ICT:A System Combination for Chinese Semantic Dependency Parsing

Hao Xiong and Qun Liu
Key Lab. of Intelligent Information Processing
Institute of Computing Technology
Chinese Academy of Sciences
P.O. Box 2704, Beijing 100190, China
{xionghao, liuqun}@ict.ac.cn

Abstract

The goal of semantic dependency parsing is to build dependency structure and label semantic relation between a head and its modifier. To attain this goal, we concentrate on obtaining better dependency structure to predict better semantic relations, and propose a method to combine the results of three state-of-the-art dependency parsers. Unfortunately, we made a mistake when we generate the final output that results in a lower score of 56.31% in term of Labeled Attachment Score (LAS), reported by organizers. After giving golden testing set, we fix the bug and rerun the evaluation script, this time we obtain the score of 62.8% which is consistent with the results on developing set. We will report detailed experimental results with correct program as a comparison standard for further research.

1 Introduction

In this year's Semantic Evaluation Task, the organizers hold a task for Chinese Semantic Dependency Parsing. The semantic dependency parsing (SDP) is a kind of dependency parsing. It builds a dependency structure for a sentence and labels the semantic relation between a head and its modifier. The semantic relations are different from syntactic relations. They are position independent, e.g., the patient can be before or behind a predicate. On the other hand, their grains are finer than syntactic relations, e.g., the syntactic subject can be agent or experiencer. Readers can refer to (Wanxiang Che, 2012) for detailed introduction.

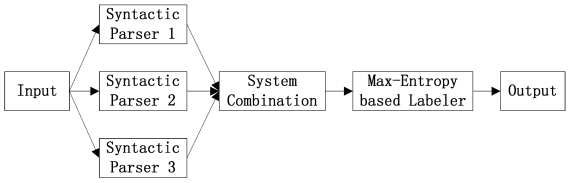

Figure 1: The pipeline of our system, where we combine the results of three dependency parsers and use max-entropy classifier to predict the semantic relations.

Different from most methods proposed in CoNLL-2008 [1] and 2009 [2], in which some researchers build a joint model to simultaneously generate dependency structure and its syntactic relations (Surdeanu et al., 2008; Hajič et al., 2009), here, we first employ several parsers to generate dependency structure and then propose a method to combine their outputs. After that, we label relation between each head and its modifier via the traversal of this refined parse tree. The reason why we use a pipeline model while not a joint model is that the number of semantic relations annotated by organizers is more than 120 types, while in the former task is only 21 types. Compared to the former task, the large number of types will obviously drop the performance of classifier. On the other hand, the performance of syntactic dependency parsing is approaching to perfect, intuitively, that better dependency structure does help to semantic parsing, thus we can concentrate on improving the accuracy of dependency structure construction.

The overall framework of our system is illustrated

[1] http://www.yr-bcn.es/conll2008/
[2] http://ufal.mff.cuni.cz/conll2009-st/

*First Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 514–518,
Montréal, Canada, June 7-8, 2012.

in figure 1, where three dependency parsers are employed to generate the dependency structure, and a maximum entropy classifier is used to predict relation for head and its modifier over combined parse tree. Final experimental results show that our system achieves 80.45% in term of unlabeled attachment score (UAS), and 62.8 % in term of LAS. Both of them are higher than the baseline without using system combinational techniques.

In the following of this paper, we will demonstrate the detailed information of our system, and report several experimental results.

2 System Description

As mentioned, we employ three single dependency parsers to generate respect dependency structure. To further improve the accuracy of dependency structure construction, we blend the syntactic outputs and find a better dependency structure. In the followings, we will first introduce the details of our strategy for dependency structure construction.

2.1 Parsers

We implement three transition-based dependency parsers with three different parsing algorithms: Nivre's arc standard, Nivre's arc eager (see Nivre (2004) for a comparison between the two Nivre algorithms), and Liang's dynamic algorithm(Huang and Sagae, 2010). We use these algorithms for several reasons: first, they are easy to implement and their reported performance are approaching to state-of-the-art. Second, their outputs are projective, which is consistent with given corpus.

2.2 Parser Combination

We use the similar method presented in Hall et al. (2011) to advance the accuracy of parses. The parses of each sentence are combined into a weighted directed graph. The left procedure is similar to traditional graph-based dependency parsing except that the number of edges in our system is smaller since we reserve best edges predicted by three single parsers. We use the popular Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds et al., 1968) to find the maximum spanning tree (MST) of the new constructed graph, which is considered as the final parse of the sentence. Specifically, we use the parsing accuracy on developing set to represent the weight of graph edge. Formally, the weight of graph edge is computed as follows,

$$w_e = \sum_{p \in P} Accuracy(p) \cdot I(e, p) \quad (1)$$

where the $Accuracy(p)$ is the parsing score of parse tree p whose value is the score of parsing accuracy on developing set, and $I(e, p)$ is an indicator, if there is such dependency in parse tree p, it returns 1, otherwise returns 0. Since the value of $Accuracy(p)$ ranges from 0 to 1, we doesn't need to normalize its value.

Thus, the detailed procedure for dependency structure construction is,

- Parsing each sentence using Nivre's arc standard, Nivre's arc eager and Liang's dynamic algorithm, respectively.
- Combining parses outputted by three parsers into weighted directed graph, and representing its weight using equation 1.
- Using Chu-Liu-Edmonds algorithm to search final parse for each sentence.

2.3 Features for Labeling

After given dependency structure, for each relation between head and its modifier, we extract 31 types of features, which are typically exploited in syntactic dependency parsing, as our basic features. Based on these basic features, we also add a additional distance metric for each features and obtain 31 types of distance incorporated features. Besides that, we use greedy hill climbing approach to select additional 29 features to obtain better performance. Table 1 shows the basic features used in our system,

And the table 2 gives the additional features. It is worth mentioning, that the distance is calculated as the difference between the head and its modifier, which is different from the calculation reported by most literatures.

2.4 Classifier

We use the classifier from Le Zhang's Maximum Entropy Modeling Toolkit[3] and use the L-BFGS

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

	Features
Basic	**mw**:modifier's word **mp**:modifier's POS tag **hw**:head's word **hp**:head's POS tag
Combination	hw|hp,mw|mp,hw|mw hp|mp,hw|mp,hp|mw hw|hp|mw hw|hp|mp hw|mw|mp hp|mw|mp hp|mp|mp-1 hp|mp|mp+1 hp|hp-1|mp hp|hp+1|mp hp|hp-1|mp-1 hp|hp-1|mp+1 hp|hp+1|mp-1 hp|hp+1|mp+1 hp-1|mp|mp-1 hp-1|mp|mp+1 hp+1|mp|mp-1 hp+1|mp|mp+1 hw|hp|mw|mp hp|hp-1|mp|mp-1 hp|hp+1|mp|mp+1 hp|hp+1|mp|mp-1 hp|hp-1|mp|mp+1

Table 1: The basic features used in our system. -1 and +1 indicate the one on the left and right of given word.

parameter estimation algorithm with gaussian prior smoothing(Chen and Rosenfeld, 1999). We set the gaussian prior to 2 and train the model in 1000 iterations according to the previous experience.

3 Experiments

The given corpus consists of 8301 sentences for training(TR), and 569 sentences for developing(DE). For tuning parameters, we just use TR portion, while for testing, we combine two parts and retrain the parser to obtain better results. Surely, we also give results of testing set trained on TR portion for comparison. In the following of this section, we will report the detailed experimental results both on developing and testing set.

	Features
Distance	**dist**:basic features with distance
Additional	**lmw**:leftmost word of modifier **rnw**:rightnearest word of modifier **gfw**:grandfather of modifier **lmp**,**rnp**,**gfp** lmw|lmp,rnw|rnp,lmw|rnw lmp|rnp,lmw|mw,lmp|mp rnw|mw,rnp|mp,gfw|mw gfp|mp,gfw|hw,gfp|hp gfw|mw|gfp|mp lmw|lmp|mw|mp rnw|rnp|mw|mp lmw|rnw|mw,lmp|rnp|mp gfw|hw|gfp|hp gfw|mw|hw,gfp|mp|hp gfw|mw|hw|gfp|mp|hp lmw|rnw|lmp|rnp|mw|mp lmw|rnw|lmp|rnp

Table 2: The additional features used in our system.

3.1 Results on Developing Set

We first report the accuracy of dependency construction on developing set using different parsing algorithms in table 3. Note that, the features used in our system are similar to that used in their published papers(Nivre, 2003; Nivre, 2004; Huang and Sagae, 2010). From table 3 we find that although using simple method for combination over three single parsers, the system combination technique still achieves 1.1 points improvement over the highest single system. Since the Liang's algorithm is a dynamic algorithm, which enlarges the searching space in decoding, while the former two Nivre's arc al-

	Precision (%)
Nivre's arc standard	78.86
Nivre's arc eager	79.11
Liang's dynamic	79.78
System Combination	**80.85**

Table 3: Syntactic precision of different parsers on developing set.

gorithms actually still are simple beam search algorithm, thus the Liang's algorithm achieves better performance than Nivre's two algorithm, which is consistent with the experiments in Liang's paper.

To acknowledge that the better dependency structure does help to semantic relation labeling, we further predict semantic relations on different dependency structures. For comparison, we also report the performance on golden structure. Since our combi-

	Precision (%)
Nivre's arc standard	60.84
Nivre's arc eager	60.76
Liang's dynamic	61.43
System Combination	**62.92**
Golden Tree	**76.63**

Table 4: LAS of semantic relations over different parses on developing set.

national algorithm requires weight for each edges, we use the developing parsing accuracy 0.7886, 0.7911, and 0.7978 as corresponding weights for each single system. Table 4 shows, that the prediction of semantic relation could benefit from the improvement of dependency structure. We also notice that even given the golden parse tree, the performance of relation labeling is still far from perfect. Two reasons could be explained for that: first is the small size of supplied corpus, second is that the relation between head and its modifier is too fine-grained to distinguish for a classifier. Moreover, here we use golden segmentation for parsing, imagining that an automatic segmenter would further drop the accuracy both on syntactic and semantic parsing.

3.2 Results on Testing Set

Since there is a bug[4] in our final results submitted to organizers, here, in order to confirm the improvement of our method and supply comparison standard for further research, we reevaluate the correct output and report its performance on different training set. Table 5 and table 6 give the results trained on different corpus. We can see that when increasing the training size, the performance is slightly improved. Also, we find the results on testing set is consistent with that on developing set, where best dependency structure achieves the best performance.

	LAS (%)	UAS(%)
Nivre's arc standard	60.38	78.19
Nivre's arc eager	60.78	78.62
Liang's dynamic	60.85	79.09
System Combination	**62.76**	**80.23**
Submitted Error Results	55.26	71.85

Table 5: LAS and UAS on testing set trained on TR.

	LAS (%)	UAS(%)
Nivre's arc standard	60.49	78.25
Nivre's arc eager	60.99	78.78
Liang's dynamic	61.29	79.59
System Combination	**62.80**	**80.45**
Submitted Error Results	56.31	73.20

Table 6: LAS and UAS on testing set trained on TR and DE.

4 Conclusion

In this paper, we demonstrate our system framework for Chinese Semantic Dependency Parsing, and report the experiments with different configurations. We propose to use system combination to better the dependency structure construction, and then label semantic relations over refined parse tree. Final experiments show that better syntactic parsing do help to improve the accuracy of semantic relation prediction.

Acknowledgments

The authors were supported by National Science Foundation of China, Contracts 90920004, and High-Technology R&D Program (863) Project No 2011AA01A207 and 2012BAH39B03. We thank Heng Yu for generating parse tree using Liang's algorithm. We thank organizers for their generous supplied resources and arduous preparation. We also thank anonymous reviewers for their thoughtful suggestions.

[4]The bug is come from that when we converting the CoNLL-styled outputs generated by our combination system into plain text. While in developing stage, we directly used CoNLL-styled outputs as our input, thus we didn't realize this mistake.

References

Stanley F. Chen and Ronald Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. Technical report, CMU-CS-99-108.

Y.J. Chu and T.H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*, 14(1396-1400):270.

J. Edmonds, J. Edmonds, and J. Edmonds. 1968. *Optimum branchings*. National Bureau of standards.

J. Hajič, M. Ciaramita, R. Johansson, D. Kawahara, M.A. Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štěpánek, et al. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–18. Association for Computational Linguistics.

J. Hall, J. Nilsson, and J. Nivre. 2011. Single malt or blended? a study in multilingual parser optimization. *Trends in Parsing Technology*, pages 19–33.

L. Huang and K. Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1077–1086. Association for Computational Linguistics.

J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT*. Citeseer.

J. Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57. Association for Computational Linguistics.

M. Surdeanu, R. Johansson, A. Meyers, L. Màrquez, and J. Nivre. 2008. The conll-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177. Association for Computational Linguistics.

Ting Liu Wanxiang Che. 2012. Semeval-2012 Task 5: Chinese Semantic Dependency Parsing. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012)*.