Distributional Phrase Structure Induction

Dan Klein and **Christopher D. Manning**
Computer Science Department
Stanford University
Stanford, CA 94305-9040
{klein, manning}@cs.stanford.edu

Abstract

Unsupervised grammar induction systems commonly judge potential constituents on the basis of their effects on the likelihood of the data. Linguistic justifications of constituency, on the other hand, rely on notions such as substitutability and varying external contexts. We describe two systems for distributional grammar induction which operate on such principles, using part-of-speech tags as the contextual features. The advantages and disadvantages of these systems are examined, including precision/recall trade-offs, error analysis, and extensibility.

1 Overview

While early work showed that small, artificial context-free grammars could be induced with the EM algorithm (Lari and Young, 1990) or with chunk-merge systems (Stolcke and Omohundro, 1994), studies with large natural language grammars have shown that these methods of completely unsupervised acquisition are generally ineffective. For instance, Charniak (1993) describes experiments running the EM algorithm from random starting points, which produced widely varying grammars of extremely poor quality. Because of these kinds of results, the vast majority of statistical parsing work has focused on parsing as a supervised learning problem (Collins, 1997; Charniak, 2000). It remains an open problem whether an entirely unsupervised method can either produce linguistically sensible grammars or accurately parse free text.

However, there are compelling motivations for unsupervised grammar induction. Building supervised training data requires considerable resources, including time and linguistic expertise. Furthermore, investigating unsupervised methods can shed light on linguistic phenomena which are implicitly captured within a supervised parser's supervisory information, and, therefore, often not explicitly modeled in such systems. For example, our system and others have difficulty correctly attaching subjects to verbs above objects. For a supervised CFG parser, this ordering is implicit in the given structure of VP and S constituents, however, it seems likely that to learn attachment order reliably, an unsupervised system will have to model it explicitly.

Our goal in this work is the induction of high-quality, linguistically sensible grammars, not parsing accuracy. We present two systems, one which does not do disambiguation well and one which does not do it at all. Both take tagged but unparsed Penn treebank sentences as input.[1] To whatever degree our systems parse well, it can be taken as evidence that their grammars are sensible, but no effort was taken to improve parsing accuracy directly.

There is no claim that human language acquisition is in any way modeled by the systems described here. However, any success of these methods is evidence of substantial cues present in the data, which could potentially be exploited by humans as well. Furthermore, mistakes made by these systems could indicate points where human acquisition is likely not being driven by these kinds of statistics.

2 Approach

At the heart of any iterative grammar induction system is a method, implicit or explicit, for deciding how to update the grammar. Two linguistic criteria for constituency in natural language grammars form the basis of this work (Radford, 1988):

1. External distribution: A constituent is a sequence of words which appears in various structural positions within larger constituents.

[1] The Penn tag and category sets used in examples in this paper are documented in Manning and Schütze (1999, 413).

2. Substitutability: A constituent is a sequence of words with (simple) variants which can be substituted for that sequence.

To make use of these intuitions, we use a distributional notion of context. Let α be a part-of-speech tag sequence. Every occurence of α will be in some context $x\ \alpha\ y$, where x and y are the adjacent tags or sentence boundaries. The distribution over contexts in which α occurs is called its *signature*, which we denote by $\sigma(\alpha)$.

Criterion 1 regards constituency itself. Consider the tag sequences IN DT NN and IN DT. The former is a canonical example of a constituent (of category PP), while the later, though strictly more common, is, in general, not a constituent. Frequency alone does not distinguish these two sequences, but Criterion 1 points to a distributional fact which does. In particular, IN DT NN occurs in many environments. It can follow a verb, begin a sentence, end a sentence, and so on. On the other hand, IN DT is generally followed by some kind of a noun or adjective.

This example suggests that a sequence's constituency might be roughly indicated by the entropy of its signature, $H(\sigma(\alpha))$. This turns out to be somewhat true, given a few qualifications. Figure 1 shows the actual most frequent constituents along with their rankings by several other measures. Tag entropy by itself gives a list that is not particularly impressive. There are two primary causes for this. One is that uncommon but possible contexts have little impact on the tag entropy value. Given the skewed distribution of short sentences in the treebank, this is somewhat of a problem. To correct for this, let $\sigma_u(\alpha)$ be the uniform distribution over the observed contexts for α. Using $H(\sigma_u(\alpha))$ would have the obvious effect of boosting rare contexts, and the more subtle effect of biasing the rankings slightly towards more common sequences. However, while $H(\sigma(\alpha))$ presumably converges to some sensible limit given infinite data, $H(\sigma_u(\alpha))$ will not, as noise eventually makes all or most counts non-zero. Let u be the uniform distribution over all contexts. The scaled entropy

$$H_s(\sigma(\alpha)) = H(\sigma(\alpha))[H(\sigma_u(\alpha))/H(u)]$$

turned out to be a useful quantity in practice. Multiplying entropies is not theoretically meaningful, but this quantity does converge to $H(\sigma(\alpha))$ given infinite (noisy) data. The list for scaled entropy still has notable flaws, mainly relatively low ranks for common NPs, which does not hurt system performance, and overly high ranks for short subject-verb sequences, which does.

Sequence	Actual	Freq	Entropy	Scaled	Boundary	GREEDY-RE
DT NN	1	2	4	2	1	1
NNP NNP	2	1	-	-	4	2
CD CD	3	9	-	-	-	6
JJ NNS	4	7	3	3	2	4
DT JJ NN	5	-	-	-	10	8
DT NNS	6	-	-	-	9	10
JJ NN	7	3	-	7	6	3
CD NN	8	-	-	-	-	-
IN NN	9	-	-	9	10	-
IN DT NN	10	-	-	-	-	-
NN NNS	-	-	5	6	3	7
NN NN	-	8	-	10	7	5
TO VB	-	-	1	1	-	-
DT JJ	-	6	-	-	-	-
MD VB	-	-	10	-	-	-
IN DT	-	4	-	-	-	-
PRP VBZ	-	-	-	-	8	9
PRP VBD	-	-	-	-	5	-
NNS VBP	-	-	2	4	-	-
NN VBZ	-	10	7	5	-	-
RB IN	-	-	8	-	-	-
NN IN	-	5	-	-	-	-
NNS VBD	-	-	9	8	-	-
NNS IN	-	-	6	-	-	-

Figure 1: Top non-trivial sequences by actual constituent counts, raw frequency, raw entropy, scaled entropy, boundary scaled entropy, and according to GREEDY-RE (see section 4.2).

The other fundamental problem with these entropy-based rankings stems from the context features themselves. The entropy values will change dramatically if, for example, all noun tags are collapsed, or if functional tags are split. This dependence on the tagset for constituent identification is very undesirable. One appealing way to remove this dependence is to distinguish only two tags: one for the sentence boundary (#) and another for words. Scaling entropies by the entropy of this reduced signature produces the improved list labeled "Boundary." This quantity was not used in practice because, although it is an excellent indicator of NP, PP, and intransitive S constituents, it gives too strong a bias against other constituents. However, neither system is driven exclusively by the entropy measure used, and duplicating the above rankings more accurately did not always lead to better end results.

Criterion 2 regards the similarity of sequences. Assume the data were truly generated by a categorically unambiguous PCFG (i.e., whenever a token of a sequence is a constituent, its label is determined) and that we were given infinite data. If so, then two sequences, restricted to those occurrences where they are constituents, would have the same signatures. In practice, the data is finite, not statistically context-free, and even short sequences can be categorically ambiguous. However, it remains true that similar raw signatures indicate similar syntactic behavior. For example, DT JJ NN and DT NN have extremely similar signatures, and both are common NPs. Also, NN IN and NN NN IN have very similar signatures, and both are primarily non-constituents.

For our experiments, the metric of similarity between sequences was the Jensen-Shannon divergence of the sequences' signatures:

$$D_{JS}(\sigma_1, \sigma_2) = \frac{1}{2}[D_{KL}(\sigma_1|\frac{\sigma_1+\sigma_2}{2}) + D_{KL}(\sigma_2|\frac{\sigma_1+\sigma_2}{2})]$$

Where D_{KL} is the Kullback-Leibler divergence between probability distributions. Of course, just as various notions of context are possible, so are various metrics between signatures. The issues of tagset dependence and data skew did not seem to matter for the similarity measure, and unaltered Jensen-Shannon divergence was used.

Given these ideas, section 4.1 discusses a system whose grammar induction steps are guided by sequence entropy and interchangeability, and section 4.2 discusses a maximum likelihood system where the objective being maximized is the quality of the constituent/non-constituent distinction, rather than the likelihood of the sentences.

2.1 Problems with ML/MDL

Viewing grammar induction as a search problem, there are three principal ways in which one can induce a "bad" grammar:

- Optimize the wrong objective function.
- Choose bad initial conditions.
- Be too sensitive to initial conditions.

Our current systems primarily attempt to address the first two points. Common objective functions include maximum likelihood (ML) which asserts that a good grammar is one which best encodes or compresses the given data. This is potentially undesirable for two reasons. First, it is strongly data-dependent. The grammar G which maximizes $P(D|G)$ depends on the corpus D, which, in some sense, the core of a given language's phrase structure should not. Second, and more importantly, in an ML approach, there is pressure for the symbols and rules in a PCFG to align in ways which maximize the truth of the conditional independence assumptions embodied by that PCFG. The symbols and rules of a natural language grammar, on the other hand, represent syntactically and semantically coherent units, for which a host of linguistic arguments have been made (Radford, 1988). None of these arguments have anything to do with conditional independence; traditional linguistic constituency reflects only grammatical possibilty of expansion. Indeed, there are expected to be strong connections across phrases (such as are captured by argument dependencies). For example, in the treebank data used, CD CD is a common object of a verb, but a very rare subject. However, a linguist would take this as a selectional characteristic of the data set, not an indication that CD CD is not an NP. Of course, it could be that the ML and linguistic criteria align, but in practice they do not always seem to, and one should not expect that, by maximizing the former, one will also maximize the latter.

Another common objective function is minimum description length (MDL), which asserts that a good analysis is a short one, in that the joint encoding of the grammar and the data is compact. The "compact grammar" aspect of MDL is perhaps closer to some traditional linguistic argumentation which at times has argued for minimal grammars on grounds of analytical (Harris, 1951) or cognitive (Chomsky and Halle, 1968) economy. However, some CFGs which might possibly be seen as the acquisition goal are anything but compact; take the Penn treebank covering grammar for an extreme example. Another serious issue with MDL is that the target grammar is presumably bounded in size, while adding more and more data will on average cause MDL methods to choose ever larger grammars.

In addition to optimizing questionable objective functions, many systems begin their search procedure from an extremely unfavorable region of the grammar space. For example, the randomly weighted grammars in Carroll and Charniak (1992) rarely converged to remotely sensible grammars. As they point out, and quite independently of whether ML is a good objective function, the EM algorithm is only locally optimal, and it seems that the space of PCFGs is riddled with numerous local maxima. Of course, the issue of initialization is somewhat tricky in terms of the bias given to the system; for example, Brill (1994) begins with a uniformly right-branching structure. For English, right-branching structure happens to be astonishingly good both as an initial point for grammar learning and even as a baseline parsing model. However, it would be unlikely to perform nearly as well for a VOS language like Malagasy or VSO languages like Hebrew.

3 Search vs. Clustering

Whether grammar induction is viewed as a search problem or a clustering problem is a matter of perspective, and the two views are certainly not mutually exclusive. The search view focuses on the recursive relationships between the non-terminals in the grammar. The clustering view, which is perhaps more applicable to the present work, focuses on membership of (terminal) sequences to classes represented by the non-terminals. For example, the non-terminal symbol NP can be thought of as a cluster of (terminal) sequences which can be generated starting from NP. This clustering is then inherently soft clustering, since sequences can be ambiguous.

Unlike standard clustering tasks, though, a sequence token in a given sentence need not be a constituent at all. For example, DT NN is an extremely common NP, and when it occurs, it is a constituent around 82% of the time in the data. However, when it occurs as a subsequence of DT NN NN it is usually not a constituent. In fact, the difficult decisions for a supervised parser, such as attachment level or coordination scope, are decisions as to which sequences are constituents, not what their tags would be if they were. For example, DT NN IN DT NN is virtually always an NP when it is a constituent, but it is only a constituent 66% of the time, mostly because the PP, IN DT NN, is attached elsewhere.

One way to deal with this issue is to have an explicit class for "not a constituent" (see section 4.2). There are difficulties in modeling such a class, mainly stemming from the differences between this class and the constituent classes. In particular, this class will not be distributionally cohesive. Also, for example, DT NN and DT JJ NN being generally of category NP seems to be a highly distributional fact, while DT NN not being a constituent in the context DT NN NN seems more properly modeled by the competing productions of the grammar.

Another approach is to model the non-constituents either implicitly or independently of the clustering model (see section 4.1). The drawback to insufficiently modeling non-constituency is that for acquisition systems which essentially work bottom-up, non-constituent chunks such as NN IN or IN DT are hard to rule out locally.

4 Systems

We present two systems. The first, GREEDY-MERGE, learns symbolic CFGs for partial parsing. The rules it learns are of high quality (see figures 3 and 4), but parsing coverage is relatively shallow. The second, CONSTITUENCY-PARSER, learns distributions over sequences representing the probability that a constituent is realized as that sequence (see figure 1). It produces full binary parses.

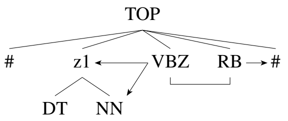

Figure 2: The possible contexts of a sequence.

4.1 GREEDY-MERGE

GREEDY-MERGE is a precision-oriented system which, to a first approximation, can be seen as an agglomerative clustering process over sequences. For each pair of sequences, a normalized divergence is calculated as follows:

$$d(\alpha, \beta) = \frac{D_{JS}(\sigma(\alpha), \sigma(\beta))}{H_s(\sigma(\alpha)) + H_s(\sigma(\beta))}$$

The pair with the least divergence is merged.[2] Merging two sequences involves the creation of a single new non-terminal category which rewrites as either sequence. Once there are non-terminal categories, the definitions of sequences and contexts become slightly more complex. The input sentences are parsed with the previous grammar state, using a shallow parser which ties all parentless nodes together under a TOP root node. Sequences are then the ordered sets of adjacent sisters in this parse, and the context of a sequence can either be the preceding and following tags or a higher node in the tree. To illustrate, in figure 2, the sequence VBZ RB could either be considered to be in context [z1 . . . #] or [NN . . . #]. Taking the highest potential context ([z1 . . . #] in this case) performed slightly better.[3]

Merging a sequence and a single non-terminal results in a rule which rewrites the non-terminal as the sequence (i.e., that sequence is added to that non-terminal's class), and merging two non-terminals involves collapsing the two symbols in the grammar (i.e., those classes are merged). After the merge, re-analysis of the grammar rule RHSs is necessary.

An important point about GREEDY-MERGE is that stopping the system at the correct point is critical. Since our greedy criterion is not a measure over entire grammar states, we have no way to detect the optimal point beyond heuristics (the same

[2]We required that the candidates be among the 250 most frequent sequences. The exact threshold was not important, but without some threshold, long singleton sequences with zero divergence are always chosen. This suggests that we need a greater bias towards quantity of evidence in our basic method.

[3]An option which was not tried would be to consider a non-terminal as a distribution over the tags of the right or left corners of the sequences belonging to that non-terminal.

category appears in several merges in a row, for example) or by using a small supervision set to detect a parse performance drop. The figures shown are from stopping the system manually just before the first significant drop in parsing accuracy.

The grammar rules produced by the system are a strict subset of general CFG rules in several ways. First, no unary rewriting is learned. Second, no nonterminals which have only a single rewrite are ever proposed, though this situation can occur as a result of later merges. The effect of these restrictions is discussed below.

4.2 CONSTITUENCY-PARSER

The second system, CONSTITUENCY-PARSER, is recall-oriented. Unlike GREEDY-MERGE, this system always produces a full, binary parse of each input sentence. However, its parsing behavior is secondary. It is primarily a clustering system which views the data as the entire set of (sequence, context) pairs (α, x) that occurred in the sentences. Each pair token comes from some specific sentence and is classified with a binary judgement c of that token's constituency in that sentence. We assume that these pairs are generated by the following model:

$$P(\alpha, x) = \sum_{c \in \{t,f\}} P(\alpha|c)P(x|c)P(c)$$

We use EM to maximize the likelihood of these pairs given the hidden judgements c, subject to the constraints that the judgements for the pairs from a given sentence must form a valid binary parse.

Initialization was either done by giving initial seeds for the probabilities above or by forcing a certain set of parses on the first round. To do the reestimation, we must have some method of deciding which binary bracketing to prefer. The chance of a pair (α, x) being a constituent is

$$P(c|\alpha, x) = P(c|\alpha)P(c|x)/P(c)$$

and we score a tree T by the likelihood product of its judgements $c(\alpha, T)$. The best tree is then

$$\arg\max_T (\prod_{(\alpha,x) \in s} P(c(\alpha, T)|\alpha, x)$$

As we are considering each pair independently from the rest of the parse, this model does not correspond to a generative model of the kind standardly associated with PCFGs, but can be seen as a random field over the possible parses, with the features being the sequences and contexts (see (Abney, 1997)). However, note that we were primarily interested in the clustering behavior, not the parsing behavior, and that the random field parameters have not been fit to any distribution over trees. The parsing model is very crude, primarily serving to eliminate systematically mutually incompatible analyses.

4.2.1 Sparsity

Since this system does not postulate any non-terminal symbols, but works directly with terminal sequences, sparsity will be extremely severe for any reasonably long sequences. Substantial smoothing was done to all terms; for the $P(c|\alpha)$ estimates we interpolated the previous counts equally with a uniform $P(c)$, otherwise most sequences would remain locked in their initial behaviors. This heavy smoothing made rare sequences behave primarily according to their contexts, removed the initial invariance problem, and, after a few rounds of re-estimation, had little effect on parser performance.

4.2.2 Parameters

CONSTITUENCY-PARSER's behavior is determined by the initialization it is given, either by initial parameter estimates, or fixed first-round parses. We used four methods: RANDOM, ENTROPY, RIGHTBRANCH, and GREEDY.

For RANDOM, we initially parsed randomly. For ENTROPY, we weighted $P(c|\alpha)$ proportionally to $H_s(\sigma(\alpha))$. For RIGHTBRANCH, we forced right-branching structures (thereby introducing a bias towards English structure). Finally, GREEDY used the output from GREEDY-MERGE (using the grammar state in figure 3) to parse initially.

5 Results

Two kinds of results are presented. First, we discuss the grammars learned by GREEDY-MERGE and the constituent distributions learned by CONSTITUENCY-PARSER. Then we apply both systems to parsing free text from the WSJ section of the Penn treebank.

5.1 Grammars learned by GREEDY-MERGE

Figure 3 shows a grammar learned at one stage of a run of GREEDY-MERGE on the sentences in the WSJ section of up to 10 words after the removal of punctuation ($\approx$ 7500 sentences). The non-terminal categories proposed by the systems are internally given arbitrary designations, but we have relabeled them to indicate the best recall match for each.

Categories corresponding to NP, VP, PP, and S are learned, although some are split into sub-categories (transitive and intransitive VPs, proper NPs and two

N-bar or zero determiner NP
zNN → NN | NNS
zNN → JJ zNN
zNN → zNN zNN

NP with determiner
zNP → DT zNN
zNP → PRP$ zNN

Proper NP
zNNP → NNP | NNPS
zNNP → zNNP zNNP

PP
zPP → zIN zNN
zPP → zIN zNP
zPP → zIN zNNP

verb groups / intransitive VPs
zV → VBZ | VBD | VBP
zV → MD VB
zV → MD RB VB
zV → zV zRB
zV → zV zVBG

Transitive VPs
(complementation)
zVP → zV JJ
zVP → zV zNP
zVP → zV zNN
zVP → zV zPP

Transitive VPs
(adjunction)
zVP → zRB zVP
ZVP → zVP zPP

Intransitive S
zS → PRP zV
zS → zNP zV
zS → zNNP zV

Transitive S
zSt → zNNP zVP
zSt → zNN zVP
zSt → PRP zVP

Figure 3: A learned grammar.

kinds of common NPs, and so on).[4] Provided one is willing to accept a verb-group analysis, this grammar seems sensible, though quite a few constructions, such as relative clauses, are missing entirely.

Figure 4 shows a grammar learned at one stage of a run when verbs were split by transitivity. This grammar is similar, but includes analyses of sentencial coordination and adverbials, and subordinate clauses. The only rule in this grammar which seems overly suspect is zVP → IN zS which analyzes complementized subordinate clauses as VPs.

In general, the major mistakes the GREEDY-MERGE system makes are of three sorts:

- Mistakes of omission. Even though the grammar shown has correct, recursive analyses of many categories, no rule can non-trivially incorporate a number (CD). There is also no analysis for many common constructions.
- Alternate analyses. The system almost invariably forms verb groups, merging MD VB sequences with single main verbs to form verb group constituents (argued for at times by some linguists (Halliday, 1994)). Also, PPs are sometimes attached to NPs below determiners (which is in fact a standard linguistic analysis (Abney, 1987)). It is not always clear whether these analyses should be considered mistakes.
- Over-merging. These errors are the most serious. Since at every step two sequences are merged, the process will eventually learn the grammar where X → X X and X → (any terminal). However, very incorrect merges are sometimes made relatively early on (such as merging VPs with PPs, or merging the sequences IN NNP IN and IN.

[4]Splits often occur because unary rewrites are not learned in the current system.

N-bar or zero-determiner NP
zNN → NN | NNS
zNN → zNN zNN
zNN → JJ zNN

Common NP with determiner
zNP → DT zNN
zNP → PRP$ zNN

Proper NP
zNNP → zNNP zNNP
zNNP → NNP

PP
zPP → zIN zNN
zPP → zIN zNP
zPP → zIN zNNP

Transitive Verb Group
zVt → VBZt | VBDt | VBPt
zVt → MD zVBt
zVt → zVt RB

Intransitive Verb Group
zVP → VBZ | VBD | VBP
zVP → MD VB
zVP → zVP zVBN [1]

VP adjunction
zVP → RB zVP
zVP → zVP RB
zVP → zVP zPP
zVP → zVP zJJ

VP complementation
zVP → zVt zNP
zVP → zVt zNN

S
zS → zNNP zVP
zS → zNN zVP
zS → zNP zVP
zS → DT zVP

zS → CC zS
zS → RB zS

S-bar
zVP → IN zS [2]

1 - wrong attachment level
2 - wrong result category

Figure 4: A learned grammar (with verbs split).

5.2 CONSTITUENCY-PARSER's Distributions

The CONSTITUENCY-PARSER's state is not a symbolic grammar, but estimates of constituency for terminal sequences. These distributions, while less compelling a representation for syntactic knowledge than CFGs, clearly have significant facts about language embedded in them, and accurately learning them can be seen as a kind of acquisiton.

Figure 5 shows the sequences whose constituency counts are most incorrect for the GREEDY-RE setting. An interesting analysis given by the system is the constituency of NNP POS NN sequences as NNP (POS NN) which is standard in linguistic analyses (Radford, 1988), as opposed to the treebank's systematic (NNP POS) NN. Other common errors, like the overcount of JJ NN or JJ NNS are partially due to parsing inside NPs which are flat in the treebank (see section 5.3).

It is informative to see how re-estimation with CONSTITUENCY-PARSER improves and worsens the GREEDY-MERGE initial parses. Coverage is improved; for example NPs and PPs involving the CD tag are consistently parsed as constituents while GREEDY-MERGE did not include them in parses at all. On the other hand, the GREEDY-MERGE sys-

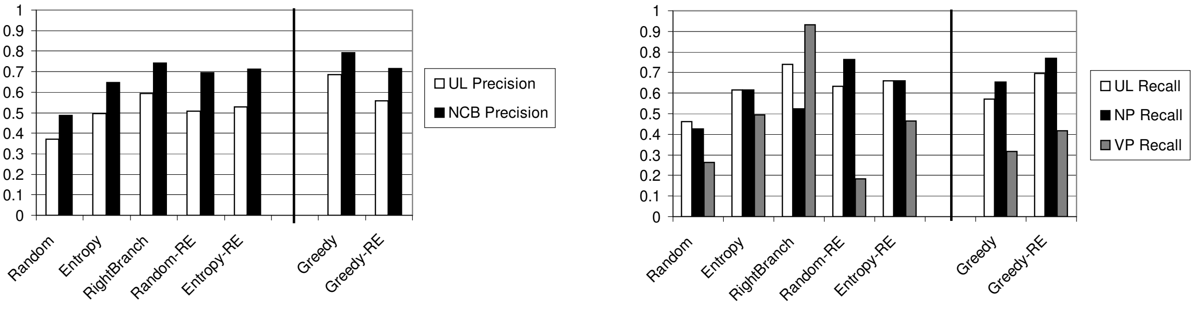

Figure 6: Unlabeled precision (left) and recall (right) values for various settings.

Sequence	Overcount	Estimated	True	Total
JJ NN	736	1099	363	1385
NN NN	504	663	159	805
NNP NNP	434	1419	985	2261
PRP VBZ	420	453	33	488
PRP VBD	392	415	23	452
PRP VBP	388	405	17	440
TO VB	324	443	119	538
MD VB	318	355	37	455
NN NNS	283	579	296	618
JJ NNS	283	799	516	836

Sequence	Undercount	Estimated	True	Total
NNP POS	127	33	160	224
VBD RB VBN	59	6	65	83
VB DT NN	53	10	63	137
NNP NNP POS	42	8	50	58
VB VBN	42	3	45	141
VB RB	39	6	45	100
VBD VBN	36	17	53	202
VBZ RB JJ	33	18	51	72
RB CD	30	26	56	117
VB DT JJ NN	29	3	32	51

Figure 5: Sequences most commonly over- and under-identified as constituents by CONSTITUENCY-PARSER using GREEDY-RE (ENTROPY-RE is similar). "Total" is the frequency of the sequence in the flat data. "True" is the frequency as a constituent in the treebank's parses. "Estimated" is the frequency as a constituent in the system's parses.

tem had learned the standard subject-verb-object attachment order, though this has disappeared, as can be seen in the undercounts of VP sequences. Since many VPs did not fit the conservative VP grammar in figure 3, subjects and verbs were often grouped together frequently even on the initial parses, and the CONSTITUENCY-PARSER has a further bias towards over-identifying frequent constituents.

5.3 Parsing results

Some issues impact the way the results of parsing treebank sentences should be interpreted. Both systems, but especially the CONSTITUENCY-PARSER, tend to form verb groups and often attach the subject below the object for transitive verbs. Because of this, certain VPs are systematically incorrect and VP accuracy suffers dramatically, substantially pulling down the overall figures.[5] Secondly, the treebank's grammar is an imperfect standard for an unsupervised learner. For example, transitive sentences are bracketed [subject [verb object]] ("The president [executed the law]") while nominalizations are bracketed [[possessive noun] complement] ("[The president's execution] of the law"), an arbitrary inconsistency which is unlikely to be learned automatically. The treebank is also, somewhat purposefully, very flat. For example, there is no analysis of the inside of many short noun phrases. The GREEDY-MERGE grammars above, however, give a (correct) analysis of the insides of NPs like DT JJ NN NN for which it will be penalized in terms of unlabeled precision (though not crossing brackets) when compared to the treebank.

An issue with GREEDY-MERGE is that the grammar learned is symbolic, not probabilistic. Any disambiguation is done arbitrarily. Therefore, even adding a linguistically valid rule can degrade numerical performance (sometimes dramatically) by introducing ambiguity to a greater degree than it improves coverage.

In figure 6, we report summary results for each system on the $\leq$ 10-word sentences of the WSJ section. GREEDY is the above snapshot of the GREEDY-MERGE system. RANDOM, ENTROPY, and RIGHTBRANCH are the behaviors of the random-parse baseline, the right-branching baseline, and the entropy-scored initialization for CONSTITUENCY-PARSER. The -RE settings are the result of context-based re-estimation from the respective baselines using CONSTITUENCY-PARSER.[6] NCB precision is the percentage of pro-

[5]The RIGHTBRANCH baseline is in the opposite situation. Its high overall figures are in a large part due to extremely high VP accuracy, while NP and PP accuracy (which is more important for tasks such as information extraction) is very low.

[6]RIGHTBRANCH was invariant under re-estimation, and RIGHTBRANCH-RE is therefore omitted.

posed brackets which do not cross a correct bracket. Recall is also shown separately for VPs and NPs to illustrate the VP effect noted above.

The general results are encouraging. GREEDY is, as expected, higher precision than the other settings. Re-estimation from that initial point improves recall at the expense of precision. In general, re-estimation improves parse accuracy, despite the indirect relationship between the criterion being maximized (constituency cluster fit) and parse quality.

6 Limitations of this study

This study presents preliminary investigations and has several significant limitations.

6.1 Tagged Data

A possible criticism of this work is that it relies on part-of-speech tagged data as input. In particular, while there has been work on acquiring parts-of-speech distributionally (Finch et al., 1995; Schütze, 1995), it is clear that manually constructed tag sets and taggings embody linguistic facts which are not generally detected by a distributional learner. For example, transitive and intransitive verbs are identically tagged yet distributionally dissimilar.

In principle, an acquisition system could be designed to exploit non-distributionality in the tags. For example, verb subcategorization or selection could be induced from the ways in which a given lexical verb's distribution differs from the average, as in (Resnik, 1993). However, rather than being exploited by the systems here, the distributional non-unity of these tags appears to actually degrade performance. As an example, the systems more reliably group verbs and their objects together (rather than verbs and their subjects) when transitive and intransitive verbs are given separate tags.

Future experiments will investigate the impact of distributional tagging, but, despite the degradation in tag quality that one would expect, it is also possible that some current mistakes will be corrected.

6.2 Individual system limitations

For GREEDY-MERGE, the primary limitations are that there is no clear halting condition, there is no ability to un-merge or to stop merging existing classes while still increasing coverage, and the system is potentially very sensitive to the tagset used. For CONSTITUENCY-PARSER, the primary limitations are that no labels or recursive grammars are learned, and that the behavior is highly dependent on initialization.

7 Conclusion

We present two unsupervised grammar induction systems, one of which is capable of producing declarative, linguistically plausible grammars and another which is capable of reliably identifying frequent constituents. Both parse free text with accuracy rivaling that of weakly supervised systems. Ongoing work includes lexicalization, incorporating unary rules, enriching the models learned, and addressing the limitations of the systems.

References

Stephen P. Abney. 1987. *The English Noun Phrase in its Sentential Aspect*. Ph.D. thesis, MIT.

Steven P. Abney. 1997. Stochastic attribute-value grammars. *Computational Linguistics*, 23(4):597–618.

E. Brill. 1994. Automatic grammar induction and parsing free text: A transformation-based approach. In *Proc. ARPA Human Language Technology Workshop '93*, pages 237–242, Princeton, NJ.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In Carl Weir, Stephen Abney, Ralph Grishman, and Ralph Weischedel, editors, *Working Notes of the Workshop Statistically-Based NLP Techniques*, pages 1–13. AAAI Press, Menlo Park, CA.

Eugene Charniak. 1993. *Statistical Language Learning*. MIT Press, Cambridge, MA.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *NAACL 1*, pages 132–139.

Noam Chomsky and Morris Halle. 1968. *The Sound Pattern of English*. Harper & Row, New York.

Michael John Collins. 1997. Three generative, lexicalised models for statistical parsing. In *ACL 35/EACL 8*, pages 16–23.

Steven P. Finch, Nick Chater, and Martin Redington. 1995. Acquiring syntactic information from distributional statistics. In J. Levy, D. Bairaktaris, J. A. Bullinaria, and P. Cairns, editors, *Connectionist models of memory and language*, pages 229–242. UCL Press, London.

M. A. K. Halliday. 1994. *An introduction to functional grammar*. Edward Arnold, London, 2nd edition.

Zellig Harris. 1951. *Methods in Structural Linguistics*. University of Chicago Press, Chicago.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Boston, MA.

Andrew Radford. 1988. *Transformational Grammar*. Cambridge University Press, Cambridge.

Philip Stuart Resnik. 1993. *Selection and Information: A Class-Based Approach to Lexical Relationships*. Ph.D. thesis, University of Pennsylvania.

Hinrich Schütze. 1995. Distributional part-of-speech tagging. In *EACL 7*, pages 141–148.

Andreas Stolcke and Stephen M. Omohundro. 1994. Inducing probabilistic grammars by Bayesian model merging. In *Grammatical Inference and Applications: Proceedings of the Second International Colloquium on Grammatical Inference*. Springer Verlag.