Reranking the Berkeley and Brown Parsers*

Mark Johnson
Department of Computing
Macquarie University
Sydney, Australia
mjohnson@science.mq.edu.au

Ahmet Engin Ural
Cognitive and Linguistic Sciences
Brown University
Providence, RI, USA
aeural@gmail.com

Abstract

The Brown and the Berkeley parsers are two state-of-the-art generative parsers. Since both parsers produce n-best lists, it is possible to apply reranking techniques to the output of both of these parsers, and to their union. We note that the standard reranker feature set distributed with the Brown parser does not do well with the Berkeley parser, and propose an extended set that does better. An ablation experiment shows that different parsers benefit from different reranker features.

1 Introduction

Syntactic parsing is the task of identifying the phrases and clauses in natural language sentences. It has been intensively studied primarily because it is generally believed that identifying syntactic structure is a first step towards semantic interpretation. This paper focuses on parsing the Wall Street Journal (WSJ) section of the University of Pennsylvania treebank corpus (Marcus et al., 1993). There are a large number of different approaches to this task. For simplicity we focus on two popular generative statistical parsing models: Charniak's "Maximum Entropy Inspired" parser (Charniak and Johnson, 2005) and Petrov's "split-merge" parser (Petrov et al., 2006). We follow conventional informal usage and refer to these as the "Brown" and the "Berkeley" parsers respectively.

Briefly, the Berkeley parser is a smoothed PCFG whose non-terminals are refinements of the original treebank grammar obtained by an automatic split-merge procedure, while the Brown parser is effectively a smoothed PCFG whose non-terminals encode a wide variety of manually chosen conditioning information, such as heads, governors, etc. The Berkeley parser is usually viewed as unlexicalized (although the preterminals may be split so finely that they may be viewed as identifying lexical clusters), while essentially every distribution used in the Brown parser conditions on lexical information. Even from this cursory description it is clear that these parsers parsers extract generalizations from the training data in different ways.

This paper applies reranking (Collins and Koo, 2005) to the n-best output of both parsers individually, as well as to an n-best list consisting of the union of the outputs of both parsers. We are interested to see whether the same kinds of features improve the performance of both the Berkeley and the Brown parsers, or whether successful reranking requires features that are specially tuned to the parser it is applied to. Finally, we are interested in the performance of the reranker trained on the union n-best lists. Combining the output of multiple parsers in other more complex ways has been previously demonstrated to improve overall accuracy, so it is interesting to see if the relatively simple method used here improves parsing accuracy as well.

The approach of Zhang et al. (2009) is closest to the work described here. They combine n-best lists produced by the same parsers as we do, but use only a relatively small set of features (the log probabil-

* We would like to thank Eugene Charniak and the other members of BLLIP for their helpful advice on this work. Naturally all errors remain our own.

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 665–668,
Los Angeles, California, June 2010.

Trees	Reranker features	
	standard	extended
Berkeley	91.6	91.7
Brown	**91.8**	91.6
Combined	**91.8**	**91.9**

Table 1: The f-scores on section 22 of rerankers trained on folds 1–18 by minimizing a regularized "MaxEnt" objective (negative log likelihood with a Gaussian regularizer) using L-BFGS. The weight of the regularizer was tuned to optimize f-score on folds 19–20.

ity of the parses plus a constituent overlap feature), while we investigate models with millions of features here. They report a higher f-score than we do when they replace the generative Brown parser with the the self-trained discriminatively-reranked parser of McClosky et al. (2006), but with inputs provided by the generative Berkeley and Brown n-best parsers they report an f-score of 91.43 on section 23, which is consistent with the results reported here.

2 Experimental setup

We ran both parsers in 50-best mode, and constructed 20-fold cross-validated training data as described in Collins and Koo (2005) and Charniak and Johnson (2005), i.e., the trees in sections 2–21 of the WSJ treebank were divided into 20 equal-sized folds, and the parses for each fold were generated by a parser trained on the trees in the other folds. Then sections 22, 23 and 24 were parsed using the standard "out-of-the-box" parser. Following the suggestion in Collins and Koo (2005), in order to avoid over-training on section 23 all reranking experiments reported here (except the final one) used folds 1–18 as training data, used folds 19–20 as development data and used section 22 as test data. (The averaged perceptron algorithm does not require development data, so the experiments using that algorithm report averages over folds 19–20 and section 22).

The Berkeley parser can be run in many modes; in order to produce the 20-fold training data we ran the Berkeley trainer with 6 splits, and ran the resulting parsers in "accurate" mode. It failed to produce any parses for 12 sentences in sections 2–21 and one sentence in section 24. The Brown parser was trained using the "out-of-the-box" settings, and produced parses for all sentences.

Using the reranker features distributed with the Brown reranker (Charniak and Johnson, 2005), which we call the "standard" set below, we obtained no overall improvement in f-score when either reranking the Berkeley parser n-best lists alone, or when the Berkeley parses were combined with the Brown parses.

However, it is possible that these results reflect the fact that the features used by the reranker were chosen because they improve the Brown parser, i.e., they are the result of feature selection based on reranking the Brown parser's n-best lists. In order to determine if this is the case, we developed an "extended" feature set that incorporates a wider set of features, specifically including features that capture global properties of the tree that might be harder for the Berkeley parser to learn.

Our extended feature set consists of 4,256,553 features, which are instances of 162 feature classes, which in turn are grouped into 20 feature "super-classes". By contrast, the standard feature set contains 1,333,950 features in 90 feature classes, grouped into 14 super-classes. A brief description of the extended feature set super-classes follows:

Parser: an indicator feature indicating which parsers generated this parse,

RelLogP: the log probability of this parse according to each parser,

InterpLogCondP: an indicator feature based on the binned log conditional probability according to each parser,

RightBranch: an indicator function of each node that lies on the right-most branch of the parse tree,

Heavy: an indicator function based on the size and location of each nonterminal (designed to identify the locations of "heavy" phrases),

LeftBranchLength: an indicator function of the binned length of each left-branching chain,

RightBranchLength: an indicator function of the binned length of each right-branching chain,

Rule: an indicator function of parent and children categories, optionally with head POS annotations,

NNGram: and indicator function of parent and n-gram sequences of children categories, optionally head

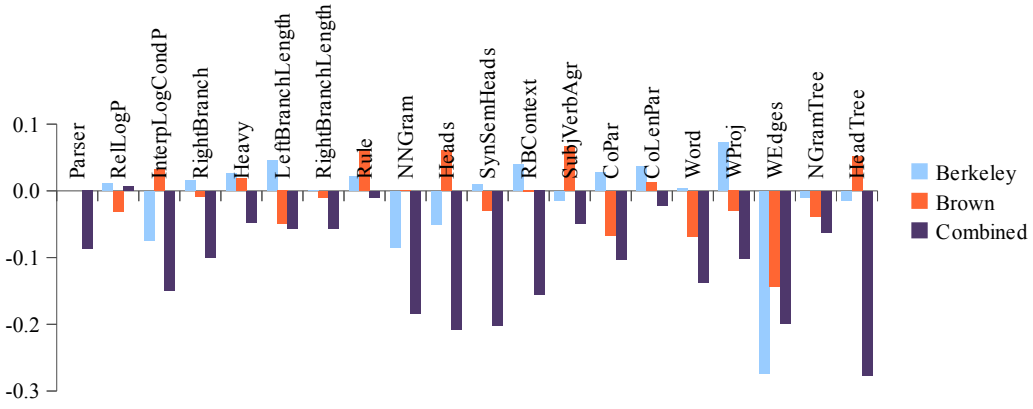

Figure 1: The average change in f-score on folds 19–20 and section 22 caused by removing a feature superclass from the extended feature set and retraining. Difference in scores less that 0.1% are probably not significant. In this experiment all rerankers were trained using the averaged perceptron algorithm. With the full extended feature set, rerankers trained with the averaged perceptron algorithm achieve f-scores of 91.2% on both the Berkeley and Brown parses, and 91.6% on the combined parses.

annotated, inspired by the n-gram rule features described in Collins and Koo (2005),

Heads: an indicator function of "head-to-head" dependencies,

SynSemHeads: an indicator function of the pair of syntactic (i.e., functional) and semantic (i.e., lexical) heads of each non-terminal,

RBContext: an indicator function of how much each subtree deviates from from right-branching,

SubjVerbAgr: an indicator function of whether subject-verb agreement is violated,

CoPar: an indicator function that fires when conjoined phrases in a coordinate structure have approximately parallel syntactic structure,

CoLenPar: an indicator function that fires when conjoined phrases in a coordinate structure have approximately the same length,

Word: an indicator function that identifies words and their preterminals,

WProj: an indicator function that identifies words and their phrasal projections up to their maximal projection,

WEdges: an indicator function that identifies the words and POS tags appearing at the edges of each nonterminal,

NGramTree: an indicator function of the subtree consisting of nodes connecting each pair of adjacent words in the parse tree, and

HeadTree: a tree fragment consisting of a head word and its projection up to its maximal projection, plus all of the siblings of each node in this sequence (this is like an auxiliary tree in a TAG).

The InterpLogCondP features were designed to capture non-linearities in the way that the Berkeley and Brown parsers assign probabilities to trees. We deliberately added features that incorporated linguistic notions such as head, governor and maximal projection, as the Berkeley parser does not explicitly condition on such information (in contrast to the Brown parser, which does).

In fact, as the reader can verify the differences in f-scores between rerankers containing the extended features and the standard features is minimal. In order to better study the importance of the various features we conducted an ablation study, in which we trained rerankers which were missing one feature superclass from the 20 superclasses of the extended feature set. In order to speed training time we used the averaged perceptron algorithm (Collins, 2002) (it converges an order of magnitude faster than the L-BFGS algorithm we used in the other experiments, but the f-score of the model estimated with the averaged perceptron is approximately 0.1% lower than when using L-BFGS). The results from this experiment are shown in Figure 1. The averaged perceptron algorithm does not rely on the development data

(folds 19–20), so the results we report are average f-scores on the development data and on section 22 (we did this because the differences are small, so a larger evaluation set may be able to detect differences more reliably).

It is interesting that linguistically-informed features (such as Heads, SynSemHeads and HeadTree) seem to be much more important when reranking the combined n-best lists than when reranking the output of either parser alone. This suggests that the log probability scores from both parsers are internally consistent, but need to be recalibrated when the parses are combined. The log probability scores from the parsers themselves (in the form of the InterpLogCondP feature) are also supplying useful information that the reranker features on their own are not providing. Finally, the WEdges feature, which identifies the words and POS at the left and right boundaries of each nonterminal, also provides extremely useful information, especially for reranking the Berkeley parser.

3 Conclusion

Reranking is a straight-forward method for improving the accuracy of n-best parsers. While one might have hoped that reranking the n-best output of the Berkeley parser, or the union of the outputs of the Berkeley and Brown parsers, would dramatically improve overall f-score, this seems not to be the case. It's possible that the features of current rerankers have been implicitly designed to work well with parsers like the Brown parser, but a reranker with a dramatically enlarged feature set performs only marginally better. This result was confirmed by training a reranker with the extended features on the union of the output of the Berkeley and Brown parsers on sections 2–21 and testing on section 23 (i.e., the standard WSJ parsing evaluation), which achieved an f-score of 91.49%; approximately 0.1% higher than a reranker with the standard feature set trained on the output of the Brown parser alone.

Acknowledgments

This research was funded by NSF awards 0530118, 0544127 and 0631667.

References

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–70.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8. Association for Computational Linguistics.

Michell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA, June. Association for Computational Linguistics.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Hui Zhang, Min Zhang, Chew Lim Tan, and Haizhou Li. 2009. K-best combination of syntactic parsers. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1552–1560, Singapore. Association for Computational Linguistics.