The Effect of Syntactic Representation on Semantic Role Labeling

Richard Johansson and **Pierre Nugues**
Lund University, Sweden
{richard, pierre}@cs.lth.se

Abstract

Almost all automatic semantic role labeling (SRL) systems rely on a preliminary parsing step that derives a syntactic structure from the sentence being analyzed. This makes the choice of syntactic representation an essential design decision. In this paper, we study the influence of syntactic representation on the performance of SRL systems. Specifically, we compare constituent-based and dependency-based representations for SRL of English in the FrameNet paradigm.

Contrary to previous claims, our results demonstrate that the systems based on dependencies perform roughly as well as those based on constituents: For the argument classification task, dependency-based systems perform slightly higher on average, while the opposite holds for the argument identification task. This is remarkable because dependency parsers are still in their infancy while constituent parsing is more mature. Furthermore, the results show that dependency-based semantic role classifiers rely less on lexicalized features, which makes them more robust to domain changes and makes them learn more efficiently with respect to the amount of training data.

1 Introduction

The role-semantic paradigm has a long and rich history in linguistics, and the NLP community has recently devoted much attention to developing accurate and robust methods for performing role-semantic analysis automatically (Gildea and Jurafsky, 2002; Litkowski, 2004; Carreras and Màrquez, 2005; Baker et al., 2007). It is widely conjectured that an increased SRL accuracy will lead to improvements in certain NLP applications, especially template-filling systems. SRL has also been used in prototypes of more advanced semantics-based applications such as textual entailment recognition.

It has previously been shown that SRL systems need a syntactic structure as input (Gildea and Palmer, 2002; Punyakanok et al., 2008). An important consideration is then what information this input should represent. By habit, most systems for automatic role-semantic analysis have used Penn-style constituents (Marcus et al., 1993) produced by Collins' (1997) or Charniak's (2000) parsers. The influence of the syntactic formalism on SRL has only been considered in a few previous articles. For instance, Gildea and Hockenmaier (2003) reported that a CCG-based parser gives improved results over the Collins parser.

Dependency syntax has only received little attention for the SRL task, despite a surge of interest in dependency parsing during the last few years (Buchholz and Marsi, 2006). Early examples of dependency-based SRL systems, which used gold-standard dependency treebanks, include Žabokrtský et al. (2002) and Hacioglu (2004). Two studies that compared the respective performances of constituent-based and dependency-based SRL systems (Pradhan et al., 2005; Swanson and Gordon, 2006), both using automatic parsers, reported that the constituent-based systems outperformed the dependency-based ones by a very wide margin. However, the figures reported in these studies can be misleading since the comparison involved a 10-year-old rule-based dependency parser versus a state-of-the-art statistical constituent parser. The recent progress in statistical dependency parsing gives grounds for a new evaluation.

© 2008. Licensed under the *Creative Commons Attribution-Noncommercial-Share Alike 3.0 Unported* license (http://creativecommons.org/licenses/by-nc-sa/3.0/). Some rights reserved.

In addition, there are a number of linguistic motivations why dependency syntax could be beneficial in an SRL context. First, complex linguistic phenomena such as *wh*-word extraction and topicalization can be transparently represented by allowing nonprojective dependency links. These links also justify why dependency syntax is often considered superior for free-word-order languages; it is even very questionable whether the traditional constituent-based SRL strategies are viable for such languages. Second, *grammatical function* such as subject and object is an integral concept in dependency syntax. This concept is intuitive when reasoning about the link between syntax and semantics, and it has been used earlier in semantic interpreters such as Absity (Hirst, 1983). However, except from a few tentative experiments (Toutanova et al., 2005), grammatical function is not explicitly used by current automatic SRL systems, but instead emulated from constituent trees by features like the constituent position and the governing category. More generally, these linguistic reasons have made a number of linguists argue that dependency structures are more suitable for explaining the syntax-semantics interface (Mel'čuk, 1988; Hudson, 1984).

In this work, we provide a new evaluation of the influence of the syntactic representation on semantic role labeling in English. Contrary to previously reported results, we show that dependency-based systems are on a par with constituent-based systems or perform nearly as well. Furthermore, we show that semantic role classifiers using a dependency parser learn faster than their constituent-based counterparts and therefore need less training data to achieve similar performances. Finally, dependency-based role classifiers are more robust to vocabulary change and outperform constituent-based systems when using out-of-domain test sets.

2 Statistical Dependency Parsing for English

Except for small-scale efforts, there is no dependency treebank of significant size for English. Statistical dependency parsers of English must therefore rely on dependency structures automatically converted from a constituent corpus such as the Penn Treebank (Marcus et al., 1993).

Typical approaches to conversion of constituent structures into dependencies are based on hand-constructed head percolation rules, an idea that has its roots in lexicalized constituent parsing (Magerman, 1994; Collins, 1997). The head rules created by Yamada and Matsumoto (2003) have been used in almost all recent work on statistical dependency parsing of English (Nivre and Scholz, 2004; McDonald et al., 2005).

Recently, Johansson and Nugues (2007) extended the head percolation strategy to incorporate long-distance links such as *wh*-movement and topicalization, and used the full set of grammatical function tags from Penn in addition to a number of inferred tags (in total 57 function tags). A dependency parser based on this syntax was used in the best-performing system in the SemEval-2007 task on Frame-semantic Structure Extraction (Baker et al., 2007), and the conversion method (in two different forms) was used for the English data in the CoNLL Shared Tasks of 2007 and 2008.

3 Automatic Semantic Role Labeling with Constituents and Dependencies

To study the influence of syntactic representation on SRL performance, we developed a framework that could be easily parametrized to process either constituent or dependency input[1]. This section describes its implementation. As the role-semantic paradigm, we used FrameNet (Baker et al., 1998).

3.1 Systems

We built SRL systems based on six different parsers. All parsers were trained on the Penn Treebank, either directly for the constituent parsers or through the LTH constituent-to-dependency converter (Johansson and Nugues, 2007). Our systems are identified as follows:

LTH. A dependency-based system using the LTH parser (Johansson and Nugues, 2008).

Malt. A dependency-based system using MaltParser (Nivre et al., 2007).

MST. A dependency-based system using MSTParser (McDonald et al., 2005).

C&J. A constituent-based system using the reranking parser (the May 2006 version) by Charniak and Johnson (2005).

Charniak. A constituent-based system using Charniak's parser (Charniak, 2000).

Collins. A constituent-based system using Collins' parser (Collins, 1997).

[1] Our implementation is available for download at `http://nlp.cs.lth.se/fnlabeler`.

MaltParser is an incremental greedy classifier-based parser based on SVMs, while the LTH parser and MSTParser use exact edge-factored search with a linear model trained using online margin-based structure learning. MaltParser and MSTParser have achieved state-of-the-art results for a wide range of languages in the 2006 and 2007 CoNLL Shared Tasks on dependency parsing, and the LTH parser obtained the best result in the 2008 CoNLL Shared Task on joint syntactic and semantic parsing. Charniak's and Collins' parsers are widely used constituent parsers for English, and the C&J parser is the best-performing freely available constituent parser at the time of writing according to published figures. Charniak's parser and the C&J parser come with a built-in part-of-speech tagger; all other systems used the Stanford tagger (Toutanova et al., 2003).

Following Gildea and Jurafsky (2002), the SRL problem is traditionally divided into two subtasks: identifying the arguments and labeling them with semantic roles. Although state-of-the-art SRL systems use sophisticated statistical models to perform these two tasks jointly (e.g. Toutanova et al., 2005, Johansson and Nugues, 2008), we implemented them as two independent support vector classifiers to be able to analyze the impact of syntactic representation on each task separately. The features used by the classifiers are traditional, although the features for the dependency-based classifiers needed some adaptation. Table 1 enumerates the features, which are described in more detail in Appendix A. The differences in the feature sets reflect the structural differences between constituent and dependency trees: The constituent-only features are based on phrase tags and the dependency-only features on grammatical functions labels.

Features	Argument identification	Argument classification
TARGETLEMMA	C,D	C,D
FEs	C,D	C,D
TARGETPOS	C,D	C,D
VOICE	C,D	C,D
POSITION	C,D	C,D
ARGWORD/POS	C,D	C,D
LEFTWORD/POS	C,D	C,D
RIGHTWORD/POS	C,D	C,D
PARENTWORD/POS	C,D	
C-SUBCAT	C	C
C-PATH	C	C
PHRASETYPE	C	C
GOVCAT	C	C
D-SUBCAT	D	D
D-PATH	D	D
CHILDDEPSET	D	D
PARENTHASOBJ	D	
RELTOPARENT	D	
FUNCTION		D

Table 1: Classifier features. The features used by the constituent-based and the dependency-based systems are marked C and D, respectively.

3.2 Dependency-based Argument Identification

The argument identification step consists of finding the arguments for a given predicate. For constituent-based SRL, this problem is formulated as selecting a subset of the constituents in a parse tree. This is then implemented in practice as a binary classifier that determines whether or not a given constituent is an argument. We approached the problem similarly in the dependency framework, applying the classifier on dependency nodes rather than constituents. In both cases, the identification step was preceded by a pruning stage that heuristically removes parse tree nodes unlikely to represent arguments (Xue and Palmer, 2004).

To score the performance of the argument identifier, traditional evaluation procedures treat the identification as a *bracketing problem*, meaning that the entities scored by the evaluation procedure are labeled snippets of text; however, it is questionable whether this is the proper way to evaluate a task whose purpose is to find *semantic relations* between logical entities. We believe that the same criticisms that have been leveled at the PARSEVAL metric for constituent structures are equally valid for the bracket-based evaluation of SRL systems. The inappropriateness of the traditional metric has led to a number of alternative metrics (Litkowski, 2004; Baker et al., 2007).

We have stuck to the traditional bracket-based scoring metric for compatibility with previous results, but since it represents the arguments as labeled spans, a conversion step is needed when using dependencies. Algorithm 1 shows how the spans are constructed from the argument dependency nodes. For each argument node, the algorithm computes the yield Y, the set of dependency nodes to include in the bracketing. This set is then partitioned into contiguous parts, which are then converted into spans. In most cases, the yield is just the subtree dominated by the argument node. However, if the argument dominates the predi-

cate, then the branch containing the predicate is removed. Also, FrameNet allows arguments to coincide with the predicate; in this case, the yield is just the predicate node.

Algorithm 1 Span creation from argument dependency nodes.

input Predicate node p, argument node a
if a does not dominate p
  $Y \leftarrow \{n; a \text{ dominates } n\}$
else if $p = a$
  $Y \leftarrow \{p\}$
else
  $c \leftarrow$ the child of a that dominates p
  $Y \leftarrow \{n; a \text{ dominates } n\} \setminus \{n; c \text{ dominates } n\}$
end if
$S \leftarrow$ partition of Y into contiguous subsets
return $\{(\text{min-index } s, \text{max-index } s); s \in S\}$

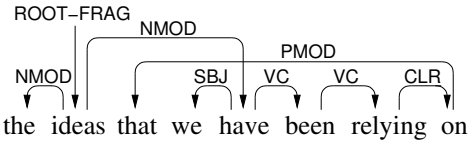

Figure 1: Example of a dependency tree containing a predicate *relying* with three arguments: *the ideas*, *we*, and *on* ... *that*.

To illustrate Algorithm 1, consider Figure 1. In this sentence, the predicate *relying* has three arguments: *the ideas*, *we*, and *on* ... *that*. The simplest of them is *we*, which does not dominate its predicate and which is not discontinuous. A more complex case is the discontinuous argument headed by *on*, where the yield $\{on, that\}$ is partitioned into two subsets that result in two separate spans. Finally, the dependency node *ideas* dominates the predicate. In this case, the algorithm removes the subtree headed by *have*, so the remaining yield is $\{the, ideas\}$.

4 Experiments

We carried out a number of experiments to compare the influence of the syntactic representation on different aspects of SRL performance. We used the FrameNet example corpus and running-text corpus, from which we randomly sampled a training and test set. The training set consisted of 134,697 predicates and 271,560 arguments, and the test set of 14,952 predicates and 30,173 arguments. This does not include null-instantiated arguments, which were removed from the training and test sets.

4.1 Argument Identification

Before evaluating the full automatic argument identification systems, we studied the effect of the span creation from dependency nodes (Algorithm 1). To do this, we measured the upper-bound recall of argument identification using the conventional span-based evaluation metric. We compared the quality of *pruned* spans (Algorithm 1) to *unpruned* spans (a baseline method that brackets the full subtree). Table 2 shows the results of this experiment. The figures show that proper span creation is essential when the traditional metrics are used: For all dependency-based systems, the upper-bound recall increases significantly. However, the dependency-based systems generally have lower figures for the upper-bound recall than constituent-based ones.

System	Pruned	Unpruned
LTH	83.9	82.1
Malt	82.1	78.3
MST	80.4	77.1
C&J	85.3	
Charniak	83.4	
Collins	81.8	

Table 2: Upper-bound recall for argument identification.

Our first experiment investigated how the syntactic representation influenced the performance of the argument identification step. Table 3 shows the result of this evaluation. As can be seen, the constituent-based systems outperform the dependency-based systems on average. However, the picture is not clear enough to draw any firm conclusion about a fundamental structural difference. There are also a number of reasons to be cautious: First, the dependency parsers were trained on a treebanks that had been automatically created from a constituent treebank, which probably results in a slight decrease in annotation quality. Second, dependency parsing is still a developing field, while constituent parsing is more mature. The best constituent parser (C&J) is a reranking parser utilizing global features, while the dependency parsers use local features only; we believe

that a reranker could be used to improve the dependency parsers as well.

System	P	R	F1
LTH	79.7	77.3	78.5
Malt	77.4	73.8	75.6
MST	73.9	71.9	72.9
C&J	81.4	77.3	79.2
Charniak	79.8	75.0	77.3
Collins	78.4	72.9	75.6

Table 3: Argument identification performance.

Differences between parsers using the same syntactic formalism are also considerable, which suggests that the attachment accuracy is probably the most important parameter when choosing a parser for this task.

4.2 Argument Classification

To evaluate the argument classification accuracies, we provided the systems with gold-standard arguments, which were then automatically classified. Table 4 shows the results.

System	Accuracy
LTH	89.6
Malt	88.5
MST	88.1
C&J	88.9
Charniak	88.5
Collins	88.3

Table 4: Semantic role classification accuracy.

Here, the situation is different: the best dependency-based system make 6.3% fewer errors than the best constituent-based one, a statistically significant difference at the 99.9% level according to a McNemar test. Again, there are no clear differences that can be attributed to syntactic formalism. However, this result is positive, because it shows clearly that SRL can be used in situations where only dependency parsers are available.

On the other hand, it may seem paradoxical that the rich set of grammatical functions used by the dependency-based systems did not lead to a clearer difference between the groups, despite the linguistic intuition that this feature should be useful for argument classification. Especially for for the second- and third-best systems (Malt and MST versus Charniak and Collins), the performance figures are almost identical. However, all systems use lexical features of the argument, and given enough training data, one may say that the grammatical function is implicitly encoded in these features. This suggests that lexical features are more important for constituent-based systems than for dependency-based ones.

4.3 Robustness of SRL Classifiers

In this section, we test the hypothesis that the SRL systems based on dependency syntax rely less heavily on lexical features. We also investigate two parameters that are influenced by lexicalization: domain sensitivity and the amount of training data required by classifiers.

Tests of Unlexicalized Models

To test the hypothesis about the reliance on lexicalization, we carried out a series of experiments where we set aside the lexical features of the argument. Table 5 shows the results.

As expected, there is a sharp drop in performance for all systems, but the results are very clear: When no argument lexical features are available, the dependency-based systems have a superior performance. The difference between MST and C&J constitutes an error reduction of 6.9% and is statistically significant at the 99.9% level.

System	Accuracy
LTH	83.0
Malt	81.9
MST	81.7
C&J	80.3
Charniak	80.0
Collins	79.8

Table 5: Accuracy for unlexicalized role classifiers. Dependency-based systems make at least 6.9% fewer errors.

Training Set Size

Since the dependency-based systems rely less on lexicalization, we can expect them to have a steeper learning curve. To investigate this, we trained semantic role classifiers using training sets of varying sizes and compared the average classification accuracies of the two groups. Figure 2 shows the reduction in classification error of the dependency-based group compared to the constituent-based group (again, all systems were lexicalized). For small training sets, the differences are large; the largest observed error reduc-

tion was 5.4% with a training set of 25,000 instances. When the training set size increases, the difference between the groups decreases. The plot is consistent with our hypothesis that the grammatical function features used by the dependency-based systems make generalization easier for statistical classifiers. We interpret the flatter learning curves for constituent-based systems as a consequence of lexicalization – these systems need more training data to use lexical information to capture grammatical function information implicitly.

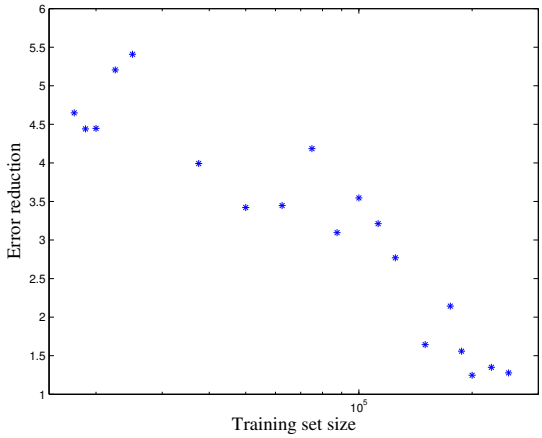

Figure 2: Error reduction of average dependency-based systems as a function of training set size.

Out-of-domain Test Sets

We finally conducted an evaluation of the semantic role classification accuracies on an out-of-domain test set: the FrameNet-annotated Nuclear Threat Initiative texts from SemEval task (Baker et al., 2007). Table 6 shows the results. This corpus contained 9,039 predicates and 15,343 arguments. The writing style is very different from the FrameNet training data, and the annotated data contain several instances of predicates and frames unseen in the training set. We thus see that all systems suffer severely from domain sensitivity, but we also see that the dependency-based systems are more resilient – the difference between MST and C&J is statistically significant at the 97.5% level and corresponds to an error reduction of 2%. The experiment reconfirms previous results (Carreras and Màrquez, 2005) that the argument classification part of SRL systems is sensitive to domain changes, and Pradhan et al. (2008) argued that an important reason for this is that the lexical features are heavily domain-dependent. Our results are consistent with this hypothesis, and suggest that the inclusion of grammatical function features is an effective way to mitigate this sensitivity.

System	Accuracy
LTH	71.1
Malt	70.1
MST	70.1
C&J	69.5
Charniak	69.3
Collins	69.3

Table 6: Classification accuracy on the NTI texts. Dependency-based systems make 2% fewer errors.

5 Discussion

We have described a set of experiments that investigate the relation between syntactic representation and semantic role labeling performance, specifically focusing on a comparison between constituent- and dependency-based SRL systems.

A first conclusion is that our dependency-based systems perform more or less as well as the more mature constituent-based systems: For the argument classification task, dependency-based systems are slightly better on average, while the constituent-based systems perform slightly higher in argument identification.

This result contrasts with previously published comparisons, which used less accurate dependency parsers, and shows that semantic analyzers can be implemented for languages where constituent parsers are not available. While traditional constituent-based SRL techniques have so far been applied to languages characterized by simple morphology and rigid word order, such as English and Chinese, we think that dependency-based SRL can be particularly useful for languages with a free word order.

For dependency-based systems, the conversion from parse tree nodes to argument spans, which are needed to use the traditional span-based evaluation method, is less trivial than in the constituent case. To make a comparison feasible, we implemented an algorithm for span creation from argument nodes. However, the fundamental problem lies in evaluation – the field needs to design new evaluation procedures that use some sort of link-based scoring method. The evaluation metrics used in the SemEval task on Frame-semantic

Structure Extraction and the 2008 CoNLL Shared Task are examples of steps in the right direction.

Our second main result is that for argument classification, dependency-based systems rely less heavily on lexicalization, and we suggest that this is because they use features based on grammatical function labels. These features make the learning curve steeper when training the classifier, and improve robustness to domain changes.

A Features Used by the Classifiers

The following subsections itemize the features used by the systems. All examples are given with respect to the sentence *she gave the horse an apple*. The constituent and dependency trees are shown in Figure 3. For this sentence, the predicate is *gave*, which has the FrameNet frame GIVING. It has three arguments: *she*, which has the DONOR role; *the horse*, the RECIPIENT; and *an apple*, the THEME.

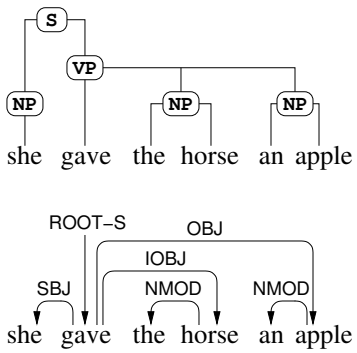

Figure 3: Examples of parse trees.

A.1 Common Features

The following features are used by both the constituent-based and the dependency-based semantic analyzers. Head-finding rules (Johansson and Nugues, 2007) were applied when heads of constituents were needed.

TARGETLEMMA. The lemma of the target word itself, e.g. *give*.

FES. For a given frame, the set of available frame elements listed in FrameNet. For instance, for *give* in the GIVING frame, we have 12 frame elements: DONOR, RECIPIENT, THEME, ...

TARGETPOS. Part-of-speech tag for the target word.

VOICE. For verbs, this feature is Active or Passive. For other types of words, it is not defined.

POSITION. Position of the head word of the argument with respect to the target word: Before, After, or On.

ARGWORD and ARGPOS. Lexical form and part-of-speech tag of the head word of the argument.

LEFTWORD and LEFTPOS. Form and part-of-speech tag of the leftmost dependent of the argument head.

RIGHTWORD and RIGHTPOS. Form and part-of-speech tag of the rightmost dependent of the argument head.

PARENTWORD and PARENTPOS. Form and part-of-speech tag of the parent node of the target.

A.2 Features Used by the Constituent-based Analyzer Only

C-SUBCAT. Subcategorization frame: corresponds to the phrase-structure rule used to expand the phrase around the target. For *give* in the example, this feature is `VP→VB NP NP`.

C-PATH. A string representation of the path through the constituent tree from the target word to the argument constituent. For instance, the path from *gave* to *she* is `↑VP-↑S-↓NP`.

PHRASETYPE. Phrase type of the argument constituent, e.g. `NP` for *she*.

GOVCAT. Governing category: this feature is either `S` or `VP`, and is found by starting at the argument constituent and moving upwards until either a `VP` or a sentence node (`S`, `SINV`, or `SQ`) is found. For instance, for *she*, this feature is `S`, while for *the horse*, it is `VP`. This can be thought of as a very primitive way of distinguishing subjects and objects.

A.3 Features Used by the Dependency-based Analyzer Only

D-SUBCAT. Subcategorization frame: the grammatical functions of the dependents concatenated. For *gave*, this feature is `SBJ+IOBJ+OBJ`.

D-PATH. A string representation of the path through the dependency tree from the target node to the argument node. Moving upwards through verb chains is not counted in this path string. In the example, the path from *gave* to *she* is `↓SBJ`.

CHILDDEPSET. The set of grammatical functions of the direct dependents of the target node. For instance, for *give*, this is { SBJ, IOBJ, OBJ }.

PARENTHASOBJ. Binary feature that is set to true if the parent of the target has an object.

RELTOPARENT. Dependency relation between the target node and its parent.

FUNCTION. The grammatical function of the argument node. For direct dependents of the target, this feature is identical to the D-PATH.

References

Baker, Collin F., Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of COLING/ACL-1998*.

Baker, Collin, Michael Ellsworth, and Katrin Erk. 2007. SemEval task 19: Frame semantic structure extraction. In *Proceedings of SemEval-2007*.

Buchholz, Sabine and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the CoNLL-X*.

Carreras, Xavier and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of CoNLL-2005*.

Charniak, Eugene and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of ACL*.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL-2000*.

Collins, Michael. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of ACL/EACL-1997*.

Gildea, Daniel and Julia Hockenmaier. 2003. Identifying semantic roles using combinatory categorial grammar. In *Proceedings of EMNLP-2003*.

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

Gildea, Daniel and Martha Palmer. 2002. The necessity of syntactic parsing for predicate argument recognition. In *Proceedings of the ACL-2002*.

Hacioglu, Kadri. 2004. Semantic role labeling using dependency trees. In *Proceedings of COLING-2004*.

Hirst, Graeme. 1983. A foundation for semantic interpretation. In *Proceedings of the ACL-1983*.

Hudson, Richard. 1984. *Word Grammar*. Blackwell.

Johansson, Richard and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Proceedings of NODALIDA 2007*.

Johansson, Richard and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of CoNLL–2008*.

Litkowski, Ken. 2004. Senseval-3 task: Automatic labeling of semantic roles. In *Proceedings of Senseval-3*.

Magerman, David M. 1994. Natural language parsing as statistical pattern recognition. Ph.D. thesis, Stanford University.

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

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

Mel'čuk, Igor A. 1988. *Dependency Syntax: Theory and Practice*. State University Press of New York.

Nivre, Joakim and Mario Scholz. 2004. Deterministic dependency parsing of English text. In *Proceedings of COLING-2004*.

Nivre, Joakim, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Pradhan, Sameer, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of ACL-2005*.

Pradhan, Sameer, Wayne Ward, and James H. Martin. 2008. Towards robust semantic role labeling. *Computational Linguistics*, 34(2):289–310.

Punyakanok, Vasin, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Swanson, Reid and Andrew S. Gordon. 2006. A comparison of alternative parse tree paths for labeling semantic roles. In *Proceedings of COLING/ACL-2006*.

Toutanova, Kristina, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL-2003*.

Toutanova, Kristina, Aria Haghighi, and Christopher D. Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of ACL-2005*.

Žabokrtský, Zdeněk, Petr Sgall, and Sašo Džeroski. 2002. A machine learning approach to automatic functor assignment in the Prague dependency treebank. In *Proceedings of LREC-2002*.

Xue, Nianwen and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP-2004*.

Yamada, Hiroyasu and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT-2003*.