Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency

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

Abstract

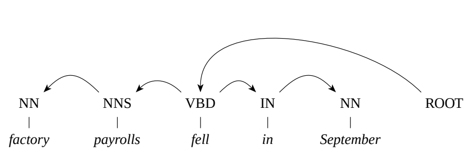
We present a generative model for the *unsupervised learning* of dependency structures which to our knowledge is the first to produce above-baseline dependency structures. We also describe the multiplicative combination of this dependency model with a model of linear constituency. The product model outperforms both components on their respective evaluation metrics, giving the best published figures for unsupervised dependency parsing *and* unsupervised constituency parsing. We also demonstrate that the combined model works and is robust cross-linguistically, being able to exploit either attachment or distributional regularities that are salient in the data.

1 Introduction

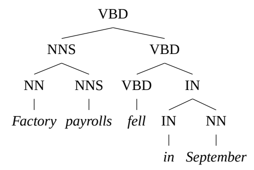
The task of statistically inducing hierarchical syntactic structure over unannotated sentences of natural language has received a great deal of attention (Carroll and Charniak, 1992a; Pereira and Schabes, 1992; Brill, 1993; Stolcke and Omohundro, 1994). Researchers have explored this problem for a variety of reasons: to argue empirically against the poverty of the stimulus (Clark, 2001), to use induction systems as a first stage in constructing large treebanks (van Zaanen, 2000), to build better language models (Baker, 1979; Chen, 1995), and to examine cognitive issues in language learning (Solan et al., 2003). An important distinction should be drawn between work primarily interested in the weak generative capacity of models, where modeling hierarchical structure is only useful insofar as it leads to improved models over observed structures (Baker, 1979; Chen, 1995), and work interested in the strong generative capacity of models, where the unobserved structure itself is evaluated (van Zaanen, 2000; Clark, 2001; Klein and Manning, 2002). This paper falls into the latter category; we will be inducing models of linguistic constituency and dependency with the goal of recovering linguistically plausible structures. We make no claims as to the congitive plausibility of the induction mechanisms we present here, however the ability of these systems to recover substantial linguistic patterns from surface yields alone does speak to the strength of support for these patterns in the data, and hence to undermine arguments based on “the poverty of the stimulus” (Chomsky, 1965).

2 Unsupervised Dependency Parsing

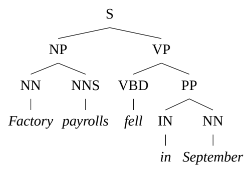
Most recent progress in unsupervised parsing has come from tree or phrase-structure grammar based models (Clark, 2001; Klein and Manning, 2002), but there are compelling reasons to reconsider unsupervised *dependency* parsing. First, most state-of-the-art *supervised* parsers make use of specific lexical information in addition to word-class level information – perhaps lexical information could be a useful source of information for unsupervised methods. Second, a central motivation for using tree structures in computational linguistics is to enable the extraction of dependencies – function-argument and modification structures – from them, and it might be more advantageous to induce such structures directly. Third, as we show below, for



(a) Classical Dependency Structure

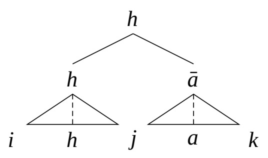


(b) Dependency Structure as Tree

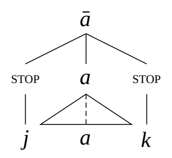


(c) PCFG Structure

Figure 1: Three kinds of parse structures.



(a)



(b)

Figure 2: Dependency configuration in a lexicalized tree. h and a are head and argument words, respectively, while i, j, and k are positions between words.

languages such as Chinese, which have few function words and for which the definition of lexical categories is much less clear, dependency structures may be easier to detect.

A dependency representation of a short sentence is shown in figure 1(a), where following the traditional dependency grammar notation, the regent or head of a dependency is marked with the tail of the dependency arrow, and the dependent is marked with the arrowhead (Mel′čuk, 1988). It will be important in what follows to see that such a representation is isomorphic (in terms of strong generative capacity) to a restricted form of phrase structure grammar, where the set of terminals and nonterminals is identical, and every rule is of the form X → X Y or X → Y X (Miller, 1999), giving the isomorpic representation of figure 1(a) shown in figure 1(b). Depending on the model, part-of-speech categories may be included in the dependency representation, as shown here, or dependencies may be directly between words. Below, we will assume an additonal reserved nonterminal ROOT, whose sole dependent is the head of the sentence. This simplifies the notation, math, and the evaluation metric. A dependency analysis will always consist of exactly as many dependencies as there are words in the sentence. For example, in the dependency structure of figure 1(b), the dependencies are {(ROOT, *fell*, ←), (*fell*, *payrolls*, →), (*fell*, *in*, →), (*in*, *September*, →), (*payrolls*, *factory*, ←)}. Its quality can be evaluated against a gold standard dependency analysis, simply by reporting a single accuracy figure which is the percentage of dependencies shared between the two analyses.

The dependency induction task has received less attention; the best known work is Carroll and Charniak (1992b), Yuret (1998), and Paskin (2002). All systems that we are aware of operate under the assumption that the probability of a dependency structure is the product of the scores of the dependencies in that structure, seen as ordered (head, dependent) pairs of words, possibly also considering a direction dir (whether the arrow points left or right). For instance, the model of Paskin (2002), which is broadly similar to the semi-probabilistic model in Yuret (1998), first chooses a dependency structure D uniformly at random, then populates it with words, starting with a fixed root symbol (assumed to be at the end of the sentence), and working down the dependency tree. The corresponding probabilistic model is

$$\mathrm{P}(S, D) = \mathrm{P}(D) \prod_{(i,j,dir)\in D} \mathrm{P}({}_{i-1}s_i|{}_{j-1}s_j, dir) .$$

Figure 3 shows accuracies for various dependency induction models on four different corpora. WSJ is the entire Penn English Treebank WSJ portion. WSJ10 is the subset of sentences which contained 10 words or less after the removal of punctuation. CTB10 is the sentences of the same length

Model	Dir.	Undir.
English (WSJ)		
Paskin 01		39.7
RANDOM		41.7
Charniak and Carroll 92-inspired		44.7
ADJACENT		53.2
DMV		54.4
English (WSJ10)		
RANDOM	30.1	45.6
ADJACENT	33.6	56.7
DMV	43.2	63.7
German (NEGRA10)		
RANDOM	21.8	41.5
ADJACENT	32.6	51.2
DMV	36.3	55.8
Chinese (CTB10)		
RANDOM	35.9	47.3
ADJACENT	30.2	47.3
DMV	42.5	54.2

Figure 3: Parsing performance (directed and undirected dependency accuracy) of various dependency models on various treebanks, along with baselines.

from the Penn Chinese treebank. NEGRA10 is the same, for the German NEGRA corpus, based on their conversion of the NEGRA corpus into Penn treebank format. In all these experiments, the provided parts-of-speech were used as the input alphabet. Where possible, we report an accuracy figure for both directed and undirected dependencies. Reporting undirected numbers has two advantages; first, it facilitates comparison with earlier work, and, more importantly, it allows one to partially obscure the effects of alternate analyses, such as the systematic choice between a modal and a main verb for the head of a sentence (in either case, the two verbs would be linked, but the direction would vary).[1]

In Paskin (2002), the model above was trained on over 30M words of raw newswire, using EM, yet the resulting parser predicted dependencies at below chance level (measured by choosing a random dependency structure). This below-random performance seems to be because the model links word pairs which have high mutual information (such as occurrences of *congress* and *bill* regardless of whether they are plausibly syntactically related). In practice, high mutual information between words is often stronger between two topically similar nouns than between, say, a preposition and its object.

One might hope that the problem is that actual lexical items are too semantically charged to represent workable units of syntactic structure. If one were to apply the Paskin (2002) model to dependency structures parameterized simply on the word-classes, the result is isomorphic to the "dependency PCFG" models described in Carroll and Charniak (1992b), where they considered PCFGs with precisely the productions discussed above that make them isomorphic to dependency grammars, with the terminal alphabet being simply parts-of-speech. Here, the rule probabilities are equivalent to $\mathrm{P}(Y|X, \mathit{right})$ and $\mathrm{P}(Y|X, \mathit{left})$ respectively. The actual experiments in Carroll and Charniak (1992b) do not report accuracies that we can compare to, but they suggest that the learned grammars were of extremely poor quality. The main issue in their experiments appears to be that they randomly initialized the production (attachment) probabilities. As a result, their learned grammars were of very poor quality and had high variance. However, one nice property of their structural constraint, which all dependency models share, is that the symbols in the grammar are not symmetric. Even with a grammar in which the productions are initially uniform, a symbol X can only possibly have non-zero posterior likelihood over spans which contain a matching terminal X. Therefore, one can start with uniform rewrites and let the interaction between the data and the model structure break the initial symmetry. If one recasts their experiments in this way, they acheive an accuracy of 44.7% on the Penn treebank, which is higher than choosing a random dependency structure, but lower than simply linking all adjacent words into a left-headed (and right-branching) structure (53.2%).

A huge limitation of the above model is that it is incapable of encoding even first-order valence facts. For example, it learns that nouns to the left of the verb (usually subjects) attach to the verb. But then, given a *noun noun verb* sequence, both nouns will attach to the verb – there is no way that the model can learn the fact that verbs have exactly one subject. We now turn to an improved depen-

[1]The Penn treebank does *not* include dependency annotations, however the automatic dependency rules from (Collins, 1999) are sufficiently accurate to be a good benchmark for unsupervised systems for the time being. Similar head-finding rules were used for Chinese and German experiments.

dency model that addresses this problem.

3 An Improved Dependency Model

The dependency models discussed above are distinct from dependency models used inside high-performance supervised probabilistic parsers in several ways. First, in supervised models, a head outward process is modeled (Eisner, 1996; Collins, 1999). In such processes, heads generate a sequence of arguments outward to the left or right, conditioning on not only the identity of the head and direction of the attachment, but also on some notion of distance or valence. Moreover, in a head-outward model, it is natural to model stop steps, where the final argument on each side of a head is always the special symbol STOP.[2] Previous work (Collins, 1999) has stressed the importance of include such termination probabilities, precisely because it does allow the modeling of required dependents.

We propose the maximally simple head-outward dependency model over word classes which includes a model of valence, which we call *DMV*. We begin at the ROOT. In the standard way, each head generates a series of non-STOP arguments to one side, then a STOP argument to that side, then a series of non-STOP arguments to the other side, then a second STOP.

In the DMV model, the probability of generating an argument a for a head h is conditioned not only on h, but also on the signed distance d between the location of the head h and the farthest extent of the argument subtrees that h has generated so far. In figure 2, this is $j - h$. The probability $P(a|h, d)$ is broken down into two steps. First, we choose whether a will be STOP based on which side of h we are on ($sign(d)$) and whether h has any arguments on that side ($val(d)$, which is 0 iff $abs(d) = 0$ and 1 otherwise). Then, the actual argument is chosen based only on the head and the direction (and the stop indicator).

$$\mathrm{P}(t) = \prod_{(h,a,d)\in t} \mathrm{P}(a|h, d)$$

$$\begin{aligned} \mathrm{P}(a|h, d) &= \mathrm{P}(stop(a)|h, sign(d), val(d)) \\ &\quad \mathrm{P}(a|h, stop(a), sign(d)) \end{aligned}$$

[2]Models like Paskin (2002) avoid modeling STOP by assuming the dependency graph is first structurally chosen uniformly, then populated with words.

We estimated these two multinomials directly without any further decomposition or smoothing for word-class dependency structures.

One can view a structure generated by this process as a lexicalized tree composed of the local binary and unary configurations shown in figure 2. Figure 2(a) shows the configuration for a head h taking a right argument $\bar{a}$. The bar indicates that a has been *sealed*; it has generated both STOP arguments and cannot generate any more arguments. A cubic dynamic program similar to those presented in Eisner (1996) can be used to iteratively re-estimate this model using EM. At all times, we constrained ROOT to have a single dependent.

Initialization is important to the success of any local search procedure. We chose to initialize EM not with an initial model, but with an initial guess at posterior distributions over dependency structures (completions). For the first-round, we constructed a rather ad-hoc “harmonic” completion where all non-ROOT words took the same number of arguments, and each took other words as arguments in inverse proportion to (a constant plus) the distance between them. The ROOT always had a single argument and took each word with equal probability. This structure had two advantages: first, when testing multiple models, it is easier to start them all off in a common way by beginning with an M-step, and, second, it allowed us to point the model in the vague general direction of what linguistic dependency structures should look like.

On the WSJ10 corpus, the DMV model recovers a substantial fraction of the broad dependency trends: 43.2% of guessed directed dependencies were correct (63.7% ignoring direction). To our knowledge, this is the first published result to break the adjacent-word heuristic (at 33.6% for this corpus). Verbs are the sentence heads, prepositions take following noun phrases as arguments, adverbs attach to verbs, and so on. The most common source of discrepancy between the test dependencies and the model’s guesses is a result of the model systematically choosing determiners as the heads of definite noun phrases, while the test trees have the rightmost noun as the head. The model’s choice is supported by a good deal of linguistic research (Abney, 1987), and is sufficiently systematic that we also report the scores that result when all NPs in the test set containing a determiner are changed to have the determiner as the correct head (this is a simple redefining of the NP headship rule). On this metric, the score jumps hugely to 55.7% directed (and 67.9% undirected).

This model also works on German and Chinese at above-baseline levels (55.8% and 54.2% undirected, respectively), with no modifications whatsoever. In German, the largest source of errors is also the systematic postulation of determiner-headed noun-phrases. In Chinese, the primary mismatch is that subjects are considered to be the heads of sentences rather than verbs.

This dependency induction model is reasonably successful. While one cannot tell this from previously published numbers, we show later that, for the task of dependency recovery, it is much more successful than the constituency model of Klein and Manning (2002). However, our intuition is still that the model can be improved by paying more attention to syntactic constituency. To this end, after briefly recapping the model of Klein and Manning (2002), we present a combined model that exploits dependencies and constituencies. As we will see, this combined model finds correct dependencies more successfully than the model above, and finds constituents more successfully than the model of Klein and Manning (2002).

4 Distributional Constituency Induction

In linear distributional clustering, items (e.g., words or word sequences) are represented by characteristic distributions over their linear contexts (e.g., multinomial models over the preceding and following words, see figure 4). These context distributions are then clustered in some way, often using standard data clustering methods. In the most common case, the items are words, and one uses distributions over adjacent words to induce word classes. Previous work has shown that even this quite simple representation allows the induction of quite high quality word classes, largely corresponding to traditional parts of speech (Finch, 1993; Schütze, 1995; Clark, 2000). A typical pattern would be that *stocks* and *treasuries* both frequently occur before the words *fell* and *rose*, and might therefore be put into the same class.

Clark (2001) and Klein and Manning (2002) show that this approach can be successfully used for discovering syntactic constituents as well. However, as one might expect, it is easier to cluster word sequences (or word class sequences) than to tell how to put them together into trees. In particular, if one is given all contiguous subsequences (*subspans*) from a corpus of sentences, most natural clusters will not represent valid constituents (to the extent that constituency of a non-situated sequence is even a well-formed notion). For example, it is easy enough to discover that DET N and DET ADJ N are similar and that V PREP DET and V PREP DET ADJ are similar, but it is much less clear how to discover that the former pair are generally constituents while the latter pair are generally not. In (Klein and Manning, 2002), we propose a *constituent-context model* (CCM) which solves this problem by building constituency decisions directly into the distributional model, by earmarking a single cluster d for non-constituents. During the calculation of cluster assignments, only a non-crossing subset of the observed word sequences can be assigned to other, constituent clusters. This integrated approach is empirically successful.

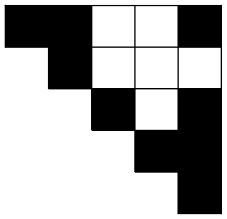
The CCM works as follows. Sentences are given as sequences S of word classes (parts-of-speech or otherwise). One imagines each sentence as a list of the $O(n^2)$ subspans (index pairs) $\langle i, j\rangle$, each followed by the corresponding subsequence ${}_is_j$ and linear context ${}_{i-1}s_i \sim {}_js_{j+1}$ (see figure 4). The model generates all constituent-context pairs, span by span.

The first stage is to choose a *bracketing* B for the sentence, which is a maximal non-crossing subset of the spans (equivalent to a binary tree). In the basic model, $P(B)$ is uniform over binary trees. Then, for each $\langle i, j\rangle$, the subsequence and context pair ${}_is_j, {}_{i-1}s_i \sim {}_js_{j+1}$ is generated via a class-conditional independence model:

$$\mathrm{P}(S, B) = \mathrm{P}(B)\prod_{\langle i,j\rangle}\mathrm{P}({}_is_j|b_{ij})\mathrm{P}({}_{i-1}s_i \sim {}_js_{j+1}|b_{ij})$$

That is, all spans guess their sequences and contexts given only a constituency decision b.[3]

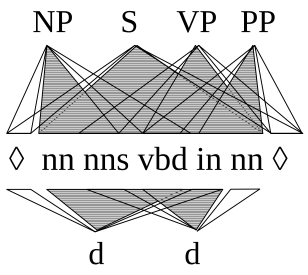
[3]As is typical of distributional clustering, positions in the



(a)

Span	Label	Constituent	Context
⟨0,5⟩	S	NN NNS VBD IN NN	◇ – ◇
⟨0,2⟩	NP	NN NNS	◇ – VBD
⟨2,5⟩	VP	VBD IN NN	NNS – ◇
⟨3,5⟩	PP	IN NN	VBD – ◇
⟨0,1⟩	NN	NN	◇ – NNS
⟨1,2⟩	NNS	NNS	NN – VBD
⟨2,3⟩	VBD	VBD	NNS – IN
⟨3,4⟩	IN	IN	VBD – NN
⟨4,5⟩	NN	NNS	IN – ◇

(b)



(c)

Figure 4: The generative process. (a) A binary tree-equivalent bracketing is chosen at random. For the two-class case considered here, this also counts as a labeling of the constituents and distituents with their category, *c* and *d* respectively. (b) and (c) *Each* span generates its yield and context (empty spans not shown here). Derivations which are not coherent are given mass zero.

This is a model $\mathrm{P}(S, B)$ over hidden bracketings and observed sentences, and it is estimated via EM to maximize the sentence likelihoods $\mathrm{P}(S)$ over the training corpus. Figure 5 shows the accuracy of the *CCM* model not only on English but for the Chinese and German corpora discussed above.[4] Results are reported at convergence; for the English case, F_1 is monotonic during training, while for the others, there is an earlier peak.

Also shown is an upper bound (the target trees are not all binary and so any all-binary system will over-propose constituents). Klein and Manning (2002) gives comparative numbers showing that the basic CCM outperforms other recent system on the ATIS corpus (which many other constituency induction systems have reported on). While absolute numbers are hard to compare across corpora, all the systems compared to in Klein and Manning (2002) parsed below a right-branching baseline, while the CCM is substantially above it.

5 A Combined Model

The two models described above have some common ground. Both can be seen as models over lexicalized trees composed of the configurations in figure 2. For the DMV, it is already a model over these structures (though the cubic dynamic program exploits a more refined decomposition). At the "attachment" rewrite for the CCM in (a), we assign the quantity:

$$\frac{\mathrm{P}({}_i s_k | true)\mathrm{P}({}_{i-1}s_i \sim {}_k s_{k+1} | true)}{\mathrm{P}({}_i s_k | \, false)\mathrm{P}({}_{i-1}s_i \sim {}_k s_{k+1} | \, false)}$$

which is the odds ratio of generating the subsequence and context for span $\langle i, j \rangle$ as a constituent as opposed to a distituent. If we multiply all trees' attachment scores by

$$\prod\nolimits_{\langle i,j \rangle} \mathrm{P}({}_i s_j | \, false)\mathrm{P}({}_{i-1}s_i \sim {}_j s_{j+1} | \, false)$$

the denominators of the odds ratios cancel, and we are left with each tree being assigned the probability it would have received under the CCM. However, this scoring function will not properly distribute the CCM mass across the various lexicalized trees that share the same unlabeled backbone (rather, each receives the same score). To correct for this, we multiply in a "head choice" factor of $1/(k - j)$ at each "sealing" configuration (b). The resulting scoring over configurations has the property that if one calculates the fraction of trees which contain a bracket $\langle i, j \rangle$ with head h ($\mathrm{P}(h, i, j|S)$), and sums out the h, one recovers the same fraction of sentences that the CCM claimed contained that unlabeled bracket.

In this way, both models can be asked to generate either constituency or dependency structures. Of course, the CCM will generate fairly random dependency structures (constrained only by bracketings). Getting constituency structures from the DMV is also problematic, because the choice of

corpus can get generated multiple times. Since derivations need not be consistent, the entire model is mass deficient when viewed as a model over sentences.

[4]Klein and Manning (2002) reported results using unlabeled bracketing statistics which gave no credit for brackets which spanned the entire sentence (raising the scores) but macro-averaged over sentences (lowering the scores). The numbers here hew more closely to the standard methods used for evaluating supervised parsers, by being micro-averaged and including full-span brackets. However, the scores are, overall, approximately the same.

which side to first attach arguments on has ramifications on constituency, even though it is an arbitrary convention for pure dependency models. For example, if we attach right arguments first, then a verb with a left subject and a right object will attach the object first, giving traditional VPs, while the other attachment order gives subject-verb groups. To avoid this bias, we alter the DMV for now so that each word has even probability for either order.

In figure 5, we give the behavior of the CCM constituency model and the DMV dependency model on both constituency and dependency induction. Unsurprisingly, their strengths are complementary. The CCM is better at recovering constituency, and the dependency model is better at recovering dependency structures. It is reasonable to hope that a combination model might exhibit the best of both. In the supervised parsing domain, for example, it has been found that scoring a lexicalized tree with the product of a simple lexical dependency model and a PCFG model leads to a combined scoring metric which outperforms each factor on their respective metrics (Klein and Manning, 2003).

In the combined model, we score each tree with the product of the probabilities from the individual models above. We use an $O(n^5)$ dynamic program (a variant of the inside-outside algorithm (Baker, 1979)) to sum over all lexicalized trees. The core operation in the dynamic program is to score the merging of two adjacent constituent signatures, such as $(h : i, j)$ and $(a : j, k)$ into a larger signature $(h : i, k)$ (shown in figure 2). At such a merge, we multiply in the relevant factors from both models. From the CCM we must generate ${}_is_k$ as a constituent and (s_{i-1}, s_{k+1}) as its context (using some constituent class if the CCM factor has more than one).On the same merge, the dependency model will assess two costs: first the cost of h taking a as a head at the distance $j - h$, and, second, the two factors for sealing the argument, $(a : j, k)$ (see figure 2b). There is an accompanying outside phase; we omit the details here. From the results of this dynamic program, we can extract the sufficient statistics needed to re-estimate either individual model.

The models were intitialized in the same way as

Model	UP	UR	UF$_1$	Dir	Undir
English (WSJ10 – 7422 Sentences)					
LBRANCH/RHEAD	25.6	32.6	28.7	33.6	56.7
RANDOM	31.0	39.4	34.7	30.1	45.6
RBRANCH/LHEAD	55.1	70.0	61.7	24.0	55.9
DMV	46.6	59.2	52.1	43.2	62.7
CCM	64.2	81.6	71.9	23.8	43.3
DMV+CCM (POS)	**69.3**	**88.0**	**77.6**	**47.5**	**64.5**
DMV+CCM (DISTR.)	65.2	82.8	72.9	42.3	60.4
UBOUND	78.8	100.0	88.1	100.0	100.0
German (NEGRA10 – 2175 Sentences)					
LBRANCH/RHEAD	27.4	48.8	35.1	32.6	51.2
RANDOM	27.9	49.6	35.7	21.8	41.5
RBRANCH/LHEAD	33.8	60.1	43.3	21.0	49.9
DMV	38.4	69.5	49.5	40.0	57.8
CCM	48.1	85.5	61.6	25.5	44.9
DMV+CCM	**49.6**	**89.7**	**63.9**	**50.6**	**64.7**
UBOUND	56.3	100.0	72.1	100.0	100.0
Chinese (CTB10 – 2437 Sentences)					
LBRANCH/RHEAD	26.3	48.8	34.2	30.2	43.9
RANDOM	27.3	50.7	35.5	35.9	47.3
RBRANCH/LHEAD	29.0	53.9	37.8	14.2	41.5
DMV	**35.9**	**66.7**	**46.7**	42.5	54.2
CCM	34.6	64.3	45.0	23.8	40.5
DMV+CCM	33.3	62.0	43.3	**55.2**	**60.3**
UBOUND	53.9	100.0	70.1	100.0	100.0

Figure 5: Parsing performance of the combined model on various treebanks, along with baselines.

when they were run individually. Sufficient statistics were taken off these completions, then the resulting models were used together from then on during re-estimation.

Figure 5 summarizes the results. The combined model beats the CCM on English F$_1$: 77.6 vs. 71.9. The figure also shows the combination model's score when using word classes which were induced entirely automatically, using the same method as Klein and Manning (2002). These classes show some degradation, e.g. 72.9 F$_1$, but, it is worth noting that these totally unsupervised numbers are better than the performance of the CCM model of Klein and Manning (2002) running off of Penn treebank word classes. Again, if we modify the gold standard so as to make determiners the head of NPs, then this model with distributional tags scores 50.6% on directed and 64.8% on undirected dependency accuracy.

On the German data, the combination again outperforms each factor alone, though while the combination was most helpful at boosting constituency quality for English, for German it provided a larger boost to the dependency structures. Finally, on the Chinese data, the combination did substantially boost dependency accuracy over either single factor, but actually suffered a small drop in constituency.[5] Overall, the combination is able to combine the individual factors in an effective way.

6 Conclusion

We have presented a successful new dependency-based model for the unsupervised induction of syntactic structure, which picks up the key ideas that have made dependency models successful in supervised statistical parsing work. We proceeded to show that it works cross-linguistically. We then demonstrated how this model could be combined with the previous best constituent-induction model to produce a combination which, in general, substantially outperforms either individal model, on either metric. The key reason that these models are capable of recovering structure more accurately than previous work is that they minimize the amount of hidden structure that must be induced. In particular, neither model attempts to learn intermediate, recursive categories with no direct connection to surface statistics. Our results here are just on the ungrounded induction of syntactic structure. Nonetheless, we see the investigation of what patterns can be recovered from corpora as important, both from a computational perspective and from a philosophical one. It demonstrates that the broad constituent and dependency structure of a language can be recovered quite successfully from a very modest amount of training data, individually or, more effectively, jointly.

References

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

James K. Baker. 1979. Trainable grammars for speech recognition. In D. H. Klatt and J. J. Wolf, editors, *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pages 547–550.

Eric Brill. 1993. Automatic grammar induction and parsing free text: A transformation-based approach. In *ACL 31*, pages 259–265.

[5]This partially seems to be due to the large number of unanalyzed fragments in the Chinese gold standard, which leave a very large fraction of the posited bracketings completely unjudged.

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

Glenn Carroll and Eugene Charniak. 1992b. 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.

Stanley F. Chen. 1995. Bayesian grammar induction for language modeling. In *ACL 33*, pages 228–235.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Alexander Clark. 2000. Inducing syntactic categories by context distribution clustering. In *The Fourth Conference on Natural Language Learning*.

Alexander Clark. 2001. Unsupervised induction of stochastic context-free grammars using distributional clustering. In *The Fifth Conference on Natural Language Learning*.

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

Jason Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 16*, pages 340–345.

Steven Paul Finch. 1993. *Finding Structure in Language*. Ph.D. thesis, University of Edinburgh.

Dan Klein and Christopher D. Manning. 2002. A generative constituent-context model for improved grammar induction. In *ACL 40*, pages 128–135.

Dan Klein and Christopher D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA. MIT Press.

Pat Langley. 1980. A production system model of first language acquisition. In *Proceedings of the Eighth International Conference on Computational Linguistics*, pages 183–189, Tokyo, Japan.

Igor Aleksandrovich Mel′čuk. 1988. *Dependency Syntax: theory and practice*. State University of New York Press, Albany, NY.

Philip H. Miller. 1999. *Strong Generative Capacity*. CSLI Publications, Stanford, CA.

Donald Cort Olivier. 1968. *Stochastic Grammars and Language Acquisition Mechanisms*. Ph.D. thesis, Harvard University.

Mark A. Paskin. 2002. Grammatical bigrams. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA. MIT Press.

Fernando Pereira and Yves Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *ACL 30*, pages 128–135.

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

Z. Solan, E. Ruppin, D. Horn, and S. Edelman. 2003. Automatic acquisition and efficient representation of syntactic structures. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA. MIT Press.

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.

Menno van Zaanen. 2000. ABL: Alignment-based learning. In *COLING 18*, pages 961–967.

J. Gerard Wolff. 1988. Learning syntax and meanings through optimization and distributional analysis. In Y. Levy, I. M. Schlesinger, and M. D. S. Braine, editors, *Categories and processes in language acquisition*, pages 179–215. Lawrence Erlbaum, Hillsdale, NJ.

Deniz Yuret. 1998. *Discovery of Linguistic Relations Using Lexical Attraction*. Ph.D. thesis, MIT.