Facilitating Treebank Annotation Using a Statistical Parser

Fu-Dong Chiou, David Chiang, and Martha Palmer
Dept of Computer and Information Science
University of Pennsylvania
200 S 33rd Street, Philadelphia, PA 19104-6389
{chioufd,dchiang,mpalmer}@linc.cis.upenn.edu

1. INTRODUCTION

Corpora of phrase-structure-annotated text, or treebanks, are useful for supervised training of statistical models for natural language processing, as well as for corpus linguistics. Their primary drawback, however, is that they are very time-consuming to produce. To alleviate this problem, the standard approach is to make two passes over the text: first, parse the text automatically, then correct the parser output by hand.

In this paper we explore three questions:

- How much does an automatic first pass speed up annotation?
- Does this automatic first pass affect the reliability of the final product?
- What kind of parser is best suited for such an automatic first pass?

We investigate these questions by an experiment to augment the Penn Chinese Treebank [15] using a statistical parser developed by Chiang [3] for English. This experiment differs from previous efforts in two ways: first, we quantify the increase in annotation speed provided by the automatic first pass (70–100%); second, we use a parser developed on one language to augment a corpus in an unrelated language.

2. THE PARSER

The parsing model described by Chiang [3] is based on stochastic TAG [13, 14]. In this model a parse tree is built up out of tree fragments (called elementary trees), each of which contains exactly one lexical item (its anchor).

In the variant of TAG used here, there are three kinds of elementary trees: initial, (predicative) auxiliary, and modifier, and three corresponding composition operations: substitution, adjunction, and sister-adjunction. Figure 1 illustrates all three of these operations. The first two come from standard TAG [8]; the third is borrowed from D-tree grammar [11].

In a stochastic TAG derivation, each elementary tree is generated with a certain probability which depends on the elementary tree itself as well as the node it gets attached to. Since every tree is

3. METHODOLOGY

For the present experiment the parsing model was trained on the entire treebank (99,720 words). We then prepared a new set of 20,202 segmented, POS-tagged words of Xinhua newswire text, which was blindly divided into 3 sets of equal size (±10 words).

Each set was then annotated in two or three passes, as summarized by the following table:

<table>
<thead>
<tr>
<th>Set</th>
<th>Pass 1</th>
<th>Pass 2</th>
<th>Pass 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>—</td>
<td>Annotator A</td>
<td>Annotators A&amp;B</td>
</tr>
<tr>
<td>2</td>
<td>parser</td>
<td>Annotator A</td>
<td>Annotators A&amp;B</td>
</tr>
<tr>
<td>3</td>
<td>revised parser</td>
<td>Annotator A</td>
<td>Annotators A&amp;B</td>
</tr>
</tbody>
</table>

Here “Annotators A&B” means that Annotator B checked the work of Annotator A, then for each point of disagreement, both annotators worked together to arrive at a consensus structure. “Parser” is Chiang’s parser, adapted to parse Chinese text as described by Bikel and Chiang [1].

“Revised parser” is the same parser with additional modifications suggested by Annotator A after correcting Set 2. These revisions primarily resulted from a difference between the artificial evaluation metric used by Bikel and Chiang [1] and this real-world task. The metric used earlier, following common practice, did not take punctuation or empty elements into account, whereas the present task ideally requires that they be present and correctly placed. Thus following changes were made:

- The parser was originally trained on data with the punctuation marks moved, and did not bother to move the punctuation marks back. For Set 3 we simply removed the preprocessing phase which moved the punctuation marks.

- Similarly, the parser was trained on data which had all empty elements removed. In this case we simply applied a rule-based postprocessor which inserted null relative pronouns.

- Finally, the parser often produced an NP (or VP) which dominated only a single NP (respectively, VP), whereas such a
Facilitating Treebank Annotation Using a Statistical Parser

Department of Computer and Information Sciences, University of Pennsylvania, Philadelphia, PA, 19104

Approved for public release; distribution unlimited
structure is not specified by the bracketing guidelines. Therefore we applied another rule-based postprocessor to remove these nodes. (This modification would have helped the original evaluation as well.)

In short, none of the modifications required major changes to the parser, but they did improve annotation speed significantly, as we will see below.

4. RESULTS

The annotation times and rates for Pass 2 are as follows:

<table>
<thead>
<tr>
<th>Set</th>
<th>Pass 1 Time (Pass 2)</th>
<th>Rate (Pass 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>—</td>
<td>28:01</td>
</tr>
<tr>
<td>2</td>
<td>parser</td>
<td>16:21</td>
</tr>
<tr>
<td>3</td>
<td>revised parser</td>
<td>14:06</td>
</tr>
</tbody>
</table>

The rate increase for Set 2 over Set 1 was about 70%; for Set 3 over Set 1, about double. Thus the time saved by the use of an automatic first pass is substantial.

Assessing the reliability of the final product is somewhat trickier. The third column reports the accuracy of Pass 1 (the parser) using the results of Pass 2 (Annotator A) as a gold standard. The fourth column reports the accuracy of Pass 2 (Annotators A&B) using the results of Pass 3 (Annotators A&B) as a gold standard.

We note several points:

- The parser revisions significantly improved the accuracy of the parser with respect to the present metric (which is sensitive to punctuation and empty elements). On Set 2 the revised parser obtained 78.98/77.39% labeled precision/recall, an error reduction of about 9%.
- Not surprisingly, errors due to large-scale structural ambiguities were the most time-consuming to correct by hand. To take an extreme example, one parse produced by the parser is shown in Figure 2. It often matches the correct parse (shown in Figure 3) at the lowest levels but the large-scale errors require the annotator to make many corrections.

5. DISCUSSION

In summary, although Chiang’s parser was not specifically designed for Chinese, and trained on a moderate amount of data (less than 100,000 words), the parses it provided were reliable enough that the annotation rate was effectively doubled.

Now we turn to our third question: what kind of parser is most suitable for an automatic first pass? Marcus et al. [10] describe the use of the deterministic parser Fidditch [6] as an automatic first pass for the Penn (English) Treebank. They cite two features of this parser as strengths:

1. It only produces a single parse per sentence, so that the annotator does not have to search through many parses.
2. It produces reliable partial parses, and leaves uncertain structures unspecified.

The Penn-Helsinki Parsed Corpus of Middle English was constructed using a statistical parser developed by Collins [4] as an automatic first pass. This parser, as well as Chiang’s, retains the first advantage but not the second. However, we suggest two ways a statistical parser might be used to speed annotation further:

1. It only produces a single parse per sentence, so that the annotator does not have to search through many parses.
2. It produces reliable partial parses, and leaves uncertain structures unspecified.

The Penn-Helsinki Parsed Corpus of Middle English was constructed using a statistical parser developed by Collins [4] as an automatic first pass. This parser, as well as Chiang’s, retains the first advantage but not the second. However, we suggest two ways a statistical parser might be used to speed annotation further:

First, the parser can be made more useful to the annotator. A statistical parser typically produces a single parse, but can also (with little additional computation) produce multiple parses. Ratnaparkhi [12] has found that choosing (by oracle) the best parse out of the 20 highest-ranked parses boosts labeled recall and precision.
These businesses also transfer and spread the intellectual property rights of 36,000 technologies to other businesses and organizations, creating an income of 4.43 billion RMB.
from about 87% to about 93%. This suggests that if the annotator had access to several of the highest-ranked parses, he or she could save time by choosing the parse with the best gross structure and making small-scale corrections.

Would such a change defeat the first advantage above by forcing the annotator to search through multiple parses? No, because the parses produced by a statistical parser are ranked. The additional lower-ranked parses can only be of benefit to the annotator. Indeed, because the chart contains information about the certainty of each subparse, a statistical parser might regain the second advantage as well, provided this information can be suitably presented.

Second, the annotator can be made more useful to the parser by means of active learning or sample selection [5, 7]. (We are assuming now that the parser and annotator will take turns in a train-parse-correct cycle, as opposed to a simple two-pass scheme.) The idea behind sample selection is that some sentences are more informative for training a statistical model than others; therefore, if we have some way of automatically guessing which sentences are more informative, these sentences are the ones we should hand-correct first. Thus the parser’s accuracy will increase more quickly, potentially requiring the annotator to make fewer corrections overall.

6. ACKNOWLEDGMENTS

We would like to thank Fei Xia, Mitch Marcus, Aravind Joshi, Mary Ellen Okurowski and John Kovarik for their helpful comments on the design of the evaluation, Beth Randall for her post-processing and error-checking code, and Nianwen Xue for serving as “Annotator B.” This research was funded by DARPA N66001-00-1-8915, DOD MDA904-97-C-0307, and NSF SBR-89-20230-15.

7. REFERENCES


