# Using LTAG-Based Features for Semantic Role Labeling

**Yudong Liu** and **Anoop Sarkar**
Computing Science Department
Simon Fraser University
British Columbia, Canada, V5A 1S6
yudongl,anoop@cs.sfu.ca

## Abstract

Semantic role labeling (SRL) methods typically use features from syntactic parse trees. We propose a novel method that uses Lexicalized Tree-Adjoining Grammar (LTAG) based features for this task. We convert parse trees into LTAG derivation trees where the semantic roles are treated as hidden information learned by supervised learning on annotated data derived from PropBank. We extracted various features from the LTAG derivation trees and trained a discriminative decision list model to predict semantic roles. We present our results on the full CoNLL 2005 SRL task.

## 1 Introduction

Semantic role labeling (SRL) is a natural extension of the syntactic parsing task. In SRL, particular syntactic constituents in a parse tree for a sentence are identified with semantic roles. The labels assigned to various types of arguments and adjuncts differ in different annotation schemes. In this paper, we use the PropBank corpus of predicate-argument structures (Palmer, Gildea and Kingsbury, 2005). We assume we are given a syntactic parse tree and a particular predicate in the sentence for which we then identify the arguments and adjuncts and their labels. In this paper we compare two models for the identification of semantic role labels in a parse tree: A model that uses a path in the parse tree (or the derived tree in TAG terminology) and various associated features related to this, and we compare this model with a model that converts the syntactic parse tree into a Lexicalized Tree-Adjoining Grammar (LTAG) derivation tree and uses features extracted from the elementary trees and the LTAG derivation tree.

In each model the features of that model are used in a discriminative model for semantic role labeling. The model is a simple decision list learner that uses tree patterns extracted from the LTAG derivation trees in order to classify constituents into their semantic roles. We present results on the full CoNLL 2005 SRL task (Carreras and Màrquez, 2005) a dataset built by combining the Treebank and parser data with the PropBank annotations.

## 2 Background about SRL

A semantic role is defined to be the relationship that a syntactic constituent has with the predicate. For example, the following sentence, taken from the PropBank corpus, shows the annotation of semantic roles:

> [A0 *Late buying*] [V *gave*] [A2 *the Paris Bourse*] [A1 *a parachute*] [AM-TMP *after its free fall early in the day*].

Here, the arguments for the predicate *gave* are defined in the PropBank Frame Scheme (Palmer, Gildea and Kingsbury, 2005) as:

| | |
|---|---|
| V: verb | A2: beneficiary |
| A0: giver | AM-TMP: temporal |
| A1: thing given | |

Recognizing and labeling semantic arguments is a key task for answering "*Who*", "*When*","*What*", "*Where*", "*Why*", etc. questions in Information Extraction, Question Answering, Summarization (Melli et al, 2005), and, in general, in all NLP tasks in which some kind of semantic interpretation is needed.

Most previous research treats the semantic role labeling task as a classification problem, and divides it into two phases: *argument identification* and *argument classification*. Argument identification involves classifying each syntactic element in a sentence into either an argument or a non-argument. Argument classification involves classifying each argument identified into a specific semantic role. A variety of machine learning methods have been applied to this task. One of the most important steps in building an accurate classifier is feature selection. Different from the widely used



*Proceedings of the 8th International Workshop on Tree Adjoining Grammar and Related Formalisms*, pages 127–132,
Sydney, July 2006. 

feature functions that are based on the syntactic parse tree (Gildea and Jurafsky, 2002), we explore the use of LTAG-based features in a simple discriminative decision-list learner.

## 3 LTAG Based Feature Extraction

In this section, we introduce the main components of our system. First, we do a pruning on the given parse trees with certain constraints. Then we decompose the pruned parse trees into a set of LTAG elementary trees. For each constituent in question, we extract features from its corresponding derivation tree. We train using these features in a decision list model.

### 3.1 Pruning the Parse Trees

Given a parse tree, the pruning component identifies the predicate in the tree and then only admits those nodes that are sisters to the path from the predicate to the root. It is commonly used in the SRL community (cf. (Xue and Palmer, 2004)) and our experiments show that 91% of the SRL targets can be recovered despite this aggressive pruning. There are two advantages to this pruning: the machine learning method used for prediction of SRLs is not overwhelmed with a large number of non-SRL nodes; and the process is far more efficient as 80% of the target nodes in a full parse tree are pruned away in this step. We make two enhancements to the pruned Propbank tree: we enrich the sister nodes with their head information, which is a part-of-speech tag and word pair: $\langle t, w \rangle$ and PP nodes are expanded to include the NP complement of the PP (including the head information). Note that the target SRL node is still the PP. Figure 1 shows the pruned parse tree for a sentence from PropBank section 24.

### 3.2 LTAG-based Decomposition

As next step, we decompose the pruned tree around the predicate using standard head-percolation based heuristic rules[1] to convert a Treebank tree into a LTAG derivation tree. We do not use any sophistical adjunct/argument or other extraction heuristics using empty elements (as we don't have access to them in the CoNLL 2005 data). Also, we do not use any substitution nodes in our elementary trees: instead we exclusively use adjunction or sister adjunction for the attachment of sub-derivations. As a result the root node in an LTAG derivation tree is a *spinal* elementary tree and the derivation tree provides the path from the predicate to the constituent in question. Figure 2 shows the resulting elementary tree after decomposition of the pruned tree. For each of the elementary trees we consider their labeling in the derivation tree to be their semantic role labels from the training data. Figure 3 is the derivation tree for the entire pruned tree.

Note that the LTAG-based decomposition of the parse tree allows us to use features that are distinct from the usual parse tree path features used for SRL. For example, the typical parse tree feature from Figure 2 used to identify constituent *(NP (NN terminal))* as A0 would be the parse tree fragment: $NP \uparrow NP \downarrow SBAR \downarrow S \downarrow VP \downarrow S \downarrow VP \downarrow VBG$ *cover* (the arrows signify the path through the parse tree). Using the LTAG-based decomposition means that our SRL model can use any features from the derivation tree such as in Figure 2, including the elementary tree shapes.

### 3.3 Decision List Model for SRL

Before we train or test our model, we convert the training, development and test data into LTAG derivation trees as described in the previous section. In our model we make an independence assumption that each semantic role is assigned to each constituent independently, conditional only on the path from the predicate elementary tree to the constituent elementary tree in the derivation tree. Different elementary tree siblings in the LTAG derivation tree do not influence each other in our current models. Figure 4 shows the different derivation trees for the target constituent *(NP (NN terminal))*: each providing a distinct semantic role labeling for a particular constituent. We use a decision list learner for identifying SRLs based on LTAG-based features. In this model, LTAG elementary trees are combined with some distance information as features to do the semantic role labeling. The rationale for using a simple DL learner is given in (Gildea and Jurafsky, 2002) where essentially it based on their experience with the setting of backoff weights for smoothing, it is stated that the most specific single feature matching the training data is enough to predict the SRL on test data. For simplicity, we only consider one intermediate elementary tree (if any) at one time instead of multiple intermediate trees along the path from the predicate to the argument.

[1] using http://www.isi.edu/∼chiang/software/treep/treep.html

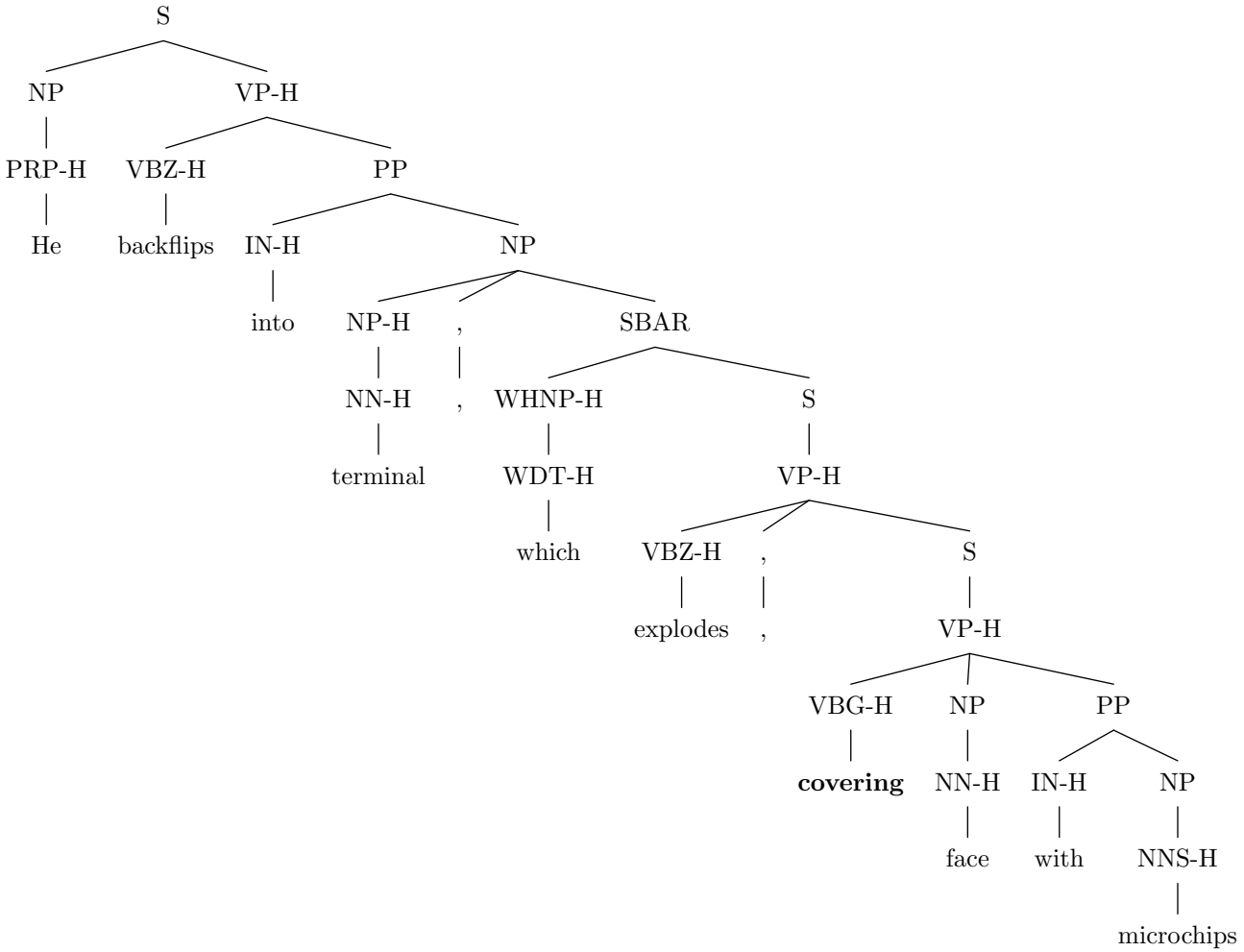


Figure 1: The pruned tree for the sentence "*He backflips into a desktop computer terminal, which explodes, covering Huntz Hall 's face with microchips.*"

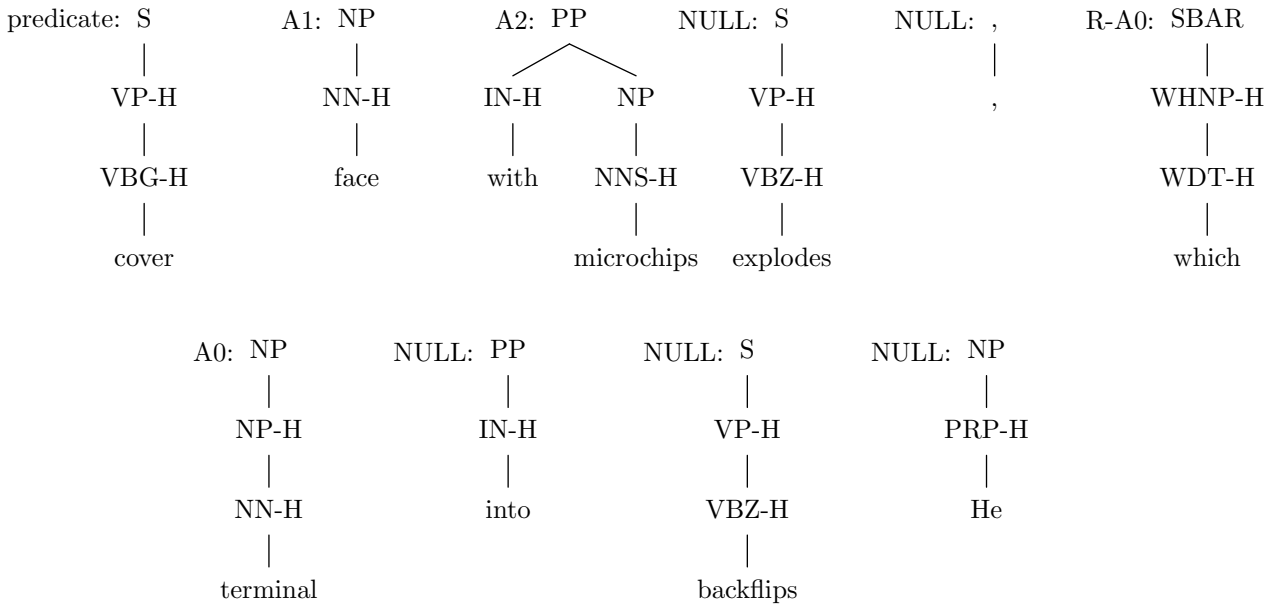


Figure 2: The resulting elementary trees after decomposition of the pruned tree.

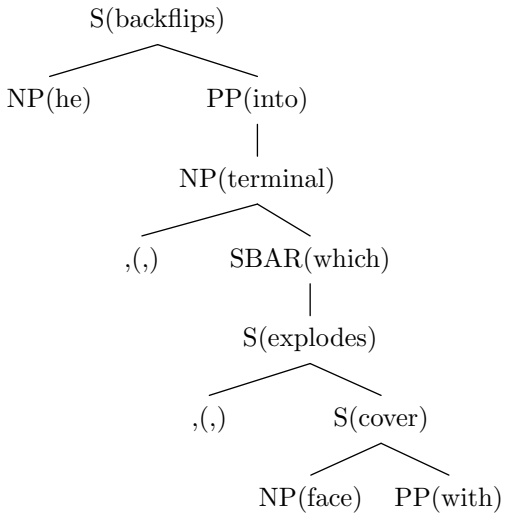


Figure 3: The LTAG derivation tree (with no semantic role labels) corresponding to the pruned tree.

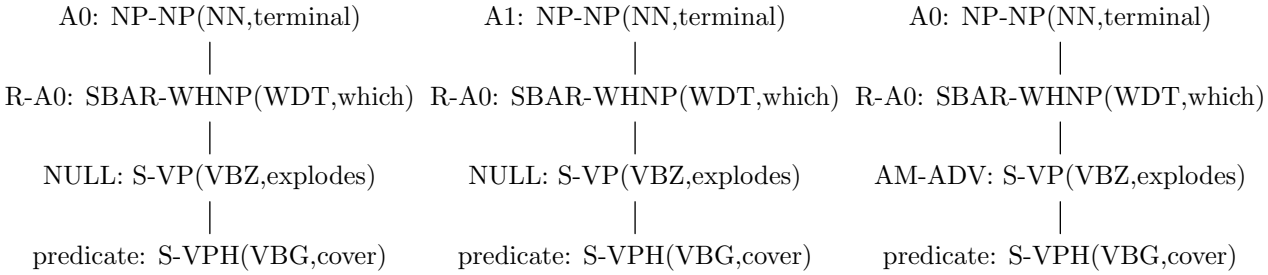


Figure 4: Different LTAG derivation trees corresponding to different assignments of semantic roles to constituents. The constituent in question is *(NP (NN terminal))*.

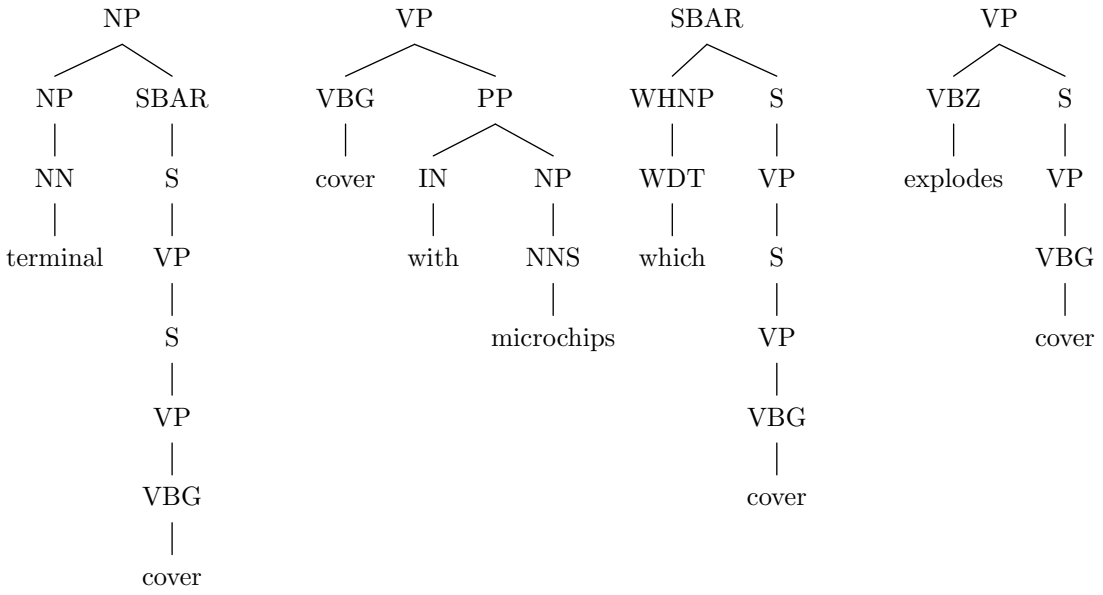


Figure 5: Tree patterns in tree pattern matching

The input to the learning algorithm is labeled examples of the form $(\mathbf{x_i}, y_i)$. $y_i$ is the label (either NULL for no SRL, or the SRL) of the $i$th example. $\mathbf{x_i}$ is a feature vector $\langle P, A, Dist, Position, R\text{-}type, t_i \in t_I, Dist_{t_i} \rangle$, where $P$ is the predicate elementary tree, $A$ is the tree for the constituent being labeled with a SRL, $t_I$ is a set of intermediate elementary trees between the predicate tree and the argument tree. Each $P, A, I$ tree consists of the elementary tree template plus the tag, word pair: $\langle t, w \rangle$.

All possible combinations of fully-lexicalized/postag/un-lexicalized elementary trees are used for each example. $Dist$ and $Dist_{t_i}$ denote the distance to the predicate from the argument tree and the intermediate elementary tree respectively. $Position$ is interpreted as the position that the target is relative to the predicate. $R\text{-}type$ denotes the relation type of the predicate and the target constituent. 3 types are defined: if the predicate dominates (directly or undirectly) the argument in the derivation tree, we have the relation of type-1; if the other way around, the argument dominates (directly or undirectly) the predicate then we have the relation of type-2; and finally type-3 means that neither the predicate or the argument dominate each other in the derivation tree and instead are dominated (again, directly or indirectly) by another elementary tree.

The output of the learning algorithm is a function $h(\mathbf{x}, y)$ which is an estimate of the conditional probability $p(y \mid \mathbf{x})$ of seeing SRL $y$ given pattern $\mathbf{x}$. $h$ is interpreted as a decision list of rules $x \Rightarrow y$ ranked by the score $h(\mathbf{x}, y)$. In testing, we simply pick the first rule that matches the particular test example $\mathbf{x}$. We trained different models using the same learning algorithm. In addition to the LTAG-based method, we also implemented a pattern matching based method on the derived (parse) tree using the same model. In this method, instead of considering each intermediate elementary tree between the predicate and the argument, we extract the whole path from the predicate to the argument. So the input is more like a tree than a discrete feature vector. Figure 5 shows the patterns that are extracted from the same pruned tree.

## 4 Experiments and Results

We use the PropBank corpus of predicate-argument structures (Palmer, Gildea and Kingsbury, 2005) as our source of annotated data for the SRL task. However, there are many different ways to evaluate performance on the PropBank, leading to incomparable results. To avoid such a situation, in this paper we use the CoNLL 2005 shared SRL task data (Carreras and Màrquez, 2005) which provides a standard train/test split, a standard method for training and testing on various problematic cases involving coordination. However, in some cases, the CoNLL 2005 data is not ideal for the use of LTAG-based features as some "deep" information cannot be recovered due to the fact that trace information and other empty categories like PRO are removed entirely from the training data. As a result some of the features that undo long-distance movement via trace information in the TreeBank as used in (Chen and Rambow, 2003) cannot be exploited in our model. Our results are shown in Table 1. Note that we test on the gold standard parse trees because we want to compare a model using features from the derived parse trees to the model using the LTAG derivation trees.

| dev = Sec24<br>test = Sec23 | p(%) | r(%) | f(%) |
|---|---|---|---|
| M1: dev | 78.42 | 77.03 | 77.72 |
| M1: test | 80.52 | 79.40 | 79.96 |
| M2: dev | 81.11 | 79.39 | 80.24 |
| M2: test | 83.47 | 81.82 | 82.64 |
| M3: dev | 80.98 | 79.56 | 80.26 |
| M3: test | 81.86 | 83.34 | 82.60 |

Table 1: Results on the CoNLL 2005 shared task using gold standard parse trees. M1 is the LTAG-based model, M2 is the derived tree pattern matching Model, M3 is a hybrid model

## 5 Related Work

In the community of SRL researchers (cf. (Gildea and Jurafsky, 2002; Punyakanok, Roth and Yih, 2005; Pradhan et al, 2005; Toutanova et al., 2005)), the focus has been on two different aspects of the SRL task: (a) finding appropriate features, and (b) resolving the parsing accuracy problem by combining multiple parsers/predictions. Systems that use parse trees as a source of feature functions for their models have typically outperformed shallow parsing models on the SRL task. Typical features extracted from a parse tree is the path from the predicate to the constituent and various generalizations based on this path (such as phrase type, position, etc.). Notably the voice (passive or

active) of the verb is often used and recovered using a heuristic rule. We also use the passive/active voice by labeling this information into the parse tree. However, in contrast with other work, in this paper we do not focus on the problem of parse accuracy: where the parser output may not contain the constituent that is required for recovering all SRLs.

There has been some previous work in SRL that uses LTAG-based decomposition of the parse tree and we compare our work to this more closely. (Chen and Rambow, 2003) discuss a model for SRL that uses LTAG-based decomposition of parse trees (as is typically done for statistical LTAG parsing). Instead of using the typical parse tree features used in typical SRL models, (Chen and Rambow, 2003) uses the path within the elementary tree from the predicate to the constituent argument. They only recover semantic roles for those constituents that are localized within a single elementary tree for the predicate, ignoring cases that occur outside the elementary tree. In contrast, we recover all SRLs regardless of locality within the elementary tree. As a result, if we do not compare the machine learning methods involved in the two approaches, but rather the features used in learning, our features are a natural generalization of (Chen and Rambow, 2003).

Our approach is also very akin to the approach in (Shen and Joshi, 2005) which uses PropBank information to recover an LTAG treebank as if it were hidden data underlying the Penn Treebank. This is similar to our approach of having several possible LTAG derivations representing recovery of SRLs. However, (Shen and Joshi, 2005) do not focus on the SRL task, and in both of these instances of previous work using LTAG for SRL, we cannot directly compare our performance with theirs due to differing assumptions about the task.

## 6 Conclusion and Future Work

In this paper, we proposed a novel model for SRL using features extracted from LTAG derivation trees. A simple decision list learner is applied to train on the tree patterns containing new features. This simple learning method enables us to quickly explore new features for this task. However, this work is still preliminary: a lot of additional work is required to be competitive with the state-of-the-art SRL systems. In particular, we do not deal with automatically parsed data yet, which leads to a drop in our performance. We also do not incorporate various other features commonly used for SRL, as our goal in this paper was to make a direct comparison between simple pattern matching features on the derived tree and compare them to features from LTAG derivation trees.

## References


X. Carreras and L. Màrquez 2005. Introduction to the CoNLL-2005 Shared Task. In Proc. of CoNLL 2005.

J. Chen and O. 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, 2003.

D. Gildea and D. Jurafsky. 2002. Automatic Labeling of Semantic Roles. Computational Linguistics, 58(3):245–288

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. Computational Linguistics, 31(1).

G. Melli and Y. Wang and Y. Liu and M. Kashani and Z. Shi and B. Gu and A. Sarkar and F. Popowich 2005. Description of SQUASH, the SFU Question Answering Summary Handler for the DUC-2005 Summarization Task. In Proceeding of Document Understanding Conference (DUC-2005)

S. Pradhan, K. Hacioglu, W. Ward, J. H. Martin, and D. Jurafsky. 2005. Semantic Role Chunking Combining Complementary Syntactic Views, In Proceedings of the 9th Conference on Natural Language Learning (CoNLL 2005), Ann Arbor, MI, 2005.

V. Punyakanok, D. Roth, and W Yih. 2005. Generalized Inference with Multiple Semantic Role Labeling Systems (shared task paper). Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL) pp. 181-184

Ruppenhofer, Josef, Collin F. Baker and Charles J. Fillmore. 2002. The FrameNet Database and Software Tools. In Braasch, Anna and Claus Povlsen (eds.), Proceedings of the Tenth Euralex International Congress. Copenhagen, Denmark. Vol. I: 371-375.

L. Shen and A. Joshi. 2005. Building an LTAG Treebank. Technical Report MS-CIS-05-15, CIS Department, University of Pennsylvania.

K. Toutanova, A. Haghighi, and C. D. Manning. 2005. Joint learning improves semantic role labeling. ACL 2005

N. Xue and M. Palmer. 2004. Calibrating Features for Semantic Role Labeling, In Proceedings of EMNLP-2004. Barcelona, Spain.