# Exploiting Full Parsing Information to Label Semantic Roles Using an Ensemble of ME and SVM via Integer Linear Programming

**Tzong-Han Tsai, Chia-Wei Wu, Yu-Chun Lin, Wen-Lian Hsu**
Institute of Information Science
Academia Sinica
Taipei 115, Taiwan
{thtsai, cwwu, sbb, hsu}@iis.sinica.edu.tw

## Abstract

In this paper, we propose a method that exploits full parsing information by representing it as features of argument classification models and as constraints in integer linear learning programs. In addition, to take advantage of SVM-based and Maximum Entropy-based argument classification models, we incorporate their scoring matrices, and use the combined matrix in the above-mentioned integer linear programs. The experimental results show that full parsing information not only increases the F-score of argument classification models by 0.7%, but also effectively removes all labeling inconsistencies, which increases the F-score by 0.64%. The ensemble of SVM and ME also boosts the F-score by 0.77%. Our system achieves an F-score of 76.53% in the development set and 76.38% in Test WSJ.

## 1 Introduction

The Semantic Role Labeling problem can be formulated as a sentence tagging problem. A sentence can be represented as a sequence of words, as phrases (chunks), or as a parsing tree. The basic units of a sentence are words, phrases, and constituents in these representations, respectively.. Pradhan et al. (2004) established that Constituent-by-Constituent (C-by-C) is better than Phrase-by-Phrase (P-by-P), which is better than Word-by-Word (W-by-W). This is probably because the boundaries of the constituents coincide with the arguments; therefore, C-by-C has the highest argument identification F-score among the three approaches.

In addition, a full parsing tree also provides richer syntactic information than a sequence of chunks or words. Pradhan et al. (2004) compared the seven most common features as well as several features related to the target constituent's parent and sibling constituents. Their experimental results show that using other constituents' information increases the F-score by 6%. Punyakanok et al. (2004) represent full parsing information as constraints in integer linear programs. Their experimental results show that using such information increases the argument classification accuracy by 1%.

In this paper, we not only add more full parsing features to argument classification models, but also represent full parsing information as constraints in integer linear programs (ILP) to resolve label inconsistencies. We also build an ensemble of two argument classification models: Maximum Entropy and SVM by combining their argument classification results and applying them to the above-mentioned ILPs.

## 2 System Architecture

Our SRL system is comprised of four stages: *pruning*, *argument classification*, *classification model incorporation*, and *integer linear programming*. This section describes how we build these stages, including the features used in training the argument classification models.

### 2.1 Pruning



*Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL),*
pages 233–236, Ann Arbor, June 2005. 

When the full parsing tree of a sentence is available, only the constituents in the tree are considered as argument candidates. In CoNLL-2005, full parsing trees are provided by two full parsers: the Collins parser (Collins, 1999) and the Charniak parser (Charniak, 2000). According to Punyakanok et al. (2005), the boundary agreement of Charniak is higher than that of Collins; therefore, we choose the Charniak parser's results. However, there are two million nodes on the full parsing trees in the training corpus, which makes the training time of machine learning algorithms extremely long. Besides, noisy information from unrelated parts of a sentence could also affect the training of machine learning models. Therefore, our system exploits the heuristic rules introduced by Xue and Palmer (2004) to filter out simple constituents that are unlikely to be arguments. Applying pruning heuristics to the output of Charniak's parser effectively eliminates 61% of the training data and 61.3% of the development data, while still achieves 93% and 85.5% coverage of the correct arguments in the training and development sets, respectively.

### 2.2 Argument Classification

This stage assigns the final labels to the candidates derived in Section 2.1. A multi-class classifier is trained to classify the types of the arguments supplied by the pruning stage. In addition, to reduce the number of excess candidates mistakenly output by the previous stage, these candidates can be labeled as null (meaning "not an argument"). The features used in this stage are as follows.

#### Basic Features

- **Predicate** – The predicate lemma.
- **Path** – The syntactic path through the parsing tree from the parse constituent being classified to the predicate.
- **Constituent Type**
- **Position** – Whether the phrase is located before or after the predicate.
- **Voice** – passive: if the predicate has a POS tag VBN, and its chunk is not a VP, or it is preceded by a form of "to be" or "to get" within its chunk; otherwise, it is active.
- **Head Word** – calculated using the head word table described by Collins (1999).
- **Head POS** – The POS of the Head Word.
- **Sub-categorization** – The phrase structure rule that expands the predicate's parent node in the parsing tree.
- **First and Last Word/POS**
- **Named Entities** – LOC, ORG, PER, and MISC.
- **Level** – The level in the parsing tree.

#### Combination Features

- **Predicate Distance Combination**
- **Predicate Phrase Type Combination**
- **Head Word and Predicate Combination**
- **Voice Position Combination**

#### Context Features

- **Context Word/POS** – The two words preceding and the two words following the target phrase, as well as their corresponding POSs.
- **Context Chunk Type** – The two chunks preceding and the two chunks following the target phrase.

#### Full Parsing Features

We believe that information from related constituents in the full parsing tree helps in labeling the target constituent. Denote the target constituent by *t*. The following features are the most common baseline features of *t*'s parent and sibling constituents. For example, Parent/ Left Sibling/ Right Sibling Path denotes *t*'s parents', left sibling's, and right sibling's Path features.

- **Parent / Left Sibling / Right Sibling Path**
- **Parent / Left Sibling / Right Sibling Constituent Type**
- **Parent / Left Sibling / Right Sibling Position**
- **Parent / Left Sibling / Right Sibling Head Word**
- **Parent / Left Sibling / Right Sibling Head POS**
- **Head of PP parent** – If the parent is a PP, then the head of this PP is also used as a feature.

#### Argument Classification Models

We use all the features of the SVM-based and ME-based argument classification models. All SVM classifiers are realized using SVM-Light with a polynomial kernel of degree 2. The ME-based model is implemented based on Zhang's MaxEnt toolkit[1] and L-BFGS (Nocedal and Wright, 1999) method to perform parameter estimation.

### 2.3 Classification Model Incorporation

We now explain how we incorporate the SVM-based and ME-based argument classification models. After argument classification, we acquire two scoring matrices, $\mathbf{P}_{ME}$ and $\mathbf{P}_{SVM}$, respectively. Incorporation of these two models is realized by weighted summation of $\mathbf{P}_{ME}$ and $\mathbf{P}_{SVM}$ as follows:

$$\mathbf{P}' = w_{ME}\mathbf{P}_{ME} + w_{SVM}\mathbf{P}_{SVM}$$

We use $\mathbf{P}'$ for the objective coefficients of the ILP described in Section 2.4.

### 2.4 Integer Linear Programming (ILP)

To represent full parsing information as features, there are still several syntactic constraints on a parsing tree in the SRL problem. For example, on a path of the parsing tree, there can be only one constituent annotated as a non-null argument. However, it is difficult to encode this constraint in the argument classification models. Therefore, we apply integer linear programming to resolve inconsistencies produced in the argument classification stage.

According to Punyakanok et al. (2004), given a set of constituents, **S**, and a set of semantic role labels, **A**, the SRL problem can be formulated as an ILP as follows:

Let $z_{ia}$ be the indicator variable that represents whether or not an argument, $a$, is assigned to any $S_i \in \mathbf{S}$; and let $p_{ia} = \text{score}(S_i = a)$. The scoring matrix **P** composed of all $p_{ia}$ is calculated by the argument classification models. The goal of this ILP is to find a set of assignments for all $z_{ia}$ that maximizes the following function:

$$\sum_{S_i \in \mathbf{S}} \sum_{a \in \mathbf{A}} p_{ia} z_{ia} .$$

Each $S_i \in \mathbf{S}$ should have one of these argument types, or no type (null). Therefore, we have

$$\sum_{a \in \mathbf{A}} z_{ia} = 1 .$$

Next, we show how to transform the constraints in the filter function into linear equalities or inequalities, and use them in this ILP.

**Constraint I: No overlapping or embedding**

For arguments $S_{j_1}, \ldots, S_{j_k}$ on the same path in a full parsing tree, only one argument can be assigned to an argument type. Thus, at least $k - 1$ arguments will be *null*, which is represented by $\phi$ in the following linear equality:

$$\sum_{i=1}^{k} z_{j_i \phi} \geq k - 1 .$$

**Constraint II: No duplicate argument classes**

Within the same sentence, A0-A5 cannot appear more than once. The inequality for A0 is therefore:

$$\sum_{i=1}^{k} z_{i\text{A0}} \leq 1 .$$

**Constraint III: R-XXX arguments**

The linear inequalities that represent A0 and its reference type R-A0 are:

$$\forall m \in \{1,\ldots,M\} : \sum_{i=1}^{k} z_{i\text{A0}} \geq z_{m\text{R-A0}} .$$

**Constraint IV: C-XXX arguments**

The continued argument XXX has to occur before C-XXX. The linear inequalities for A0 are:

$$\forall m \in \{2,\ldots,M\} : \sum_{i=1}^{m-1} z_{j_i\text{A0}} \geq z_{m\text{C-A0}} .$$

**Constraint V: Illegal arguments**

For each verb, we look up its allowed roles. This constraint is represented by summing all the corresponding indicator variables to 0.

## 3 Experiment Results

### 3.1 Data and Evaluation Metrics

The data, which is part of the PropBank corpus, consists of sections from the Wall Street Journal part of the Penn Treebank. All experiments were carried out using Section 2 to Section 21 for training, Section 24 for development, and Section 23 for testing. Unlike CoNLL-2004, part of the Brown corpus is also included in the test set.

### 3.2 Results

Table 1 shows that our system makes little difference to the development set and Test WSJ. However, due to the intrinsic difference between the WSJ and Brown corpora, our system performs better on Test WSJ than on Test Brown.

[1] http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html

| | Precision | Recall | $F_{\beta=1}$ |
|---|---|---|---|
| Development | 81.13% | 72.42% | 76.53 |
| Test WSJ | 82.77% | 70.90% | 76.38 |
| Test Brown | 73.21% | 59.49% | 65.64 |
| Test WSJ+Brown | 81.55% | 69.37% | 74.97 |

| Test WSJ | Precision | Recall | $F_{\beta=1}$ |
|---|---|---|---|
| Overall | 82.77% | 70.90% | 76.38 |
| A0 | 88.25% | 84.93% | 86.56 |
| A1 | 82.21% | 72.21% | 76.89 |
| A2 | 74.68% | 52.34% | 61.55 |
| A3 | 78.30% | 47.98% | 59.50 |
| A4 | 84.29% | 57.84% | 68.60 |
| A5 | 100.00% | 60.00% | 75.00 |
| AM-ADV | 64.19% | 47.83% | 54.81 |
| AM-CAU | 70.00% | 38.36% | 49.56 |
| AM-DIR | 38.20% | 40.00% | 39.08 |
| AM-DIS | 83.33% | 71.88% | 77.18 |
| AM-EXT | 86.67% | 40.62% | 55.32 |
| AM-LOC | 63.71% | 41.60% | 50.33 |
| AM-MNR | 63.36% | 48.26% | 54.79 |
| AM-MOD | 98.00% | 97.64% | 97.82 |
| AM-NEG | 99.53% | 92.61% | 95.95 |
| AM-PNC | 44.44% | 17.39% | 25.00 |
| AM-PRD | 50.00% | 20.00% | 28.57 |
| AM-REC | 0.00% | 0.00% | 0.00 |
| AM-TMP | 83.21% | 61.09% | 70.45 |
| R-A0 | 91.08% | 86.61% | 88.79 |
| R-A1 | 79.49% | 79.49% | 79.49 |
| R-A2 | 87.50% | 43.75% | 58.33 |
| R-A3 | 0.00% | 0.00% | 0.00 |
| R-A4 | 0.00% | 0.00% | 0.00 |
| R-AM-ADV | 0.00% | 0.00% | 0.00 |
| R-AM-CAU | 100.00% | 25.00% | 40.00 |
| R-AM-EXT | 0.00% | 0.00% | 0.00 |
| R-AM-LOC | 92.31% | 57.14% | 70.59 |
| R-AM-MNR | 25.00% | 16.67% | 20.00 |
| R-AM-TMP | 72.73% | 61.54% | 66.67 |
| V | 97.32% | 97.32% | 97.32 |

Table 1. Overall results (top) and detailed results on the WSJ test (bottom).

| | Precision | Recall | $F_{\beta=1}$ |
|---|---|---|---|
| ME w/o parsing | 77.28% | 70.55% | 73.76% |
| ME | 78.19% | 71.08% | 74.46% |
| ME with ILP | 79.57% | 71.11% | 75.10% |
| SVM | 79.88% | 72.03% | 75.76% |
| Hybrid | **81.13%** | **72.42%** | **76.53%** |

Table 2. Results of all configurations on the development set.

From Table 2, we can see that the model with full parsing features outperforms the model without the features in all three performance matrices. After applying ILP, the performance is improved further. We also observe that SVM slightly outperforms ME. However, the hybrid argument classification model achieves the best results in all three metrics.

## 4 Conclusion

In this paper, we add more full parsing features to argument classification models, and represent full parsing information as constraints in ILPs to resolve labeling inconsistencies. We also integrate two argument classification models, ME and SVM, by combining their argument classification results and applying them to the above-mentioned ILPs. The results show full parsing information increases the total F-score by 1.34%. The ensemble of SVM and ME also boosts the F-score by 0.77%. Finally, our system achieves an F-score of 76.53% in the development set and 76.38% in Test WSJ.

## Acknowledgement

We are indebted to Wen Shong Lin and Prof. Fu Chang for their invaluable advice in data pruning, which greatly speeds up the training of our machine learning models.

## References

X. Carreras and L. Màrquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *Proceedings of the CoNLL-2005.*

E. Charniak. 2000. A Maximum-Entropy-Inspired Parser. *Proceedings of the NAACL-2000*.

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

J. Nocedal and S. J. Wright. 1999. *Numerical Optimization*, Springer.

S. Pradhan, K. Hacioglu, V. Kruglery, W. Ward,J. H. Martin, and D. Jurafsky. 2004. Support Vector Learning for Semantic Argument Classification. *Journal of Machine Learning*.

V. Punyakanok, D. Roth, and W. Yih. 2005. The Necessity of Syntactic Parsing for Semantic Role Labeling. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05).*

V. Punyakanok, D. Roth, W. Yih, and D. Zimak. 2004. Semantic Role Labeling via Integer Linear Programming Inference. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04).*

N. Xue and M. Palmer. 2004. Calibrating Features for Semantic Role Labeling. In *Proceedings of the EMNLP 2004*.