Tree Representations for Chinese Semantic Role Labeling

Ni Lao
Language Technologies Institute
Carnegie Mellon University
nlao@cs.cmu.edu

Teruko Mitamura
Language Technologies Institute
Carnegie Mellon University
teruko@cs.cmu.edu

Eric Nyberg
Language Technologies Institute
Carnegie Mellon University
ehn@cs.cmu.edu

Abstract

We compare different parse tree representations for the task of Chinese Semantic Role Labeling (SRL), including dependency and constituency parse trees, two tree pruning methods, and neighbor features. Three learning models are compared. By using SVM classifier with neighbor features and pruning tree to phrase level we achieve significantly better speed and accuracy than state of the art Chinese SRL systems.

1 Introduction

Despite of the large body of existing work on SRL in English, there has not been much work on exploring different tree representations for Chinese. For tree structures derived for a sentence, the node labels are co-dependent. One way to express this codependency is to add global constraints (Punyakanok et al., 2004; Tromble & Eisner, 2006). Another way is to do joint prediction with model such as tree CRF (Cohn & Blunsom, 2005), where decision of one node is dependent to its neighbors on the tree. In this study we compare a tree CRF approach with a third approach that adds to each node features from its neighbor nodes. In this way, information propagates around the tree without the use of a joint decisions model. Furthermore, we found in our study that simply pruning tree to phrase level effectively reduced the training time without sacrificing accuracy.

2 Description of Approach

2.1 System Architecture

Most current SRL approaches include three stages: first, annotate the sentences by constituency or dependency parser, then apply a classifier (like SVM) to each constituent based on carefully engineered features, and finally perform joint scoring based on hard or soft constraints (Punyakanok et al., 2004; Tromble & Jason Eisner, 2006). Xue and Palmer (2004) decomposed the second stage into three steps. First, some negative nodes are filtered out with heuristics that exploit the syntactic structures represented in the parse tree. Second, a binary classifier is applied to further separate argument nodes from non-argument nodes ("argument identification"). Finally a multi-category classifier is applied to assign the semantic role labels to the positive samples ("argument classification").

In this study we use the same basic approach as Pradhan et al. (2004). The classifier is an SVM, and the joint scoring step simply removes overlapping constituents. The predicates are given in the task, and for each predicate in a sentence, stages 2 and 3 are applied.

2.2 Tree Pruning

Tree pruning is known to improve both speed and accuracy of SRL. We explored two pruning strategies. One is to prune the tree to the phrase level (to the lowest NP nodes in parse tree). Another is Xue and Palmer's (2004) more aggressive rule: first include all nodes along the path from the predicate to the root, then include all their children; if any of these nodes is PP, then also include its immediate children.

2.3 Features

SRL differs from lower-level NLP tasks such as POS tagging in that it has a fairly large feature space; as a result, linguistic knowledge is crucial in designing effective features for this task. New features have continuously been proposed in the past years (Gildea & Jurafsky, 2002; Surdeanu et al., 2003; Pradhan et al., 2004; Xue & Palmer, 2004, 2005). In this study, we implemented many of the above features, excluding those that need more than a syntax parser can produce (e.g. named entity, voice, and etc). Furthermore, we explore adding features from neighbor nodes like mother, left/right sibling. All the implemented features are listed as following:

1. **POS:** Part-of-speech, or non-terminal tags
2. **W:** The word form. For non-leaf node (with span length <=3), this feature is the sequence of words covered by its span.
3. **h.{POS, W}:** Features from the head word
4. **NVO:** Coarse grain POS, which only distinguish Noun, Verb, and Other. It is automatically produced from the first letter of POS (N, V, or other)
5. **p.{POS, POS+W, NVO}:** Path from each node to the predicate, decorated with POS, POS + headword, or NVO
6. **p.{POS, POS+W, NVO}.{b, a}:** Half path before/after common ancestor of a node and the predicate
7. **position:** If the span of a node is before, over, or after the predicate {b,o,a}
8. **level:** Level of this node on the tree {0,1,…}
9. **pred:** If this node is the predicate {true,false}
10. **verb:** Text of the predicate
11. **{l,r,m}.{POS,W}:** POS or word from neighbors (left/right siblings, and mother)
12. **VC:** Verb sub-category from FrameNet (Fillmore, et al, 2001)

2.4 MaxEnt and Tree CRF Model

As a way to express codependency among nodes, tree CRFs have been used for English SRL (Cohn and Blunsom, 2005), with the model structure derived from pruned constituency parse trees. Each non-terminal node (constituent) in the parse tree maps to a node in the graphical model, and edges in the graphical model are the same as the ones in the parse trees.

Formally, in a CRF the conditional probability for the labels given the observation is defined as:

$$p(y \mid x) = \frac{1}{Z(x)} \exp(\sum_{c \in C} \sum_{i=1..k} \lambda_i f_i(c, y_c, x))$$

where C is the set of cliques, f_i and λ_i are the factors and corresponding weights. Z(x) is the partition function. For a node clique in our model, each factor is defined over the label and a feature defined in the previous section (e.g. y_3=ARG1&POS=S). For an edge clique, each factor is defined over labels of both nodes and a feature on the child (e.g. y_1.label=NULL & y_3.label=ARG1 & y_3.POS=S) (Figure 1). For dependent tree, features come from the highest collapsed constituency node. When only node clique is used, the model falls back to maximum entropy model. Different than SVM mode, no overlapping filter is applied at the joint scoring stage, for MaxEnt and tree-CRF.

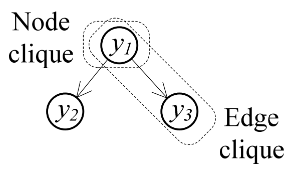

Figure 1: Node and edge cliques on parse tree

3 Experiment

3.1 Setup

In this study we use gold standard constituency tree from Chinese TreeBank 5.0. We also derive dependency tree by applying Dan Bikel's(2004) head finding rules to the constituency trees. We first identify the head child of each node, and then recursively move head children up the tree. Therefore, the resulting trees only have POSs and words tagged on the nodes, and no dependency relations. Similar to Hacioglu (2004), a node is tagged as an argument if its word span matches a span specified in PropBank 1.0 (Xue and Palmer, 2003).

We use 99 files (1.fid to 99.fid) for testing and all the remaining 791 files (100.fid to 1151.fid) for training. We also use TreeBank gold standard parses, and report the standard precision, recall and F-measure.

For the SVM implementation, we modified the implementation of ASSERT[1] (Pradhan et al., 2004) to accept our features, and the same parameter settings are applied. For the MaxEnt and CRF implementations, we used Sutton's GRMM toolkit[2], with tree belief propagation.

For this study, Xue and Palmer (2005) and CASSERT performance would be natural baselines. In order to compare to CASSERT, we retrain and test it with the same data settings. Xue and Palmer (2005) use pre-released version of PropBank, therefore only 661 files (100.fid to 931.fid) are used for training (about 84% of ours).

[1] http://oak.colorado.edu/assert/
[2] http://mallet.cs.umass.edu

Run	Representation				SVM				MaxEnt				TreeCRF			
ID	**Feature**	**T1**	**T2**	**#nodes/tree**	**PR**	**RC**	**F1**	**Time**	**PR**	**RC**	**F1**	**Time**	**PR**	**RC**	**F1**	**Time**
1	C	no	no	86.8	89.6	85.7	87.6	90	89.0	85.2	87.1	185	89.1	85.1	87.1	2750
2	C	yes	no	48.1	90.0	88.2	89.1	41	89.5	87.4	88.4	141	89.3	87.2	88.2	1360
3	C+vc	yes	no	48.1	90.3	88.3	89.3	44	89.8	87.4	88.6	149	89.5	87.3	88.4	1384
4	C+vc+nb	yes	no	48.1	**91.0**	**89.0**	**90.0**	45	**90.5**	**88.2**	**89.3**	154	**90.1**	**88.0**	**89.0**	1394
5	C	no	yes	17.7	89.0	87.4	88.2	8	88.3	86.9	87.6	56	88.3	87.0	87.6	475
6	C+vc	no	yes	17.7	89.3	87.8	88.5	9	88.7	87.0	87.8	61	88.4	87.0	87.7	492
7	C+vc+nb	no	yes	17.7	90.1	88.2	89.1	11	89.4	88.0	88.7	57	89.3	87.8	88.5	460
8	C	yes	yes	13.9	89.0	87.0	88.0	6	89.1	**88.2**	88.7	28	89.1	**88.1**	88.6	233
9	D	no	no	40.8	89.2	80.0	84.4	26	85.7	85.0	85.3	168	86.4	84.4	85.4	1748
10	D	yes	no	20.6	89.4	**85.0**	**87.1**	10	**87.8**	**85.3**	**86.5**	76	**87.7**	**86.4**	**87.0**	519
11	D	no	yes	14.2	89.6	81.1	85.1	7	86.5	84.8	85.6	46	86.6	85.2	85.9	344
12	D	yes	yes	9.8	**90.0**	81.0	85.3	5	84.9	83.3	84.1	35	86.9	84.6	85.8	263

Table 1: Feature Exploration with SVM. D=dependency tree, C=constituency tree, vc=verb class, nb= neighbor features, **T1**= Prune to phrase, **T2**= Prune to predicate path. Boldfaced figures are statistically significant within same model and same type of tree. Training time measured in minutes (dual-core 3.0GHz CPUs).

3.2 Comparing Tree Representations

For the purpose of exploration, we randomly selected 10% of the sentences from the training set for training. Testing is still on the 99 file test set. Table 1 compares performance with different tree representations and learning models.

Comparing run 1 with runs 2~7 we can see that pruning the tree not only dramatically reduces the tree size and training time, but also improves the classification accuracy. Both pruning strategies improve the result. Pruning to the phrase level (T2) significantly improves both precision and recall. Pruning to predicate path (T1) although improves recall, slightly hurts precision. T1 can cut down tree size more drastically than T2 (resulting in a 75% training time reduction) but reduces F1 from 0.5%~0.9%.

Comparing runs 2, 3 and 5, 6 we can see that the verb class feature (vc) slightly improves the result. This is similar to the observation from Xue and Palmer (2005). Neighbor features (nb) improve F1 by about 0.7%. However, contrary to our expectation, the tree CRF does not improve accuracy via joint inference. Its accuracy is slightly worse than the linear models, both with or without neighbor features.

From runs 9~12 we can see that dependency trees are smaller and thus are trained faster. However, their accuracy is worse than constituency trees. For the SVM model, dependency trees generally produce similar precision, but recall is hurt by the overlapping filter. For MaxEnt and TreeCRF where there is no overlapping filter, precision and recall are still slightly worse than their constituency tree counterparts. We believe that the constituency tree provides a more natural representation for the SRL task than dependency tree. For example in Figure 2, the top NP node and the NN node spanning 文件 can each have distinctive feature set in the constituency tree, which gives model finer grained information for its decision. However, these two nodes are merged into one in the dependency tree and the distinction is lost.

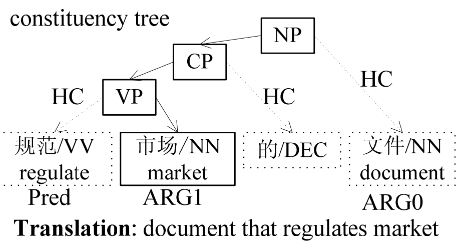

Figure 2: Example, HC=head child

3.3 Comparing Different Learners

In this subsection we compare the SVM, MaxEnt, and CRF models with the best tree representation of the previous section (run 4 with C+vc+nb+T1). Table 2 shows the main result with different models and data sizes. To reduce variance for small training data sets, we down-sample 10 independent training sets for experiments with 0.1%~10% data, and average the scores. We can see that although the SVM method is not as good as the other methods when training data size is small, it gets the best result as the data size gets larger. Considering that we use the same SVM classifier as CAssert, and features used are very similar, the gain in time and F1 are attributed to tree pruning and the use of neighbor features. CAssert has better performance with small data sizes, but lower performance with larger data sizes. This might be because CAssert uses fewer features. After accounting for the fact that Xue & Palmer (2005) use less training data, our SVM approach still performs better by 2% (Table 2).

We had expected the tree CRF model to out perform SVM and MaxEnt as it performs joint prediction. However, it is true only when the data size is small. SVM and MaxEnt compensate local decisions by better utilizing neighbor features (mother, head, and siblings) and outperform tree CRFs when larger data set is available.

The training times for all the methods are quite predictable. The CRF model is one order of magnitude slower than the MaxEnt model. Both SVM and MaxEnt are relatively simple models. The differences between their training time may be due to difference of optimization procedure, or just the differences between C++ and Java implementations. All the methods have empirical complexities around O($n^{1.5}$) (measured in minuets), where n is the number of training samples.

Data	**Method**	**PR**	**RC**	**F1**	**Time**
32,363 (100%)	**SVM**	**94.1**	**93.4**	**93.8**	1064
	MaxEnt	93.5	92.6	93.1	3051
	CRF	93.5	92.4	93.0	34142
	CAssert	90.0	90.2	90.1	1502
3,236 (10%)	**SVM**	**91.0**	**89.0**	**90.0**	45
	MaxEnt	90.5	88.2	89.3	154
	CRF	90.1	88.0	89.0	1394
	CAssert	87.6	86.7	87.1	100
324 (1%)	**SVM**	**86.3**	78.3	**82.1**	1.6
	MaxEnt	85.8	77.8	81.6	5.4
	CRF	85.9	78.4	**82.0**	61
	CAssert	82.4	**80.3**	81.3	3.3
32 (0.1%)	**SVM**	72.9	51.2	60.2	0.05
	MaxEnt	73.4	52.2	61.0	0.23
	CRF	**75.8**	54.3	63.3	1.8
	CAssert	74.3	**56.5**	**64.2**	0.16
84%	**X&P05**	91.4	91.1	91.3	N/A

Table 2: Compare Different Learners. Boldfaced figures are statistically significant within same data size

4 Conclusion and Future Directions

This study compares different tree representations and models for the task of Chinese SRL. We found that simpler models like SVM and MaxEnt can out perform tree CRF, if enough training data is given. However, using dependency trees hurt accuracy. SVM classifier with neighbor features and pruning tree to phrase level can achieve significantly better speed and accuracy than state of the art Chinese SRL systems. One potential improvement to the current approach is to apply joint scoring approaches like those described by Punyakanok et al. (2004) or Tromble & Eisner (2006).

References

Dan Bikel, On the Parameter Space of Generative Lexicalized Statistical Parsing Models. Phd thesis, University of Pennsylvania. 2004

Trevor Cohn and Philip Blunsom. 2005. Semantic role labelling with tree Conditional Random Fields. In Proceedings of CoNLL.

Charles J. Fillmore, et al., Building a large lexical databank which provides deep semantics. In Proceedings of the Pacific Asian Conference on Language, Information and Computation 2001.

Pascale Fung, et al., 2004. A Maximum-Entropy Chinese Parser Augmented by Transformation-Based Learning. ACM Transactions on Asian Language Processing 3(2): 159-168. June 2004.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. Computational Linguistics 28:3, 245-288. Taipei, Taiwan, 2002.

Kadri Hacioglu, et al., 2003. Shallow semantic parsing using support vector machines. Technical Report TR-CSLR-2003-1, Center for Spoken Language Research, Boulder, Colorado.

Martha Palmer, Dan Gildea, and Paul Kingsbury. The Proposition Bank: An annotated corpus of semantic roles. In Computational Linguistics, 31(1), 2005.

Sameer S. Pradhan, et al., Shallow Semantic Parsing Using Support Vector Machines. In Proceedings of HLT-NAACL, 2004

Vasin Punyakanok, Dan Roth, Wen-tau Yih, Dav Zimak, Semantic Role Labeling via Integer Linear Programming, In Proceedings of COLING 2004.

Roy W. Tromble and Jason Eisner. 2006. A fast finite-state relaxation method for enforcing global constraints on sequence decoding. Proceedings of HLT-NAACL, pages 423-430, New York, June.

Nianwen Xue and Martha Palmer. Annotating the Propositions in the Penn Chinese Treebank. In The Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing, Sapporo, Japan, 2003.

Nianwen Xue and Martha Palmer. Calibrating features for semantic role labeling. In Proceedings of 2004 Conference on Empirical Methods in Natural Lan-guage Processing, Barcelona, Spain, 2004.

Nianwen Xue and Martha Palmer. 2005. Automatic Semantic Role Labeling for Chinese Verbs, in Proceedings of the 19th International Joint Conference on Artificial Intelligence. Edinburgh, Scotland