Combining Constituent Parsers

Victoria Fossum
Dept. of Computer Science
University of Michigan
Ann Arbor, MI 48104
vfossum@umich.edu

Kevin Knight
Information Sciences Institute
University of Southern California
Marina del Rey, CA 90292
knight@isi.edu

Abstract

Combining the 1-best output of multiple parsers via parse selection or parse hybridization improves f-score over the best individual parser (Henderson and Brill, 1999; Sagae and Lavie, 2006). We propose three ways to improve upon existing methods for parser combination. First, we propose a method of parse hybridization that recombines *context-free productions* instead of *constituents*, thereby preserving the structure of the output of the individual parsers to a greater extent. Second, we propose an efficient linear-time algorithm for computing expected f-score using Minimum Bayes Risk parse selection. Third, we extend these parser combination methods from multiple 1-best outputs to multiple n-best outputs. We present results on WSJ section 23 and also on the English side of a Chinese-English parallel corpus.

1 Introduction

Parse quality impacts the quality of downstream applications such as syntax-based machine translation (Quirk and Corston-Oliver, 2006). Combining the output of multiple parsers can boost the accuracy of such applications. Parses can be combined in two ways: *parse selection* (selecting the best parse from the output of the individual parsers) or *parse hybridization* (constructing the best parse by recombining sub-sentential components from the output of the individual parsers).

1.1 Related Work

(Henderson and Brill, 1999) perform parse selection by maximizing the expected precision of the selected parse with respect to the set of parses being combined. (Henderson and Brill, 1999) and (Sagae and Lavie, 2006) propose methods for parse hybridization by recombining constituents.

1.2 Our Work

In this work, we propose three ways to improve upon existing methods for parser combination.

First, while constituent recombination (Henderson and Brill, 1999; Sagae and Lavie, 2006) gives a significant improvement in f-score, it tends to flatten the structure of the individual parses. To illustrate, Figures 1 and 2 contrast the output of the Charniak parser with the output of constituent recombination on a sentence from WSJ section 24. We recombine *context-free productions* instead of *constituents*, producing trees containing only context-free productions that have been seen in the individual parsers' output (Figure 3).

Second, the parse selection method of (Henderson and Brill, 1999) selects the parse with maximum expected *precision*; here, we present an efficient, linear-time algorithm for selecting the parse with maximum expected *f-score* within the Minimum Bayes Risk (MBR) framework.

Third, we extend these parser combination methods from 1-best outputs to n-best outputs. We present results on WSJ section 23 and also on the English side of a Chinese-English parallel corpus.

Proceedings of NAACL HLT 2009: Short Papers, pages 253–256,
Boulder, Colorado, June 2009.

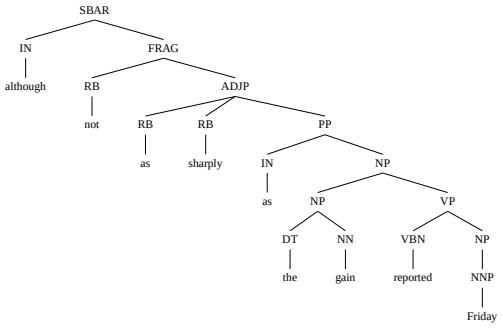

Figure 1: Output of Charniak Parser

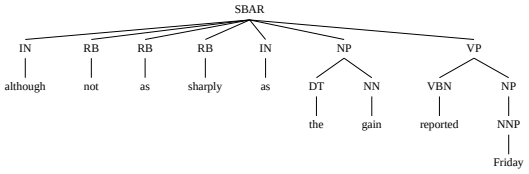

Figure 2: Output of Constituent Recombination

2 Parse Selection

In the MBR framework, although the true reference parse is unknown, we assume that the individual parsers' output forms a reasonable distribution over possible reference parses. We compute the expected f-score of each parse tree p_i using this distribution:

$$expected\ f(p_i) = \sum_{p_j} f(p_i, p_j) \cdot pr(p_j)$$

where $f(p_i, p_j)$ is the f-score of parse p_i with respect to parse p_j and $pr(p_j)$ is the prior probability of parse p_j. We estimate $pr(p_j)$ as follows: $pr(p_j) = pr(parser_k) \cdot pr(p_j|parser_k)$, where $parser_k$ is the parser generating p_j. We set $pr(parser_k)$ according to the proportion of sentences in the development set for which the 1-best output of $parser_k$ achieves the highest f-score of any individual parser, breaking ties randomly.

When $n = 1$, $pr(p_j|parser_k) = 1$ for all p_j; when $n > 1$ we must estimate $pr(p_j|parser_k)$, the distribution over parses in the n-best list output by any given parser. We estimate this distribution using the model score, or log probability, given by $parser_k$ to each entry p_j in its n-best list:

$$pr(p_j|parser_k) = \frac{e^{\alpha * score_{j,k}}}{\sum_{j'=1}^{n} e^{\alpha * score_{j',k}}}$$

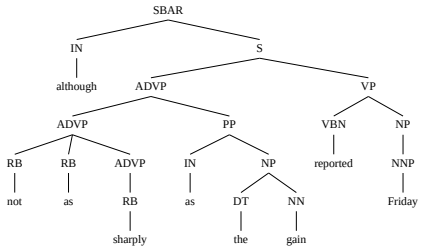

Figure 3: Output of Context-Free Production Recombination

Parser	wsj		ce	
	dev	test	dev	test
Berkeley (Petrov and Klein, 2007)	88.6	89.3	82.9	83.5
Bikel–Collins Model 2 (Bikel, 2002)	87.0	88.2	81.2	80.6
Charniak (Charniak and Johnson, 2005)	90.6	**91.4**	84.7	**84.1**
Soricut–Collins Model 2 (Soricut, 2004)	87.3	88.4	82.3	82.1
Stanford (Klein and Manning, 2003)	85.4	86.4	81.3	80.1

Table 1: F-Scores of 1-best Output of Individual Parsers

We tune α on a development set to maximize f-score,[1] and select the parse p_i with highest expected f-score.

Computing exact expected f-score requires $O(m^2)$ operations per sentence, where m is the number of parses being combined. We can compute an approximate expected f-score in $O(m)$ time. To do so, we compute expected precision for all parses in $O(m)$ time by associating with each unique constituent c_i a list of parses in which it occurs, plus the total probability q_i of those parses. For each parse p associated with c_i, we increment the expected precision of that parse by $q_i/size(p)$. This computation yields the same result as the $O(m^2)$ algorithm. We carry out a similar operation for expected recall. We then compute the harmonic mean of expected precision and expected recall, which closely approximates the true expected f-score.

[1] A low value of α creates a uniform distribution, while a high value concentrates probability mass on the 1-best entry in the n-best list. In practice, tuning α produces a higher f-score than setting α to the value that exactly reproduces the individual parser's probability distribution.

Parse Selection: Minimum Bayes Risk												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	91.7	90.5	91.1	92.5	91.8	92.0	87.1	84.6	85.8	86.7	83.7	85.2
n=10	92.1	90.8	**91.5**	92.4	91.7	92.0	87.9	85.3	86.6	87.7	84.4	**86.0**
n=25	92.1	90.9	**91.5**	92.4	91.7	92.0	88.0	85.4	**86.7**	87.4	84.2	85.7
n=50	92.1	91.0	**91.5**	92.4	91.7	**92.1**	88.0	85.3	86.6	87.6	84.3	85.9

Table 2: Precision, Recall, and F-score Results from Parse Selection

3 Constituent Recombination

(Henderson and Brill, 1999) convert each parse into constituents with syntactic labels and spans, and weight each constituent by summing $pr(parser_k)$ over all parsers k in whose output the constituent appears. They include all constituents with weight above a threshold $t = \frac{m+1}{2}$, where m is the number of input parses, in the combined parse.

(Sagae and Lavie, 2006) extend this method by tuning t on a development set to maximize f-score.[2] They populate a chart with constituents whose weight meets the threshold, and use a CKY-style parsing algorithm to find the heaviest tree, where the weight of a tree is the sum of its constituents' weights. Parsing is not constrained by a grammar; any context-free production is permitted. Thus, the combined parses may contain context-free productions not seen in the individual parsers' outputs. While this failure to preserve the structure of individual parses does not affect f-score, it may hinder downstream applications.

To extend this method from 1-best to n-best lists, we weight each constituent by summing $pr(parser_k) \cdot pr(p_j|parser_k)$ over all parses p_j generated by $parser_k$ in which the constituent appears.

4 Context-Free Production Recombination

To ensure that all context-free productions in the combined parses have been seen in the individual parsers' outputs, we recombine context-free productions rather than constituents. We convert each parse into context-free productions, labelling each constituent in the production with its span and syntactic category and weighting each production by summing $pr(parser_k) \cdot pr(p_j|parser_k)$ over all parses p_j generated by $parser_k$ in which the production appears. We re-parse the sentence with these productions, returning the heaviest tree (where the weight of a tree is the sum of its context-free productions' weights). We optimize f-score by varying the trade-off between precision and recall using a derivation length penalty, which we tune on a development set.[3]

5 Experiments

Table 1 illustrates the 5 parsers used in our combination experiments and the f-scores of their 1-best output on our data sets. We use the n-best output of the Berkeley, Charniak, and Soricut parsers, and the 1-best output of the Bikel and Stanford parsers. All parsers were trained on the standard WSJ training sections. We use two corpora: the WSJ (sections 24 and 23 are the development and test sets, respectively) and English text from the LDC2007T02 Chinese-English parallel corpus (the development and test sets contain 400 sentences each).

6 Discussion & Conclusion

Results are shown in Tables 2, 3, and 4. On both test sets, constituent recombination achieves the best f-score (1.0 points on WSJ test and 2.3 points on Chinese-English test), followed by context-free production combination, then parse selection, though the differences in f-score among the combination methods are not statistically significant. Increasing the n-best list size from 1 to 10 improves parse selection and context-free production recombination,

[2] A high threshold results in high precision, while a low threshold results in high recall.

[3] By subtracting higher(lower) values of this length penalty from the weight of each production, we can encourage the combination method to favor trees with shorter(longer) derivations and therefore higher precision(recall) at the constituent level.

Parse Hybridization: Constituent Recombination												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	92.5	90.3	91.4	93.0	91.6	92.3	89.2	84.6	86.8	89.1	83.6	86.2
n=10	92.6	90.5	**91.5**	93.1	91.7	**92.4**	89.9	84.4	**87.1**	89.9	83.2	**86.4**
n=25	92.6	90.5	**91.5**	93.2	91.7	**92.4**	89.9	84.4	87.0	89.7	83.4	**86.4**
n=50	92.6	90.5	**91.5**	93.1	91.7	**92.4**	89.9	84.4	**87.1**	89.7	83.2	86.3

Table 3: Precision, Recall, and F-score Results from Constituent Recombination

Parse Hybridization: Context-Free Production Recombination												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	91.7	91.0	91.4	92.1	91.9	92.0	86.9	85.4	86.2	86.2	84.3	85.2
n=10	92.1	90.9	91.5	92.5	91.8	**92.2**	87.8	85.1	**86.4**	86.2	84.3	**86.1**
n=25	92.2	91.0	**91.6**	92.5	91.8	**92.2**	87.8	85.1	**86.4**	87.6	84.6	**86.1**
n=50	92.1	90.8	91.4	92.4	91.7	92.1	87.6	84.9	86.2	87.7	84.6	**86.1**

Table 4: Precision, Recall, and F-score Results from Context-Free Production Recombination

though further increasing n does not, in general, help.[4] Chinese-English test set f-score gets a bigger boost from combination than WSJ test set f-score, perhaps because the best individual parser's baseline f-score is lower on the out-of-domain data.

We have presented an algorithm for parse hybridization by recombining context-free productions. While constituent recombination results in the highest f-score of the methods explored, context-free production recombination produces trees which better preserve the syntactic structure of the individual parses. We have also presented an efficient linear-time algorithm for selecting the parse with maximum expected f-score.

Acknowledgments

We thank Steven Abney, John Henderson, and Kenji Sagae for helpful discussions. This research was supported by DARPA (contract HR0011-06-C-0022) and by NSF ITR (grant IIS-0428020).

[4]These diminishing gains in f-score as n increases reflect the diminishing gains in f-score of the oracle parse produced by each individual parser as n increases.

References

Daniel M. Bikel. 2004. *Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine.* In Proceedings of HLT.

Eugene Charniak and Mark Johnson. 2005. *Coarse-to-fine n-best parsing and MaxEnt discriminative reranking*. In Proceedings of ACL.

Michael Collins and Terry Koo. 2005. *Discriminative Reranking for Natural Language Parsing*. Computational Linguistics, 31(1):25-70.

John C. Henderson and Eric Brill. 2000. *Exploiting Diversity in Natural Language Processing: Combining Parsers.* In Proceedings of EMNLP.

Dan Klein and Christopher D. Manning. 2003. *Accurate Unlexicalized Parsing.* In Proceedings of ACL.

Slav Petrov and Dan Klein. 2007. *Improved Inference for Unlexicalized Parsing*. In Proceedings of HLT-NAACL.

Chris Quirk and Simon Corston-Oliver. 2006. *The Impact of Parse Quality on Syntactically-Informed Statistical Machine Translation.* In Proceedings of EMNLP.

Kenji Sagae and Alon Lavie. 2006. *Parser Combination by Reparsing*. In Proceedings of HLT-NAACL.

Radu Soricut. 2004. *A Reimplementation of Collins' Parsing Models*. Technical report, Information Sciences Institute, Department of Computer Science, University of Southern California.