Higher-Order Constituent Parsing and Parser Combination*

Xiao Chen and Chunyu Kit
Department of Chinese, Translation and Linguistics
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong SAR, China
{cxiao2,ctckit}@cityu.edu.hk

Abstract

This paper presents a higher-order model for constituent parsing aimed at utilizing more local structural context to decide the score of a grammar rule instance in a parse tree. Experiments on English and Chinese treebanks confirm its advantage over its first-order version. It achieves its best F1 scores of 91.86% and 85.58% on the two languages, respectively, and further pushes them to 92.80% and 85.60% via combination with other high-performance parsers.

1 Introduction

Factorization is crucial to discriminative parsing. Previous discriminative parsing models usually factor a parse tree into a set of parts. Each part is scored separately to ensure tractability. In dependency parsing (DP), the number of dependencies in a part is called the *order* of a DP model (Koo and Collins, 2010). Accordingly, existing graph-based DP models can be categorized into tree groups, namely, the first-order (Eisner, 1996; McDonald et al., 2005a; McDonald et al., 2005b), second-order (McDonald and Pereira, 2006; Carreras, 2007) and third-order (Koo and Collins, 2010) models.

Similarly, we can define the *order* of constituent parsing in terms of the number of grammar rules in a part. Then, the previous discriminative constituent parsing models (Johnson, 2001; Henderson, 2004; Taskar et al., 2004; Petrov and Klein, 2008a; Petrov and Klein, 2008b; Finkel et al., 2008) are the first-order ones, because there is only one grammar rule in a part. The discriminative re-scoring models (Collins, 2000; Collins and Duffy, 2002; Charniak and Johnson, 2005; Huang, 2008) can be viewed as previous attempts to higher-order constituent parsing, using some parts containing more than one grammar rule as non-local features.

In this paper, we present a higher-order constituent parsing model[1] based on these previous works. It allows multiple adjacent grammar rules in each part of a parse tree, so as to utilize more local structural context to decide the plausibility of a grammar rule instance. Evaluated on the PTB WSJ and Chinese Treebank, it achieves its best F1 scores of 91.86% and 85.58%, respectively. Combined with other high-performance parsers under the framework of constituent recombination (Sagae and Lavie, 2006; Fossum and Knight, 2009), this model further enhances the F1 scores to 92.80% and 85.60%, the highest ones achieved so far on these two data sets.

2 Higher-order Constituent Parsing

Discriminative parsing is aimed to learn a function $f : \mathcal{S} \rightarrow \mathcal{T}$ from a set of sentences $\mathcal{S}$ to a set of valid parses $\mathcal{T}$ according to a given CFG, which maps an input sentence $s \in \mathcal{S}$ to a set of candidate parses $\mathcal{T}(s)$. The function takes the following discriminative form:

$$f(s) = \arg\max_{t \in \mathcal{T}(s)} g(t, s) \quad (1)$$

*The research reported in this paper was partially supported by the Research Grants Council of HKSAR, China, through the GRF Grant 9041597 (CityU 144410).

[1] http://code.google.com/p/gazaparser/

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 1–5,
Jeju, Republic of Korea, 8-14 July 2012.

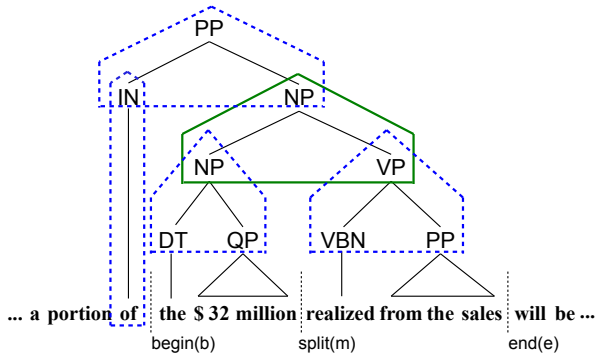

Figure 1: A part of a parse tree centered at NP → NP VP

where $g(t, s)$ is a scoring function to evaluate the event that t is the parse of s. Following Collins (2002), this scoring function is formulated in the linear form

$$g(t, s) = \theta \cdot \Psi(t, s), \tag{2}$$

where $\Psi(t, s)$ is a vector of features and θ the vector of their associated weights. To ensure tractability, this model is factorized as

$$g(t, s) = \sum_{r \in t} g(\mathcal{Q}(r), s) = \sum_{r \in t} \theta \cdot \Phi(\mathcal{Q}(r), s), \tag{3}$$

where $g(\mathcal{Q}(r), s)$ scores $\mathcal{Q}(r)$, a part centered at grammar rule instance r in t, and $\Phi(\mathcal{Q}(r), s)$ is the vector of features for $\mathcal{Q}(r)$. Each $\mathcal{Q}(r)$ makes its own contribution to $g(t, s)$. A part in a parse tree is illustrated in Figure 1. It consists of the center grammar rule instance NP → NP VP and a set of immediate neighbors, i.e., its parent PP → IN NP, its children NP → DT QP and VP → VBN PP, and its sibling IN → of. This set of neighboring rule instances forms a local structural context to provide useful information to determine the plausibility of the center rule instance.

2.1 Feature

The feature vector $\Phi(\mathcal{Q}(r), s)$ consists of a series of features $\{\phi_i(\mathcal{Q}(r), s))|i \geq 0\}$. The first feature $\phi_0(\mathcal{Q}(r), s)$ is calculated with a PCFG-based generative parsing model (Petrov and Klein, 2007), as defined in (4) below, where r is the grammar rule instance $\mathcal{A} \rightarrow \mathcal{B}\ \mathcal{C}$ that covers the span from the b-th to the e-th word, splitting at the m-th word, x, y and z are latent variables in the PCFG-based model, and $I(\cdot)$ and $O(\cdot)$ are the inside and outside probabilities, respectively.

All other features $\phi_i(\mathcal{Q}(r), s)$ are binary functions that indicate whether a configuration exists in $\mathcal{Q}(r)$ and s. These features are by their own nature in two categories, namely, lexical and structural. All features extracted from the part in Figure 1 are demonstrated in Table 1. Some back-off structural features are used for smoothing, which cannot be presented due to limited space. With only lexical features in a part, this parsing model backs off to a first-order one similar to those in the previous works. Adding structural features, each involving a least a neighboring rule instance, makes it a higher-order parsing model.

2.2 Decoding

The factorization of the parsing model allows us to develop an exact decoding algorithm for it. Following Huang (2008), this algorithm traverses a parse forest in a bottom-up manner. However, it determines and keeps the best derivation for every grammar rule instance instead of for each node. Because all structures above the current rule instance is not determined yet, the computation of its nonlocal structural features, e.g., parent and sibling features, has to be delayed until it joins an upper level structure. For example, when computing the score of a derivation under the center rule NP → NP VP in Figure 1, the algorithm will extract child features from its children NP → DT QP and VP → VBN PP. The parent and sibling features of the two child rules can also be extracted from the current derivation and used to calculate the score of this derivation. But parent and sibling features for the center rule will not be computed until the decoding process reaches the rule above, i.e., PP → IN NP.

This algorithm is more complex than the approximate decoding algorithm of Huang (2008). However, its efficiency heavily depends on the size of the parse forest it has to handle. Forest pruning (Char-

$$\phi_0(\mathcal{Q}(r), s) = \frac{\sum_x \sum_y \sum_z O(\mathcal{A}_x, b, e)\mathcal{P}(\mathcal{A}_x \rightarrow \mathcal{B}_y\ \mathcal{C}_z)I(\mathcal{B}_y, b, m)I(\mathcal{C}_z, m, e)}{I(\mathcal{S}, 0, n)} \tag{4}$$

Template		Description		Comments
Lexical feature	N-gram on inner /outer edge	$w_{b/e+l}$(l=0,1,2,3,4)	& b/e & l & NP	Similar to the *distributional similarity cluster bigrams* features in Finkel et al. (2008)
		$w_{b/e-l}$(l=1,2,3,4,5)	& b/e & l & NP	
		$w_{b/e+l}w_{b/e+l+1}$(l=0,1,2,3)	& b/e & l & NP	
		$w_{b/e-l-1}w_{b/e-l}$(l=1,2,3,4)	& b/e & l & NP	
		$w_{b/e+l}w_{b/e+l+1}w_{b/e+l+2}$($l$=0,1,2)	& b/e & l & NP	
		$w_{b/e-l-2}w_{b/e-l-1}w_{b/e-l}$($l$=1,2,3)	& b/e & l & NP	
	Bigram on edges	$w_{b/e-1}w_{b/e}$	& NP	Similar to the lexical span features in Taskar et al. (2004) and Petrov and Klein (2008b)
	Split pair	$w_{m-1}w_m$	& NP → NP VP	
	Inner/Outer pair	$w_b w_{e-1}$	& NP → NP VP	
		$w_{b-1}w_e$	& NP → NP VP	
	Rule bigram	Left & NP	& NP	Similar to the *bigrams* features in Collins (2000)
		Right & NP	& NP	
Structural feature	Parent	PP → IN NP	& NP → NP VP	Similar to the *grandparent rules* features in Collins (2000)
	Child	NP → DT QP & VP → VBN PP	& NP → NP VP	
		NP → DT QP	& NP → NP VP	
		VP → VBN PP	& NP → NP VP	
	Sibling	Left & IN → of	& NP → NP VP	

Table 1: Examples of lexical and structural feature

niak and Johnson, 2005; Petrov and Klein, 2007) is therefore adopted in our implementation for efficiency enhancement. A parallel decoding strategy is also developed to further improve the efficiency without loss of optimality. Interested readers can refer to Chen (2012) for more technical details of this algorithm.

3 Constituent Recombination

Following Fossum and Knight (2009), our constituent weighting scheme for parser combination uses multiple outputs of independent parsers. Suppose each parser generates a k-best parse list for an input sentence, the weight of a candidate constituent c is defined as

$$\omega(c) = \sum_i \sum_k \lambda_i \delta(c, t_{i,k}) f(t_{i,k}), \qquad (5)$$

where i is the index of an individual parser, λ_i the weight indicating the confidence of a parser, $\delta(c, t_{i,k})$ a binary function indicating whether c is contained in $t_{i,k}$, the k-th parse output from the i-th parser, and $f(t_{i,k})$ the score of the k-th parse assigned by the i-th parser, as defined in Fossum and Knight (2009).

The weight of a recombined parse is defined as the sum of weights of all constituents in the parse. However, this definition has a systematic bias towards selecting a parse with as many constituents as possible for the highest weight. A pruning threshold ρ, similar to the one in Sagae and Lavie (2006), is therefore needed to restrain the number of constituents in a recombined parse. The parameters λ_i and ρ are tuned by the Powell's method (Powell, 1964) on a development set, using the F1 score of PARSEVAL (Black et al., 1991) as objective.

	English	Chinese
Train.	Section 2-21	Art. 1-270,400-1151
Dev.	Section 22/24	Art. 301-325
Test.	Section 23	Art. 271-300

Table 2: Experiment Setup

4 Experiment

Our parsing models are evaluated on both English and Chinese treebanks, i.e., the WSJ section of Penn Treebank 3.0 (LDC99T42) and the Chinese Treebank 5.1 (LDC2005T01U01). In order to compare with previous works, we opt for the same split as in Petrov and Klein (2007), as listed in Table 2. For parser combination, we follow the setting of Fossum and Knight (2009), using Section 24 instead of Section 22 of WSJ treebank as development set.

In this work, the lexical model of Chen and Kit (2011) is combined with our syntactic model under the framework of product-of-experts (Hinton, 2002). A factor λ is introduced to balance the two models. It is tuned on a development set using the gold sec-

	English			Chinese		
	R(%)	P(%)	F1(%)	R(%)	P(%)	F1(%)
Berkeley parser	89.71	90.03	*89.87*	82.00	84.48	*83.22*
First-order	91.33	91.79	91.56	84.14	86.23	85.17
Higher-order	91.62	92.11	91.86	84.24	86.54	85.37
Higher-order+λ	91.60	92.13	**91.86**	84.45	86.74	**85.58**
Stanford parser	-	-	-	77.40	79.57	78.47
C&J parser	91.04	91.76	91.40	-	-	-
Conbination	92.02	93.60	**92.80**	82.44	89.01	**85.60**

Table 3: The performance of our parsing models on the English and Chinese test sets.

System	F1(%)	EX(%)
Single		
Charniak (2000)	89.70	
Berkeley parser	89.87	36.7
Bod (2003)	90.70	
Carreras et al. (2008)	91.1	
Re-scoring		
Collins (2000)	89.70	
Charniak and Johnson (2005)	91.02	
The parser of Charniak and Johnson	91.40	43.54
Huang (2008)	91.69	43.5
Combination		
Fossum and Knight (2009)	92.4	
Zhang et al. (2009)	92.3	
Petrov (2010)	91.85	41.9
Self-training		
Zhang et al. (2009) (s.t.+combo)	92.62	
Huang et al. (2010) (single)	91.59	40.3
Huang et al. (2010) (combo)	92.39	43.1
Our single	91.86	40.89
Our combo	**92.80**	41.60

Table 4: Performance comparison on the English test set

System	F1(%)	EX(%)
Single		
Charniak (2000)	80.85	
Stanford parser	78.47	26.44
Berkeley parser	83.22	31.32
Burkett and Klein (2008)	84.24	
Combination		
Zhang et al. (2009) (combo)	85.45	
Our single	85.56	31.61
Our combo	**85.60**	29.02

Table 5: Performance comparison on the Chinese test set

tion search algorithm (Kiefer, 1953). The parameters θ of each parsing model are estimated from a training set using an averaged perceptron algorithm, following Collins (2002) and Huang (2008).

The performance of our *first-* and *higher-order* parsing models on all sentences of the two test sets is presented in Table 3, where λ indicates a tuned balance factor. This parser is also combined with the parser of Charniak and Johnson (2005)[2] and the Stanford. parser[3] The best combination results in Table 3 are achieved with k=70 for English and k=100 for Chinese for selecting the k-best parses. Our results are compared with the best previous ones on the same test sets in Tables 4 and 5. All scores listed in these tables are calculated with `evalb`,[4] and EX is the *complete match rate*.

5 Conclusion

This paper has presented a higher-order model for constituent parsing that factorizes a parse tree into larger parts than before, in hopes of increasing its power of discriminating the true parse from the others without losing tractability. A performance gain of 0.3%-0.4% demonstrates its advantage over its first-order version. Including a PCFG-based model as its basic feature, this model achieves a better performance than previous single and re-scoring parsers, and its combination with other parsers performs even better (by about 1%). More importantly, it extends the existing works into a more general framework of constituent parsing to utilize more lexical and structural context and incorporate more strength of various parsing techniques. However, higher-order constituent parsing inevitably leads to a high computational complexity. We intend to deal with the efficiency problem of our model with some advanced parallel computing technologies in our future works.

[2] ftp://ftp.cs.brown.edu/pub/nlparser/

[3] http://nlp.stanford.edu/software/lex-parser.shtml

[4] http://nlp.cs.nyu.edu/evalb/

References

E. Black, S. Abney, D. Flickenger, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of DARPA Speech and Natural Language Workshop*, pages 306–311.

Rens Bod. 2003. An efficient implementation of a new DOP model. In *EACL 2003*, pages 19–26.

David Burkett and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *EMNLP 2008*, pages 877–886.

Xavier Carreras, Michael Collins, and Terry Koo. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *CoNLL 2008*, pages 9–16.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *EMNLP-CoNLL 2007*, pages 957–961.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *ACL 2005*, pages 173–180.

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

Xiao Chen and Chunyu Kit. 2011. Improving part-of-speech tagging for context-free parsing. In *IJCNLP 2011*, pages 1260–1268.

Xiao Chen. 2012. *Discriminative Constituent Parsing with Localized Features*. Ph.D. thesis, City University of Hong Kong.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *ACL 2002*, pages 263–270.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *ICML 2000*, pages 175–182.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP 2002*, pages 1–8.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 1996*, pages 340–345.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *ACL-HLT 2008*, pages 959–967.

Victoria Fossum and Kevin Knight. 2009. Combining constituent parsers. In *NAACL-HLT 2009*, pages 253–256.

James Henderson. 2004. Discriminative training of a neural network statistical parser. In *ACL 2004*, pages 95–102.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800.

Zhongqiang Huang, Mary Harper, and Slav Petrov. 2010. Self-training with products of latent variable grammars. In *EMNLP 2010*, pages 12–22.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *ACL-HLT 2008*, pages 586–594.

Mark Johnson. 2001. Joint and conditional estimation of tagging and parsing models. In *ACL 2001*, pages 322–329.

J. Kiefer. 1953. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society*, 4:502–506.

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *ACL 2010*, pages 1–11.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *EACL 2006*, pages 81–88.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *ACL 2005*, pages 91–98.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *EMNLP-HLT 2005*, pages 523–530.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *NAACL-HLT 2007*, pages 404–411.

Slav Petrov and Dan Klein. 2008a. Discriminative log-linear grammars with latent variables. In *NIPS 20*, pages 1–8.

Slav Petrov and Dan Klein. 2008b. Sparse multi-scale grammars for discriminative latent variable parsing. In *EMNLP 2008*, pages 867–876.

Slav Petrov. 2010. Products of random latent variable grammars. In *NAACL-HLT 2010*, pages 19–27.

M. J. D. Powell. 1964. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer Journal*, 7(2):155–162.

Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *NAACL-HLT 2006*, pages 129–132.

Ben Taskar, Dan Klein, Mike Collins, Daphne Koller, and Christopher Manning. 2004. Max-margin parsing. In *EMNLP 2004*, pages 1–8.

Hui Zhang, Min Zhang, Chew Lim Tan, and Haizhou Li. 2009. K-best combination of syntactic parsers. In *EMNLP 2009*, pages 1552–1560.