Tree-bank Grammars

Eugene Charniak
Department of Computer Science, Brown University
Providence RI 02912-1910
ec@cs.brown.edu

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 outperforms all other non-word-based statistical parsers/grammars on this corpus. That is, it outperforms parsers that consider the input as a string of tags and ignore the actual words of the corpus.

Introduction

Recent years have seen many natural-language processing (NLP) projects aimed at producing grammars/parsers capable of assigning reasonable syntactic structure to a broad swath of English. Naturally, judging the creations of your parser requires a "gold standard," and NLP researchers have been fortunate to have several corpora of hand-parsed sentences for this purpose, of which the so-called "Penn tree-bank" [7] is perhaps the best known. It is also the corpus used in this study. (In particular, we used the Wall Street Journal portion of the tree bank which consists of about one million words of hand-parsed sentences.)

However, when a convenient standard exists, the research program subtly shifts: the goal is no longer to create any-old parser, but rather to create one that mimics the Penn tree-bank parses. Fortunately, while there is no firm NLP consensus on the exact form a syntactic parse should take, the Penn trees are reasonably standard and disagreements are usually about less common, or more detailed, features. Thus the attempt to find Penn-style trees seems a reasonable one, and this paper is a contribution to this effort.

Of those using tree banks as a starting point, a significant sub-community is interested in using them to support supervised learning schemes so that the grammar/parser can be created with minimal human intervention [1,2,5,6,8]. The benefits of this approach are twofold: learning obviates the need for grammar writers, and such grammars may well have better coverage (assign parses to more sentences) than the hand-tooled variety. At any rate, this is the game we have chosen.

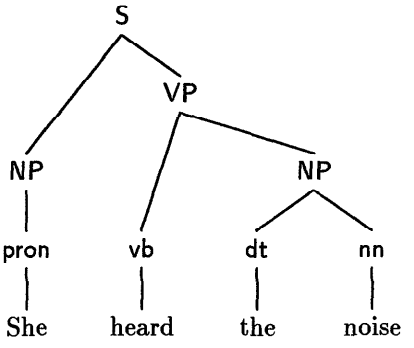

Figure 1: A simple parsed entry in a tree-bank

Now the simplest way to "learn" a context-free grammar from a tree-bank is to read the grammar off the parsed sentences. That is, we can read the following rules off the parsed sentence in Figure 1

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

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

It is common wisdom that tree-bank grammars do not work well. We 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 the common expectations.

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. On 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.

The Experiment

We used as our tree bank the Penn parsed Wall Street Journal corpus, release 2.[1] We divided the sentences into two separate corpora, about 100,000 words for testing and about 1,000,000 words for training. We ignored all sentences in the testing data of length greater than 40 because of processing-time considerations; at any rate, the actual number of such sentences is quite low, as the overall average sentence length is about 22 words and punctuation. Of the 100,000 words of testing data, half were used for preliminary testing and the other half for "official" testing — the results reported here.

With the exception of the right-bracketing bias 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 were ignored. To create a probabilistic CFG, a PCFG, we assigned a probability to each rule by observing how often it was used in the training corpus. Let $| r |$ be the number of times rule r occurred in the parsed training corpus and $\lambda(r)$ be the non-terminal that r expands. Then the probability assigned to r is

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

After training we test our parser/grammar on the test data. The input to the tester is the parsed sentence with each word assigned its (presumably) correct part of speech (or *tag*). Naturally the parse is ignored by the parser and only used to judge the parsers output. Also, our grammar does not use lexical information, but only the tags. Thus the actual words of the sentence are irrelevant as far as our parser is concerned; it only notices the tag sequence specified by the tree-bank. For example, the sentence in Figure 1 would be "pron vb dt nn."

We used as our set of non-terminals those specified in the tree-bank documentation, which is roughly the set specified in [7]. It was necessary to add a new start symbol, S1, as all the parses in our version of the tree bank have the following form:

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

Note the topmost unlabeled bracketing with the single S subconstituent, but no label of its own. We handled such cases by labeling this bracket S1.[2]

We use 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 (an aux in the "ing" form). 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 15953 rules of which only 6785 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:

1. precision: the percentage of all non-terminal bracketings appearing in map parses that also appear as a non-terminal bracketing in the corresponding tree-bank parse,
2. recall: the percentage of all non-empty non-terminal bracketings from the tree bank that also appeared as non-terminal bracketings in the map parse, and
3. accuracy: the percentage of all bracketings from the map parses that do not cross over the bracketings in the tree-bank parse.

The results obtained are shown in Figure 2.

Sentence Lengths	Average Length	Precision	Recall	Accuracy
2-12	8.0	91.5	89.1	96.9
2-16	11.5	89.6	87.1	95.0
2-20	13.9	87.3	84.9	92.9
2-25	16.3	85.5	83.3	91.2
2-30	18.8	83.6	81.6	89.7
2-40	22.0	82.0	80.0	88.0

Figure 2: Parsing results for the tree-bank grammar

At about sixteen thousand rules, our grammar is rather large. We also ran some tests using only the

[1] An earlier draft of this paper was based upon a preliminary version of this corpus. As this earlier version was about one-third the size and somewhat less "clean," this version of the paper sports (a) a larger tree-bank grammar (because of more training sentences), and (b) somewhat better results (primarily because of the cleaner test data).

[2] One interesting question is whether this outermost bracketing should be counted when evaluating the precision and recall of the grammar against the tree-bank. We have not counted it in this paper. Note that this bracketing always encompasses the entire sentence, so it is impossible to get wrong. Including it would improve our results by about 1%, i.e., precision would increase from the current 82% to about 83%.

Sentence Lengths	Grammar Size	Precision	Recall	Accuracy
2-16	Full	89.6	87.1	95.0
	Reduced	89.3	87.2	94.9
2-25	Full	85.5	83.3	91.2
	Reduced	85.1	83.3	91.1
2-40	Full	82.0	80.0	88.0
	Reduced	81.6	80.0	87.8

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

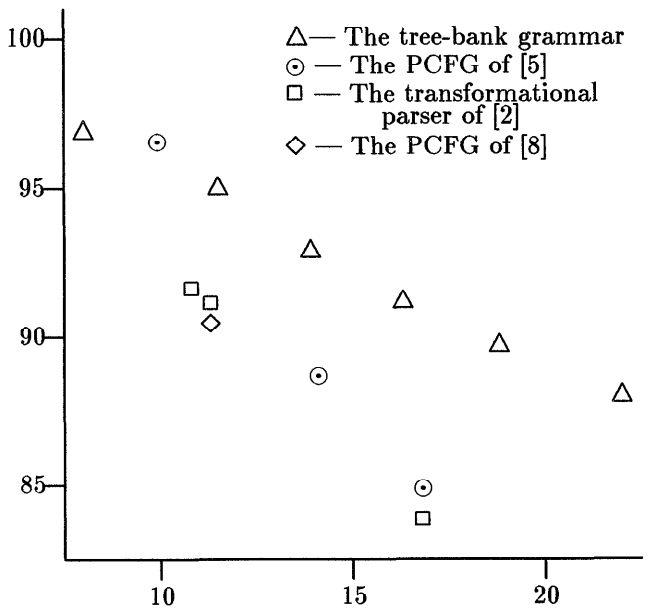

Figure 4: Accuracy vs. average sentence length for several parsers

subset of rules that occurred more than once. As noted earlier, this reduced the number of rules in the grammar to 6785. Interestingly, this reduction had almost no impact on the parsing results. Figure 3 gives first the results for the full grammar followed by the results with the 6785-rule subset; the differences are small.

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.

It seems clear that the tree-bank grammar is more accurate than the others, particularly when the average sentence length increases — i.e., when longer sentences are allowed into the testing corpus, The only data point that matches our current results is one for an earlier grammars of ours [5], and that only for very short sentences.

This is not to say, however, that there are no better grammars/parsers. Magerman [6] 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. We believe this shows the importance of including lexical information, a point to which we return below.

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" parse for a sentence:

1. the necessary rules are not in the grammar,
2. the rules are there, but the 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 to $400 price range", which has the tag sequence dt $ cd to $ cd nn nn. Our tree-bank grammar does not have the corresponding NP rule (or any reasonable combination of rules as far as we can tell) and thus could not assign a correct parse to a sentence that contained this NP. For this reason we gave some thought to how new rules might be introduced and assigned non-zero probability. Indeed, we started on this work becuase we believed we had a interesting way to do this. In the event, however, no such complications proved necessary. First, our 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

Sentence Lengths	Data Used	Precision	Recall	Accuracy
2-16	Testing	89.6	87.1	95.0
	Training	90.7	88.6	95.4
2-25	Testing	85.5	83.3	91.2
	Training	86.7	84.0	91.6
2-40	Testing	82.0	80.0	88.0
	Training	83.7	81.1	88.6

Figure 5: Parsing results for the tree-bank grammar

backward single quote mark (neither of which occurred in our corpus), sym (symbol), uh (interjection), • (final punctuation), and). Any string in x* (where "*" is the normal Kleene star operator) 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 can 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.[3]

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.

We noted earlier that the tree-bank grammar not only overgenerates, but also places almost no constraints on what part of speech may 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 when, in preliminary testing on the training corpus, a significant number of sentences were not 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 were able to use some new techniques on our tree-bank grammar. We achieved the performance indicated in Figure 2 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) $t_j \ldots t_{k-1}$

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

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., [4]), 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 advantages of this particular figure of merit (but see [3]). Rather, we simply want to note the difficulty of obtaining parses, not to mention high-probability parses, in the face of extreme ambiguity. It is possible that some of the negative "common wisdom" about tree-bank grammars stems from this source.

Right-branching Bias

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 arose 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 [2] "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 $a^n b^n$, 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

[3] So tedious that after proving this fact for the tree-bank grammar used in the first draft of this paper, we could not muster the enthusiasm necessary to confirm it for the current grammar. However, since the new grammar is larger than that in the earlier draft, the above theorem or a similar one will surely hold.

propensity to create ending constituents than is warranted by correct parses. If this is the case, we want to bias our probabilities to create more ending constituents and fewer middle ones.

The "unbiased" probabilities are those assigned by the normal PCFG rules for assigning probabilities:

$$p(\pi) = \prod_{c \in \pi} p(\text{rule}(c)) \quad (3)$$

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 our unbiased parser makes 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 has v percent.

We hypothesized that $y > x$ and $u > v$. Furthermore it seems reasonable to bias the probabilities to account for the underproduction of ending constituents by 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:

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

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 is the same in 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 collect only one further number, which we have chosen as 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. Not using this bias yields the "Unbiased" data shown here:

	Precision	Recall	Accuracy
With bias	82.0	80.0	88.0
Unbiased	79.6	77.3	85.4
Difference	2.4	2.7	2.6

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 is nearly free.

It is also interesting to speculate whether such a bias would work for grammars other than tree-bank grammars. On the one hand, the arguments that 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.

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 grammars/parsers. We then suggested two possible reasons for the mistaken impression 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 have 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. Last, we discussed a modification 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. We also noted that the improvement came at essentially no cost in program complexity.

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. It seems clear to us that 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 [6] supports this contention. On the other hand, [6] 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 81%, indicates that the new grammatical mechanism is not the important thing in those results. That is to say, we estimate that introducing word-based statis-

tics on top of our tree-bank grammar should be able to make up the 5% gap. Showing this is the next step of our research.

Acknowledgements

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

References

1. BOD, R. *Using an annotated language corpus as a virtual stochastic grammar.* In *Proceedings of the Eleventh National Conference on Artificial Intelligence.* AAAI Press/MIT Press, Menlo Park, 1993, 778–783.
2. 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.
3. CARABALLO, S. AND CHARNIAK, E. Figures of merit for best-first probabilistic chart parsing. Department of Computer Science, Brown University, Technical Report, forthcoming.
4. CHARNIAK, E. *Statistical Language Learning.* MIT Press, Cambridge, 1993.
5. CHARNIAK, E. Parsing with context-free grammars and word statistics. Department of Computer Science, Brown University, Technical Report CS-95-28, 1995.
6. 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.
7. 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.
8. 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.