The Effect of Corpus Size in Combining Supervised and Unsupervised Training for Disambiguation

Michaela Atterer
Institute for NLP
University of Stuttgart
atterer@ims.uni-stuttgart.de

Hinrich Schütze
Institute for NLP
University of Stuttgart
hinrich@hotmail.com

Abstract

We investigate the effect of corpus size in combining supervised and unsupervised learning for two types of attachment decisions: relative clause attachment and prepositional phrase attachment. The supervised component is Collins' parser, trained on the Wall Street Journal. The unsupervised component gathers lexical statistics from an unannotated corpus of newswire text. We find that the combined system only improves the performance of the parser for small training sets. Surprisingly, the size of the unannotated corpus has little effect due to the noisiness of the lexical statistics acquired by unsupervised learning.

1 Introduction

The best performing systems for many tasks in natural language processing are based on supervised training on annotated corpora such as the Penn Treebank (Marcus et al., 1993) and the prepositional phrase data set first described in (Ratnaparkhi et al., 1994). However, the production of training sets is expensive. They are not available for many domains and languages. This motivates research on combining supervised with unsupervised learning since unannotated text is in ample supply for most domains in the major languages of the world. The question arises how much annotated and unannotated data is necessary in combination learning strategies. We investigate this question for two attachment ambiguity problems: relative clause (RC) attachment and prepositional phrase (PP) attachment. The supervised component is Collins' parser (Collins, 1997), trained on the Wall Street Journal. The unsupervised component gathers lexical statistics from an unannotated corpus of newswire text.

The sizes of both types of corpora, annotated and unannotated, are of interest. We would expect that large annotated corpora (training sets) tend to make the additional information from unannotated corpora redundant. This expectation is confirmed in our experiments. For example, when using the maximum training set available for PP attachment, performance decreases when "unannotated" lexical statistics are added.

For unannotated corpora, we would expect the opposite effect. The larger the unannotated corpus, the better the combined system should perform. While there is a general tendency to this effect, the improvements in our experiments reach a plateau quickly as the unlabeled corpus grows, especially for PP attachment. We attribute this result to the noisiness of the statistics collected from unlabeled corpora.

The paper is organized as follows. Sections 2, 3 and 4 describe data sets, methods and experiments. Section 5 evaluates and discusses experimental results. Section 6 compares our approach to prior work. Section 7 states our conclusions.

2 Data Sets

The unlabeled corpus is the Reuters RCV1 corpus, about 80,000,000 words of newswire text (Lewis et al., 2004). Three different subsets, corresponding to roughly 10%, 50% and 100% of the corpus, were created for experiments related to the size of the unannotated corpus. (Two weeks after Aug 5, 1997, were set apart for future experiments.)

The labeled corpus is the Penn Wall Street Journal treebank (Marcus et al., 1993). We

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 25–32,
Sydney, July 2006.

created the 5 subsets shown in Table 1 for experiments related to the size of the annotated corpus.

unlabeled R	
100%	20/08/1996–05/08/1997 (351 days)
50%	20/08/1996–17/02/1997 (182 days)
10%	20/08/1996–24/09/1996 (36 days)
labeled WSJ	
50%	sections 00–12 (23412 sentences)
25%	lines 1 – 292960 (11637 sentences)
5%	lines 1 – 58284 (2304 sentences)
1%	lines 1 – 11720 (500 sentences)
0.05%	lines 1 – 611 (23 sentences)

Table 1: Corpora used for the experiments: unlabeled Reuters (R) corpus for attachment statistics, labeled Penn treebank (WSJ) for training the Collins parser.

The test set, sections 13-24, is larger than in most studies because a single section does not contain a sufficient number of RC attachment ambiguities for a meaningful evaluation.

which-clauses subset	highA	lowA	total
develop set (sec 00-12)	71	211	282
test set (sec 13-24)	71	193	264
PP subset	verbA	nounA	total
develop set (sec 00-12)	5927	6560	12487
test set (sec 13-24)	5930	6273	12203

Table 2: RC and PP attachment ambiguities in the Penn Treebank. Number of instances with high attachment (highA), low attachment (lowA), verb attachment (verbA), and noun attachment (nounA) according to the gold standard.

All instances of RC and PP attachments were extracted from development and test sets, yielding about 250 RC ambiguities and 12,000 PP ambiguities per set (Table 2). An RC attachment ambiguity was defined as a sentence containing the pattern `NP1 Prep NP2 which`. For example, the relative clause in Example 1 can either attach to *mechanism* or to *System.*

(1) ... the exchange-rate mechanism of the European Monetary System, which links the major EC currencies.

A PP attachment ambiguity was defined as a subtree matching either [VP [NP PP]] or [VP NP PP]. An example of a PP attachment ambiguity is Example 2 where either the approval or the transaction is performed by written consent.

(2) ...a majority ...have approved the transaction by written consent ...

Both data sets are available for download (Web Appendix, 2006). We did not use the PP data set described by (Ratnaparkhi et al., 1994) because we are using more context than the limited context available in that set (see below).

3 Methods

Collins parser. Our baseline method for ambiguity resolution is the Collins parser as implemented by Bikel (Collins, 1997; Bikel, 2004). For each ambiguity, we check whether the attachment ambiguity is resolved correctly by the 5 parsers corresponding to the different training sets. If the attachment ambiguity is not recognized (e.g., because parsing failed), then the corresponding ambiguity is excluded for that instance of the parser. As a result, the size of the effective test set varies from parser to parser (see Table 4).

Minipar. The unannotated corpus is analyzed using minipar (Lin, 1998), a partial dependency parser. The corpus is parsed and all extracted dependencies are stored for later use. Dependencies in ambiguous PP attachments (those corresponding to [VP NP PP] and [VP [NP PP]] subtrees) are not indexed. An experiment with indexing both alternatives for ambiguous structures yielded poor results. For example, indexing both alternatives will create a large number of spurious verb attachments of *of*, which in turn will result in incorrect high attachments by our disambiguation algorithm.

For relative clauses, no such filtering is necessary. For example, spurious subject-verb dependencies due to RC ambiguities are rare compared to a large number of subject-verb dependencies that can be extracted reliably.

Inverted index. Dependencies extracted by minipar are stored in an inverted index (Witten et al., 1999), implemented in Lucene (Lucene, 2006). For example, "john subj buy", the analysis returned by minipar for *John buys*, is stored as "john buy john<subj

subj<buy john<subj<buy". All words, dependencies and partial dependencies of a sentence are stored together as one document. This storage mechanism enables fast on-line queries for lexical and dependency statistics, e.g., how many sentences contain the dependency "john subj buy", how often does *john* occur as a subject, how often does *buy* have *john* as a subject and *car* as an object etc. Query results are approximate because double occurrences are only counted once and structures giving rise to the same set of dependencies (*a piece of a tile of a roof of a house* vs. *a piece of a roof of a tile of a house*) cannot be distinguished. We believe that an inverted index is the most efficient data structure for our purposes. For example, we need not compute expensive joins as would be required in a database implementation. Our long-term goal is to use this inverted index of dependencies as a versatile component of NLP systems in analogy to the increasingly important role of search engines for association and word count statistics in NLP.

A total of three inverted indexes were created, one each for the 10%, 50% and 100% Reuters subset.

Lattice-Based Disambiguation. Our disambiguation method is Lattice-Based Disambiguation (LBD, (Atterer and Schütze, 2006)). We formalize a possible attachment as a triple $< R, i, X >$ where X is (the parse of) a phrase with two or more possible attachment nodes in a sentence S, i is one of these attachment nodes and R is (the relevant part of a parse of) S with X removed. For example, the two attachments in Example 2 are represented as the triples:
$<$ approved$_{i_1}$ the transaction$_{i_2}, i_1,$ by consent $>$,
$<$ approved$_{i_1}$ the transaction$_{i_2}, i_2,$ by consent $>$.

We decide between attachment possibilities based on pointwise mutual information, the well-known measure of how surprising it is to see R and X together given their individual frequencies:

$$\mathrm{MI}(< R, i, X >) = \log_2 \frac{P(<R,i,X>)}{P(R)P(X)}$$
$$\text{for } P(< R, i, X >), P(R), P(X) \neq 0$$
$$\mathrm{MI}(< R, i, X >) = 0 \quad \text{otherwise}$$

where the probabilities of the dependency structures $< R, i, X >$, R and X are estimated on the unlabeled corpus by querying the inverted index. Unfortunately, these structures will often not occur in the corpus. If this is the case we back off to generalizations of R and X. The generalizations form a lattice as shown in Figure 1 for PP attachment. For example, MN:pMN corresponds to *commercial transaction by unanimous consent*, N:pM to *transaction by unanimous* etc. For 0:p we compute MI of the two events "noun attachment" and "occurrence of p". Points in the lattice in Figure 1 are created by successive elimination of material from the complete context R:X. A child c directly dominated by a parent p is created by removing exactly one contextual element from p, either on the right side (the attachment phrase) or on the left side (the attachment node). For RC attachment, generalizations other than elimination are introduced such as the replacement of a proper noun (e.g., *Canada*) by its category (*country*) (see below).

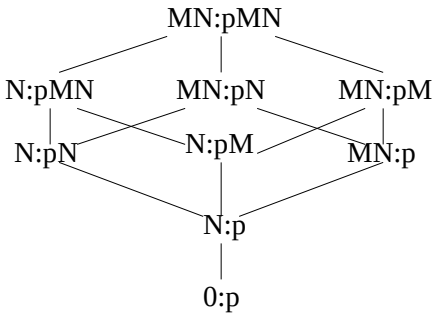

Figure 1: Lattice of pairs of potential attachment site (NP) and attachment phrase (PP). M: premodifying adjective or noun (upper or lower NP), N: head noun (upper or lower NP), p: Preposition.

The MI of each point in the lattice is computed. We then take the maximum over all MI values of the lattice as a measure of the *affinity* of attachment phrase and attachment node. The intuition is that we are looking for the strongest evidence available for the attachment. The strongest evidence is often not provided by the most specific context (MN:pMN in the example) since contextual elements like modifiers will only add noise to the attachment decision in some cases. The actual syntactic disambiguation is performed by computing the affinity (maximum over MI values in the lattice) for each possible attachment and selecting the attachment with highest affinity. (The

default attachment is selected if the two values are equal.) The second lattice for PP attachment, the lattice for attachment to the verb, has a structure identical to Figure 1, but the attachment node is SV instead of MN, where S denotes the subject and V the verb. So the supremum of that lattice is SV:pMN and the infimum is 0:p (which in this case corresponds to the MI of verb attachment and occurrence of the preposition).

LBD is motivated by the desire to use as much context as possible for disambiguation. Previous work on attachment disambiguation has generally used less context than in this paper (e.g., modifiers have not been used for PP attachment). No change to LBD is necessary if the lattice of contexts is extended by adding additional contextual elements (e.g., the preposition between the two attachment nodes in RC, which we do not consider in this paper).

4 Experiments

The Reuters corpus was parsed with minipar and all dependencies were extracted. Three inverted indexes were created, corresponding to 10%, 50% and 100% of the corpus.[1] Five parameter sets for the Collins parser were created by training it on the WSJ training sets in Table 1. Sentences with attachment ambiguities in the WSJ corpus were parsed with minipar to generate Lucene queries. (We chose this procedure to ensure compatibility of query and index formats.) The Lucene queries were run on the three indexes. LBD disambiguation was then applied based on the statistics returned by the queries. LBD results are applied to the output of the Collins parser by simply replacing all attachment decisions with LBD decisions.

4.1 RC attachment

The lattice for LBD in RC attachment is shown in Figure 2. When disambiguating an RC attachment, two instances of the lattice are formed, one for NP1 and one for NP2 in `NP1 Prep NP2 RC`. Figure 2 shows the maximum possible lattice. If contextual elements are not present in a context (e.g., a modifier), then the lattice will be smaller. The supremum of the lattice corresponds to a query that includes the entire NP (including modifying adjectives and nouns)[2], the verb and its object. Example: *exchange_rate<nn<mechanim && mechanism<subj<link && currency<obj<link.*

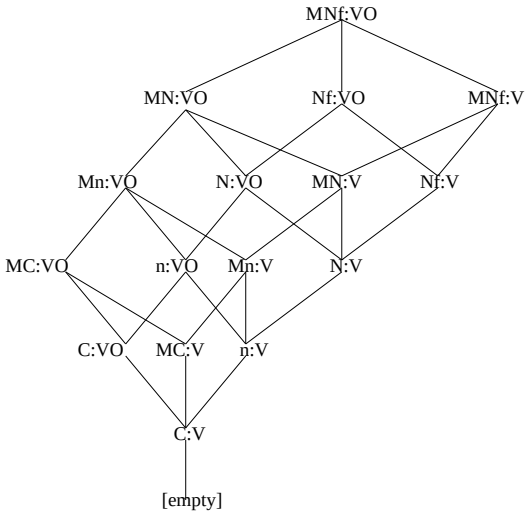

Figure 2: Lattice of pairs of potential attachment site NP and relative clause X. M: premodifying adjective or noun, Nf: head noun with lexical modifiers, N: head noun only, n: head noun in lower case, C: class of NP, V: verb in relative clause, O: object of verb in the relative clause.

To generalize contexts in the lattice, the following generalization operations are employed:

- strip the NP of the modifying adjective/noun (*weekly report → report*)
- use only the head noun of the NP (*Catastrophic Care Act → Act*)
- use the head noun in lower case (*Act → act*)
- for named entities use a hypernym of the NP (*American Bell Telephone Co. → company*)
- strip the object from X (*company have subsidiary → company have*)

The most important dependency for disam-

[1] In fact, two different sets of inverted indexes were created, one each for PP and RC disambiguation. The RC index indexes all dependencies, including ambiguous PP dependencies. Computing the RC statistics on the PP index should not affect the RC results presented here, but we didn't have time to confirm this experimentally for this paper.

[2] From the minipar output, we use all adjectives that modify the NP via the relation *mod*, and all nouns that modify the NP via the relation *nn*.

biguation is the noun-verb link, but the other dependencies also improve the accuracy of disambiguation (Atterer and Schütze, 2006). For example, light verbs like *make* and *have* only provide disambiguation information when their objects are also considered.

Downcasing and hypernym generalizations were used because proper nouns often cause sparse data problems. Named entity classes were identified with LingPipe (LingPipe, 2006). Named entities identified as companies or organizations are replaced with *company* in the query. Locations are replaced with *country*. Persons block RC attachment because *which*-clauses do not attach to person names, resulting in an attachment of the RC to the other NP.

query	MI
+exchange_rate⟨nn⟨mechanism +mechanism⟨subj⟨link +currency⟨obj⟨link	12.2
+exchange_rate⟨nn⟨mechanism +mechanism⟨subj⟨link	4.8
+mechanism⟨subj⟨link +currency⟨obj⟨link	10.2
mechanism⟨subj⟨link	3.4
+European_Monetary_System⟨subj⟨link +currency⟨obj⟨link	0
+System⟨subj⟨link +currency⟨obj⟨link	0
European_Monetary_System⟨subj⟨link	0
System⟨subj⟨link	0
+system⟨subj⟨link +currency⟨obj⟨link	0
system⟨subj⟨link	1.2
+company⟨subj⟨link +currency⟨obj⟨link	0
company⟨subj⟨link	-1.1
empty	3

Table 3: Queries for computing high attachment (above) and low attachment (below) for Example 1.

Table 3 shows queries and mutual information values for Example 1. The highest values are 12.2 for high attachment (*mechanism*) and 3 for low attachment (*System*). The algorithm therefore selects high attachment.

The value 3 for low attachment is the default value for the empty context. This value reflects the bias for low attachment: the majority of relative clauses are attached low. If all MI-values are zero or otherwise low, this procedure will automatically result in low attachment.[3]

[3]We experimented with a number of values (2, 3, and 4) on the development set. Accuracy of the algorithm was best for a value of 3. The results presented here differ slightly from those in (Atterer and Schütze, 2006) due to a coding error.

Decision list. For increased accuracy, LBD is embedded in the following decision list.

1. If minipar has already chosen high attachment, choose high attachment (this only occurs if `NP1 Prep NP2` is a named entity).
2. If there is agreement between the verb and only one of the NPs, attach to this NP.
3. If one of the NPs is in a list of person entities, attach to the other NP.[4]
4. If possible, use LBD.
5. If none of the above strategies was successful (e.g. in the case of parsing errors), attach low.

4.2 PP attachment

The two lattices for LBD applied to PP attachment were described in Section 3 and Figure 1. The only generalization operation used in these two lattices is elimination of contextual elements (in particular, there is no downcasing and named entity recognition). Note that in RC attachment, we compare affinities of two instances of the same lattice (the one shown in Figure 2). In PP attachment, we compare affinities of two different lattices since the two attachment points (verb vs. noun) are different. The basic version of LBD (with the untuned default value 0 and without decision lists) was used for PP attachment.

5 Evaluation and Discussion

Evaluation results are shown in Table 4. The lines marked *LBD* evaluate the performance of LBD separately (without Collins' parser). LBD is significantly better than the baseline for PP attachment ($p < 0.001$, all tests are χ^2 tests). LBD is also better than baseline for RC attachment, but this result is not significant due to the small size of the data set (264). Note that the baseline for PP attachment is 51.4% as indicated in the table (upper right corner of PP table), but that the baseline for RC attachment is 73.1%. The difference between 73.1% and 76.1% (upper right corner of RC table) is due to the fact that for RC attachment LBD proper is embedded in a decision list. The decision list alone, with an

[4]This list contains 136 entries and was semiautomatically computed from the Reuters corpus: Antecedents of *who* relative clauses were extracted, and the top 200 were filtered manually.

RC attachment						
Train data	#	Coll. only	100% R	50% R	10% R	0% R
LBD	264		78.4%	78.0%	76.9%	76.1%
50%	251	71.7%	78.5%	78.1%	76.9%	76.1%
25%	250	70.0%	78.0%	77.6%	76.4%	76.4%
5%	238	68.9%	78.2%	77.7%	76.9%	76.1%
1%	245	67.8%	78.8%	78.4%	77.1%	76.7%
0.05%	194	60.8%	76.8%	76.3%	75.8%	73.7%

PP attachment						
Train data	#	Coll. only	100% R	50% R	10% R	0% R
LBD	12203		73.4%	73.4%	73.0%	51.4%
50%	11953	82.8%	73.6%	73.6%	73.2%	51.7%
25%	11950	81.5%	73.6%	73.7%	73.3%	51.7%
5%	11737	77.4%	74.1%	74.2%	73.7%	52.3%
1%	11803	72.9%	73.6%	73.6%	73.2%	51.6%
0.05%	8486	58.0%	73.9%	73.8%	74.0%	52.8%

Table 4: Experimental results. Results for LBD (without Collins) are given in the first lines. # is the size of the test set. The baselines are 73.1% (RC) and 51.4% (PP). The combined method performs better for small training sets. There is no significant difference between 10%, 50% and 100% for the combination method ($p < 0.05$).

unlabeled corpus of size 0, achieves a performance of 76.1%.

The bottom five lines of each table evaluate combinations of a parameter set trained on a subset of WSJ (0.05% – 50%) and a particular size of the unlabeled corpus (100% – 0%). In addition, the third column gives the performance of Collins' parser without LBD. Recall that test set size (second column) varies because we discard a test instance if Collins' parser does not recognize that there is an ambiguity (e.g., because of a parse failure). As expected, performance increases as the size of the training set grows, e.g., from 58.0% to 82.8% for PP attachment.

The combination of Collins and LBD is consistently better than Collins for RC attachment (not statistically significant due to the size of the data set). However, this is not the case for PP attachment. Due to the good performance of Collins' parser for even small training sets, the combination is only superior for the two smallest training sets (significant for the smallest set, $p < 0.001$).

The most surprising result of the experiments is the small difference between the three unlabeled corpora. There is no clear pattern in the data for PP attachment and only a small effect for RC attachment: an increase between 1% and 2% when corpus size is increased from 10% to 100%.

We performed an analysis of a sample of incorrectly attached PPs to investigate why unlabeled corpus size has such a small effect. We found that the noisiness of the statistics extracted from Reuters were often responsible for attachment errors. The noisiness is caused by our filtering strategy (ambiguous PPs are not used, resulting in undercounting), by the approximation of counts by Lucene (Lucene overcounts and undercounts as discussed in Section 3) and by minipar parse errors. Parse errors are particularly harmful in cases like *the impact it would have on prospects*, where, due to the extraction of the NP *impact*, minipar attaches the PP to the verb. We did not filter out these more complex ambiguous cases. Finally, the two corpora are from distinct sources and from distinct time periods (early nineties vs. mid-nineties). Many topic- and time-specific dependencies can only be mined from more similar corpora.

The experiments reveal interesting differences between PP and RC attachment. The dependencies used in RC disambiguation rarely occur in an ambiguous context (e.g., most subject-verb dependencies can be reliably extracted). In contrast, a large proportion of the dependencies needed in PP disambiguation (verb-prep and noun-prep dependencies) do occur in ambiguous contexts. Another difference is that RC attachment is syntactically more complex. It interacts with agreement, passive and long-distance depen-

dencies. The algorithm proposed for RC applies grammatical constraints successfully. A final difference is that the baseline for RC is much higher than for PP and therefore harder to beat.[5]

An innovation of our disambiguation system is the use of a search engine, lucene, for serving up dependency statistics. The advantage is that counts can be computed quickly and dynamically. New text can be added on an ongoing basis to the index. The updated dependency statistics are immediately available and can benefit disambiguation performance. Such a system can adapt easily to new topics and changes over time. However, this architecture negatively affects accuracy. The unsupervised approach of (Hindle and Rooth, 1993) achieves almost 80% accuracy by using partial dependency statistics to disambiguate ambiguous sentences in the unlabeled corpus. Ambiguous sentences were excluded from our index to make index construction simple and efficient. Our larger corpus (about 6 times as large as Hindle et al.'s) did not compensate for our lower-quality statistics.

6 Related Work

Other work combining supervised and unsupervised learning for parsing includes (Charniak, 1997), (Johnson and Riezler, 2000), and (Schmid, 2002). These papers present integrated formal frameworks for incorporating information learned from unlabeled corpora, but they do not explicitly address PP and RC attachment. The same is true for uncorrected colearning in (Hwa et al., 2003).

Conversely, no previous work on PP and RC attachment has integrated specialized ambiguity resolution into parsing. For example, (Toutanova et al., 2004) present one of the best results achieved so far on the WSJ PP set: 87.5%. They also integrate supervised and unsupervised learning. But to our knowledge, the relationship to parsing has not been explored before – even though application to parsing is the stated objective of most work on PP attachment.

[5]However, the baseline is similarly high for the PP problem if the most likely attachment is chosen per preposition: 72.2% according to (Collins and Brooks, 1995).

With the exception of (Hindle and Rooth, 1993), most unsupervised work on PP attachment is based on superficial analysis of the unlabeled corpus without the use of partial parsing (Volk, 2001; Calvo et al., 2005). We believe that dependencies offer a better basis for reliable disambiguation than cooccurrence and fixed-phrase statistics. The difference to (Hindle and Rooth, 1993) was discussed above with respect to analysing the unlabeled corpus. In addition, the decision procedure presented here is different from Hindle et al.'s. LBD uses more context and can, in principle, accommodate arbitrarily large contexts. However, an evaluation comparing the performance of the two methods is necessary.

The LBD model can be viewed as a backoff model that combines estimates from several "backoffs". In a typical backoff model, there is a single more general model to back off to. (Collins and Brooks, 1995) also present a model with multiple backoffs. One of its variants computes the estimate in question as the average of three backoffs. In addition to the maximum used here, testing other combination strategies for the MI values in the lattice (e.g., average, sum, frequency-weighted sum) would be desirable. In general, MI has not been used in a backoff model before as far as we know.

Previous work on relative clause attachment has been supervised (Siddharthan, 2002a; Siddharthan, 2002b; Yeh and Vilain, 1998).[6] (Siddharthan, 2002b)'s accuracy for RC attachment is 76.5%.[7]

7 Conclusion

Previous work on specific types of ambiguities (like RC and PP) has not addressed the integration of specific resolution algorithms into a generic statistical parser. In this paper, we have shown for two types of ambiguities, relative clause and prepositional phrase attachment ambiguity, that integration into a statistical parser is possible and that the com-

[6]Strictly speaking, our experiments were not completely unsupervised since the default value and the most frequent attachment were determined based on the development set.

[7]We attempted to recreate Siddharthan's training and test sets, but were not able to based on the description in the paper and email communication with the author.

bined system performs better than either component by itself. However, for PP attachment this only holds for small training set sizes. For large training sets, we could only show an improvement for RC attachment.

Surprisingly, we only found a small effect of the size of the unlabeled corpus on disambiguation performance due to the noisiness of statistics extracted from raw text. Once the unlabeled corpus has reached a certain size (5-10 million words in our experiments) combined performance does not increase further.

The results in this paper demonstrate that the baseline of a state-of-the-art lexicalized parser for specific disambiguation problems like RC and PP is quite high compared to recent results for stand-alone PP disambiguation. For example, (Toutanova et al., 2004) achieve a performance of 87.6% for a training set of about 85% of WSJ. That number is not that far from the 82.8% achieved by Collins' parser in our experiments when trained on 50% of WSJ. Some of the supervised approaches to PP attachment may have to be reevaluated in light of this good performance of generic parsers.

References

Michaela Atterer and Hinrich Schütze. 2006. A lattice-based framework for enhancing statistical parsers with information from unlabeled corpora. In *CoNLL*.

Daniel M. Bikel. 2004. Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4):479–511.

Hiram Calvo, Alexander Gelbukh, and Adam Kilgarriff. 2005. Distributional thesaurus vs. WordNet: A comparison of backoff techniques for unsupervised PP attachment. In *CICLing*.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *AAAI/IAAI*, pages 598–603.

Michael Collins and James Brooks. 1995. Prepositional attachment through a backed-off model. In *Third Workshop on Very Large Corpora*. Association for Computational Linguistics.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *ACL*.

Donald Hindle and Mats Rooth. 1993. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120.

Rebecca Hwa, Miles Osborne, Anoop Sarkar, and Mark Steedman. 2003. Corrected co-training for statistical parsers. In *Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, ICML*.

Mark Johnson and Stefan Riezler. 2000. Exploiting auxiliary distributions in stochastic unification-based grammars. In *NAACL*.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5.

Dekang Lin. 1998. Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain.

LingPipe. 2006. http://www.alias-i.com/lingpipe/.

Lucene. 2006. http://lucene.apache.org.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large natural language corpus of English: the Penn treebank. *Computational Linguistics*, 19:313–330.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *HLT*.

Helmut Schmid. 2002. Lexicalization of probabilistic grammars. In *Coling*.

Advaith Siddharthan. 2002a. Resolving attachment and clause boundary ambiguities for simplifying relative clause constructs. In *Student Research Workshop, ACL*.

Advaith Siddharthan. 2002b. Resolving relative clause attachment ambiguities using machine learning techniques and wordnet hierarchies. In *4th Discourse Anaphora and Anaphora Resolution Colloquium*.

Kristina Toutanova, Christopher D. Manning, and Andrew Y. Ng. 2004. Learning random walk models for inducing word dependency distributions. In *ICML*.

Martin Volk. 2001. Exploiting the WWW as a corpus to resolve pp attachment ambiguities. In *Corpus Linguistics 2001*.

Web Appendix. 2006. http://www.ims.uni-stuttgart.de/∼schuetze/colingacl06/apdx.html.

Ian H. Witten, Alistair Moffat, and Timothy C. Bell. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufman.

Alexander S. Yeh and Marc B. Vilain. 1998. Some properties of preposition and subordinate conjunction attachments. In *Coling*.