Tree-bank Grammars

Eugene Charniak

Department of Computer Science
Brown University
Providence, Rhode Island 02912

CS-96-02
January 1996

Tree-bank Grammars*

Eugene Charniak
Department of Computer Science, Brown University

January 5, 1996

Abstract

By a "tree-bank grammar" we mean a context-free grammar created by reading the production rules directly from hand-parsed sentences in a tree bank. Common wisdom has it that such grammars do not perform well, though we know of no published data on the issue. The primary purpose of this paper is to show that the common wisdom is wrong. In particular we present results on a tree-bank grammar based on the Penn Wall Street Journal tree bank. To the best of our knowledge, this grammar out-performs all other non-word-based statistical parsers/grammars on this corpus. That is, it out-performs parsers that consider the input as a string of tags and ignore the actual words of the corpus.

1 Introduction

The simplest way to "learn" a context-free grammar from a parsed corpus (a "tree bank"), is to read the grammar off the parsed sentences. That is, if we have the sentence diagrammed in Figure 1 we can read the following rules off this diagram:

$$\begin{array}{lcl} \mathsf{S} & \rightarrow & \mathsf{NP\ VP} \\ \mathsf{NP} & \rightarrow & \mathsf{pron} \\ \mathsf{VP} & \rightarrow & \mathsf{vb\ NP} \\ \mathsf{NP} & \rightarrow & \mathsf{dt\ nn} \end{array}$$

*This research was supported in part by NSF grant IRI-9319516.

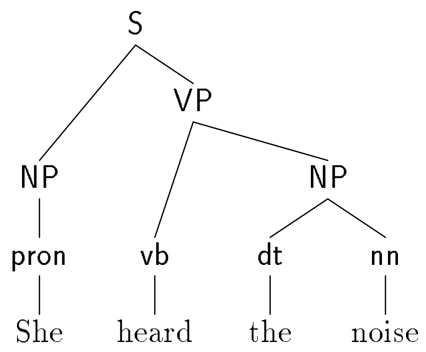

Figure 1: A simple parsed entry in a tree-bank

We call grammars obtained in this fashion "tree-bank grammars."

It is common wisdom that tree-bank grammars do not work well. I have heard this from several well-known researchers in the statistical NLP community, and the complete lack of any performance results on such grammars suggests that if they have been researched the results did not warrant publication. The primary purpose of this paper is to refute this common wisdom. The next section does this by presenting some results for a tree-bank grammar. Section 3 compares these results to prior work and addresses why our results differ from what common expectations would predict.

The parser used in our experiments is, for the most part, a standard chart parser. It does differ from the standard, however, in two ways. One is an efficiency matter — we improved its ability to search for the most probable parse. This is discussed briefly in section 3 as well. The second difference is more unusual. From impressionistic evidence, we have come to believe that standard PCFGs do not match English's preference for right-branching structures. In section 4 we present some ideas on how this might be corrected and show how these ideas contribute to the performance results of section 2.

2 The Experiment

We used as our tree bank a preliminary version of the Penn parsed Wall Street Journal text. We divided the sentences into two separate corpora, about 30000 words for testing and about ten times that for training. We ignored all sentences of length greater than 40 in the testing data. The actual number of such sentences is quite low, as the overall average sentence length is about 22 words and punctuation.

With the exception of the right-bracketing correction to be discussed later, the training was particularly simple. We obtained a context-free grammar (CFG) by reading the rules off all the sentences in the training data. Trace elements indicated in the parse where ignored. To create a probabilistic CFG, a PCFG, we assigned a probability to each rule by observing how often each rule was used in the training corpus. So, if r is a rule, let $\mid r \mid$ be the number of times r occurred in the parsed corpus and $\lambda(r)$ be the non-terminal that r expands. Then the probability assigned to r is given by

$$p(r) = \frac{\mid r \mid}{\sum_{r' \in \{r' \mid \lambda(r') = \lambda(r)\}} \mid r' \mid} \tag{1}$$

Originally we used as our set of non-terminals those specified in [6]. However, it was found that other non-terminals were used in the tree bank as well. Two of these (ORD and PRT) we added to the grammar, but for the majority we simply ignored any rule in which they occurred. It was also necessary to add a new start symbol, S1, as many of the parses in our version of the tree bank had the following form:

((S (NP The dog) (VP chewed (NP the bone))) .)

Note the topmost unlabeled bracketing with (in this case) two constituents, the S and the final period. We handled such cases by labeling this bracket S1.

We used the full set of Penn-tree-bank terminal parts of speech augmented by two new parts of speech, the auxiliary verb categories aux and auxg. We introduced these by assigning all occurrences of the most common aux-verbs (e.g., have, had, is, am, are, etc.) to their respective categories.

The grammar obtained had 10,605 rules of which only 3943 occurred more than once. We used all the rules, though we give some results in which only a subset are used.

We obtained the most probable parse of each sentence using the standard extension of the HMM Viterbi algorithm to PCFGs. We call this parse the map (maximum a posteriori) parse. We then compared the map parse to the one given in the tree-bank testing data. We measured performance by three observations:

Sentence Lengths	Average Length	Precision	Recall	Accuracy
2-12	8.7	88.6	91.7	97.9
2-16	11.4	85.0	87.7	94.5
2-20	13.8	83.5	86.2	92.8
2-25	16.3	82.0	84.0	90.8
2-30	18.7	80.6	82.5	89.5
2-40	21.9	78.8	80.4	87.7

Figure 2: Parsing results for the tree-bank grammar

1. precision: the percentage of non-terminal bracketings in the map parse that also appeared in the tree-bank parse,
2. recall: the percentage of non-empty non-terminal bracketings from the tree bank that also appeared in the map parse, and
3. accuracy: the percentage of bracketings from the map parse that did not cross over the bracketings in the tree-bank parse.

The results obtained are shown in Figure 2.

At about eleven thousand rules, our grammar is rather large. We also ran some tests using only the subset of rules that occurred more than once. As noted earlier, this reduced the number of rules in the grammar to 3943. Interestingly, this reduction had almost no impact on the parsing results, as shown in Figure 3, which gives first the results for the full grammar followed by the results with the 4000-rule subset. The differences are small.

3 Discussion

To put the experimental results into perspective it is useful to compare them to previous results on Wall Street Journal data. Figure 4 compares the accuracy figures for our tree-bank grammar with those of three earlier grammars/parsers that also used Wall Street Journal text for testing purposes. We compare only accuracy figures because the earlier work did not give precision and recall figures.

Sentence Lengths	Grammar Size	Precision	Recall	Accuracy
2-16	Full	85.0	87.7	94.5
	Reduced	84.3	87.9	94.5
2-25	Full	82.0	84.0	90.8
	Reduced	81.6	84.7	91.0
2-40	Full	78.8	80.4	87.7
	Reduced	78.2	80.7	87.6

Figure 3: Parsing results for a reduced tree-bank grammar

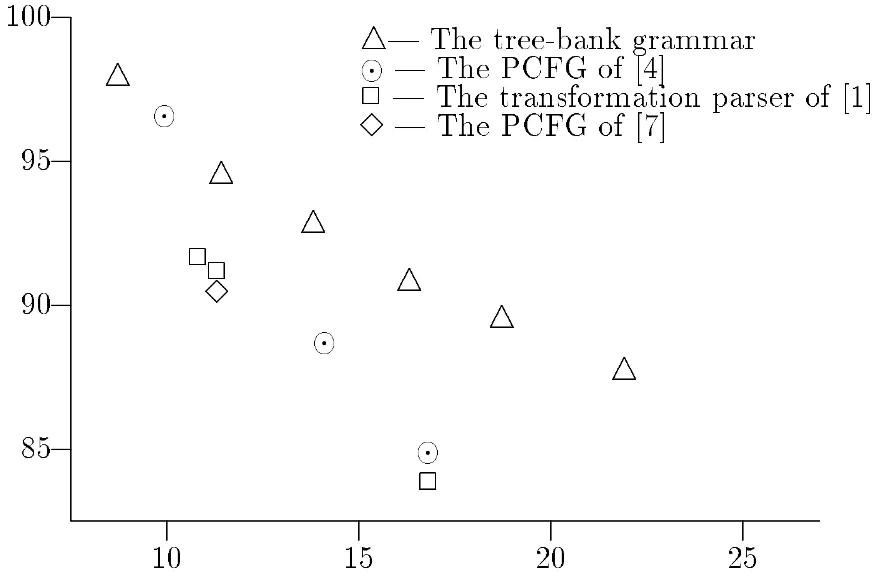

Figure 4: Accuracy vs. average sentence length for several parsers

It seems clear that the tree-bank grammar outperforms the others, particularly when the average sentence length gets higher — i.e., when longer sentences are allowed into the testing corpus, The only data point that matches our current results is one for one of our earlier grammars [4], and then only for very short sentences.

This is not to say, however, that there are no better grammars/parsers. Magerman [5] reports precision and accuracy figures of 86% for WSJ sentences of length 40 and less. The difference is that Magerman's parser uses statistics based upon the actual words of the sentence, while ours, and the others shown in Figure 4 use only the tags of the words. Obviously this shows the importance of including lexical information, a point to which we return in the conclusion.

Next we turn to the discrepancy between our results and the prevailing expectations. Roughly speaking, one can identify five reasons why a parser does not identify the "correct" parser for a sentence.

1. the necessary rules are not in the grammar
2. the rules are there, but their probabilities are incorrect
3. the probabilities are correct, but the tag sequence by itself does not provide sufficient information to select the correct parse
4. the information is sufficient, but because the parser could not consider all of the possible parses, it did not find the correct parse,
5. it found the correct parse, but the the tree-bank "gold standard" was wrong (or the correct parse is simply not clear).

Of these (3) and (5) are important but not relevant to the current discussion. Of the rest, we believe that (1) is a major component of the low expectations for tree-bank grammars. Certainly it was our major concern. Penn-style trees tend to be be rather shallow, and the 40-odd parts of speech allow many possible combinations. For example, consider the NP "the $200 hat", which has the tag sequence dt $ cd nn. Our tree-bank grammar does not have the corresponding NP rule and thus could not assign a correct parse to a sentence that contained this NP. For these reasons we gave some thought to how new rules might be introduced and assigned non-zero probability. In the event, however, no such complications proved necessary. First, our grammar

Sentence Lengths		Precision	Recall	Accuracy
2-16	Test Data	85.0	87.7	94.5
	Training Data	88.3	89.5	94.0
2-25	Test Data	82.0	84.0	90.8
	Training Data	86.9	85.1	91.5
2-40	Test Data	78.8	80.4	87.7
	Training Data	83.3	81.8	88.4

Figure 5: Parsing results for the tree-bank grammar

was able to parse all of the test sentences. Second, it is not too hard to show that coverage is not a first-order problem

In retrospect, our concerns about coverage were not well thought out because of a second property of our tree-bank grammar, its extreme overgeneration. In particular, the following fact is true:

> Let x be the set of the tree-bank parts of speech minus the following parts of speech: forward and backward single quote mark (neither of which occurred in our corpus), sym (symbol), uh (interjection), • (final punctuation), and). Any string in x^* is a legitimate prefix to a sentence in the language of our tree-bank-grammar, and furthermore, any non-terminal may start immediately following x^*.

In other words, our grammar effectively rules out no strings at all, and every possible part of speech could start at almost any point in the sentence. The proof of this fact is by induction on the length of the string and is straightforward, but tedious.

Of course, that our grammar comes up with *some* parse for a sentence does not mean that it is immune to missing rules. However, we can show that possible missing rules are not a first-order problem for our grammar by applying it to sentences from the training corpus. This gives an upper bound on the performance we can expect when we have all of the necessary rules (and the correct probabilities). The results are given in Figure 5. Looking at the data for all sentences of length less than or equal to 40 we see that having all of the necessary rules makes little difference, particularly in recall and accuracy (though why precision is affected so much more is an interesting question).

We noted earlier that the tree-bank grammar not only overgenerates, but places almost no constraints on what part of speech might occur at any point in the sentence. This fact suggests a second reason for the bad reputation of such grammars — they can be hard on parsers. We noticed this in preliminary testing on the training corpus when a significant number of sentences were not being parsed — this despite the fact that our standard parser used a simple best-first mechanism. That is, the parser chooses the next constituent to work on by picking the one with the highest "figure of merit." In our case this is the geometric mean of the inside probability of the constituent.

Fortunately we have been also working on improved best-first chart parsing, and we were able to use some new techniques on our tree-bank grammar. We achieved the above performance using the following figure of merit for a constituent $N^i_{j,k}$, that is, a constituent headed by the ith non-terminal, which covers the terms (parts of speech) $t_j \ldots t_{k-1}$

$$p(N^i_{j,k} \mid t_{0,n}) \approx \frac{p(N^i \mid t_{j-1}) p(t_{j,k} \mid N^i) p(t_k \mid N^i)}{p(t_{j,k+1})} \tag{2}$$

Here $p(t_{j,k+1})$ is the probability of the sequence of terms $t_j \ldots t_k$ and is estimated by a tri-tag model $p(t_{j,k} \mid N^i)$ is the inside probability of $N^i_{j,k}$ and is computed in the normal fashion (see, e.g., [3]) and $p(N^i \mid t_{j-1})$ and $p(t_k \mid N^i)$ are estimated by gathering statistics from the training corpus.

It is not our purpose here to discuss the benefits of this particular figure of merit (but see [2]). Rather we simply want to note the difficulty of obtaining parses, and particularly, high-probability parses, in the face of extreme ambiguity. It is possible that some of the negative "common wisdom" with respect to tree-bank grammars stems from this source.

4 Right-Branching Corrections

Earlier we noted that we made one modification to our grammar/parser other than the purely efficiency-related ones discussed in the last section.

This modification stemmed from our long standing belief that our context-free parsers seemed, at least from our non-systematic observations, to tend more toward center-embedding constructions than is warranted in English. It is generally recognized that English is a right-branching language. For example, consider the following right-branching bracketing of the sentence "The cat licked several pans."

((The (cat (licked (several pans)))) .)

While the bracketing starting with "cat" is quite absurd, note how many of the bracketings are correct. This tendency has been exploited by Brill's [1] "transformational parser," which starts with the right-branching analysis of the sentence and then tries to improve on it.

On the other hand, context-free grammars have no preference for right-branching structures. Indeed, those familiar with the theory of computation will recognize that the language $\mathsf{a}^n\mathsf{b}^n$, the canonical center embedded language, is also the canonical context-free language. It seemed to us that a tree-bank grammar, because of the close connection between the "gold-standard" correct parses and the grammar itself, offered an opportunity to test this hypothesis.

As a starting point in our analysis, note that a right-branching parse of a sentence has all of the closing parentheses just prior to the final punctuation. We call constituents that end just prior to the final punctuation "ending constituents," and the rest "middle constituents." We suspect that our grammar has a smaller propensity to create ending constituents that is warranted by correct parses. If this is the case, we want to redress this bias.

The uncorrected probabilities that would lead to this bias are those assigned by the normal PCFG rules for assigning probabilities:

$$p(\pi) = \prod_{c \in \pi} p(\mathrm{rule}(c)) \tag{3}$$

Here π is a parse of the tag sequence, c is a non-terminal constituent of this parse, and rule(c) is the grammar rule used to expand this constituent in the parse. Assume that we observe our uncorrected parser making x percent of the constituents ending constituents whereas the correct parses have y percent, and that conversely it makes u percent of the constituents middle constituents whereas the correct parse found v percent.

We hypothesized that one would find $y > x$ and $u > v$. Furthermore it seems reasonable to "correct" the probabilities to account for this bias by (in the case of an ending constituent) dividing out by x to get an "uninfluenced" version and then multiplying by the correct probability y to make the influence match the reality (and similarly for middle constituents). This gives the following equation for the probability of a parse:

$$p(\pi) = \prod_{c \in \pi} p(\mathrm{rule}(c)) \cdot \left\{ \begin{array}{ll} y/x & \text{if c is ending} \\ v/u & \text{otherwise} \end{array} \right\} \tag{4}$$

Note that the deviation of this equation from the standard context-free case is heuristic in nature: it derives not from any underlying principles, but rather from our intuition. The best way to understand it is simply to note that if the grammar tends to underestimate the number of ending constituents and overestimate middle constituents, the above equation will multiply the former by y/x, a number greater than one, and the latter by v/u, a number less than one.

Furthermore, if we assume that on the average the total number of constituents are the same in for both the map-parse and the tree-bank parse (a pretty good assumption), and that y and u (the numbers for the correct parses) are collected from the training data, we need only collect one further number, which we have chosen to be the ending-factor $\mathcal{E} = y/x$.

To test our theory, we estimated $\mathcal{E}$ from some held-out data. It came out 1.2 (thus confirming, at least for this test sample, our hypothesis that the map-parses would underestimate the number of ending constituents). We modified our parse probability equation to correspond to Equation 4. The data we reported earlier is the result. If we do not use this correction we get the "No correction" data shown here:

	Precision	Recall	Accuracy
With correction	78.8	80.4	87.7
No correction	77.1	78.1	86.0
Difference	1.7	2.3	1.7

The data is for sentences of lengths 2-40. The differences are not huge, but they are significant — both in the statistical sense and in the sense that they make up a large portion of the improvement over the other grammars in Figure 4. Furthermore, the modification required to the parsing algorithm is trivial (a few lines of code), so the improvement comes nearly for free.

It is also interesting to speculate whether such a bias would work for grammars other than tree-bank grammars. On the one hand, our basic arguments that might lead one to suspect a problem with context-free grammars are not peculiar to tree-bank grammars. On the other, mechanisms like counting the percentage of ending constituents assume that the parser's grammar and that of the gold standard are quite similar, as otherwise one is comparing incomparables. Some experimentation might be warranted.

5 Conclusion

We have presented evidence that tree-bank grammars perform much better than one might at first expect and, in fact, seem to outperform other non-word-based grammar/parsers. We then suggested two possible reasons for the mistaken impressions of tree-bank grammars' inadequacies. The first of these is the fear that missing grammar rules will prove fatal. Here we observed that our grammar was able to parse all of our test data, and by reparsing the training data showed that the real limits of the parsers performance must lie elsewhere (probably in the lack of information provided by the tags alone). The second possible reason behind the mistaken current wisdom is the high level of ambiguity of Penn tree-bank grammars. The ambiguity makes it hard to obtain a parse because the number of possible partial constituents is so high, and similarly makes it hard to find the best parse even should one parse be found. Here we simply pointed to some work we have done on best-first parsing and suggested that this may have tamed this particular problem. Lastly we discussed a correction made to the probabilities of the parses to encourage more right-branching structures and showed how this led to a small but significant improvement in our results.

However, because of the informational poverty of tag sequences, we recognize that context-free parsing based only upon tags is not sufficient for high precision, recall, and accuracy. Rather, we need to include lexical items in the information mix upon which we base our statistics. Certainly the 86% precision and recall achieved by Magerman [5] supports this contention. On the other hand, [5] abjures grammars altogether, preferring a more complicated (or at least, more unusual) mechanism that, in effect, makes up the rules as it goes along. We would suggest that the present work, with its accuracy and recall of about 80%, indicates that the new grammatical mechanism is not the important thing in those results. That is to say, we estimate that introducing word-based statistics on top of our tree-bank grammar should be able to make up the 6% gap. Showing this is the next step of our research.

References

1. Brill, E. *Automatic grammar induction and parsing free text: a transformation-based approach.* In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics.* 1993, 259–265.
2. Caraballo, S. and Charniak, E. Figures of Merit for Best-First Probabilistic Chart Parsing. Brown Univeristy Technical Report, forthcoming.
3. Charniak, E. *Statistical Language Learning.* MIT Press, Cambridge, 1993.
4. Charniak, E. Parsing with context-free grammars and word statistics. Department of Computer Science, Brown University, Technical Report CS-95-28, 1995.
5. Magerman, D. M. *Statistical decision-tree models for parsing.* In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics.* 1995, 276–283.
6. Marcus, M. P., Santorini, B. and Marcinkiewicz, M. A. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics 19* (1993), 313–330.
7. Pereira, F. and Schabes, Y. *Inside-outside reestimation from partially bracketed corpora.* In *27th Annual Meeting of the Association for Computaitonal Linguistics.* ACL, 1992, 128–135.