Assigning Function Labels to Unparsed Text

Gabriele Musillo
Depts of Linguistics and Computer Science
musillo4@etu.unige.ch
University of Geneva
2 rue de Candolle
1211 Geneva 4
Switzerland

Paola Merlo*
Department of Linguistics
merlo@lettres.unige.ch
University of Geneva
2 rue de Candolle
1211 Geneva 4
Switzerland

Abstract

In this paper, we propose a novel solution to the problem of assigning function labels to syntactic constituents. This task is a useful intermediate step between syntactic parsing and semantic role labelling. What distinguishes our proposal from other attempts in function or semantic role labelling is that we perform the learning of function labels at the same time as parsing. We reach state-of-the-art performance both on parsing and function labelling. Our results indicate that function label information is located in the lower levels of the parse tree, and that, similarly to other function and semantic labelling results, the main difficulty lies in distinguishing constituents that bear a function label from constituents that do not.

1 Introduction

Recent successes in statistical parsing indicate that the time is ripe to solve deeper natural language understanding tasks using similar techniques (Collins 99; Charniak 00; Henderson 03). To achieve this goal, lexical semantic resources such as Framenet and Propbank are being annotated with semantic roles, as a form of shallow semantic annotation (Baker *et al.* 98; Kingsbury & Palmer 02), and a great deal of work has already been proposed to solve the problem of semantic role labelling (Gildea & Jurafsky 02; Nielsen & Pradhan 04; Xue & Palmer 04). See also the common task of (Senseval 04; CoNLL 04). Semantic information will be useful in information extraction applications (Surdeanu *et al.* 03), dialogue (Stallard 00), question-answering, and machine translation systems, among others.

A level of annotation similar to semantic role labels is already present in the the Penn Treebank (PTB) WSJ corpus (Marcus *et al.* 93) in the form of function labels. For instance, in the sentence *The Government's borrowing authority dropped at midnight Tuesday to 2.80 trillion from 2.87 trillion*[1], the constituent *The Government's borrowing authority* bears the function label SBJ and the PP *at midnight* the function label TMP. Table 1 provides the complete list of function labels in the PTB corpus. Function labels represent an intermediate level between syntactic phrase structure and semantic roles, and they have not yet been fully exploited, as observed in (Blaheta & Charniak 00). Function labels expressing grammatical roles, such as LGS, are useful in recovering argument structure. Semantically oriented labels, such as DIR, carry semantic role information.

Syntactic Labels		**Semantic Labels**	
DTV	dative	ADV	adverbial
LGS	logical subject	BNF	benefactive
PRD	predicate	DIR	direction
PUT	locative complement of *put*	EXT	extent
SBJ	surface subject	LOC	locative
VOC	vocative	MNR	manner
Miscellaneous Labels		NOM	nominal
CLF	*it*-cleft	PRP	purpose or reason
HLN	headline	TMP	temporal
TTL	title		
CLR	closely related	**Topic Labels**	
		TPC	topicalized

Table 1: Complete set of function labels in the Penn Treebank.

In this paper, we illustrate how to learn function labels during parsing, annotating parse trees with a richer set of non-terminal labels set than the standard PTB label set. Few other attempts have been made to automatically learn PTB function labels (Blaheta & Charniak 00; Blaheta 04; Jijkoun & deRijke 04)[2]. What distinguishes our

* We thank the Swiss National Science Foundation for supporting this research under grant number 101411-105286. We also would like to thank the reviewers for their helpful comments and James Henderson for his useful discussions.

[1] PTB, section 00.

[2] Recent attempts at automatically generating parsing systems consisting of a Lexical-Functional Grammar (LFG) have dealt with the problem of learning f-structures (Riezler *et al.* 02; Cahill *et al.* 04). Labels in LFG f-structures encode predicate-argument relations, similarly

proposal from previous attempts – and from most existing work on semantic role labelling – is that we perform the learning at the same time as parsing.

Our proposal tests the hypothesis that the function label of a constituent can be determined based only on the structural position it occupies in a labelled parse forest, and that a fully connected parse tree is not required to predict it. Also, it assumes that function labels depend on the same context as the usual non-terminal labels. This proposal is, therefore, more constrained than other methods that assign function or semantic role labels in two steps. These other methods have access to a full parse tree, including the context at the right of the node to be labelled. Moreover, the set of features they input to the function or semantic learner could be specialised and be very different from the input features for syntactic parsing.

It is interesting to test a more constrained hypothesis, because its results are of wide applicability. In particualr, since the function labelling is done incrementally, these results could be used in language modelling and interactive applications, where entire parse trees are not available.

2 The Learning Method

Our method is an extension of a robust statistical parser developed on the PTB, whose properties make it particularly adaptive to new tasks (Henderson 03).

2.1 The Function Label Set

The bracketting guidelines for the PTB II list 20 function labels, shown in Table 1 (Bies *et al.* 95). Based on their description in the PTB guidelines, we partition the set of function labels into four classes, as indicated in the table. Following (Blaheta & Charniak 00), we refer to the first class as syntactic function labels, and to the second class as semantic function labels. In the rest of the paper, we will ignore the other two classes, for they do not intersect with PropBank labels, and they do not form natural classes. Like previous work, we complete the sets of syntactic and semantic labels by labelling constituents that do not bear any function label with a NULL label.[3]

2.2 The Parser

Recall that our main hypothesis says that function labels can be successfully and automatically learned and recovered while parsing. It could be objected that this way the parsing task becomes more difficult. Moreover, the independence assumptions of parsing models might not be justified for this new task, rendering such models inappropriate and their parameters more difficult to estimate. It is therefore important to choose a statistical parser that can meet such objections. We use a family of statistical parsers, the Simple Synchrony Network (SSN) parsers (Henderson 03), which crucially do not make any explicit independence assumptions, and are therefore likely to adapt without much modification to the current problem. This architecture has shown state-of-the-art performance.

SSN parsers comprise two components, one which estimates the parameters of a stochastic model for syntactic trees, and one which searches for the most probable syntactic tree given the parameter estimates. As with many others statistical parsers (Collins 99; Charniak 00), the model of parsing is history-based. Its events are derivation moves. The set of well-formed sequences of derivation moves in this parser is defined by a Predictive LR pushdown automaton (Nederhof 94), which implements a form of left-corner parsing strategy.[4]

The probability of a phrase-structure tree can be equated to the probability of a finite (but unbounded) sequence of derivation moves. To bound the number of parameters, standard history-based models partition the set of well-formed sequences of transitions into equivalence classes. While such a partition makes the problem of searching for the most probable parse polynomial, it introduces hard independence assumptions: a derivation move only depends on the equivalence class to which its history belongs.

to our syntactic function labels, but no labels corresponding to the PTB semantic function labels are produced. While these attempts are indeed among the few that output richer annotations than the standard PTB labels, they can not be directly compared to our work.

[3] Strictly speaking, this label corresponds to two NULL labels: the SYN-NULL and the SEM-NULL. A node bearing the SYN-NULL label is a node that does not bear any other syntactic label. Analogously, the SEM-NULL label completes the set of semantic labels. Note that both the SYN-NULL label and the SEM-NULL are necessary, since both a syntactic and a semantic label can label a given constituent.

[4] The derivation moves include: projecting a constituent with a specified label, attaching one constituent to another, and shifting a tag-word pair onto the pushdown stack.

SSN parsers, on the other hand, do not state any explicit independence assumptions: they induce a finite history representation of an unbounded sequence of moves, so that the representation of a move $i-1$ is included in the inputs to the represention of the next move i, as explained in more detail in (Henderson 03). However, SSN parsers impose soft inductive biases to capture relevant properties of the derivation. The art of designing SSN parsers consists in selecting and introducing such biases. To this end, it is sufficient to specify features that extract some information relevant to the next derivation move from previous ones, or some set of nodes that are structurally local to the node on top of the stack. These features and these nodes are input to the computation of a hidden history representation of the sequence of previous derivation moves. Given the hidden representation of a derivation, a log-linear distribution over possible next moves is computed. Thus, the set D of structurally local nodes and the set f of predefined features determine the inductive bias of an SSN system. Unless stated otherwise, for each of the experiments reported here, the set D that is input to the computation of the history representation of the derivation moves $d_1, \ldots, d_{i-1}$ includes the following nodes: top_i, the node on top of the pushdown stack before the ith move; the left-corner ancestor of top_i; the leftmost child of top_i; and the most recent child of top_i, if any. The set of features f includes the last move in the derivation, the label or tag of top_i, the tag-word pair of the most recently shifted word, the leftmost tag-word pair that top_i dominates.

2.3 Evaluation Measures

To evaluate the performance of our function parsing experiments, we extend standard Parseval measures of labelled recall and precision to include function labels. Note that the maximal precision or recall score of function labelling is strictly smaller than one-hundred percent if the precision or the recall of the parser is less than one-hundred percent. Following (Blaheta & Charniak 00), incorrectly parsed constituents will be ignored (roughly 11% of the total) in the evaluation of the precision and recall of the function labels, but not in the evaluation of the parser. Of the correctly parsed constituents, some bear function labels, but the overwhelming majority do not bear any label, or rather, in our notation, they bear a NULL label. To avoid calculating excessively optimistic scores, constituents bearing the NULL label are not taken into consideration for computing overall recall and precision figures. NULL-labelled constituents are only needed to calculate the precision and recall of other function labels. (In other words, NULL-labelled constituents never contribute to the numerators of our calculations.) For example, consider the confusion matrix M in Table 2, which reports scores for syntactic labels of Model 1. Precision is computed as

$$\frac{\sum_{i=\text{DTV},\ldots,\text{VOC}} M[i,i]}{\sum_{j=\text{DTV},\ldots,\text{VOC}} M[\text{SUM}, j]}$$

Recall is computed by setting the denominator to $M[j, \text{SUM}_{gold}]$. Notice that $M[\text{NULL}, \text{NULL}]$ is never taken into account.

	DTV	LGS	PRD	PUT	SBJ	VOC	NULL	SUM_{gold}
DTV	0	0	0	0	0	0	12	12
LGS	0	98	0	0	0	0	19	117
PRD	0	0	482	0	0	0	62	544
PUT	0	0	0	0	0	0	7	7
SBJ	0	0	8	0	2590	0	97	2695
VOC	0	0	0	0	0	0	0	0
NULL	0	20	23	0	59	0	18825	18927
SUM	0	118	513	0	2649	0	19022	22302

Table 2: Confusion matrix for Model 1, calculated on the validation set. The NULL index in the matrix refers to the SYN-NULL label.

3 Experiments

In this section, we report the results of three experiments testing hypotheses concerning function labelling. All SSN function parsers were trained on sections 2-21 from the PTB and validated on section 24. All models are trained on parse trees whose labels include syntactic and semantic function labels. Both parsing results taking function labels into account (FLBL) and results not taking them into account (FLBL-less) are reported in Table 3. For the model that yields the best results on the validation set, we also report results on the test set, section 23 of the PTB. Results indicating performance on function labelling alone are reported in Table 4 below.

3.1 The Models

Model 1 Our hypothesis states that function labelling can be performed incrementally while parsing. First of all, we need to assess the complexity and relevance of the task. We need to

	Validation Set					
	FLBL			FLBL-less		
	F	R	P	F	R	P
Model 1	83.4	82.8	83.9	87.7	87.1	88.2
Model 2	83.8	83.2	84.4	87.9	87.3	88.5
Model 3	84.6	84.0	85.2	88.1	87.5	88.7
	Test Set					
	FLBL			FLBL-less		
Model 3	86.1	85.8	86.5	88.9	88.6	89.3

Table 3: Percentage F-measure (F), recall (R), and precision (P) of SSN parsers.

	Validation Set					
	Syntactic Labels			Semantic Labels		
	F	R	P	F	R	P
Model 1	95.3	93.9	96.7	73.1	70.2	76.3
Model 2	95.6	94.6	96.7	74.5	73.0	76.0
Model 3	95.7	95.0	96.5	80.1	77.0	83.5
	Test Set					
Model 3	96.4	95.3	97.4	86.3	82.4	90.5

Table 4: Results of different models for function labelling, separated for syntactic and semantic labels.

show that the function labelling problem is challenging, as it is not simply derivable from the parsing labels. To show this, we run a simple function parsing model that consists of the original SSN parser trained and tested on a more complex set of nonterminal labels which includes function labels. If function labelling is not easily predictable from parsing, we should have a degradation of the parser model with more complex labels.

For this model, 136 non-terminal labels were needed, in total. Of these labels, 103 consist of a standard non-terminal label and a sequence of one or more function labels. This SSN used all tag-word pairs which occur at least 200 times in the training set, resulting in 508 tag-word pairs.[5]

This first experiment yields two results that provide the starting point of our investigation, shown in the first lines of tables 3 and 4. First, it confirms that function labelling is not easily derived from parsing, as the difference in performance between function labelling (FLBL column) and parsing (FLBL-less column) in Table 3 illustrates. This motivates the task, as it shows that function labels require specific modelling to be properly learnt. The degradation in performance of the initial parser will have to be eliminated for our method to be competitive with other methods which learn function or semantic labels based on the output of a parser. These techniques do not modify the behaviour of the parser in any way, and therefore do not run the risk of improving their performance at the expense of the accuracy of the parser. Instead, we could trivially improve our function labeller by simply reducing the output of the parser to the few cases on which it is very confident.

[5]SSN parsers do not tag their input sentence. To provide the pre-terminal tags used in our first two models, we used (Ratnaparkhi 96)'s POS tagger.

	Syntactic Labels			Semantic Labels		
	F	R	P	F	R	P
BC00	95.7	95.8	95.5	79.0	77.6	80.4
B04 FT	95.9	95.3	96.4	83.4	80.3	86.7
B04 KP	98.7	98.4	99.0	78.0	73.2	83.5

Table 5: Results of Blaheta and Charniak's model for function labelling, separated for syntactic and semantic labels. The feature trees (FT) and kernel perceptrons (KP) are optimised separately for the two different sets of labels. Results are calculated on the test set of the PTB.

The second informative observation derives from a comparison with results reported by Blaheta and Charniak's paper and in Blaheta's dissertation, shown in Table 5. As can be noticed by comparing the results of Model 1 (Table 4), our results are lower.

For all these reasons, we develop two other models to improve performance, concentrating in particular on improving recall, which is particularly poor. We will see that the more function labelling improves, the more the parser improves, reducing the distance from the level of performance of the parser without function labels (Table 3, FLBL-less column).

Model 2 Our first SSN parser was designed to discriminate only among constituents bearing syntactic or semantic labels, and did not discriminate those constituents bearing the NULL label. Our second parser was designed to make such a distinction.

In this model, we hypothesize that the label NULL (i.e. the conjunction of the SYN-NULL and

SEM-NULL labels) is a mixture of types, which will be more accurately learnt separately. As can be observed by the confusion matrix in Table 2, most of the confusion occurs between the function labels and the NULL. If the label NULL is learnt more precisely, the recall of the other labels is expected to increase.

The NULL label was split into the mutually exclusive labels CLR, OBJ and OTHER. Constituents were assigned the OBJ label according to the conditions stated in (Collins 99).[6] As a result, 52 non-terminal labels were added, yielding a total of 188 non-terminals.[7]

As can be observed from the results concerning Model 2 in tables 3 and 4, our hypothesis is weakly confirmed. However, while it is true that all performance indicators increase, our method is still not as good as other methods. We think performance could be improved even further by finer-grained modelling of function labels.

Model 3 We observe that SSNs tend to project NULL labels more than any other label. Since SSNs decide the syntactic label of a non-terminal at projection, this behaviour indicates that the parser does not have enough information at this point in the parse to project the correct function label. We hypothesize that finer-grained labelling will improve parsing performance. This observation is consistent with results reported in (Klein & Manning 03), who showed that tags occurring in the Treebank are not fine-grained enough to discriminate between preterminals. For example, the tag TO labels both the preposition *to* and the infinitival marker. Extending (Klein & Manning 03)'s technique to function labelling, we split some POS tags into tags marked with semantic function labels. More precisely, the function labels DIR, LOC, MNR, PRP or TMP attached to a non-terminal were propagated down to the POS tag of the head word of the non-terminal, provided that the non-terminal is projected from the POS tag of its head.[8] As a result, 83 new part-of-speech (POS) tags were introduced to partition the original tagset of the Treebank. The vocabulary consists of 819 tag-word pairs. The non-terminal label set also includes the labels CLR, OBJ and OTHER introduced with our second model.

To provide Model 3 with tagged input sentences, we trained an SVM tagger whose features and parameters are described in detail in (Gimenez & Marquez 04). Trained on section 2-21, the tagger reaches a performance of 95.8% on the test set (section 23) of the PTB using our new tag set. As can be observed from the results concerning Model 3 in tables 3 and 4, this experiment indicates that function labelling of non-terminal labels can be done very accurately, if the parser is provided finer-grained POS tags. Concerning function labels, notice that our performance is better than the model in (Blaheta & Charniak 00) on all accounts. This is the only model which is trained on the same set of features for syntactic and semantic labels, like our model. The specialised models, reported in Table 5, optimise either their input features or their parameters separately for syntactic or semantic labels. They perform a little better than our model on syntactic labels, while they do worse than our model on semantic labels. In particular, the very time-consuming kernel models (Table 4, B04 KP) do not seem to provide any interesting added value for semantic labels. Also, the differential between the parser outputting complex labels (FLBL, Table 3) and the parser evaluated only on the standard non-terminal labels (FLBL-less, Table 3) has considerably decreased. Furthermore, the resulting parser achieves state-of-the-art parsing performance (88.9% F-measure).

4 Discussion and Comparison to Related Work

The work reported in the previous sections is directly related to a small number of other pieces of work on function labelling (Blaheta & Charniak 00; Blaheta 04), and more indirectly on all the recent work on semantic role labelling, of which we discuss the few who have reported results on function labelling (Jijkoun & deRijke 04) or who discuss issues relevant to ours here (Gildea & Palmer 02; Punyakanok *et al.* 05). In work that predates

[6] Roughly, an OBJ non-terminal is an NP, SBAR or S whose parent is an S, VP or SBAR. Any such non-terminal must not bear either syntactic or semantic function labels, or the CLR label. In addition, the first child following the head of a PP is marked with the OBJ label.

[7] This second SSN also used all tag-word pairs which occurs at least 200 times in the training set (508).

[8] In most cases, this condition was implemented by requiring that the non-terminal immediately dominates the POS tag. This condition was relaxed in a few cases to capture constructs such as coordinated PPs (e.g. $[_{\text{PP-LOC}}[_{\text{PP}}[_{\text{IN}}at]\ldots][_{\text{CC}}and][_{\text{PP}}[_{\text{IN}}in]\ldots]\ldots]$ or infinitival clauses (e.g. $[_{\text{S-PRP}}[_{\text{VP}}[_{\text{TO}}to][_{\text{VP}}[_{\text{VB}}\ldots]\ldots]\ldots]$).

the availability of Framenet and Propbank and explores many issues that also apply to semantic role labelling, (Blaheta & Charniak 00) define the task of function labelling (that they refer to as the *function tagging* task) for the first time, and highlight its relevance for NLP. Their method is in two-steps. First, they parse the PTB using a state-of-the-art parser (Charniak 00). Then, they assign function labels using features from the local context, mostly limited to two levels up the tree and only one next label. (Blaheta 04) extends on this method by developing specialised feature sets for the different subproblems of function labelling and slightly improves the results, as reported in Table 5. (Jijkoun & deRijke 04) approach the problem of enriching the output of a parser in several steps. The first step applies memory-based learning to the output of a parser mapped to dependency structures. This step learns function labels. Only results for all function labels, and not for syntactic or semantic labels alone, are provided. Although they cannot be compared directly to our results, it is interesting to notice that they are slightly better in F-measure than Blaheta's (88.5% F-measure).

In comparing different models for function and semantic role labelling, very important properties of the model are the features used by the learner and the domain of locality these features define in the tree. Both (Blaheta & Charniak 00; Blaheta 04) and (Jijkoun & deRijke 04) find that lexical heads are very useful features. (Blaheta 04) finds in particular that the head of the PP-internal noun improves results considerably for semantic function labels, which are often assigned to PPs. This is in contrast to our results. Our SSN parsers do not incorporate any inductive bias towards phrasal heads. This design was chosen because heads were not found to be useful. An SSN parser with an explicit representation of phrasal head was designed to investigate this issue directly.[9] Results are slightly worse than Model 2 above, both for syntactic and for semantic labels. (F-measure of 95.7%, recall 94.9%, and precision 96.8% for syntactic labels, F-measure of 74.0%, recall of 72.7%, and precision 75.3% for semantic labels.) While this result is in contradiction with other methods, it confirms published results on the parser we use (Henderson 03), and is therefore to be interpreted as an inherent property of the learning and parsing regime. It appears then that head-lexicalisation is not as essential as it was thought, as confirmed also by recent findings in (Bikel 04) and (Dubey & Keller 04).

The other interesting area of comparison lies in the locality of the nodes that are available to the learner. Since other methods take parsed trees as their input, in principle, they have access to all nodes in the tree. This differs crucially from assigning labels while parsing, where in most cases the parser has access only to the current level of recursion and the nodes to the right of the current node are not yet available. In practice, (Blaheta & Charniak 00; Blaheta 04) make limited use of context, and use the next label only to predict syntactic function labels. The domain of locality is therefore limited, and it defines topologies in the tree similar to ours. Such constrained methods are needed for language modelling and interactive applications.

Finally, our results provide some new insights into the discussion about the necessity of parsing for function or semantic role labelling (Gildea & Palmer 02; Punyakanok *et al.* 05). Comparing semantic role labelling based on chunked input to the better semantic role labels retrieved based on parsed trees, (Gildea & Palmer 02) conclude that parsing is necessary. In an extensive experimental investigation of the different learning stages usually involved in semantic role labelling, (Punyakanok *et al.* 05) find instead that sophisticated chunking can achieve state-of-the-art results. However, they identify pre-labelling pruning as the stage in which parsing provides an improvement that even sophisticated chunking techniques are not able to match. Pruning eliminates the nodes that almost certainly will not bear a semantic role, thus simplifying role labelling. These results are coherent with our findings. Our last experiment indicates that function labels tend to be situated very low in the tree and that tag-splitting techniques do a large amount of the work, if appropriately exploited. This suggests that most of the information is available, in principle, to a chunker, albeit a sophisticated one that recognises some phrase-internal structure. However, we also find that most errors are misclassifications between nodes that bear a function label and those that do not, affecting recall in particular. This indicates that, although a parser

[9] This model, which we do not have space to describe in detail, implements two additional derivation moves that project or attach head children of a constituent.

can identify nodes that do not need a label better than a chunker, argument identification remains the most difficult aspect of the task to be performed based on local information. Future research will lie in improving this stage of function and semantic role labelling.

5 Conclusions and Future Work

This paper has tested the hypothesis that function labelling can be successfully performed while parsing. The main result of the paper indicates that information related to function labels lies in lower level of syntactic trees and can be accurately projected from fine-grained POS tags. Future work lies in using function labels as input for semantic role labelling. Consider the semantic role labels of PropBank. Semantic function labels are straightforward predictors of the ARGM labels. Syntactic function labels, such as SBJ or LGS, encode grammatical function explicitly, and are therefore less noisy predictors of argument labels (ARG0..ARG6 in PropBank) than the indirect encoding of grammatical functions, like subject or object provided by the commonly used feature *path*.

References

(Baker *et al.* 98) Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In Christian Boitet and Pete Whitelock, editors, *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics (ACL-COLING'98)*, pages 86–90, Montreal, Canada, 1998. Morgan Kaufmann Publishers.

(Bies *et al.* 95) Ann Bies, M. Ferguson, K.Katz, and Robert MacIntyre. Bracketing guidelines for Treebank II style. Technical report, University of Pennsylvania, 1995.

(Bikel 04) Dan Bikel. Intricacies of collins' parsing model. *Computational Linguistics*, pages 479–511, 2004.

(Blaheta & Charniak 00) Don Blaheta and Eugene Charniak. Assigning function tags to parsed text. In *Proceedings of NAACL-00*, 2000.

(Blaheta 04) Don Blaheta. *Function Tagging.* Unpublished PhD thesis, Department of Computer Science, Brown University, 2004.

(Cahill *et al.* 04) Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. Long-distance dependency resolution in automatically acquired wide-coverage pcfg-based lfg approximations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL'04)*, pages 320–327, Barcelona, Spain, 2004.

(Charniak 00) Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 1st Meeting of North American Chapter of Association for Computational Linguistics*, pages 132–139, Seattle, Washington, 2000.

(Collins 99) Michael Collins. *Head-Driven Statistical Models for Natural Language Parsing.* Unpublished PhD thesis, Department of Computer Science, University of Pennsylvania, 1999.

(CoNLL 04) CoNLL. Eighth conference on computational natural language learning (conll-2004). http://cnts.uia.ac.be/conll2004, 2004.

(Dubey & Keller 04) Amit Dubey and Frank Keller. Probabilistic parsing for german using sister-head dependencies. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 96–103, Sapporo,Japan, 2004.

(Gildea & Jurafsky 02) Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3), 2002.

(Gildea & Palmer 02) Daniel Gildea and Martha Palmer. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 239–246, Philadelphia, PA, 2002.

(Gimenez & Marquez 04) Jesus Gimenez and Lluis Marquez. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, 2004.

(Henderson 03) Jamie Henderson. Inducing history representations for broad-coverage statistical parsing. In *Proceedings of the Joint Meeting of the North American Chapter of the Association for Computational Linguistics and the Human Language Technology Conference (NAACL-HLT'03)*, pages 103–110, Edmonton, Canada, 2003.

(Jijkoun & deRijke 04) Valentin Jijkoun and Maarten de Rijke. Enriching the output of a parser using memory-based learning. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Barcelona,Spain, 2004.

(Kingsbury & Palmer 02) Paul Kingsbury and Martha Palmer. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC2002)*, Las Palmas, Spain, 2002.

(Klein & Manning 03) Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the ACL*, pages 423–430, Sapporo, Japan, 2003.

(Marcus *et al.* 93) Mitch Marcus, Beatrice Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330, 1993.

(Nederhof 94) Mark Jan Nederhof. *Linguistic Parsing and Program Transformations.* Unpublished PhD thesis, Department of Computer Science, University of Nijmegen, 1994.

(Nielsen & Pradhan 04) Rodney Nielsen and Sameer Pradhan. Mixing weak learners in semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, pages 80–87, Barcelona, Spain, July 2004.

(Punyakanok *et al.* 05) V. Punyakanok, D. Roth, and W. Yih. The necessity of syntactic parsing for semantic role labeling. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI'05)*, 2005.

(Ratnaparkhi 96) Adwait Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proceedings of the First Conference on Empirical Methods in Natural Language Processing*, pages 133–142, Philadelphia, PA, 1996.

(Riezler *et al.* 02) Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 271–278, Philadephia, PA, 2002.

(Senseval 04) Senseval. Third international workshop on the evaluation of systems for the semantic analysis of text (acl 2004). http://www.senseval.org/senseval3, 2004.

(Stallard 00) David Stallard. Talk'n'travel: A conversational system for air travel planning. In *Proceedings of the Sixth Applied Natural Language Processing Conference (ANLP'00)*, pages 68–75, 2000.

(Surdeanu *et al.* 03) Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003.

(Xue & Palmer 04) Nianwen Xue and Martha Palmer. Calibrating features for semantic role labeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 88–94, 2004.