Calibrating Features for Semantic Role Labeling

Nianwen Xue
CIS, University of Pennsylvania
Levine Hall, 3330 Walnut Street
Philadelphia, PA 19104-6389
U.S.A.
xueniwen@linc.cis.upenn.edu

Martha Palmer
CIS, University of Pennsylvania
Levine Hall, 3330 Walnut Street
Philadelphia, PA 19104-6389
U.S.A.
mpalmer@linc.cis.upenn.edu

Abstract

This paper takes a critical look at the features used in the semantic role tagging literature and show that the information in the input, generally a syntactic parse tree, has yet to be fully exploited. We propose an additional set of features and our experiments show that these features lead to fairly significant improvements in the tasks we performed. We further show that different features are needed for different subtasks. Finally, we show that by using a Maximum Entropy classifier and fewer features, we achieved results comparable with the best previously reported results obtained with SVM models. We believe this is a clear indication that developing features that capture the right kind of information is crucial to advancing the state-of-the-art in semantic analysis.

1 Introduction

There has been growing interest in domain-independent semantic analysis, fed off recent efforts in semantic annotation. The availability of semantically annotated corpora such as the Proposition Banks (Kingsbury and Palmer, 2002; Xue and Palmer, 2003) and FrameNet (Baker et al., 1998) have enabled the development of a rapidly growing list of statistical semantic analyzers (Gildea and Jurafsky, 2002; Gildea and Palmer, 2002; Chen and Rambow, 2003; Pradhan et al., 2003; Pradhan et al., 2004; Sun and Jurafsky, 2004; Palmer et al., submitted). The shared task of the CoNLL-2004 is devoted to semantic role labeling (Carreras and Màrquez, 2004). Most of these systems generally take as input a syntactic parse tree and use the syntactic information as features to tag the syntactic constituents with semantic role labels. Although these systems have shown great promise, we demonstrate that the features used in previous work have not fully exploited the information that a parse tree provides. In this paper we prepose an additional set of features and show that these features lead to fairly significant improvements in the tasks we performed.

This paper is organized as follows. In the next section, we briefly describe the annotation of the Proposition Bank, the data for our automatic semantic role labeling experiments. Section 3 describes the architecture of our system. We take a critical look at the previously used features against each subtask and propose a new set of features in Section 4. Section 5 presents experimental results that show the effectiveness of these new features and a comparison with previous results. We conclude in Section 6.

2 The PropBank and Semantic Role Labeling

The PropBank adds a layer of semantic annotation to the Treebank II (Marcus et al., 1993; Marcus et al., 1994) to capture generalizations that are not adequately represented in the treebank parse trees. For example, in both *John broke the window into a million pieces yesterday* and *The window broke into a million pieces yesterday*, *the window* plays the same role with regard to the verb *break* in both sentences even though they occur in different syntactic positions. The PropBank annotation captures this regularity by assigning a semantic role label to each argument of the verb independently of its syntactic position. This means a fixed set of roles are specified for each verb and a different label is assigned to each role. In PropBank annotations, these roles are labeled with a sequence of integers, starting with 0[1] and prefixed with *ARG*. For example, the verb *break*, has four such numbered arguments: *ARG0: the breaker*, *ARG1: thing broken*, *ARG2: instrument* and *ARG3: pieces*. It is worth pointing out that even though the same numbers (0-5) are used to label the semantic roles of all verbs, these roles can only be interpreted in a verb-specific

[1] There are some exceptions.

manner. That is, an argument marked with the same number, e.g. *ARG2*, may not share any semantic similarities for different verbs.

In addition to the numbered arguments, which are considered to be core to a verb, there are also elements that are less closely related to the verb. This roughly parallels the argument/adjunct dichotomy but the distinction may not be drawn along the same lines as in the theoretic linguistics literature. These adjunct-like elements are labeled *ARGM*, followed by a secondary tag indicating the type of adjunct. For example, *yesterday* in those abovementioned sentences is not specific to the verb *break* and instead it applies to a wide variety of verbs. Therefore it will be marked as *ARGM*, followed by a secondary tag *-TMP*, indicating the temporal nature of this constituent. The secondary tags are effectively a global classification of adjunct-like elements. There are 12 secondary tags for *ARGM*s in the Proposition Bank: *DIR, LOC, MNR, TMP, EXT, REC, PRD, PRP, DIS, ADV, MOD, NEG*[2].

Some verbs require different sets of arguments for different senses, and accurately characterizing the semantic roles of their arguments necessitates first distinguishing these senses. For example, the verb "pass" takes three arguments, *legislative body*, *bill* and *law* when it means "vote and pass", while it takes only two arguments *entity moving ahead* and *entity falling behind* when it means "overtake". Each sense of this verb is likely to be realized in a set of distinct subcategorization frames and is therefore called a *frameset*.

(1) a. The congress passed the bill into law.

b. A couple of my law clerks were going to pass me in three or four years.

Semantic role tagging There are different ways to formulate the semantic role tagging task based on the annotation of the PropBank, depending on what type information one wants to learn automatically. For comparison purposes we ignore the frameset information for now, following the practice of Gildea and Palmer (Gildea and Jurafsky, 2002; Pradhan et al., 2003) and others. For each verb, we will predict the core arguments *ARG[0-5]*, as well as the secondary tags for *ARGM*s. The total tagset will be *ARG[0-5]*, *ARGa*[3] *ARGM* $\times$ *secondary tags*. There are also constituents that are not semantic arguments (by semantic arguments we mean both numbered arguments and *ARGM*s) to a given verb and we will label such constituents *NULL*. Semantic role tagging is thus an one of N classification task.

3 System architecture

Although it is conceivable that one can simply treat this as a multi-category classification problem, there are at least two reasons why such a simple approach will not work effectively. One is that for a given verb, the majority of the constituents in a syntactic tree are not its semantic arguments. When negative samples (constituents marked *NULL*) overwhelm positive samples, the current machine-learning algorithms will not be effective. The second reason, which is more subtle, is that information that is effective in separating arguments from *NULL* elements may not be as effective in distinguishing different types of arguments and *vice versa*, as we will show in our experiments. Based on these considerations, we will adopt a three-stage architecture:

Stage 1: To save training time, we use a simple algorithm to filter out constituents that are clearly not semantic arguments to the predicate in question.

Stage 2: We then classify the candidates derived from the first stage as either semantic arguments or non-arguments.

Stage 3: Finally we run a multi-category classifier to classify the constituents that are labeled as arguments into one of the classes plus *NULL*.

Pruning Algorithm The first stage is done with a simple algorithm and it is illustrated with Figure 1:

Step 1: Designate the predicate as the current node and collect its sisters (constituents attached at the same level as the predicate) unless its sisters are *coordinated* with the predicate. If a sister is a PP, also collect its immediate children.

[2]Modals (MOD) and negation markers (NEG) are clearly not adjuncts. They are included because they are critical to the interpretation of the events

[3]A limited number of arguments in the PropBank are labeled *ARGA*, meaning causative agent for induced action.

Step 2: Reset the current node to its parent and repeat Step 1 till it reaches the top level node.

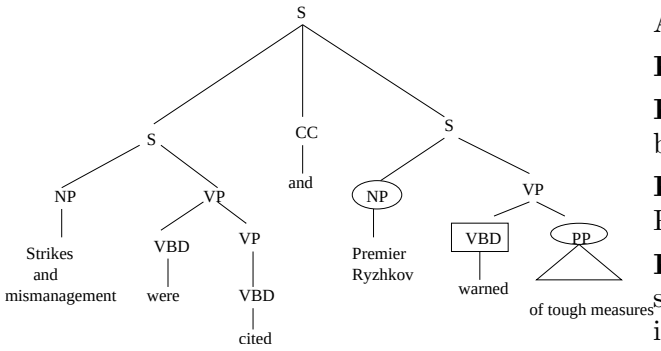

Figure 1: Pruning

For the second and third stages, we use a Maximum Entropy classifier with a tunable Gaussian prior from the Mallet Toolkit[4]. The Maximum Entropy classifier is a multi-category classifier that can be applied to the problem here in a straightforward manner without additional post-processing steps. The classifier can be tuned to minimize overfitting by adjusting the Gaussian prior. In addition, training a Maximum Entropy classifier is much faster than training SVM classifiers, even though the latter have thus far achieved the best results in semantic parsing (Pradhan et al., 2004; Sun and Jurafsky, 2004). Since our main goal in this paper is to demonstrate the crucial role of suitable features, training time will be of critical importance. We believe the features we propose will be to a large degree independent of the specific machine-learning methods and make meaningful contributions to semantic parsing regardless of which classifier we use.

In the next section, we will take a critical look at some of the "standard" features used in previous work in the context of argument identification (binary classification between arguments and non-arguments) and (multi-category) classification respectively to motivate a new set of features.

4 Features: a critical look

Most of the previous work uses the set of features proposed in Gildea and Jurafsky (Gildea and Jurafsky, 2002), in both argument identification and classification tasks. We go over these features briefly and then show that argument identification and classification call for different features.

4.1 "Standard" features

Almost all systems use the following features:

Predicate The predicate itself.

Path The minimal path from the constituent being classified to the predicate.

Phrase Type The syntactic category (NP, PP, etc.) of the constituent being classified.

Position The relative position of the constituent being classified with regard to the predicate (before or after)

Voice Whether the predicate is active or passive.

Head Word The head word of the constituent being classified.

Sub-categorization The phrase structure rule expanding the parent of the predicate.

4.2 Argument Identification

For argument identification, the *path* feature is clearly important because in a treebank-style parse tree, all constituents that are semantic arguments to a predicate are in a certain syntactic configuration. They are either sisters of the predicate or adjoined to the parent (or grandparent, great grandparent, etc.) of the predicate, within a certain syntactic domain. The path feature directly encodes this information. For example, the feature "NP↑S↓VP↓VV" suggests that the NP constituent being classified is adjoined to the VP under S and thus has a high probability of being an argument to the predicate. In contrast, an ADJP constituent that has a path "ADJP↑NP↑S↓VP↓VV" is highly unlikely to be an argument because it is deeply embedded within an NP.

In addition to the path feature, we also find that head words and their part-of-speech are also helpful presumably because constituents with certain heads are more likely to be arguments than others. However, the *position*, *predicate*, *voice*, and *subcategorization* features are not discriminative or just marginally discriminative in separating arguments from non-arguments.

The phrase type feature has little effect when used by itself, but it becomes much more effective when it is used together with the predicate. One explanation might be that the predicate itself is not a good indicator of whether

[4]http://mallet.cs.umass.edu

or not a constituent is an argument, but for a given predicate, certain types of constituents are much more likely to be arguments than others. We also find that the distance between the predicate and the constituent being classified is a good feature when the predicate is specified. The same is true for head words: they are more effective features when they are used in conjunction with the predicate. To summarize, the six feature types we use for argument identification are *path*, *head word*, *head word part-of-speech*, *predicate - phrase type combination*, *predicate-head word combination*, *distance between constituent and predicate, with the predicate specified.*

4.3 Argument Classification

In this section we show that features that are discriminative for argument identification may not be as effective for argument classification. We will examine *path*, *subcat*, *voice* and *phrase type* and *head word* features.

Path It has been pointed out by several researchers (Palmer et al., submitted) that *path* is not an very effective feature for argument classification but little explanation has been provided as to why this is the case. We suggest one reason why the path feature is not effective is that it does not discriminate between sisters that are attached at the same level in a parse tree, as illustrated in Figure 2. Both NPs following the predicate *give* will have the same path "VBD↑VP↓NP", even though they have different argument labels. A more subtle problem is that although the path feature is used to capture grammatical notions like "subject" and "object", it does not discriminate between subject of transitive verbs and subject of intransitive verbs, which are crucial notions in defining arguments of certain types of verbs.

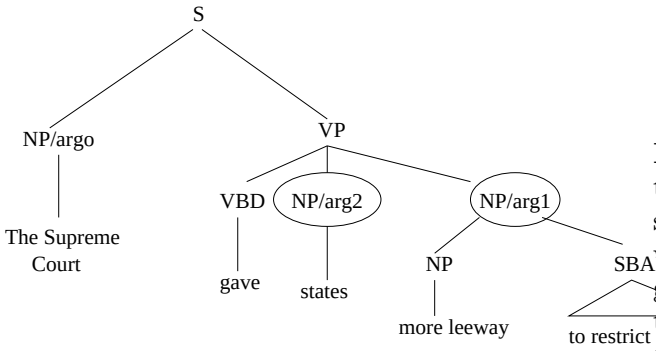

Figure 2: NPs as syntactic pivots

Subcategorization The subcategorization feature has been defined as the rule that expands the VP headed by the predicate. It is a feature shared by all constituents in a parse tree. As a result, it does not discriminate between constituents appearing in different syntactic positions.

Voice Like the subcategorization features, the voice feature is also shared by all constituents in a parse tree. The voice feature defined this way is useful in that it captures a certain bias of the existence or absence of certain types of arguments when it is set to the value *passive* or *active*, but it would be more discriminative if the position of the constituent under consideration is also specified. For example, a subject NP is very likely to be *ARG0* in an active sentence whereas it is more likely to be an ARG1 in a passive sentence.

Phrase Type The phrase type of the constituent being classified is a useful feature because different syntactic categories demonstrate tendencies of being different types of arguments, but such tendencies are even stronger when the predicate itself is also known. This is illustrated in Figure 3. The phrase type SBAR itself can be assigned any number of argument labels but its argument label is more definitive when the predicate is "ask".

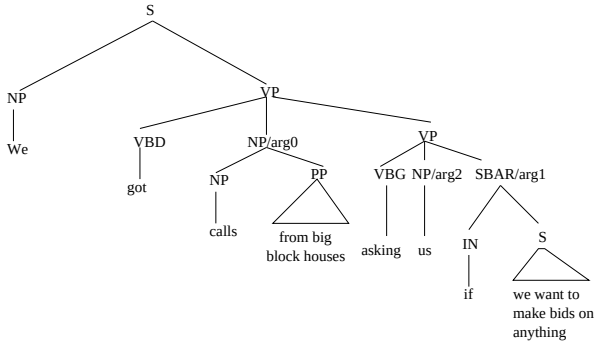

Figure 3: Lexicalizing phrase type features

Head Word It has been well-documented that the head word features are very useful, presumably because constituents with certain head words are more likely to be certain types of arguments. However, it is clear from Figure 4 that a head word is more discriminative when the predicate is also known. For example, if it is known that the head of the PP is *in* given the predicate *put*, the PP has a higher probability of being *ARG2*.

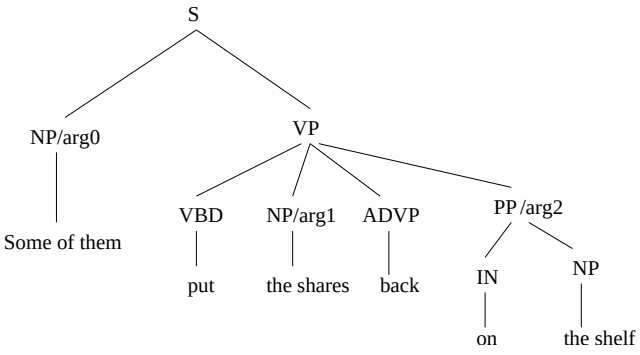

Figure 4: Lexicalizing head word features

4.3.1 New Features

We propose the following new features to address these inadequacies:

Syntactic frame To complement the path and subcat features, we propose a syntactic frame feature that varies with the constituent being classified. This feature designates the predicate and NPs as syntactic "pivots" and other constituents are defined in relation to them. For example, the syntactic frame feature of the constituent *states* in Figure 2 would be *np_v_NP_np* while the syntactic frame feature for the constituent *more leeway to restrict abortions* is *np_v_np_NP*. We also add a more general form of this feature by not specifying the syntactic category, in which case the features for the two NPs would be respectively *np_v_CUR_np* and *np_v_np_CUR*. Another variant of this feature is a lexicalized syntactic frame, where the predicate lemma is provided: *np_give_CUR_np* and *np_give_np_CUR*.

Lexicalized constituent type This feature uses the combination of the predicate lemma and the phrase type, rather than the phrase type itself, e.g. *give_np*.

Lexicalized head word We also use the predicate lemma and the head word combination as a feature, e.g. give_states.

Voice position combination We also use the voice position combination as a feature, e.g. passive_before.

Head of PP parent If the parent of the current constituent is a PP, then the head of this PP, the preposition is also used as a feature. This is a function of this version of the Propbank annotation, where the extent of an oblique argument excludes the preposition itself[5]. This is illustrated in 2, where the extent of *arg4* excludes the preposition *to*. The preposition is generally a good predictor of the argument label for the constituent.

(2) [**arg1** Sales] [**rel** rose] [**arg2** 4%] to [**arg4-to** 3.28 billion] [**argm-TMP** last year]

5 Results and Discussion

For all of our experiments, we use the version of the Propbank released on 02/04/2004. Sections 2-21 are used as training data and Section 23 is used as test data. When the gold standard parses from the Penn Treebank are used, 99.3% (or 12,628) of the 12,715 human annotated arguments are recovered after the initial pruning procedure (discussed in Section 3). Out of 242,306 non-empty constituents, only 49,469 are remaining after the initial pruning and are passed along to the next stage, which is the binary argument identification task. When the test data is parsed with the Collins parser (Collins, 1999) instead, 88.9% (or 11306) of the 12,715 arguments from the human annotation are recovered. Out of the 241,813 constituents produced by the parser, 48,870 are passed along to the next stage after the initial pruning.

5.1 Argument Identification

After this initial pruning, a binary Maximum Entropy classifier is trained to further separate semantic arguments to a verb from *NULL* elements. These experiments are performed under two conditions. Under the first condition (Row 1 in Table 1), a straightforward classification is performed to determine the effectiveness of this binary classifier. Under the second experiment condition (Row 2), a probability value is subtracted from the probability that a constituent is labeled *NULL* to maximize the recall and pass along the maximum number of arguments to the multi-category classification stage. When Gold Standard parses are used, the adjustment value is 0.80, while 0.90 is used when the Collins parser is used. Table 1 presents the results before and after this adjustment.

5.2 Argument classification

In the third and final stage, a multi-category classifier is trained to assign a semantic role label (out of 19 labels described in Section 2) to each argument of the verb. There are again two

[5] A later version, which will be released soon, will include the preposition for oblique arguments.

	Gold Standard			Collins Parser		
	p(%)	r(%)	f(%)	p(%)	r(%)	f(%)
1	95.2	92.4	93.8	84.0	78.8	81.3
2	85.5	97.2	91.0	68.0	84.1	75.1
	NULL -.80			NULL -.90		

Table 1: Argument identification results: (1) before adjustment (2) after adjustment

experiment conditions. In the first experiment, the constituents that are arguments to a verb is already known, and the task is only to assign the correct semantic role label to the constituents. In the second experiment, this same task is performed on the output of the argument identification task presented in Table 1. The same experiments are repeated using automatic parses produced by the Collins parser. The results are presented in Table 2. Row 1 presents results of all arguments when functional tags of the ArgMs are predicted, while Row 2 presents results of all arguments when functional tags are ignored. Finally Row 3 presents results when only the core arguments (numbered arguments) are calculated.

data set	GS Acc.	GS (f%)	CP (f%)
all (ArgM+)	92.95	88.51	76.21
all (ArgM)	95.42	90.55	77.76
Core	94.96 (f)	90.58	78.16

Table 2: Classification results: GS Acc. = Gold Standard accuracy, GS (f) = Gold Standard f-score, CP = Collins Parser

Feature performance Table 3 shows the performance of the new features. The baseline system uses the original features proposed in (Gildea and Palmer, 2002) and each row shows the improvement over the baseline when that feature is added to the baseline features. The results are on known (when constituents that are semantic arguments are given) and unknown (when constituents that are arguments have to be identified first before being classified) constituents respectively using Gold Standard Treebank parses. It is clear that the syntactic frame feature results in the most improvement (more than 1.7%) over the baseline, with the head of the PP parent feature being a close second. It is also worth noting that although the feature combining position and voice results in an improvement when the constituents are known, it actually results in a small loss when the constituents are unknown. This indicates that the slight change in the classification task (for classification of unknown constituents, an additional category *NULL* is added) could change the feature performance. The last three features are from (Pradhan et al., 2004), and they also result in an improvement in performance.

Features	accu.	Gold(f)
baseline	88.09	82.89
syn frame	89.82	84.64
prd-hw	88.69	83.77
prd-pt	89.12	83.81
v-p	88.44	**82.57**
PP parent	89.53	84.34
First word	88.60	83.01
Last word	88.64	83.51
Left sister	89.20	83.74
all	92.95	88.51

Table 3: Results with different feature sets

5.3 Comparison with other systems

Rapid progress has been made in semantic role labeling since the Propbank annotation became first available in 2002. The progress can be attributed to better modeling techniques, more relevant features and in a small measure, cleaner annotation. The first system trained on the Propbank is by Gildea and Palmer (2002), who reported 82.8% in accuracy on Gold Standard parses when the constituents that are semantic arguments are given, 67.6% and 53.6% (f-measure) using Gold Standard and automatic parses respectively when the constituents for the arguments have to be first identified. Since then, various degrees of improvement have been reported (Gildea and Hockenmaier, 2003; Pradhan et al., 2003; Chen and Rambow, 2003). As far as we know the best results so far are reported by (Pradhan et al., 2004), where a wide range of features, including features extracted from named entities, verb clusters and verb senses, temporal cue words, dynamic context, are tested with an SVM classifier. Their system achieved an accuracy of 93.0% on known constituents and 89.4% (f-measure) on unknown constituents using Gold Standard parses. They did not report results that use automatic parses with this version of the data, but using a previous version of the data, they reported an f-score of 79.4% using automatic parses (Charniak, 2001). By carefully designing features that can all be directly extracted from the treebank parse trees, our system achieved very com-

parable results using a Maxent classifier and a much smaller feature set: 92.95% on known constituents, 88.51% on unknown constituents and 76.21% when the Collins parser is used. The results on known constituents are almost identical and the larger difference when automatic parses are used could be attributed to the different parsers, as we used output from an earlier version of the Collins parser.

6 Conclusions and Future work

This paper takes a critical look at the features used in the semantic role tagging literature and show that the information in the input, generally a syntactic parse tree, has yet to be fully exploited. We propose an additional set of features and our experiments show that these features lead to fairly significant improvements in the tasks we performed. We further show that different features are needed for different subtasks. Finally, we show that using a maximum entropy classifier and fewer features, we achieved results that are comparable to the best previously reported results obtained with SVM models. We believe this is a clear indication that developing features that capture the right kind of information is crucial to advancing the state-of-the-art in semantic analysis. We also believe that the features we proposed here are to a large extent complementary to those proposed in a recent work by Pradhan et al (2004) and we intend to incorporate them in our system.

7 Acknowledgements

We would like to thank Scott Cotton for providing the PropBank API [6] which greatly simplifies the implementation of our system. This work is funded in part by the DOD via grant MDA904-02-C-0412, and in part by the NSF ITR via grant 130-1303-4-541984-XXXX-2000-1070. .

References

C. Baker, C. Fillmore, and J. Lowe. 1998. The berkeley framenet project. In *Proceedings of COLING-ACL*, Singapore.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling. In *Proceedings of CoNLL 2004*.

E. Charniak. 2001. Immediate-head Parsing for Language Models. In *ACL-01*.

John Chen and Owen Rambow. 2003. Use of Deep Linguistic Features for the Recognition and Labeling of Semantic Arguments. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, Sapporo, Japan.

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

Dan Gildea and Julia Hockenmaier. 2003. Identifying Semantic Roles Using Combinatory Categorial Grammar. In *EMNLP-03*, Sapporo, Japan.

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

Dan Gildea and Martha Palmer. 2002. The Necessity of Parsing for Predicate Argument Recognition. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics*, Philadelphia, PA.

Paul Kingsbury and Martha Palmer. 2002. From Treebank to PropBank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC2002)*, Las Palmas, Spain.

M. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, et al. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *Proc of ARPA speech and Natural language workshop*.

Martha Palmer, Dan Gildea, and Paul Kingsbury. submitted. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*.

Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2003. Semantic Role Parsing: Adding Semantic Structure to Unstructured Text. In *Proceedings of the International Conference on Data Mining (ICDM-2003)*.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, and Daniel Jurafsky. 2004. Shallow Semantic Parsing Using Support Vector Machines. In *Proceedings of NAACL-HLT 2004*, Boston, Mass.

Honglin Sun and Daniel Jurafsky. 2004. Shallow semantic parsing of chinese. In *Proceedings of NAACL 2004*, Boston, USA.

Nianwen Xue and Martha Palmer. 2003. An-

[6] Available at http://chronis.philli.net/ scott/code

notating the Propositions in the Penn Chinese Treebank. In *The Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing*, Sapporo, Japan.