Adding Noun Phrase Structure to the Penn Treebank

David Vadas and **James R. Curran**
School of Information Technologies
University of Sydney
NSW 2006, Australia
{dvadas1, james}@it.usyd.edu.au

Abstract

The Penn Treebank does not annotate within base noun phrases (NPs), committing only to flat structures that ignore the complexity of English NPs. This means that tools trained on Treebank data cannot learn the correct internal structure of NPs.

This paper details the process of adding gold-standard bracketing within each noun phrase in the Penn Treebank. We then examine the consistency and reliability of our annotations. Finally, we use this resource to determine NP structure using several statistical approaches, thus demonstrating the utility of the corpus. This adds detail to the Penn Treebank that is necessary for many NLP applications.

1 Introduction

The Penn Treebank (Marcus et al., 1993) is perhaps the most influential resource in Natural Language Processing (NLP). It is used as a standard training and evaluation corpus in many syntactic analysis tasks, ranging from part of speech (POS) tagging and chunking, to full parsing.

Unfortunately, the Penn Treebank does not annotate the internal structure of base noun phrases, instead leaving them flat. This significantly simplified and sped up the manual annotation process.

Therefore, any system trained on Penn Treebank data will be unable to model the syntactic and semantic structure inside base-NPs.

The following NP is an example of the flat structure of base-NPs within the Penn Treebank:

```
(NP (NNP Air) (NNP Force) (NN contract))
```

Air Force is a specific entity and should form a separate constituent underneath the NP, as in our new annotation scheme:

```
(NP
  (NML (NNP Air) (NNP Force))
  (NN contract))
```

We use NML to specify that *Air Force* together is a nominal modifier of *contract*. Adding this annotation better represents the true syntactic and semantic structure, which will improve the performance of downstream NLP systems.

Our main contribution is a gold-standard labelled bracketing for every ambiguous noun phrase in the Penn Treebank. We describe the annotation guidelines and process, including the use of named entity data to improve annotation quality. We check the correctness of the corpus by measuring inter-annotator agreement, by reannotating the first section, and by comparing against the sub-NP structure in DepBank (King et al., 2003).

We also give an analysis of our extended Treebank, quantifying how much structure we have added, and how it is distributed across NPs. Finally, we test the utility of the extended Treebank for training statistical models on two tasks: NP bracketing (Lauer, 1995; Nakov and Hearst, 2005) and full parsing (Collins, 1999).

This new resource will allow any system or annotated corpus developed from the Penn Treebank, to represent noun phrase structure more accurately.

Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 240–247,
Prague, Czech Republic, June 2007.

2 Motivation

Many approaches to identifying base noun phrases have been explored as part of chunking (Ramshaw and Marcus, 1995), but determining sub-NP structure is rarely addressed. We could use multi-word expressions (MWEs) to identify some structure. For example, knowing *stock market* is a MWE may help bracket *stock market prices* correctly, and Named Entities (NEs) can be used the same way. However, this only resolves NPs dominating MWEs or NEs.

Understanding base-NP structure is important, since otherwise parsers will propose nonsensical noun phrases like *Force contract* by default and pass them onto downstream components. For example, Question Answering (QA) systems need to supply an NP as the answer to a factoid question, often using a parser to identify candidate NPs to return to the user. If the parser never generates the correct sub-NP structure, then the system may return a nonsensical answer even though the correct dominating noun phrase has been found.

Base-NP structure is also important for annotated data derived from the Penn Treebank. For instance, CCGbank (Hockenmaier, 2003) was created by semi-automatically converting the Treebank phrase structure to Combinatory Categorial Grammar (CCG) (Steedman, 2000) derivations. Since CCG derivations are binary branching, they cannot directly represent the flat structure of the Penn Treebank base-NPs.

Without the correct bracketing in the Treebank, strictly right-branching trees were created for all base-NPs. This has an unwelcome effect when conjunctions occur within an NP (Figure 1). An additional grammar rule is needed just to get a parse, but it is still not correct (Hockenmaier, 2003, p. 64). The awkward conversion results in bracketing (a) which should be (b):

(a) *(consumer ((electronics) and (appliances (retailing chain))))*

(b) *((((consumer electronics) and appliances) retailing) chain)*

We have previously experimented with using NEs to improve parsing performance on CCGbank. Due to the mis-alignment of NEs and right-branching NPs, the increase in performance was negligible.

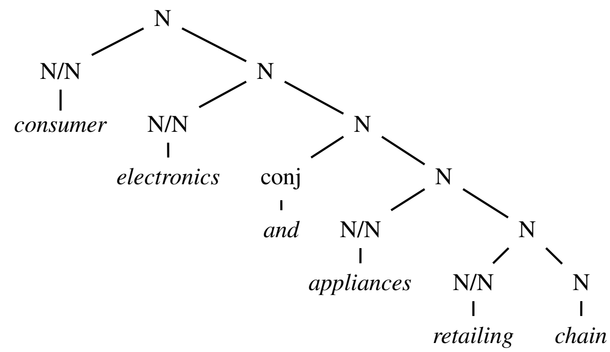

Figure 1: CCG derivation from Hockenmaier (2003)

3 Background

The NP bracketing task has often been posed in terms of choosing between the left or right branching structure of three word noun compounds:

(a) *(world (oil prices))* – Right-branching

(b) *((crude oil) prices)* – Left-branching

Most approaches to the problem use unsupervised methods, based on competing association strength between two of the words in the compound (Marcus, 1980, p. 253). There are two possible models to choose from: dependency or adjacency. The *dependency model* compares the association between words 1-2 to words 1-3, while the *adjacency model* compares words 1-2 to words 2-3.

Lauer (1995) has demonstrated superior performance of the dependency model using a test set of 244 (216 unique) noun compounds drawn from Grolier's encyclopedia. This data has been used to evaluate most research since. He uses Roget's thesaurus to smooth words into semantic classes, and then calculates association between classes based on their counts in a "training set" also drawn from Grolier's. He achieves 80.7% accuracy using POS tags to indentify bigrams in the training set.

Lapata and Keller (2004) derive estimates from web counts, and only compare at a lexical level, achieving 78.7% accuracy. Nakov and Hearst (2005) also use web counts, but incorporate additional counts from several variations on simple bigram queries, including queries for the pairs of words concatenated or joined by a hyphen. This results in an impressive 89.3% accuracy.

There have also been attempts to solve this task using supervised methods, even though the lack of gold-standard data makes this difficult. Girju et al.

(2005) draw a training set from raw WSJ text and use it to train a decision tree classifier achieving 73.1% accuracy. When they shuffled their data with Lauer's to create a new test and training split, their accuracy increased to 83.1% which may be a result of the ∼10% duplication in Lauer's test set.

We have created a new NP bracketing data set from our extended Treebank by extracting all rightmost three noun sequences from base-NPs. Our initial experiments are presented in Section 6.1.

4 Corpus Creation

According to Marcus et al. (1993), asking annotators to markup base-NP structure significantly reduced annotation speed, and for this reason base-NPs were left flat. The bracketing guidelines (Bies et al., 1995) also mention the considerable difficulty of identifying the correct scope for nominal modifiers. We found however, that while there are certainly difficult cases, the vast majority are quite simple and can be annotated reliably. Our annotation philosophy can be summarised as:

1. most cases are easy and fit a common pattern;
2. prefer the implicit right-branching structure for difficult decisions. Finance jargon was a common source of these;
3. mark very difficult to bracket NPs and discuss with other annotators later;

During this process we identified numerous cases that require a more sophisticated annotation scheme. There are genuine flat cases, primarily names like *John A. Smith*, that we would like to distinguish from implicitly right-branching NPs in the next version of the corpus. Although our scheme is still developing, we believe that the current annotation is already useful for statistical modelling, and we demonstrate this empirically in Section 6.

4.1 Annotation Process

Our annotation guidelines[1] are based on those developed for annotating full sub-NP structure in the biomedical domain (Kulick et al., 2004). The annotation guidelines for this biomedical corpus (an addendum to the Penn Treebank guidelines) introduce the use of NML nodes to mark internal NP structure.

[1]The guidelines and corpus are available on our webpages.

In summary, our guidelines leave right-branching structures untouched, and insert labelled brackets around left-branching structures. The label of the newly created constituent is NML or JJP, depending on whether its head is a noun or an adjective. We also chose not to alter the existing Penn Treebank annotation, even though the annotators found many errors during the annotation process. We wanted to keep our extended Treebank as similar to the original as possible, so that they remain comparable.

We developed a bracketing tool, which identifies ambiguous NPs and presents them to the user for disambiguation. An ambiguous NP is any (possibly non-base) NP with three or more contiguous children that are either single words or another NP. Certain common patterns, such as three words beginning with a determiner, are unambiguous, and were filtered out. The annotator is also shown the entire sentence surrounding the ambiguous NP.

The bracketing tool often suggests a bracketing using rules based mostly on named entity tags, which are drawn from the BBN corpus (Weischedel and Brunstein, 2005). For example, since *Air Force* is given ORG tags, the tool suggests that they be bracketed together first. Other suggestions come from previous bracketings of the same words, which helps to keep the annotator consistent.

Two post processes were carried out to increase annotation consistency and correctness. 915 difficult NPs were marked by the annotator and were then discussed with two other experts. Secondly, certain phrases that occurred numerous times and were non-trivial to bracket, e.g. *London Interbank Offered Rate*, were identified. An extra pass was made through the corpus, ensuring that every instance of these phrases was bracketed consistently.

4.2 Annotation Time

Annotation initially took over 9 hours per section of the Treebank. However, with practice this was reduced to about 3 hours per section. Each section contains around 2500 ambiguous NPs, i.e. annotating took approximately 5 seconds per NP. Most NPs require no bracketing, or fit into a standard pattern which the annotator soon becomes accustomed to, hence the task can be performed quite quickly.

For the original bracketing of the Treebank, annotators performed at 375–475 words per hour after a

	PREC.	RECALL	F-SCORE
Brackets	89.17	87.50	88.33
Dependencies	96.40	96.40	96.40
Brackets, revised	97.56	98.03	97.79
Dependencies, revised	99.27	99.27	99.27

Table 1: Agreement between annotators

few weeks, and increased to about 1000 words per hour after gaining more experience (Marcus et al., 1993). For our annotation process, counting each word in every NP shown, our speed was around 800 words per hour. This figure is not unexpected, as the task was not large enough to get more than a month's experience, and there is less structure to annotate.

5 Corpus Analysis

5.1 Inter-annotator Agreement

The annotation was performed by the first author. A second Computational Linguistics PhD student also annotated Section 23, allowing inter-annotator agreement, and the reliability of the annotations, to be measured. This also maximised the quality of the section used for parser testing.

We measured the proportion of matching brackets and dependencies between annotators, shown in Table 1, both before and after they discussed cases of disagreement and revised their annotations. The number of dependencies is fixed by the length of the NP, so the dependency precision and recall are the same. Counting matched brackets is a harsher evaluation, as there are many NPs that both annotators agree should have no additional bracketing, which are not taken into account by the metric.

The disagreements occurred for a small number of repeated instances, such as this case:

```
(NP                            (NP (NNP Goldman)
  (NML (NNP Goldman)             (, ,)
    (, ,)                        (NNP Sachs)
    (NNP Sachs) )                (CC &) (NNP Co) )
  (CC &) (NNP Co) )
```

The first annotator felt that `Goldman , Sachs` should form their own `NML` constituent, while the second annotator did not.

We can also look at exact matching on NPs, where the annotators originally agreed in 2667 of 2908 cases (91.71%), and after revision, in 2864 of 2907 cases (98.52%). These results demonstrate that high agreement rates are achievable for these annotations.

	MATCHED	TOTAL	%
By dependency	1409 (1315)	1479	95.27 (88.91)
By noun phrase	562 (489)	626	89.78 (78.12)
By dependency, only annotated NPs	578 (543)	627	92.19 (86.60)
By noun phrase, only annotated NPs	186 (162)	229	81.22 (70.74)

Table 2: Agreement with DepBank

5.2 DepBank Agreement

Another approach to measuring annotator reliability is to compare with an independently annotated corpus on the same text. We used the PARC700 Dependency Bank (King et al., 2003) which consists of 700 Section 23 sentences annotated with labelled dependencies. We use the Briscoe and Carroll (2006) version of DepBank, a 560 sentence subset used to evaluate the RASP parser.

Some translation is required to compare our brackets to DepBank dependencies. We map the brackets to dependencies by finding the head of the NP, using the Collins (1999) head finding rules, and then creating a dependency between each other child's head and this head. This does not work perfectly, and mismatches occur because of which dependencies DepBank marks explicitly, and how it chooses heads. The errors are investigated manually to determine their cause.

The results are shown in Table 2, with the number of agreements before manual checking shown in parentheses. Once again the dependency numbers are higher than those at the NP level. Similarly, when we only look at cases where we have inserted some annotations, we are looking at more difficult cases and the score is not as high.

The results of this analysis are quite positive. Over half of the disagreements that occur (in either measure) are caused by how company names are bracketed. While we have always separated the company name from post-modifiers such as *Corp* and *Inc*, DepBank does not in most cases. These results show that consistently and correctly bracketing noun phrase structure is possible, and that inter-annotator agreement is at an acceptable level.

5.3 Corpus Composition and Consistency

Looking at the entire Penn Treebank corpus, the annotation tool finds 60959 ambiguous NPs out of the 432639 NPs in the corpus (14.09%). 22851 of

LEVEL	COUNT	POS TAGS	EXAMPLE
NP	1073	JJ JJ NNS	big red cars
	1581	DT JJ NN NN	a high interest rate
	1693	JJ NN NNS	high interest rates
	3557	NNP NNP NNP	John A. Smith
NML	1468	NN NN	(interest rate) rises
	1538	JJ NN	(heavy truck) rentals
	1650	NNP NNP NNP	(A. B. C.) Corp
	8524	NNP NNP	(John Smith) Jr.
JJP	82	JJ JJ	(dark red) car
	114	RB JJ	(very high) rates
	122	JJ CC JJ	(big and red) apples
	160	“ JJ ”	(“ smart ”) cars

Table 3: Common POS tag sequences

these (37.49%) had brackets inserted by the annotator. This is as we expect, as the majority of NPs are right-branching. Of the brackets added, 22368 were NML nodes, while 863 were JJP.

To compare, we can count the number of existing NP and ADJP nodes found in the NPs that the bracketing tool presents. We find there are 32772 NP children, and 579 ADJP, which are quite similar numbers to the amount of nodes we have added. From this, we can say that our annotation process has introduced almost as much structural information into NPs as there was in the original Penn Treebank.

Table 3 shows the most common POS tag sequences for NP, NML and JJP nodes. An example is given showing typical words that match the POS tags. For NML and JJP, we also show the words bracketed, as they would appear under an NP node.

We checked the consistency of the annotations by identifying NPs with the same word sequence and checking whether they were always bracketed identically. After the first pass through, there were 360 word sequences with multiple bracketings, which occurred in 1923 NP instances. 489 of these instances differed from the majority case for that sequence, and were probably errors.

The annotator had marked certain difficult and commonly repeating NPs. From this we generated a list of phrases, and then made another pass through the corpus, synchronising all instances that contained one of these phrases. After this, only 150 instances differed from the majority case. Inspecting these remaining inconsistencies showed cases like:

```
(NP-TMP (NML (NNP Nov.) (CD 15))
  (, ,)
  (CD 1999))
```

where we were inconsistent in inserting the NML node because the Penn Treebank sometimes already has the structure annotated under an NP node. Since we do not make changes to existing brackets, we cannot fix these cases. Other inconsistencies are rare, but will be examined and corrected in a future release.

The annotator made a second pass over Section 00 to correct changes made after the beginning of the annotation process. Comparing the two passes can give us some idea of how the annotator changed as he grew more practiced at the task.

We find that the old and new versions are identical in 88.65% of NPs, with labelled precision, recall and F-score being 97.17%, 76.69% and 85.72% respectively. This tells us that there were many brackets originally missed that were added in the second pass. This is not surprising since the main problem with how Section 00 was annotated originally was that company names were not separated from their post-modifier (such as *Corp*). Besides this, it suggests that there is not a great deal of difference between an annotator just learning the task, and one who has had a great deal of experience with it.

5.4 Named Entity Suggestions

We have also evaluated how well the suggestion feature of the annotation tool performs. In particular, we want to determine how useful named entities are in determining the correct bracketing.

We ran the tool over the original corpus, following NE-based suggestions where possible. We find that when evaluated against our annotations, the F-score is 50.71%. We need to look closer at the precision and recall though, as they are quite different. The precision of 93.84% is quite high. However, there are many brackets where the entities do not help at all, and so the recall of this method was only 34.74%. This suggests that a NE feature may help to identify the correct bracketing in one third of cases.

6 Experiments

Having bracketed NPs in the Penn Treebank, we now describe our initial experiments on how this additional level of annotation can be exploited.

6.1 NP Bracketing Data

The obvious first task to consider is noun phrase bracketing itself. We implement a similar system to

CORPUS	# ITEMS	LEFT	RIGHT
Penn Treebank	5582	58.99%	41.01%
Lauer's	244	66.80%	33.20%

Table 4: Comparison of NP bracketing corpora

N-GRAM	MATCH
Unigrams	51.20%
Adjacency bigrams	6.35%
Dependency bigrams	3.85%
All bigrams	5.83%
Trigrams	1.40%

Table 5: Lexical overlap

Lauer (1995), described in Section 3, and report on results from our own data and Lauer's original set.

First, we extracted three word noun sequences from all the ambiguous NPs. If the last three children are nouns, then they became an example in our data set. If there is a NML node containing the first two nouns then it is left-branching, otherwise it is right-branching. Because we are only looking at the right-most part of the NP, we know that we are not extracting any nonsensical items. We also remove all items where the nouns are all part of a named entity to eliminate flat structure cases.

Statistics about the new data set and Lauer's data set are given in Table 4. As can be seen, the Penn Treebank based corpus is significantly larger, and has a more even mix of left and right-branching noun phrases. We also measured the amount of lexical overlap between the two corpora, shown in Table 5. This displays the percentage of n-grams in Lauer's corpus that are also in our corpus. We can clearly see that the two corpora are quite dissimilar, as even on unigrams barely half are shared.

6.2 NP Bracketing Results

With our new data set, we began running experiments similar to those carried out in the literature (Nakov and Hearst, 2005). We implemented both an adjacency and dependency model, and three different association measures: raw counts, bigram probability, and χ^2. We draw our counts from a corpus of n-gram counts calculated over 1 trillion words from the web (Brants and Franz, 2006).

The results from the experiments, on both our and Lauer's data set, are shown in Table 6. Our results

ASSOC. MEASURE	LAUER	PTB
Raw counts, adj.	75.41%	77.46%
Raw counts, dep.	77.05%	68.85%
Probability, adj.	71.31%	76.42%
Probability, dep.	**80.33%**	69.56%
χ^2, adj.	71.31%	**77.93%**
χ^2, dep.	74.59%	68.92%

Table 6: Bracketing task, unsupervised results

FEATURES	LAUER	10-FOLD CROSS
All features	80.74%	**89.91%** (1.04%)
Lexical	71.31%	**84.52%** (1.77%)
n-gram counts	75.41%	82.50% (1.49%)
Probability	72.54%	78.34% (2.11%)
χ^2	75.41%	80.10% (1.71%)
Adjacency model	72.95%	79.52% (1.32%)
Dependency model	**78.69%**	72.86% (1.48%)
Both models	76.23%	79.67% (1.42%)
-Lexical	79.92%	85.72% (0.77%)
-n-gram counts	80.74%	89.11% (1.39%)
-Probability	79.10%	**89.79%** (1.22%)
$-\chi^2$	80.74%	**89.79%** (0.98%)
-Adjacency model	81.56%	89.63% (0.96%)
-Dependency model	81.15%	89.72% (0.86%)
-Both models	**81.97%**	89.63% (0.95%)

Table 7: Bracketing task, supervised results

on Lauer's corpus are similar to those reported previously, with the dependency model outperforming the adjacency model on all measures. The bigram probability scores highest out of all the measures, while the χ^2 score performed the worst.

The results on the new corpus are even more surprising, with the adjacency model outperforming the dependency model by a wide margin. The χ^2 measure gives the highest accuracy, but still only just outperforms the raw counts. Our analysis shows that the good performance of the adjacency model comes from the large number of named entities in the corpus. When we remove all items that have any word as an entity, the results change, and the dependency model is superior. We also suspect that another cause of the unusual results is the different proportions of left and right-branching NPs.

With a large annotated corpus, we can now run supervised NP bracketing experiments. We present two configurations in Table 7: training on our corpus and testing on Lauer's set; and performing 10-fold cross validation using our corpus alone.

The feature set we explore encodes the information we used in the unsupervised experiments. Ta-

	OVERALL			ONLY NML JJP			NOT NML JJP		
	PREC.	RECALL	F-SCORE	PREC.	RECALL	F-SCORE	PREC.	RECALL	F-SCORE
Original	88.93	88.90	88.92	–	–	–	88.93	88.90	88.92
NML and JJP bracketed	88.63	88.29	88.46	77.93	62.93	69.63	88.85	88.93	88.89
Relabelled brackets	88.17	87.88	88.02	91.93	51.38	65.91	87.86	88.65	88.25

Table 8: Parsing performance

ble 7 shows the performance with: all features, followed by the individual features, and finally, after removing individual features.

The feature set includes: lexical features for each n-gram in the noun compound; n-gram counts for unigrams, bigrams and trigrams; raw probability and χ^2 association scores for all three bigrams in the compound; and the adjacency and dependency results for all three association measures. We discretised the non-binary features using an implementation of Fayyad and Irani's (1993) algorithm, and classify using MegaM[2].

The results on Lauer's set demonstrate that the dependency model performs well by itself but not with the other features. In fact, a better result comes from using every feature *except* those from the dependency and adjacency models. It is also impressive how good the performance is, considering the large differences between our data set and Lauer's.

These differences also account for the disparate cross-validation figures. On this data, the lexical features perform the best, which is to be expected given the nature of the corpus. The best model in this case comes from using all the features.

6.3 Collins Parsing

We can also look at the impact of our new annotations upon full statistical parsing. We use Bikel's implementation (Bikel, 2004) of Collins' parser (Collins, 1999) in order to carry out these experiments, using the non-deficient Collins settings. The numbers we give are labelled bracket precision, recall and F-scores for all sentences. Bikel mentions that base-NPs are treated very differently in Collins' parser, and so it will be interesting to observe the results using our new annotations.

Firstly, we compare the parser's performance on the original Penn Treebank and the new NML and JJP bracketed version. Table 8 shows that the new brackets make parsing marginally more difficult overall (by about 0.5% in F-score).

The performance on only the new NML and JJP brackets is not very high. This shows the difficulty of correctly bracketing NPs. Conversely, the figures for all brackets except NML and JJP are only a tiny amount less in our extended corpus. This means that performance for other phrases is hardly changed by the new NP brackets.

We also ran an experiment where the new NML and JJP labels were relabelled as NP and ADJP. These are the labels that would be given if NPs were originally bracketed with the rest of the Penn Treebank. This meant the model would not have to discriminate between two different types of noun and adjective structure. The performance, as shown in Table 8, was even lower with this approach, suggesting that the distinction is larger than we anticipated. On the other hand, the precision on NML and JJP constituents was quite high, so the parser is able to identify at least some of the structure very well.

7 Conclusion

The work presented in this paper is a first step towards accurate representation of noun phrase structure in NLP corpora. There are several distinctions that our annotation currently ignores that we would like to identify correctly in the future. Firstly, NPs with genuine flat structure are currently treated as implicitly right branching. Secondly, there is still ambiguity in determining the head of a noun phrase. Although Collins' head finding rules work in most NPs, there are cases such as *IBM Australia* where the head is not the right-most noun. Similarly, apposition is very common in the Penn Treebank, in NPs such as *John Smith , IBM president*. We would like to be able to identify these multi-head constructs properly, rather than simply treating them as a single entity (or even worse, as two different entities).

Having the correct NP structure also means that we can now represent the true structure in CCGbank, one of the problems we described earlier. Transfer-

[2] Available at http://www.cs.utah.edu/ hal/megam/

ring our annotations should be fairly simple, requiring just a few changes to how NPs are treated in the current translation process.

The addition of consistent, gold-standard, noun phrase structure to a large corpus is a significant achievement. We have shown that the these annotations can be created in a feasible time frame with high inter-annotator agreement of 98.52% (measuring exact NP matches). The new brackets cause only a small drop in parsing performance, and no significant decrease on the existing structure. As NEs were useful for suggesting brackets automatically, we intend to incorporate NE information into statistical parsing models in the future.

Our annotated corpus can improve the performance of any system that relies on NPs from parsers trained on the Penn Treebank. A Collins' parser trained on our corpus is now able to identify sub-NP brackets, making it of use in other NLP systems. QA systems, for example, will be able to exploit internal NP structure. In the future, we will improve the parser's performance on NML and JJP brackets.

We have provided a significantly larger corpus for analysing NP structure than has ever been made available before. This is integrated within perhaps the most influential corpus in NLP. The large number of systems trained on Penn Treebank data can all benefit from the extended resource we have created.

Acknowledgements

We would like to thank Matthew Honnibal, our second annotator, who also helped design the guidelines; Toby Hawker, for implementing the discretiser; Mark Lauer for releasing his data; and the anonymous reviewers for their helpful feedback. This work has been supported by the Australian Research Council under Discovery Projects DP0453131 and DP0665973.

References

Ann Bies, Mark Ferguson, Karen Katz, and Robert MacIntyre. 1995. Bracketing guidelines for Treebank II style Penn Treebank project. Technical report, University of Pennsylvania.

Dan Bikel. 2004. *On the Parameter Space of Generative Lexicalized Statistical Parsing Models*. Ph.D. thesis, University of Pennsylvania.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram version 1. Linguistic Data Consortium.

Ted Briscoe and John Carroll. 2006. Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank. In *Proceedings of the Poster Session of COLING/ACL-06*. Sydney, Australia.

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

Usama M. Fayyad and Keki B. Irani. 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Joint Conference on Artifical Intelligence (IJCAI–93)*, pages 1022–1029. Chambery, France.

Roxana Girju, Dan Moldovan, Marta Tatu, and Daniel Antohe. 2005. On the semantics of noun compounds. *Journal of Computer Speech and Language - Special Issue on Multiword Expressions*, 19(4):313–330.

Julia Hockenmaier. 2003. *Data and Models for Statistical Parsing with Combinatory Categorial Grammar*. Ph.D. thesis, University of Edinburgh.

Tracy Holloway King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. 2003. The PARC700 dependency bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*. Budapest, Hungary.

Seth Kulick, Ann Bies, Mark Libeman, Mark Mandel, Ryan McDonald, Martha Palmer, Andrew Schein, and Lyle Ungar. 2004. Integrated annotation for biomedical information extraction. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*. Boston.

Mirella Lapata and Frank Keller. 2004. The web as a baseline: Evaluating the performance of unsupervised web-based models for a range of NLP tasks. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 121–128. Boston.

Mark Lauer. 1995. Corpus statistics meet the compound noun: Some empirical results. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. Cambridge, MA.

Mitchell Marcus. 1980. *A Theory of Syntactic Recognition for Natural Language*. MIT Press, Cambridge, MA.

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

Preslav Nakov and Marti Hearst. 2005. Search engine statistics beyond the n-gram: Application to noun compound bracketing. In *Proceedings of CoNLL-2005, Ninth Conference on Computational Natural Language Learning*. Ann Arbor, MI.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*. Cambridge MA, USA.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, MA.

Ralph Weischedel and Ada Brunstein. 2005. BBN pronoun coreference and entity type corpus. Technical report, Linguistic Data Consortium.