Anna Rafferty
TongKe Xue

**Increasing Accuracy While Maintaining Minimal Grammars in CKY Parsing**

Significant work in both lexicalized and unlexicalized parsing has been done in the past ten years. F1 measures of accuracy of over 90% have been achieved (Bikel, 2005), and linguistic notions of lexical dependencies and using head words have been harnessed to create significant improvements in probabilistic CFG (PCFG) parsers (Bikel 2005; Collins 1996, 1997, 1999; Klein and Manning 2003). Klein and Manning (2003) note, however, that many of the techniques for improving lexicalized parsing create relatively little gain while creating more complex algorithms. While Collins (1999) parser is extremely useful is F1 is of the utmost importance, Klein and Manning's (2003) parser achieves F1 within 5% of that parser without invoking lexicalization.

For our project, we attempted to increase the precision and recall (F1) of the CKY parser built for project four. Our goal was to significantly improve F1 while keep the parser unlexicalized and maintaining a relatively small number of non-terminals. Although such a parser might not be as accurate as lexicalized models or unlexicalized models with larger grammars, we wished to show that acceptable F1 scores can be attained using minimal grammars and no lexicalization; we hypothesized that an F1 score within 1% of the best unlexicalized parser we found in the literature (Klein and Manning, 2003), with an F1 of 86.36%, was achievable. The minimal grammar allows for fast parsing; given a fully optimized parser, this grammar might be used for extremely quick trials that approximate the results of larger grammars that produce more accurate parsing or for pre-processing purposes.

In increasing the F1 of our parser, we modeled our changes closely on those described in "Accurate Unlexicalized Parsing" (2003) by Dan Klein and Chris Manning. This paper provided the clearest suggestions for improving unlexicalized parsing and allowed us to explore linguistic patterns that are useful in parsing. All of our improvements were made through annotating the grammar in various ways to reflect external areas of the parse tree that might affect the current non-terminal's behavior and internal properties of this particular non-terminal or the structure below it.

Our baseline parser was that created for project four. This parser included second-order vertical Markovization and first order horizontal Markovization. Additionally, it annotated preterminal nodes with the tag of their parents, just as vertical Markovization annotates other nodes with their parent tag. This parser produced a baseline F1 of 81.92%. Given previous results suggesting that increasing the order of vertical Markovization vastly increases the number of tags and is difficult to further annotate without creating problems of sparseness, we chose to include only second-order vertical Markovization in our improved parser rather than experiment further with this variable. We also limited the model to first order horizontal Markovization rather than second order as Klein and Manning (2003) did to limit the number of non-terminals, which increases significantly with higher order horizontal Markovization, and thus test out hypothesis that a relatively small number of non-terminals can be used to achieve high F1.

*Unary Annotations*

One feature we used to produce annotations was whether or not a node produced only one child. Nodes producing only one child are fairly rare and tend to occur within specific constructions, so marking nodes that produced only one child allowed recognition of these patterns. Including this trait increased our F1 to 84.05%, an absolute increase in F1 of 2.13%. This notation actually produced the largest increase of any individual annotation.

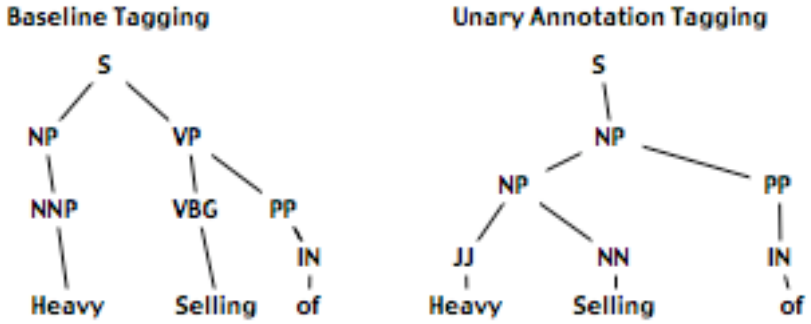


Figure 1

Consider Figure 1, an example of part of a sentence that was parsed incorrectly in the baseline but correctly when unary annotations were added. The baseline parser includes three instances of non-preterminals that have only one child; in contrast, the parser with unary annotations includes only one such instance. By learning when a unary child is likely to occur, the unary parser has lower probabilities for the unary children that are falsely created in the baseline parse, decreasing the probability that such a parse will be chosen. Many similar fixes occurred in other sentences in the test set.

As an extension to this unary tagging, we chose to annotate certain tags that were only children themselves. Based again on Klein and Manning (2003), we experimented with annotating determiners and adverbs that were only children. By annotating only these specific types of tags, we limited the number of non-terminals in our grammar, a concern for reasons of parsing speed, while increasing our F1 to 84.20%.

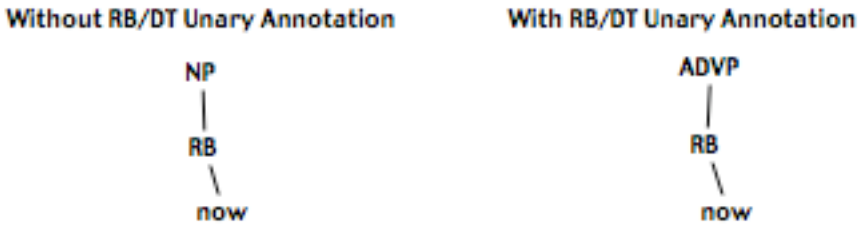


Figure 2

Although clearly this annotation was not as successful as the previous in improving parsing, it was good at fixing errors like that in Figure 2. In this case it is appropriate for RB to be a single child, but adding the annotation allows the parser to differentiate what sort of words appear as children of RBs that are only children. This differentiation creates the improvement in parsing by choosing ADVP as the parent of RB rather than NP.

*Head Tag Annotations*

We attempted a few annotations based on the tag of the head word of a phrase ("head tag"). Initially, we were going to identify head words based on hand-taggings. However, we felt this was in some ways working with more data than was contained within the model in general, and we wished to be able to use our parser to learn a grammar from corpora that did not include such hand-taggings. Additionally, we thought it would be interesting to examine how much of a gain in accuracy we could produce from head words that were only approximated. Thus, based on some rough linguistic analysis and the discussion of Collins' parser in Bikel (2004) that suggested head words tend to be the word that appears at the beginning of a phrase, we use the tag of this first word in the phrase as the proxy for the correct head tag. An additional limitation on our use of head tags was that we did not propagate them up the tree more than one level. For instance, we did not perform any annotations based on head tags for nodes located above the parents of preterminals in the tree. This limits the amount of information that can be gained from using head tags but also limits the growth of non-terminals. We felt such a limitation was appropriate given the approximate nature of the head words chosen for each phrase.

We attempted to improve F1 via annotating nodes with their head tag for noun phrases (NPs) and verb phrases (VPs). For noun phrases, we attempted to duplicate the results of Klein and Manning (2003) by annotating possessive noun phrases. This was done by adding an annotation to nodes for noun phrases with possessive "head tags" (approximated as described above). Unfortunately, this innovation failed to improve F1, probably due to the lack of propagation of the head tag to noun phrases higher in the tree. For verb phrases, we again began by using Klein and Manning's (2003) suggestion to annotate all verb phrases with their head tag except for those whose head is an infinitive verb, past participle verb, or present participle verb; these three categories of head tags were merged into one annotation. This implementation of annotating verb phrases was also not as successful as Klein and Manning's (2003) implementation, but it increased F1 to 84.38%.

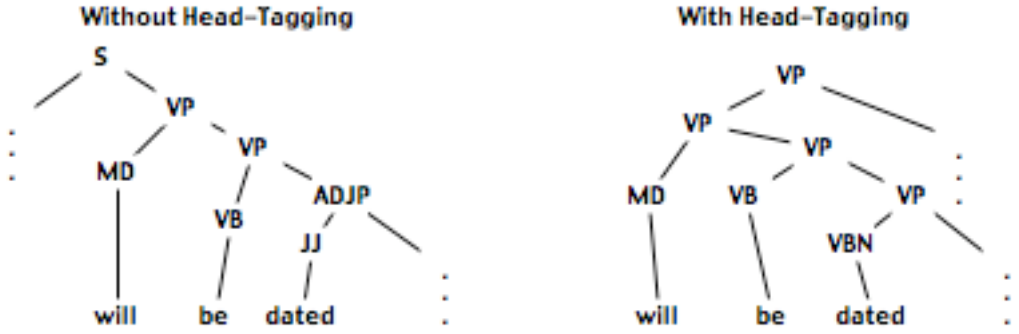


Figure 3

Figure 3 demonstrates the relatively large effects that head-tagging can have on verb phrase structure, despite the limited form of head-tagging that we used. The parser with head-tagging correctly parsed this portion of the sentence, in which every label was a type of verb, while the parser without tagging did not include as much structure as the correct parse and mislabeled one VP as an ADJP and "dated" as JJ rather than VBN. Although verb head-tagging did not produce a great deal of improvement in our parser, it

was particularly useful for situations such as Figure 3 that had areas of complicated local structure with many verb forms.

Given that neither of these approximations to head-tagging were particularly successful in our final model, we considered possible improvements to our method of approximation. Upward propagation seemed to be the most likely method of improving the results of head-tagging, although the upward propagation may result in erroneous head tags having greater influence in the tree than previously. However, upward propagation has the advantage of giving nodes much higher in the tree access to important linguistic information. Thus, we decided to implement the same annotations discussed previously with the addition of upward propagation.

Unfortunately, upward propagation still failed to increase F1 when used with noun phrases. This result suggests that at least for noun phrases, a more exact method of picking head words is required. Additionally, given that Klein and Manning (2003) state "First, possessive NPs have a very different distribution than other NPs – in particular, NP -> NP $\alpha$ rules are only used in the treebank when the leftmost child is possessive" (429), it may be that other features of our model already take these distinctions into account. Thus, distinguishing other types of NPs with unique distributions might be more successful using approximated head tags.

Upward propagation of head tags failed to improve F1 as much as the limited version of head tags for verb phrases (84.34% instead of 84.38%). Thus, neither of our attempts to increase F1 through head-tagging were successful. Several factors could contribute to this phenomena. First, our head-tagging approximation may be a poor approximation for head tags in general; in this case, it would make sense to incorporate a model that more accurately identifies head words. In this case, it might also be appropriate to simply acknowledge that identifying head words requires learning from a hand-tagged corpus and include such tags in the training trees. Another possible reason for the relatively poor results of our head-tagging is that the unary features and vertical Markovization (including tagging preterminals) may have implicitly recorded the features that create the benefit in head-tagging. While this seems unlikely given that Klein and Manning (2003) also had these features, it is important to note that the difference in F1 between Klein and Manning's parser and our best parser is less than the difference between the gain that they achieved through head tag annotations and the gain that we achieved through such annotations. Thus, it is possible that we incorporated some of the features in the head tags implicitly prior to this point in our model.

*Subdividing Part-of-Speech Tags*

Many of the annotations to our grammar involved splitting one part-of-speech label into several that more finely categorize specific types of words. This was conducted for tags CC (coordinating conjunctions), IN (subordinating conjunctions, conjunction complementizers, and prepositions other than "to"), various verb tags and SYM (symbols).

CC tags were split based on Klein and Manning's (2003) comment that the coordinating conjunctions "but" and "&" operate differently than other coordinating conjunctions. Thus, we annotated CC preterminals that had "but" or &" as their associated terminals. With these annotations, F1 actually fell from 84.38% to 84.21%. F1 can decrease when too many non-terminals are created without distinct patterns,

causing a sparsity of data during training. In an attempt to improve these results and gain information by splitting CC tags, we decided to annotate "and" as well, since like "but", this word is more common than the other coordinating conjunctions and our hypothesis was that it also operated somewhat differently linguistically. While this increase F1 to 84.26%, this result was still lower than without any splitting of the CC tag. Given that annotating the CC tag associated with "and" had increased F1, we decided to try only annotating those tags to determine if any overall increase in F1 could be gained through these simple splits in CC. Unfortunately, while this annotation caused the least harm in F1 of the CC splits, it still dropped F1 by 0.06%. Thus, our final parser did not perform any splits on the CC tag.

IN tags were split into six categories based on Klein and Manning (2003) as well as some linguistic research. Four of the categories were single word categories based on the word beneath the IN tag: "that", "of", "if", and "as". "That" is a complementizer which selects declarative clauses. Initially, we also included "for", another complementizer, in the category with "that", but this was less effective than using "that" as a single category. "For" was additionally tried as its own category, but this also failed to be as effective as not annotating it at all. "Of", "if", and "as" were similarly chosen to have their own categories based on the unique distributions these words have compared to other words tagged IN. The fifth category consisted of those words that are consistently as subordinate conjunctions. We included many words in this category based on grammar references suggesting possible words for inclusion. Finally, the sixth category included all other words labeled IN. These tags splits resulted in an F1 of 84.56%.

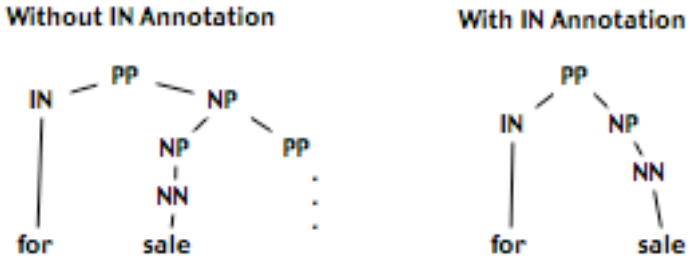


Figure 4

Figure 4 shows an instance that was incorrectly parsed prior to splitting the IN tag. Although it might seem odd that this split would fix a mistake associated with "for", which was actually not annotated through the IN split, this behavior makes sense when one considers that certain unique distributions have been removed from the IN tag. By removing atypical words from being categorized by IN, the distribution of most IN words can be learned by the parser.

The most successful part-of-speech tag annotation we tried was separating the auxiliary verbs "do", "have", and "be" from other verbs. We annotated any preterminal associated with a verb with an auxiliary base (*i.e.*, any form of "do", "have", or "be" was tagged, regardless of tense or other grammatical influences), and the three types of auxiliary verbs were annotated with separate tags. Auxiliary verbs combine with other verbs to express grammatical mood and often notions of time and tense. "Have", "do", "and "be" are the most common auxiliary verbs and are associated with specific grammatical structures. For instance, "have" is always followed by the past tense of the main verb (although the two words may be separated by an adverb or other intervening

word), a phenomena that can have specific consequences for how a sentence is parsed. Additionally, the nature of auxiliary words as occurring with a main verb in the sentence changes the typical parsing patterns. In most cases, two verbs will not appear in the same phrase, but in the case of auxiliary verbs, such a pattern is expected. Although there are more auxiliary verbs than these three, the others are tagged as modal verbs; since these three can serve several purposes, they were not originally tagged differently from other verbs.

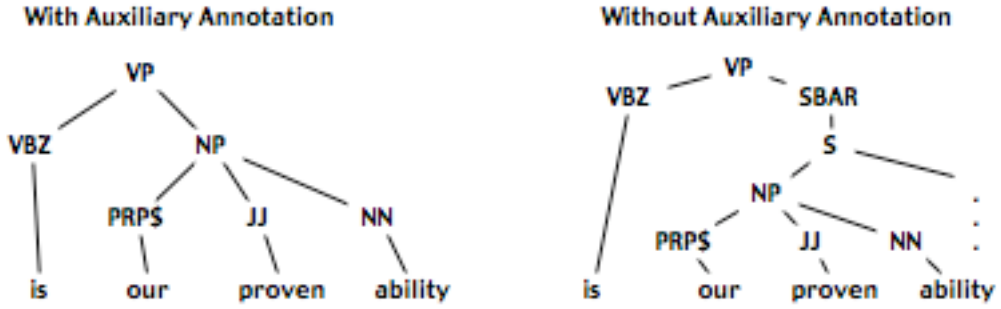


Figure 5

Adding auxiliary tags increased F1 to 85.51%, mainly by simplifying structures as shown in Figure 5. Like our other annotations, this annotation allowed us to separate the distributions, in this case the auxiliary verb distributions, with unique characteristics that differed from the distributions in which these tags were normally included.

Finally, our annotation to the SYM tag was the simple change of annotating the "%" symbol to reflect that it occurs only in particular contexts. This mirrors the contexts that "$" appears in, yet "$" has its own tag while "%" is lumped together with a multitude of other symbols. This change was intended to alleviate that asymmetry and was suggested by Klein and Manning (2003). Overall, it achieved an increase of 0.08% from the previous parser. This was our final annotation, creating a parser with F1 of 86.59%.

| | F1 | Number of Non-Terminals |
|---|---|---|
| Baseline * | 81.92% | 3403 |
| Unary - Parents w/ 1 Child * | 84.05% | 3907 |
| Unary - Only Child * | 84.20% | 3925 |
| Head-Tag - NPs | 84.20% | 3927 |
| Head-Tag – VPs * | 84.38% | 4248 |
| Head-Tag - NPs w/ U.P. | 84.20% | 3925 |
| Head-Tag - VPs w/ U.P. | 84.34% | 4266 |
| CC-Split | 84.21% | 4323 |
| CC-Split w/ "and" | 84.26% | 4407 |
| CC-Split "and" only | 84.32% | 4344 |
| IN Split * | 84.56% | 4427 |
| Auxiliaries * | 85.51% | 4581 |
| SYM-Split * | 85.59% | 4596 |
| Klein and Manning (2003) | 86.36% | 9255 |

Figure 6: Final results for parser in terms of F1 and number of non-terminals. Changes marked with * were included in the final parser. Those changes farther down in the list (with the exception of Klein and Manning (2003) which refers to a separate parser) occurred after those higher in the list; F1 is cumulative.

*Increase in Non-Terminals*

While we include all annotations in our final parser that increased F1, some of these annotations were significantly more helpful than others. Especially if this parser were to be used for pre-processing data or some other application in which both F1 and speed are important, one might want to leave some of the annotations out due to relatively little gain but relatively large increases in non-terminals. Given that one annotation may influence the amount of gain from including another, removing annotations should be done with caution and attention to the effects on F1. However, given these caveats, this analysis suggests that certain features are more important than others. For instance, both adding annotations to RB/DT tags that are only children and head-tagging VPs increases F1 by less than 0.2% (0.15% and 0.18%, respectively). The former annotation adds 18 new non-terminals while the latter adds 323 new non-terminals. Given that the complexity of the CKY algorithm is dependent on the number of non-terminals, the trade-off between F1 gained by head-tagging of VPs and extra time spent due to increased non-terminals may not be worthwhile for many applications. Additionally, our parser demonstrates that with fewer than four thousand non-terminals, an F1 of 84.2% can be achieved. Although this is not as successful as our final model, which achieved F1 of 85.59% with 4569 non-terminals, it demonstrates that high parsing accuracy can be achieved using a relatively minimal grammar. The simple annotations required to create such a grammar also require little processing time, so overall such a parser would make an excellent choice for pre-processing applications or applications in which speed is more important than absolute accuracy.

*Comparison of Current Parser to Klein and Manning's (2003) Results*

Several of our results differed significantly from Klein and Manning's (2003) results. Given that many of the annotations we made were based directly on their work, this result was surprising and led to a consideration of how the two grammars differed. Overall, the 2003 parser had an F1 of 0.8% higher than the best F1 we were able to achieve, but it also included almost 100% more non-terminals than our model. One of the reasons for this disparity was that their model included second-order horizontal Markovization. This type of Markovization includes more non-terminals because children are annotated with the sibling immediately to the left of them. Originally, we chose to omit this procedure in order to explore how well a parser with a small grammar could perform compared to other parsers with larger grammars. Based on the differences in the results of annotations that were made in both grammars, it seems that different annotations may be more helpful for grammars without sibling information than for grammars with (limited) sibling information. For instance, head-tagging was much more successful for Klein and Manning, and splitting the CC tag actually helped their parser while decreasing the F1 in our implementation. Thus, the magnitude of changes in F1 based on annotations not only differs between the two models, but the sign does as well. Although some of the disparities could be the result of combining the annotations in different orders, since we examined not gain in F1 from baseline but incremental gain from the previous annotated parser while Klein and Manning examined both measures of

F1, the number, magnitude, and direction off differences suggest that at least some of the differences are a result of differences in modeling and not testing.

The main reason we believe that these results occur is that annotations combine to provide information; the increase in F1 from adding two annotations may be greater than the sum of the increases created by adding each annotation separately. Certain constructions may require both sibling information and the annotation in order to be helpful. For example, specific head-tags might occur frequently in two types of constructions, but the sibling structures within those constructions might be distinct, leading to an increase in F1 only when sibling structure is identified. Including head-tags without sibling structure might fail to provide any information in this case but lead to a sparsity of training data, thus contributing to a decline in F1.

*Further Study*

Given the high F1 achieved by this parser and the suggestions above that the annotations appropriate for larger grammars may not be as appropriate for smaller grammars, further exploration of how the grammar may be annotated to produce better results without dramatic increases in size should be conducted. This exploration might include adding more accurate head tags as well as applying more linguistic information concerning different classes of words and phrases to produce more and better annotations. Further study concerning how a parser with a small grammar and thus faster running time might be used as a preprocessor for a lexicalized grammar or as a way to improve applications that need real time parsing would also be useful in order to take advantage of the strengths of this small grammar, unlexicalized parser.

Additionally, work to correct errors in the current version of the parser is necessary. This parser still makes a number of errors in the part-of-speech tags it assigns individual words. These errors tend to propagate, resulting in incorrect phrase labels that lead to incorrect parsing of phrases. Thus, a better method of tagging is needed. This parser also does a poor job in classifying types of clause (S, SINV, SBAR) and adding these clauses to the parse tree. Further research concerning the linguistic properties of each of these clauses is needed in order to determine what linguistic annotations might be added to fix these errors.

*Works Cited*

Bikel, Daniel M. Intricacies of Collins' Parsing Model. Computational Linguistics, forthcoming. http://citeseer.ist.psu.edu/626929.html

Collins, Michael. 1996. A new statistical parser based on bigram lexical dependencies. In Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics, pages 184–191.

Collins, Michael. 1997. Three generative, lexicalised models for statistical parsing. In Proceedings of ACL-EACL '97, pages 16–23.

Collins, Michael John. 1999. Head-Driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania.

Klein D., Manning C. (2003) Accurate Unlexicalized Parsing. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003), 423-430. http://citeseer.ist.psu.edu/klein03accurate.html