A Joint Model for Semantic Role Labeling

Aria Haghighi
Dept of Computer Science
Stanford University
Stanford, CA, 94305
aria42@stanford.edu

Kristina Toutanova
Dept of Computer Science
Stanford University
Stanford, CA, 94305
kristina@cs.stanford.edu

Christopher D. Manning
Dept of Computer Science
Stanford University
Stanford, CA, 94305
manning@cs.stanford.edu

Abstract

We present a semantic role labeling system submitted to the closed track of the CoNLL-2005 shared task. The system, introduced in (Toutanova et al., 2005), implements a joint model that captures dependencies among arguments of a predicate using log-linear models in a discriminative re-ranking framework. We also describe experiments aimed at increasing the robustness of the system in the presence of syntactic parse errors. Our final system achieves F1-Measures of 76.68 and 78.45 on the development and the WSJ portion of the test set, respectively.

1 Introduction

It is evident that there are strong statistical patterns in the syntactic realization and ordering of the arguments of verbs; for instance, if an active predicate has an `A0` argument it is very likely to come before an `A1` argument. Our model aims to capture such dependencies among the labels of nodes in a syntactic parse tree.

However, building such a model is computationally expensive. Since the space of possible joint labelings is exponential in the number of parse tree nodes, a model cannot exhaustively consider these labelings unless it makes strong independence assumptions. To overcome this problem, we adopt a discriminative re-ranking approach reminiscent of (Collins, 2000). We use a local model, which labels arguments independently, to generate a smaller number of likely joint labelings. These candidate labelings are in turn input to a joint model which can use global features and re-score the candidates. Both the local and global re-ranking models are log-linear (maximum entropy) models.

In the following sections, we briefly describe our local and joint models and the system architecture for combining them. We list the features used by our models, with an emphasis on new features, and compare the performance of a local and a joint model on the CoNLL shared task. We also study an approach to increasing the robustness of the semantic role labeling system to syntactic parser errors, by considering multiple parse trees generated by a statistical parser.

2 Local Models

Our local model labels nodes in a parse tree independently. We decompose the probability over labels (all argument labels plus `NONE`), into a product of the probability over `ARG` and `NONE`, and a probability over argument labels given that a node is an `ARG`. This can be seen as chaining an *identification* and a *classification* model. The identification model classifies each phrase as either an argument or non-argument and our classification model labels each potential argument with a specific argument label. The two models use the same features.

Previous research (Gildea and Jurafsky, 2002; Pradhan et al., 2004; Carreras and Màrquez, 2004) has identified many useful features for local identification and classification. Below we list the features and hand-picked conjunctions of features used in our local models. The ones denoted with asterisks (*) were not present in (Toutanova et al., 2005). Although most of these features have been described in previous work, some features, described in the next section, are – to our knowledge – novel.

- **Phrase-Type** Syntactic category of node
- **Predicate Lemma** Stemmed target verb
- **Path** Sequence of phrase types between the predicate and node, with ↑, ↓ to indicate direction
- **Position** Before or after predicate
- **Voice** Voice of predicate
- **Head-Word of Phrase**
- **Head-POS** POS tag of head word
- **Sub-Cat** CFG expansion of predicate's parent
- **First/Last Word**
- **Left/Right Sister Phrase-Type**
- **Left/Right Sister Head-Word/Head-POS**
- **Parent Phrase-Type**
- **Parent POS/Head-Word**
- **Ordinal Tree Distance** Phrase-type concatenated with the length of the **Path** feature
- **Node-LCA Partial Path** Path from the node to the lowest common ancestor of the predicate and the node
- **PP Parent Head-Word** If the parent of the node is a PP, the parent's head-word
- **PP NP Head-Word/Head-POS** For a PP, retrieve the head-word /head-POS of its rightmost NP
- **Temporal Keywords*** Is the head of the node a temporal word e.g 'February' or 'afternoon'
- **Missing subject*** Is the predicate missing a subject in the"standard" location
- **Projected path*** Path from the maximal extended projection of the predicate to the node
- **Predicate Lemma & Path**
- **Predicate Lemma & Head-Word**
- **Predicate Lemma & Phrase-Type**
- **Voice & Position**
- **Predicate Lemma & PP Parent Head-Word**
- **Path & Missing subject***
- **Projected path & Missing subject***

2.1 Additional Local Features

We found that a large source of errors for `A0` and `A1` stemmed from cases such as those illustrated in Figure 1, where arguments were dislocated by raising or controlling verbs. Here, the predicate, *expected*, does not have a subject in the typical position – indicated by the empty NP – since the auxiliary *is* has raised the subject to its current position. In order to capture this class of examples, we use a binary feature, **Missing Subject**, indicating whether the predicate is "missing" its subject, and use this feature in conjunction with the **Path** feature, so that we learn typical paths to raised subjects conditioned on the absence of the subject in its typical position.

In the particular case of Figure 1, there is another instance of an argument being quite far from

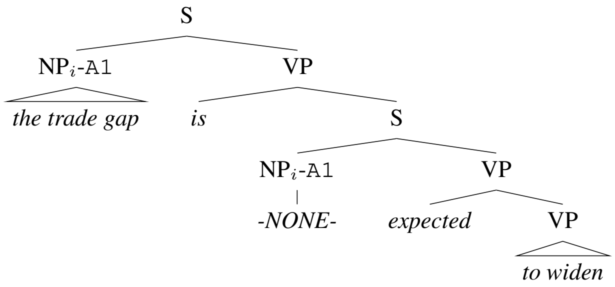

Figure 1: Example of displaced arguments

its predicate. The predicate *widen* shares *the trade gap* with *expect* as a `A1` argument. However, as *expect* is a raising verb, *widen*'s subject is not in its typical position either, and we should expect to find it in the same positions as *expected*'s subject. This indicates it may be useful to use the path relative to *expected* to find arguments for *widen*. In general, to identify certain arguments of predicates embedded in auxiliary and infinitival VPs we expect it to be helpful to take the path from the maximum extended projection of the predicate – the highest VP in the chain of VP's dominating the predicate. We introduce a new path feature, **Projected Path**, which takes the path from the maximal extended projection to an argument node. This feature applies only when the argument is not dominated by the maximal projection, (e.g., direct objects). These features also handle other cases of discontinuous and non-local dependencies, such as those arising due to controller verbs. For a local model, these new features and their conjunctions improved F1-Measure from 73.80 to 74.52 on the development set. Notably, the F1-Measure of `A0` increased from 81.02 to 83.08.

3 Joint Model

Our joint model, in contrast to the local model, collectively scores a labeling of all nodes in the parse tree. The model is trained to re-rank a set of N likely labelings according to the local model. We find the exact top N consistent[1] most likely local model labelings using a simple dynamic program described in (Toutanova et al., 2005).

Most of the features we use are described in more detail in (Toutanova et al., 2005). Here we briefly

[1]A labeling is consistent if satisfies the constraint that argument phrases do not overlap.

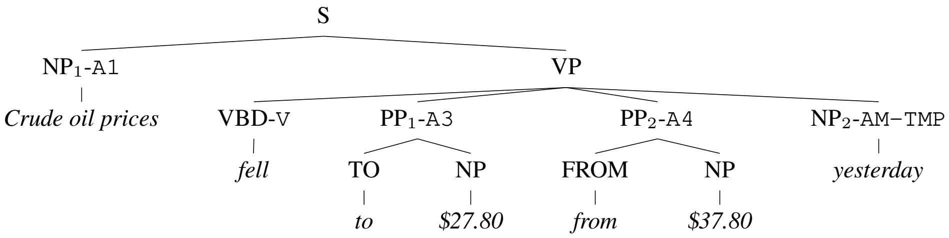

Figure 2: An example tree with semantic role annotations.

describe these features and introduce several new joint features (denoted by *). A labeling L of all nodes in the parse tree specifies a candidate argument frame – the sequence of all nodes labeled with a non-NONE label according to L. The joint model features operate on candidate argument frames, and look at the labels and internal features of the candidate arguments. We introduce them in the context of the example in Figure 2. The candidate argument frame corresponding to the correct labeling for the tree is: [NP_1-A1,VBD-V,PP_1-A3,PP_2-A4,NP_2-AM-TMP].

- **Core arguments label sequence:** The sequence of labels of core arguments concatenated with the predicate voice. Example: [voice:active: A1,V,A3,A4] A back-off feature which substitutes specific argument labels with a generic argument (A) label is also included.
- **Flattened core arguments label sequence*:** Same as the previous but merging consecutive equal labels.
- **Core arguments label and annotated phrase type sequence:** The sequence of labels of core arguments together with annotated phrase types. Phrase types are annotated with the head word for PP nodes, and with the head POS tag for S and VP nodes. Example: [voice:active: NP-A1,V,PP-to-A3,PP-from-A4]. A back-off to generic A labels is also included. Also a variant that adds the predicate stem.
- **Repeated core argument labels with phrase types:** Annotated phrase types for nodes with the same core argument label. This feature captures, for example, the tendency of WHNP referring phrases to occur as the second phrase having the same label as a preceding NP phrase.
- **Repeated core argument labels with phrase types and sister/adjacency information*:** Similar to the previous feature, but also indicates whether all repeated arguments are sisters in the parse tree, or whether all repeated arguments are adjacent in terms of word spans. These features can provide robustness to parser errors, making it more likely to label adjacent phrases incorrectly split by the parser with the same label.

4 Combining Local and Joint Models

It is useful to combine the joint model score with a local model score, because the local model has been trained using all negative examples, whereas the joint model has been trained only on likely argument frames . Our final score is given by a mixture of the local and joint model's log-probabilities: $score_{SRL}(L|t) = \alpha\, score_\ell(L|t) + score_J(L|t)$, where $score_\ell(L|t)$ is the local score of L, $score_J(L|t)$ is the corresponding joint score, and α is a tunable parameter. We search among the top N candidate labelings proposed by the local model, for the labeling that maximizes the final score.

5 Increasing Robustness to Parser Errors

Semantic role labeling is very sensitive to the correctness of the given parse tree. If an argument does not correspond to a constituent in a parse tree, our model will not be able to consider the correct phrase.

One way to address this problem is to utilize alternative parses. Recent releases of the Charniak parser (Charniak, 2000) have included an option to provide the top k parses of a given sentence according to the probability model of the parser. We use these alternative parses as follow: Suppose $t_1, \ldots, t_k$ are trees for sentence s with given probabilities $P(t_i|s)$ by the parser. Then for a fixed predicate v, let L_i denote the best joint labeling of tree t_i, with score $score_{SRL}(L_i|t_i)$ according to our final joint model.

Then we choose the labeling L which maximizes:

$$\arg\max_{i \in \{1,\ldots,k\}} \beta \log P(t_i|S) + score_{SRL}(L_i|t_i)$$

Considering top $k = 5$ parse trees using this algorithm resulted in up to 0.4 absolute increase in F-Measure. In future work, we plan to experiment with better ways to combine information from multiple parse trees.

6 Experiments and Results

For our final results we used a joint model with $\alpha = 1.5$ (local model weight), $\beta = 1$ (parse tree log-probability weight) , $N = 15$ (candidate labelings from the local model to consider) , and $k = 5$ (number of alternative parses). The whole training set for the CoNLL-2005 task was used to train the models. It takes about 2 hours to train a local identification model, 40 minutes to train a local classification model, and 7 hours to train a joint re-ranking model.[2]

In Table 1, we present our final development and test results using this model. The percentage of perfectly labeled propositions for the three sets is 55.11% (development), 56.52% (test), and 37.06% (Brown test). The improvement achieved by the joint model relative to the local model is about 2 points absolute in F-Measure, similar to the improvement when gold-standard syntactic parses are used (Toutanova et al., 2005). The relative error reduction is much lower for automatic parses, possibly due to a lower upper bound on performance. It is clear from the drop in performance from the WSJ to Brown test set that our learned model's features do not generalize very well to related domains.

	Precision	Recall	$F_{\beta=1}$
Development	77.66%	75.72%	76.68
Test WSJ	79.54%	77.39%	78.45
Test Brown	70.24%	65.37%	67.71
Test WSJ+Brown	78.34%	75.78%	77.04

Test WSJ	Precision	Recall	$F_{\beta=1}$
Overall	79.54%	77.39%	78.45
A0	88.32%	88.30%	88.31
A1	78.61%	78.40%	78.51
A2	72.55%	68.11%	70.26
A3	73.08%	54.91%	62.71
A4	77.42%	70.59%	73.85
A5	100.00%	80.00%	88.89
AM-ADV	58.20%	51.19%	54.47
AM-CAU	63.93%	53.42%	58.21
AM-DIR	52.56%	48.24%	50.31
AM-DIS	76.56%	80.62%	78.54
AM-EXT	73.68%	43.75%	54.90
AM-LOC	61.52%	55.92%	58.59
AM-MNR	58.33%	56.98%	57.65
AM-MOD	97.85%	99.09%	98.47
AM-NEG	97.41%	98.26%	97.84
AM-PNC	49.50%	43.48%	46.30
AM-PRD	100.00%	20.00%	33.33
AM-REC	0.00%	0.00%	0.00
AM-TMP	74.85%	67.34%	70.90
R-A0	92.63%	89.73%	91.16
R-A1	81.53%	82.05%	81.79
R-A2	61.54%	50.00%	55.17
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%	50.00%	66.67
R-AM-EXT	0.00%	0.00%	0.00
R-AM-LOC	85.71%	57.14%	68.57
R-AM-MNR	28.57%	33.33%	30.77
R-AM-TMP	61.54%	76.92%	68.38
V	97.32%	97.32%	97.32

Table 1: Overall results (top) and detailed results on the WSJ test set (bottom) for the closed track.

Acknowledgements

This work was supported in part by the Advanced Research and Development Activity (ARDA)'s Advanced Question Answering for Intelligence (AQUAINT) Program.

References

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of CoNLL-2004*.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL*, pages 132–139.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of ICML-2000*.

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

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Dan Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proceedings of HLT/NAACL-2004*.

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

[2] On a 3.6GHz machine with 4GB of RAM.