropBank annotations, this study aims at establishing a method of directly comparing the outputs of HPSG parsing with the PropBank annotation in order to explicitly demonstrate the availability of deep parsers. That is, we do not apply statistical model nor machine learning to the post-processing of the output of HPSG parsing. By eliminating the effect of post-processing, we can directly evaluate the accuracy of deep linguistic analysis.

Section 2 introduces recent advances in deep linguistic analysis and the development of semantically annotated corpora. Section 3 describes the details of the implementation of an HPSG parser evaluated in this study. Section 4 discusses a problem in adopting PropBank for the performance evaluation of deep linguistic parsers and proposes its solution. Section 5 reports empirical evaluation of the accuracy of the HPSG parser.

2 Deep linguistic analysis and semantically annotated corpora

Riezler et al. (2002) reported the successful application of a hand-crafted LFG (Bresnan, 1982) grammar to the parsing of the Penn Treebank (Marcus et al., 1994) by exploiting various techniques for robust parsing. The study was impressive because most researchers had believed that deep linguistic analysis of real-world text was impossible. Their success owed much to a consistent effort to maintain a wide-coverage LFG grammar, as well as var-

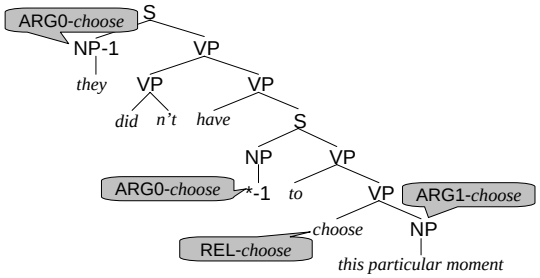

Figure 1: Annotation of the PropBank

ious techniques for robust parsing.

However, the manual development of wide-coverage linguistic grammars is still a difficult task. Recent progress in deep linguistic analysis has mainly depended on the acquisition of lexicalized grammars from annotated corpora (Xia, 1999; Chen and Vijay-Shanker, 2000; Chiang, 2000; Hockenmaier and Steedman, 2002a; Cahill et al., 2002; Frank et al., 2003; Miyao et al., 2004). This approach not only allows for the low-cost development of wide-coverage grammars, but also provides the training data for statistical modeling as a by-product. Thus, we now have a basis for integrating statistical language modeling with deep linguistic analysis. To date, accurate parsers have been developed for LTAG (Chiang, 2000), CCG (Hockenmaier and Steedman, 2002b; Clark et al., 2002; Hockenmaier, 2003), and LFG (Cahill et al., 2002; Burke et al., 2004). Those studies have opened up the application of deep linguistic analysis to practical use.

However, the accuracy of those parsers was still below PCFG parsers (Collins, 1999; Charniak, 2000) in terms of the PARSEVAL score, i.e., labeled bracketing accuracy of CFG-style parse trees. Since one advantage of deep parsers is that they can output a sort of semantic representation, e.g. predicate-argument structures, several studies have reported the accuracy of predicate-argument relations (Hockenmaier and Steedman, 2002b; Clark et al., 2002; Hockenmaier, 2003; Miyao et al., 2003). However, their evaluation employed a treebank developed for a specific grammar formalism. Hence, those results cannot be compared fairly with parsers based on other formalisms including PCFG parsers.

At the same time, following the great success of machine learning approaches in NLP, many research efforts are being devoted to developing various annotated corpora. Notably, several projects are underway to annotate large corpora with semantic information such as semantic relations of words and coreferences.

PropBank (Kingsbury and Palmer, 2002) and FrameNet (Baker et al., 1998) are large English corpora annotated with the semantic relations of words in a sentence. Figure 1 shows an example of the annotation of the PropBank. As the target text of the PropBank is the same as the Penn Treebank, a syntactic structure is given by the Penn Treebank. The PropBank includes additional annotations representing a predicate and its semantic arguments in a syntactic tree. For example, in Figure 1, REL denotes a predicate, *"choose"*, and ARGX represents its semantic arguments: *"they"* for the 0th argument (i.e., subject) and *"this particular moment"* for the 1st argument (i.e., object).

Existing studies applied statistical classifiers to the identification of the PropBank or FrameNet annotations. Similar to many methods of applying machine learning to NLP tasks, they first formulated the task as identifying in a sentence each argument of a given predicate. Then, parameters of the identifier were learned from the annotated corpus. Features of a statistical model were defined as a pattern on a partial structure of the syntactic tree output by an automatic parser (Gildea and Palmer, 2002; Gildea and Jurafsky, 2002).

Several studies proposed the use of deep linguistic features, such as predicate-argument relations output by a CCG parser (Gildea and Hockenmaier, 2003) and derivation trees output by an LTAG parser (Chen and Rambow, 2003). Both studies reported that the identification accuracy improved by introducing such deep linguistic features. Although deep analysis has not outperformed PCFG parsers in terms of the accuracy of surface structure, these results are implicitly supporting the necessity of deep linguistic analysis for the recognition of semantic relations.

However, these results do not directly reflect the performance of deep parsers. Since these corpora provide deeper structure of a sentence than surface parse trees, they would be suitable for the evaluation of deep parsers. In Section 4, we explore the possibility of using the PropBank for the evaluation of an HPSG parser.

3 Implementation of an HPSG parser

This study evaluates the accuracy of a general-purpose HPSG parser that outputs predicate argument structures. While details have been explained in other papers (Miyao et al., 2003; Miyao et al., 2004), in the remainder of this section, we briefly review the grammar and the disambiguation model of our HPSG parser.

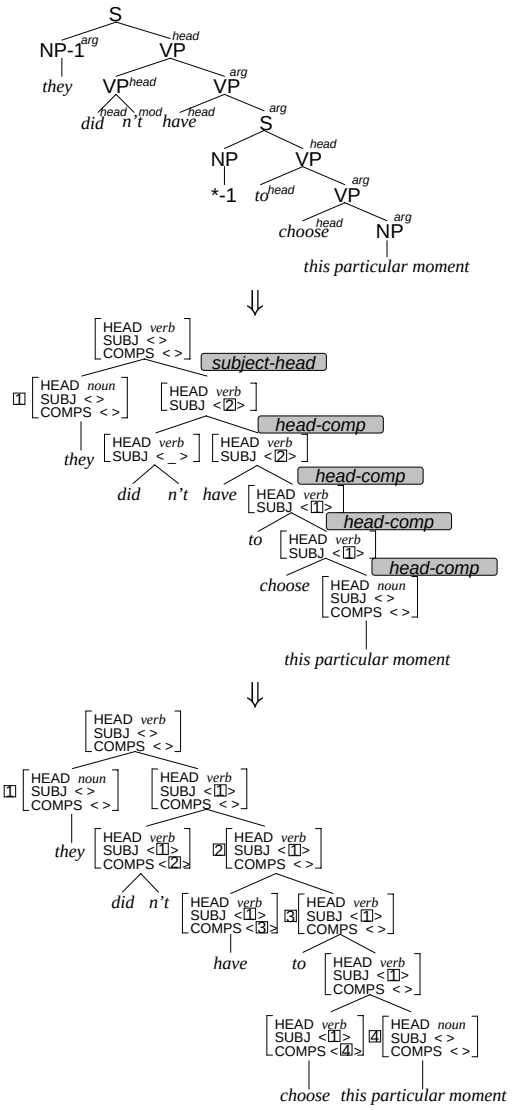

Figure 2: Extracting HPSG lexical entries from the Penn Treebank-style parse tree

3.1 Grammar

The grammar used in this paper follows the theory of HPSG (Pollard and Sag, 1994), and is extracted from the Penn Treebank (Miyao et al., 2004). In this approach, a treebank is annotated with partially specified HPSG derivations using heuristic rules. By inversely applying schemata to the derivations, partially specified constraints are percolated and integrated into lexical entries, and a large HPSG-style lexicon is extracted from the treebank.

Figure 2 shows an example of extracting HPSG lexical entries from a Penn Treebank-style parse tree. Firstly, given a parse tree (the top of the figure), we annotate partial specifications on an HPSG derivation (the middle). Then, HPSG schemata are applied to each branching in the derivation. Finally,

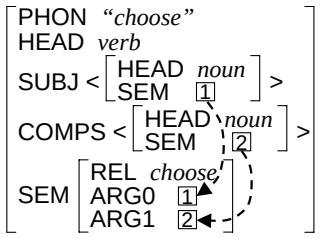

Figure 3: Mapping from syntactic arguments to semantic arguments

we get lexical entries for all of the words in the tree (the bottom).

As shown in the figure, we can also obtain complete HPSG derivation trees, i.e., an HPSG treebank. It is available for the machine learning of disambiguation models, and can also be used for the evaluation of HPSG parsing.

In an HPSG grammar, syntax-to-semantics mappings are implemented in lexical entries. For example, when we have a lexical entries for *"choose"* as shown in Figure 3, the lexical entry includes mappings from syntactic arguments (SUBJ and COMPS features) into a predicate-argument structure (ARG0 and ARG1 features). Argument labels in a predicate-argument structure are basically defined in a left-to-right order of syntactic realizations, while if we had a cue for a movement in the Penn Treebank, arguments are put in its canonical position in a predicate-argument structure.

3.2 Disambiguation model

By grammar extraction, we are able to obtain a large lexicon together with complete derivation trees of HPSG, i.e, an HPSG treebank. The HPSG treebank can then be used as training data for the machine learning of the disambiguation model.

Following recent research about disambiguation models on linguistic grammars (Abney, 1997; Johnson et al., 1999; Riezler et al., 2002; Clark and Curran, 2003; Miyao et al., 2003; Malouf and van Noord, 2004), we apply a log-linear model or maximum entropy model (Berger et al., 1996) on HPSG derivations. We represent an HPSG sign as a tuple $S = \langle l, p, s \rangle$, where l is a lexical sign of the head word, p is a part-of-speech, and s is a symbol representing the structure of the sign (mostly corresponding to nonterminal symbols of the Penn Treebank). Given an HPSG schema σ and the distance δ between the head words of the head/non-head daughter constituents, each (binary) branching of an HPSG derivation is represented as a tuple $B = \langle \sigma, \delta, l_H, p_H, s_H, l_N, p_N, s_N \rangle$, where H, N

σ	δ	l_H	p_H	s_H	l_N	p_N	s_N
√	√	√	√	√	√	√	√
√	–	√	√	√	√	√	√
√	√	–	√	√	–	√	√
√	–	–	√	√	–	√	√
√	√	√	√	–	√	√	–
√	–	√	√	–	√	√	–
√	√	–	√	–	–	√	–
√	–	–	√	–	–	√	–

σ	l_H	p_H	s_H
√	√	√	√
√	–	√	√
√	√	√	–
√	–	√	–

Table 1: Feature function templates used in the disambiguation model of HPSG parsing: for binary schema applications (top) and for unary ones (bottom)

denote head/non-head daughters.[1] Since an HPSG derivation T is represented by a set of B, a probability of T assigned to sentence W is defined as follows:

$$\begin{aligned} p(T|W) &= p(L|W)p(T|L,W) \\ &= p(L|W)\frac{1}{Z}\exp(\sum_{B\in T}\sum_i \lambda_i f_i(B)). \end{aligned}$$

$p(L|W)$ is a probability of a sequence of lexical entries, and is defined as the product of unigram probabilities $p(l_i|w_i)$, where l_i is a lexical entry assigned to word w_i. We divided the probability into $p(L|W)$ and $P(T|L,W)$ in order to accelerate the estimation of the probability model by using $p(L|W)$ as a reference distribution (Miyao et al., 2003), because the direct estimation of $P(T|W)$ was computationally expensive.

Feature function f_i returns 1 when a certain part of tuple B is observed. Table 1 lists templates of feature functions used in the disambiguation model, where a check means that the corresponding element in the tuple is seen. For example, when we have a branching $\langle$*head_comp*, 5, *trans*, VB, VP, *noun*, NNS, NP$\rangle$, [2] the following feature functions return 1, while all the other features are 0:

$\langle$*head_comp*, 5, *trans*, VB, VP, *noun*, NNS, NP$\rangle$
$\langle$*head_comp*, _, *trans*, VB, VP, *noun*, NNS, NP$\rangle$
$\langle$*head_comp*, 5, _, VB, VP, _, NNS, NP$\rangle$
$\langle$*head_comp*, _, _, VB, VP, _, NNS, NP$\rangle$
$\langle$*head_comp*, 5, *trans*, VB, _, *noun*, NNS, _$\rangle$
$\langle$*head_comp*, _, *trans*, VB, _, *noun*, NNS, _$\rangle$
$\langle$*head_comp*, 5, _, VB, _, _, NNS, _$\rangle$
$\langle$*head_comp*, _, _, VB, _, _, NNS, _$\rangle$

Given the HPSG treebank as training data, the model parameters λ_i are efficiently estimated using a dynamic programming algorithm for maximum entropy estimation (Miyao and Tsujii, 2002; Geman and Johnson, 2002).

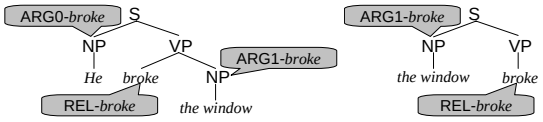

Figure 4: Annotation of an ergative verb in the PropBank

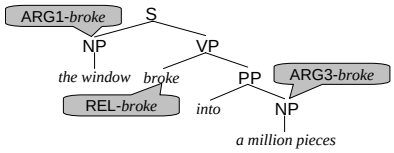

Figure 5: Annotation of another usage of "broke"

4 Evaluating HPSG parsing with semantically annotated corpora

Our study aims toward the fair evaluation of deep linguistic parsers, thus we want to directly compare the output of HPSG parsing with hand-annotated test data. However, disagreements between the output of HPSG parser and the PropBank prevents us from a direct comparison.

In the PropBank annotation, semantic arguments can occur in multiple syntactic realizations, as in the following example (Figure 4).

1. He broke the window.
2. The window broke.

In the first example, a semantic object appears in a syntactic object position, while in the second sentence it becomes the syntactic subject. This alternation is caused by two reasons: *syntactic alternations* such as passive constructions and long-distance dependencies, and *lexical alternations* such as ergative verbs. It should also be noted that the assignment of argument labels have some arbitrariness.

[1] A unary branching is represented by $\langle \sigma, l_H, p_H, s_H \rangle$.

[2] In this example, *head_comp* and *trans* stand for the Head Complement Schema and a transitive verb. In our probabilistic model, lexical entry templates are more fine-grained (as shown in Section 5, a grammar has more than 1,000 templates), while we used a simple example here.

For example, Figure 5 shows the PropBank annotation for *"The window broke into a million pieces."*, where a phrase "a million pieces" is annotated with ARG3, not with ARG2. This is because ARG2 is reserved for an *instrument* argument (e.g. "with a rock"). However, the choice of selecting ARG2 or ARG3 for "a million pieces" is arbitrary. Existing studies exploited statistical methods to mend these alternations and arbitrariness.

Basically, deep linguistic parsers derived from the Penn Treebank can handle syntactic alternations owing to trace annotation in the treebank. However, lexical alternations and arbitrariness of assignments of argument labels will be a problem when we directly compare the output of an HPSG parser with the PropBank.

However, we can see that the remaining disagreements are about the labels of argument labels. In general, we can assume that argument labels can be uniquely determined if a syntactic class of the predicate is given.[3] In the example given in Section 2, *"the window"* always occurs in the object position when *"broke"* is transitive, while it appears in the subject position when it is intransitive. Since syntactic classes are expressed by lexical entries in HPSG, this indicates that we can establish a unique mapping from an HPSG lexical entry into PropBank semantic roles.

Following this idea, we developed a mapping from HPSG argument labels into PropBank argument labels. This mapping was developed with a very simple algorithm as follows. We first computed predicate-argument structures from an HPSG treebank. We then compared the obtained predicate-argument structures with the PropBank annotations, and for each pair of a surface form of a word and its syntactic class, the mapping from argument labels of a predicate-argument structure into those of PropBank was registered. When we found a conflict, that is, multiple mappings were found for a pair, a mapping found later was simply discarded.

Our method is much simpler than existing studies, and it should be noted that PropBank was not used for training the probabilistic model or statistical identifier. This might be a handicap for our evaluation, but this method can clearly show the lower bound of the accuracy that has been attained by HPSG parsing.

[3]There exist some exceptions as follows:

- "He opened the bottles."
- "The can opener opens the bottles."

In the PropBank, "he" is assigned ARG0, while "the can opener" is assigned ARG2 (instrument).

	G_{penn}	G_{prop}
# words	8,539	8,496
# lexical entry template	1,106	1,178
# template per word	3.00	3.16
# features	50,158	52,151
Size of the training data	124 MB	131 MB
Estimation time	68 min	51 min

Table 2: Specifications of the HPSG grammar and the disambiguation model

5 Experimental results

In this section, we evaluate the accuracy of HPSG parsing using the November 2002 release of PropBank (Kingsbury and Palmer, 2002). An HPSG grammar was extracted from Section 02-21 and a disambiguation model was trained using the same data. Table 2 shows specifications of the grammar and the disambiguation model, where the size of the training data shows the file size of a compressed training data and the estimation time represents a user time required for estimating $P(T|L, W)$. We prepared two grammars for the evaluation: G_{penn} was extracted from the Penn Treebank with the original algorithm (Miyao et al., 2004), and G_{prop} was extracted using the PropBank annotations for argument/modifier distinction by a method similar to Chen and Rambow (2003). That is, constituents annotated with ARGX were treated as an argument in the grammar extraction. In G_{penn}, prepositional phrases are basically treated as modifiers since we have no cue to detect argument/modifier distinction in the original Penn Treebank. Section 02-21 was also used for developing HPSG-to-PropBank mapping. Note that the PropBank annotation was used only for this purpose, and was not used for training a statistical disambiguation model. This is very different from existing methods of identifying PropBank-style annotations where they trained the identification model using the PropBank. In the following, Section 22 of the PropBank was used for the development of the parser, while Section 23 was used for the final evaluation.

The accuracy of HPSG parsing was measured against the *core-argument* annotations (i.e., ARG0, ..., ARG5) of the PropBank. Each predicate-argument relation output by the parser was represented as a tuple $\langle pred, arg\_label, arg \rangle$, where $pred$ was a predicate, arg_label was the label of an argument position (i.e., one of ARG0, ..., ARG5), and arg was the head word of the argument of $pred$. Each tuple was compared to the annotations in the PropBank. We used a mapping table described in

	LP	*LR*	*UP*	*UR*
G_{penn}	70.3	56.0	86.7	69.2
G_{prop}	68.3	59.0	85.6	73.9
Gold parses	79.5	67.1	97.2	82.0

Table 3: Accuracy of PropBank annotations (head words of core arguments, without HPSG-to-PropBank mapping)

	LP	*LR*	*UP*	*UR*
G_{penn}	80.3	64.1	86.7	69.2
G_{prop}	79.6	68.7	85.6	73.9
Gold parses	91.2	76.9	97.2	82.0

Table 4: Accuracy of PropBank annotations (head words of core arguments, with HPSG-to-PropBank mapping)

Section 4 for mapping the argument labels of HPSG into the PropBank-style.

Table 3 shows the accuracy of semantic arguments output by the HPSG parser without mapping HPSG outputs to PropBank-style, while Table 4 shows the accuracy with the HPSG-to-PropBank mapping. LP/LR columns represent labeled precision/recall while UP/UR represent unlabeled precision/recall. "Labeled" here means the label of argument positions. That is, a predicate-argument relation was judged to be correct if $\langle pred, arg\_label, arg \rangle$ was correctly output. "Unlabeled" means that the head word of the argument was correctly output regardless of the argument position, i.e., $pred$ and arg were correctly output. The *"Gold parses"* row represents the accuracy attained when correct HPSG derivations are given. That is, it represents the accuracy when Section 23 of the HPSG treebank was given. This represents the upper bound of this measure in this evaluation.

First of all, we can see that labeled precision/recall significantly increased with the HPSG-to-PropBank mapping. This means that the low accuracy of the naive evaluation (Table 3) was mainly due to the disagreements of the representation of semantic structures.

As shown in Table 4, despite not employing the PropBank for the machine learning of a disambiguation model, the labeled precision/recall attained by G_{prop} were superior to an existing study using the Collins parser (75.9/69.6) (Gildea and Hockenmaier, 2003), and the results were approaching existing studies on the same task using a CCG parser (76.1/73.5) (Gildea and Hockenmaier, 2003). Although the results cannot directly be compared with another work using LTAG (Chen and Rambow, 2003) because their target annotations were limited to those localized in an elementary tree, considering that their target annotations were 87% of core-arguments, our results are competitive with their results (82.57/71.41).

6 Conclusion

In this paper, the accuracy of HPSG parsing was evaluated in terms of the identification of predicate-argument relations. By assuming unique mapping from HPSG predicate argument structures into the PropBank annotation of semantic arguments, we could directly compare the output of an HPSG parser with PropBank. Despite not using PropBank for the training of a disambiguation model, the HPSG parser achieved a high accuracy competitive with the previous studies on the identification of PropBank annotations. This result reveals the accurate identification of predicate-argument relations by HPSG parsing.

Although this study directly compared the HPSG output with PropBank, we may require an additional machine learning step as in the existing studies to obtain higher accuracy because the accuracy attained by gold parses showed a limitation of our approach. Another possibility is to directly extract PropBank-style semantic representations by reforming the grammar extraction algorithm (Chen and Rambow, 2003), and to estimate a disambiguation model using the PropBank.

References

Steven P. Abney. 1997. Stochastic attribute-value grammars. *Computational Linguistics*, 23(4).

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

Adam L. Berger, Stephen A. Della Pietra, and Vincent. J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Joan Bresnan, editor. 1982. *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA.

Michael Burke, Aoife Cahill, Ruth O'Donovan, Josef van Genabith, and Andy Way. 2004. Treebank-based acquisition of wide-coverage, probabilistic LFG resources: Project overview, results and evaluation. In *Proc. IJCNLP-04 Workshop "Beyond Shallow Analyses"*.

Aoife Cahill, Mairead McCarthy, Josef van Genabith, and Andy Way. 2002. Parsing with PCFGs and automatic f-structure annotation. In *Proc. 7th*

International Lexical-Functional Grammar Conference.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. NAACL-2000*, pages 132–139.

John Chen and Owen Rambow. 2003. Use of deep linguistic features for the recognition and labeling of semantic arguments. In *Proc. EMNLP 2003*.

John Chen and K. Vijay-Shanker. 2000. Automated extraction of TAGs from the Penn Treebank. In *Proc. 6th IWPT*.

David Chiang. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proc. 38th ACL*, pages 456–463.

Stephen Clark and James R. Curran. 2003. Log-linear models for wide-coverage CCG parsing. In *Proc. EMNLP 2003*, pages 97–104.

Stephen Clark, Julia Hockenmaier, and Mark Steedman. 2002. Building deep dependency structures with a wide-coverage CCG parser. In *Proc. 40th ACL*, pages 327–334.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, Univ. of Pennsylvania.

Anette Frank, Louisa Sadler, Josef van Genabith, and Andy Way. 2003. From treebank resources to LFG f-structures: Automatic f-structure annotation of treebank trees and CFGs extracted from treebanks. In Anne Abeille, editor, *Building and Using Syntactically Annotated Corpora*, pages 367–389. Kluwer Academic Publishers.

Stuart Geman and Mark Johnson. 2002. Dynamic programming for parsing and estimation of stochastic unification-based grammars. In *Proc. 40th ACL*, pages 279–286.

Daniel Gildea and Julia Hockenmaier. 2003. Identifying semantic roles using Combinatory Categorial Grammar. In *Proc. EMNLP 2003*.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proc. 40th ACL*.

Julia Hockenmaier and Mark Steedman. 2002a. Acquiring compact lexicalized grammars from a cleaner treebank. In *Proc. 3rd LREC*.

Julia Hockenmaier and Mark Steedman. 2002b. Generative models for statistical parsing with Combinatory Categorial Grammar. In *Proc. 40th ACL*, pages 335–342.

Julia Hockenmaier. 2003. Parsing with generative models of predicate-argument structure. In *Proc. 41st ACL*, pages 359–366.

Mark Johnson, Stuart Geman, Stephen Canon, Zhiyi Chi, and Stefan Riezler. 1999. Estimators for stochastic "unification-based" grammars. In *Proc. ACL '99*, pages 535–541.

Paul Kingsbury and Martha Palmer. 2002. From Treebank to PropBank. In *Proc. 3rd LREC*.

Robert Malouf and Gertjan van Noord. 2004. Wide coverage parsing with stochastic attribute value grammars. In *Proc. IJCNLP-04 Workshop "Beyond Shallow Analyses"*.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *ARPA Human Language Technology Workshop*.

Yusuke Miyao and Jun'ichi Tsujii. 2002. Maximum entropy estimation for feature forests. In *Proc. HLT 2002*.

Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. 2003. Probabilistic modeling of argument structures including non-local dependencies. In *Proc. RANLP 2003*, pages 285–291.

Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. 2004. Corpus-oriented grammar development for acquiring a Head-driven Phrase Structure Grammar from the Penn Treebank. In *Proc. IJCNLP-04*.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press.

Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell III, and Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and discriminative estimation techniques. In *Proc. 40th ACL*.

Yves Schabes, Anne Abeillé, and Aravind K. Joshi. 1988. Parsing strategies with 'lexicalized' grammars: Application to tree adjoining grammars. In *Proc. 12th COLING*, pages 578–583.

Mark Steedman. 2000. *The Syntactic Process*. The MIT Press.

Fei Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proc. 5th NLPRS*.