Hybrid Constituent and Dependency Parsing with Tsinghua Chinese Treebank

Rui Wang[†], **Yi Zhang**[†‡]

† Department of Computational Linguistics, Saarland University
‡ LT-Lab, German Research Center for Artificial Intelligence
Saarbrücken, Germany
{*rwang,yzhang*}*@coli.uni-sb.de*

Abstract

In this paper, we describe our hybrid parsing model on Mandarin Chinese processing. The model combines the mainstream constitute and dependency parsing and the dataset we use it the Tsinghua Chinese Treebank, whose annotation has both constitutes and head information. We show the adaption of this annotation scheme to the normal constitute structure, dependency structure, and the integration of both. We achieve a f1-score of 85.23% for the constitute parsing, 82.35% for the partial head information, and 74.27% for the complete head information.

1. Background

Data-driven statistical natural language parsing techniques have seen great advance in the past decade. While most of the initial research were carried out on English, with several large-scale treebanks for different languages made available in recent years, many recent studies have been focusing on the multi-linguality of the parsing systems. Among the rich literature on parsing, several data-driven systems have claimed to be largely language-independent and easily adaptable for different treebanks. However, recent investigations have shown that the performance of these systems usually varies significantly when different datasets are used for training the models.

From a data-driven view point, one might argue that such variation does not alway reflect the difficulty in parsing the underlying language, for similar variation of performance is also observed for datasets of the same language but adopting different annotation schemes. However, it is undeniable that certain linguistic properties (e.g. morphology, word order, various types of agreements, etc.) of the language do have strong influence on the performance of particular parsing models. Therefore, in-depth evaluation of parsing systems have to be connected to particular languages and phenomena.

Mandarin Chinese, as the language with most native speakers, has received much attention in the study of parsing. Several large-scale treebanks have been developed since the mid-1990s, and now being used to develop various parsing systems. Due to the lack of inflectional morphology and relatively fixed word order in Chinese, many parsing models developed for English have been successfully adapted (despite the extra difficulties in word segmentation and part-of-speech tagging), when compared to their limited success on languages like German and Arabic.

On the other hand, in recent years the field of computational linguistics has seen many competing grammar frameworks which are used as the formalisms of syntactic parsing systems. Fair comparison of parsing systems based on different frameworks become a very difficult and important task. Most attempts on cross-framework parser evaluation either try to define (normally noisy and lossy) mappings between different representations, or resort to the feedback from the performance of applications which uses the parser in specific NLP scenarios. Two grammar frameworks have gained prominent roles in the study of parsing: constituent-based and dependency-based frameworks. Both of them express (different) views on the fundamental structures of the language, and other "deeper" frameworks can normally be considered as variants from one of them, or a mixture of both. Parsing techniques developed on these two frameworks can hopefully be adapted for other derived formalisms.

2. Tsinghua Chinese Treebank (TCT)

For Mandarin Chinese, the most well-known public available treebank is the Penn Chinese Treebank (CTB) (Xue et al., 2005). Adopting a constituent-based view, the annotation scheme of CTB is similar to that of the English PTB. Recent release of the treebank contains over 28K sentences and 781K words, but annotation errors were also frequently observed and reported. Alternative public available treebanks are the Sinica Treebank (Huang et al., 2000), and the Tsinghua Chinese Treebank (TCT) (Zhou, 2004). The latter is recently released as training and testing corpus in the Chinese parsing evaluation called CIPS-ParsEval-2009.[1] Annotated with constituent-based analyses, TCT also contain *head* information in the phrase structure trees, making it an interesting treebank for both constituent-based and dependency parsing.

It should be noted that the annotation in TCT are very different on many levels from other Chinese treebanks, e.g. CTB. Needless to mention are the differences on segmentation and POS tagsets. On the syntactic level, while most other treebanks provide complete tree annotation for sentences, the data provided in CIPS-ParsEval-2009 separate two levels of annotation, i.e. the detection of "event description units" (clauses), and syntactic tree annotation within each event description units. This is a practical annotation decision, for many chinese sentences are composed of multiple "event description units", whose syntactic structures are largely independent from each other. The detection of clause boundaries is a relatively easy task compared to the syntactic parsing task. By separating the sentences

[1] http://www.ncmmsc.org/CIPS-ParsEval-2009/index.asp

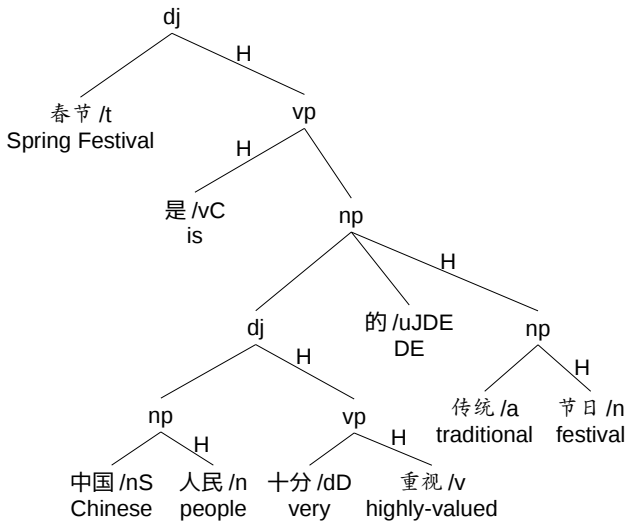


Figure 1: An Example TCT Tree in CIPS-ParsEval Format

into smaller units, the average length of the parser inputs is much shorter than usual. As an example, Figure 1 shows a headed phrase structure tree from TCT (in CIPS-ParsEval format).

3. The Hybrid Parsing Model

In this paper, we report on a series of parsing experiments carried out on the TCT, using existing data-driven constituent and dependency parsers with barest linguistic assumption, to see their (dis)similarity and mutual influences:

1. Constituent Parsing on TCT with partial head information
2. Dependency Parsing on TCT with partial phrasal information (i.e. dependency structure extraction from TCT)
3. Integration of the two structures (i.e. the conversion from the dependency structure back to the constituent structure)

3.1. Constituent Parsing

The TCT annotation in CIPS-ParsEval-2009 uses constituent trees together with head indices. It is possible to consider the phrase category and head information as a whole, and re-encode the head daughter indices in a way that can be integrated with data-driven constituent parsers. Several preprocessing steps are taken to prepare the training data for two alternative constituent tree parsing models.

For the first model, we remove the head information from the trees, and only use the constituent parser to produce the phrase boundaries and category. This results in a small PCFG grammar with only 16 different phrase categories.

In the second model, we convert the head indexing digits to the following four types,

-l left, if the head is the left-most daughter of the constituent;

-r right, if the head is the right-most daughter of the constituent;

-m middle, if the constituent has a single head that is neither on the left-most or right-most position;

-c complex, if the constituent has more than one heads.

This information is combined with the phrase categories, so that e.g. a nominal phrase with its head daughter on the right-most position will be marked as *np-r*. Obviously, the label *"-l"* and *"-r"* can be easily reverted to digit based head indices. For constituents marked with *"-m"* and have more than three daughters, further information is needed to locate the correct head. The same applies to *"-c"* that normally occurs with coordination phenomena. In the dataset, there are also phrases marked with no head *xp-*, which will be copied as it is in our conversion. After this step of conversion, the treebank contains a total of 55 different phrasal categories annotated with head indices.

Given the relatively small number of phrasal categories in both models, we decide to apply the so-called state-splitting approach to induces PCFG grammars that introduce additional subcategories automatically. More specifically, we follow the split-merge approach described in (Petrov and Klein, 2007), where new sub-categories are introduced only for more heterogeneous categories.

3.2. Dependency Extraction

In general, the extraction algorithm works recursively to acquire the lexical head of each constituent or word token, and link them to their corresponding dependent heads. Following the different constituent markers we have in the previous section (e.g. *"-l"*, *"-r"*, etc.), we deal with them separately.

The normal cases, which have single heads, can be converted directly by linking all the non-headed components to the head. For instance, "(np-2 X Y Z)" can be converted to Z→X and Z→Y. This includes both the single-headed cases and the xp-m cases mentioned in the previous section.

A more complex case is the xp-c case (e.g. "np-024", meaning a nominal phrase whose daughters whose indices are 0, 2, and 4 are heads), where the TCT annotation scheme allows multiple heads (e.g. the coordination). We firstly binarize the tree and then convert it recursively by taking the first head as the head of each constituent. For example, "(np-024 X , Y and Z)" will be binarized to "(np-0 X , (np-0 Y and Z))". After the conversion, it becomes the the same as the normal cases.

There is one case left, which is xp- without any head information. This occurs frequently when a long sentence consists of several shorter clauses. To be different from the coordination, we simply link all the clauses to the first one, e.g. "(dj- X , Y , Z)" will be converted to "X→,", "X→Y", "X→,", and "X→Z".

Notice that the conversion here is *not fully* reversible. For instance, "(np-02 X , Y)" and "(np-0 X , Y)" will be converted into the same dependency tree, "X→," and "X→Y". Consequently, in the final stage, we apply some heuristics to restore some of the information loss.

Besides the dependency structure conversion, we also need to consider the labels being given to the dependencies. Enlightened by (Hall, 2008)'s conversion algorithm, we not only annotate the dependency label with constituent names, but more, in order to preserve the information as much as possible. We use a richer annotation on each dependency label, which contains two elements,

Path All the constituent names from the head word up to the last constituent where the two words share the same ancestor on the tree;

Index The index of the shared ancestor on the corresponding constituent name path.

After applying all these steps mentioned above, an example of a complete conversion is shown in Figure 2.

3.3. Integration of the Two Structures

After obtaining both the constituent tree and the dependency tree (for the same sentence) from the parsers, we could then combine them and convert the result back to the original TCT representation. There are mainly four categories of conversion:

xp-l or xp-r For those cases with constituent names, xp-l or xp-r, we simply take the leftmost component or the rightmost component as the heads.

xp-m For the xp-m cases, we use the dependency parser's result as a hint to obtain the head. If there are only three components in that constituent, we take the second component (the middle one) as the head; if there are more than three components, we use the left-most head given by the dependency parser as the head, but it cannot be the left-most daughter of the constituent.

xp-c For the xp-c cases, we again use the dependency parser's result as a hint. The tricky part here is that sometimes the dependency parser cannot discover all the heads correctly. Therefore, we apply heuristics to fix the errors. If the number of the components is an odd number, we add all the components at the odd positions to the head list, when the dependency parser misses some of them; if the number of the components is an even number, we do the same , except for the second last component. These two rules are aiming to capture coordination like *"X , Y and Z"* and *"X , Y , and Z"*.

xp- For the *xp-* cases, we simply keep them as they are.

4. Experiments

In practice, for both mainstream statistical (syntactic) parsers, we use open source softwares, the Berkeley parser[2] (Petrov and Klein, 2007) and the MSTParser[3] (McDonald et al., 2005), partially based on our empirical study on the parser performance comparison (Zhang and Wang, 2009). The Berkeley parser is purely data-driven and it does not have linguistic assumptions. This allows us to retrain the model with minimum efforts. Some internal comparison has shown that it outperforms the modified Dan Bikel's parser[4] (Bikel, 2004), because the latter system heavily depends on the linguistic or annotation assumptions from the Penn-style treebanks. The MSTParser is a graph-based dependency parer. The best parse tree is acquired by searching for a spanning tree which maximizes the score on either a partially or a fully connected graph with all words in the sentence as nodes (Eisner, 1996; McDonald et al., 2005). For the Berkeley parser, we use the default settings. For the MSTParser, based on our experience in participating in the CoNLL 2009 shared task (Zhang et al., 2009), we use the second order setting of the parser, which includes features over pairs of adjacent edges as well as features over single edges in the graph, and other settings are default. Table 1 shows the parsing accuracy of our system on CIPS-ParsEval-2009 dataset.

From the results, we observe that the Berkeley's parser works reasonably fine and the claim is largely confirmed with a different treebank with different annotation. After 6 split-merge iterations, the resulting grammar induced complex hierarchical categories on nouns and verbs while maintaining very few subtypes on closed categories like particles and conjunction words.

In the meanwhile, multi-headed constructions are quite challenging to handle, though we apply several heuristic rules besides using the dependency parser to restore the head information. The improvement of using the dependency parser results as the backup for the constitute parsers can be seen from comparing the two rows below and above. Although there is some loss during the conversion from the dependency structure back to the constitute structure, the gain on the overall performance is substantial. After taking a comparison of the two rows above, we find that the heuristic rules do not help us much, and even decrease the partial-head f-score; while for the two rows below, the improvement is more obvious. This indicate that the heuristics can further fix the errors brought from the dependency-constitute conversion.

For instance, the following NP

> [np-0-2-4 田 赋(land-tax)/n 、(,)/wD 徭役(forced labor)/n 和(and)/cC [np-1 其他(other)/rN 杂税(misc. taxes)/n]]

is mis-parsed into

> [np-2 [np-2 田 赋(land-tax)/n 、(,)/wD 徭役(forced labor)/n] 和(and)/cC [np-1 其他(other)/rN 杂税(misc. taxes)/n]]

This is a typical error caused by the multi-headed coordination. Due to the binarinization process, the three components "田赋(land-tax)", "徭役(forced labor)", and "其他(other) 杂税(misc. taxes)" connected by "、(,)" and "和(and)" are treated as a recursive construction by our parser. We apply the heuristics to fix some of such errors, but a systematic way of handling coordination is still an open question.

Although the errors of the constitute structure parsing is not addressed in this work, we still find quite a large portion of the first type of errors could be potentially fixed by

[2] http://code.google.com/p/berkeleyparser/

[3] http://sourceforge.net/projects/mstparser/

[4] http://www.cis.upenn.edu/ dbikel/software.html

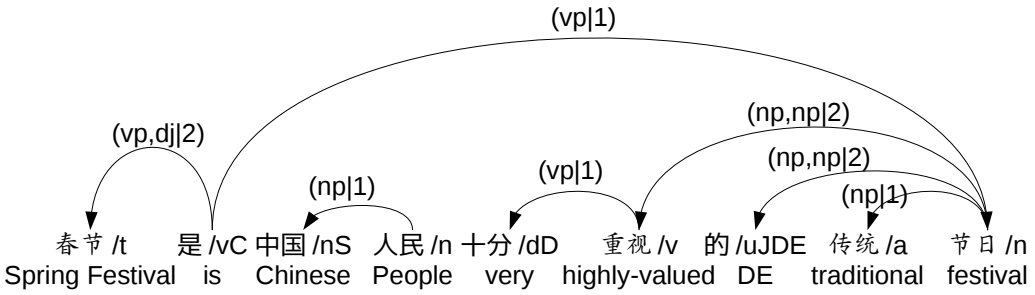


Figure 2: An example dependency tree with complex labels

Settings		F1-Scores		
With Conversion	With Heuristics	No-head	Partial-head	Full-head
No	No	84.41	81.29	73.21
No	Yes	84.41	81.21	73.75
Yes	No	85.23	80.05	72.79
Yes	Yes	85.23	82.35	74.27

Table 1: Parsing Accuracy of Our System on TCT using Different Evaluation Metrics

the dependency structure. For example, the following left-branching NP structure in the treebank

> [np-1 [np-1 财 政(finance)/n 分配(distribution)/vN] 活动(activity)/n]

but is parsed as a right-branching binary structure:

> [np-1 财 政(finance)/n [np-1 分配(distribution)/vN 活动(activity)/n]

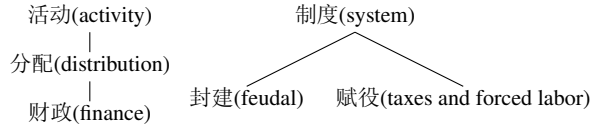
Another similar example is a right-branching NP

> [np-1 封 建(feudal)/n [np-1 赋 役(taxes and forced labor)/n 制度(system)/n]]

being mis-parsed as left-branching

> [np-1 [np-1 封 建(feudal)/n 赋 役(taxes and forced labor)/n] 制度(system)/n]]

These two complex noun phrases look very similar to each other from the constitute perspective, although the POSes are slightly different. The POS of the word "分配(distribution)" is a "vN" instead of a normal noun (i.e. "n"), meaning it has both verbal and nominal functions. Notice that in the gold standard, there is no fixed rule about combining the noun on the left or right side of the "vN" first. However, if we obtain the dependency structure of these two phrases, "财政(financial) 分配(distribution) 活动(activity)" and "封 建(feudal) 赋役(taxes and forced labor) 制度(system)", the distinction becomes obvious.



The first structure shows that "财政(finance)" and "分配(distribution)" should be combined together first and treated as modifier to "活动(activity)"; while the second structure suggests that both "封建(feudal)" and "赋役(taxes and forced labor)" are separate modifiers of "制度(system)". Unfortunately, our current parsing model does not allow the dependency structure to modify the predicted constitutes, which is on the top list of our future work.

5. Conclusion and Future Work

Although these results are still preliminary, it is interesting to have this alternative view of Chinese parsing (compared with CTB). Our first trial of combining two mainstream grammar frameworks has shown promising results.

The ongoing and future work includes 1) a closer investigation of the performance of constituent to dependency (and vice versa) conversion based on (Hall, 2008); 2) whether the richer annotation on the labels (head information on constituents and phrasal information on dependency labels) are useful for constituent and dependency parsing; and 3) how to feed the pure data-driven parsing models with more fine-grained linguistic knowledge, such as hand-crafted grammars, in order to further improve the performance.

Aknowledgements

The first author is funded by the PIRE PhD scholarship program. The second author thanks the German Excellence Cluster of Multimodal Computing and Interaction for the support of the work. We also thank the colleagues at UDS-SJTU Joint Research Lab for Language Technology for inspiring discussions.

6. References

Daniel M. Bikel. 2004. A distributional analysis of a lexicalized statistical parsing model. In *In Proceedings of*

Conference on Empirical Methods in Natural Language Processing (EMNLP 2004).

Jason Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 340–345, Copenhagen, Denmark.

Johan Hall. 2008. *Transition-Based Natural Language Parsing with Dependency and Constituency Representations*. Phd thesis, Computer Science, Växjö University.

Chu-Ren Huang, Feng-Yi Chen, Keh-Jiann Chen, Zhao ming Gao, and Kuang-Yu Chen. 2000. Sinica treebank: design criteria, annotation guidelines, and on-line interface. In *Proceedings of the second workshop on Chinese language processing: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics*, volume 12, pages 29–37, Hong Kong, China. Association for Computational Linguistics.

R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP 2005*, pages 523–530, Vancouver, Canada.

S. Petrov and D. Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HLT-NAACL 2007*, Rochester, NY, USA, April 22-27.

Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.

Yi Zhang and Rui Wang. 2009. Correlating natural language parser performance with statistical measures of the text. In *Proceedings of the 32nd Annual Conference on Artificial Intelligence (KI 2009)*, Paderborn, Germany, September.

Yi Zhang, Rui Wang, and Stephan Oepen. 2009. Hybrid multilingual parsing with hpsg for srl. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009)*, Boulder, CO, USA, June.

Qiang Zhou. 2004. Annotation scheme for chinese treebank. *Journal of Chinese Information Processing*, 18(4):1–8.