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

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

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 psychological 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 Distributional Syntax Induction

In linear context 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 1). 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 have shown that even this quite simple representation allows the induction of quite high quality word classes, largely corresponding to traditional parts of speech, sometimes with a somewhat more semantic flavor (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.

Since context distributional methods have proven successful in inducing certain kinds of syntatic word classes, it is reasonable to expect them to be useful for discovering constituent classes as well, for the purposes of learning hierarchical tree structures. For example, distributional methods might show that DETERMINER NOUN and DETERMINER ADJECTIVE NOUN occur in similar contexts and therefore could be put into the same class (definite noun phrases) – this echoes the traditional linguistic constituency test of substitutability (Radford, 1988). Clark (2001) and Klein and Manning (2001) show that this is indeed the case. 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 DETERMINER NOUN and DETERMINER ADJECTIVE NOUN are similar and that VERB PREPOSITION DETERMINER and VERB PREPOSITION DETERMINER ADJECTIVE 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.

Clark (2001) proposes an information-theoretic measure of the constituency of a sequence, correlating constituency with the mutual information between the preceding and following contexts (one of several formulations of Harris (1951)'s basic ideas). In Klein and Manning (2002), we instead build 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. Our integrated approach appears to be empirically more successful.

The model works as follows. Sentences are given as sequences of word classes (we experimented both with human-assigned parts-of-speech and automatically induced categories). One imagines each sentence as a list of the $O(n^2)$ subspans inside the sentence, each paired with its linear context (see figure 1). The model generates all $O(n^2)$ subspans (see figure 2).[1]

The first stage is to choose a *bracketing* B for the sentence. A bracketing is a specification of which spans are to be generated as constituent and which are not. $P(B)$ is not modeled; we use a uniform distribution over bracketings which correspond to well-formed (non-crossing) binary trees. This stage amounts to choosing a binary tree for the sentence. Then, each span-context pair $\langle s, c\rangle$ is generated by the simple distributional model:

$$\mathrm{P}(s, c|b) = \mathrm{P}(l|b)\mathrm{P}(s|l)\mathrm{P}(c|l)$$

That is, given a constituency decision b, a label l is chosen (with the constraint that a special label d is assigned to only and all spans with b = false – non-constituents). The span and context are independently filled in, both conditioned on the label.

If $\mathrm{P}(B)$ puts mass 1 on the "full" bracketing B, in which every span is declared a constituent, this process would be plain distributional clustering – the spans would be assigned completely independently. However, our choice of $\mathrm{P}(B)$ is such that the model only allows sets of spans which nest up into binary trees to be assigned categories other than d.

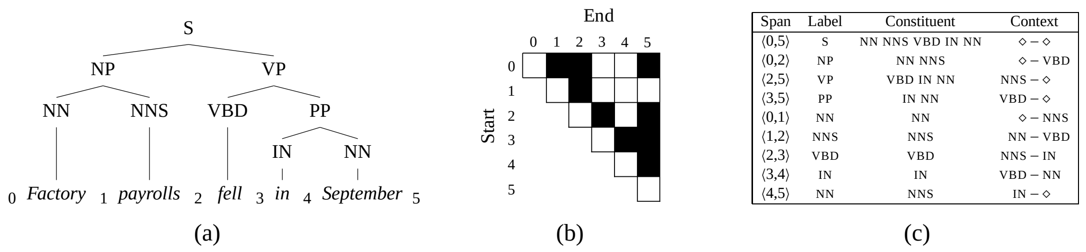
This is a model $\mathrm{P}(B, S)$ over bracketings and sentences, and is estimated via EM to maximize the sentence likelihoods $\mathrm{P}(S)$ over the training corpus. It can then be used to extract most-likely parses for sentences by finding the bracketing B which maximizes $\mathrm{P}(B|S)$. These parses consist of unlabeled binary trees, which can then be compared to human-created gold standard trees from a treebank. We trained this model on the set of 7422 sentences of length up to 10 (without punctuation) in the WSJ section of the Penn treebank (Marcus et al., 1993). The results of comparing the learned structures to the gold-standard treebank parses are shown in figure 3.[2] This model (CCM) substantially outperforms random and right-branching baselines (most unsupervised tree induction systems outperform a random baseline but not a right-branching one). The two higher-scoring models are a supervised PCFG model and an upper bound (the target trees are not all binary and so any all-binary system will over-propose constituents). Comparative figures from other experiments show that it also outperforms the other wide-coverage tree induction systems for which we had treebank parsing results (Klein and Manning, 2001).

3 Nested Models vs. Recursive Models

It is worth some comments about how this kind of approach compares to attempting to induce a recursive model of language, for example the (P)CFG induction work of Olivier (1968), Wolff (1988), Langley (1980), *inter alia*. It is our experience that PCFG induction is much harder than the tree induction described above; apparently the success of distribu-

[1]This is typical of distributional clustering, where corpus words generally get generated at least once as a data item and once as a context element of some other item.

[2]These experiments reflect training on part-of-speech sequences; see Klein and Manning (2001) for experiments with induced classes and Klein and Manning (in publication) for experiments with languages other than English.



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 – ◇

(c)

Figure 1: (a) Example parse tree with (b) its associated bracketing and (c) the yields and contexts for each constituent span in that bracketing. Distituent yields and contexts are not shown, but *are* modeled.

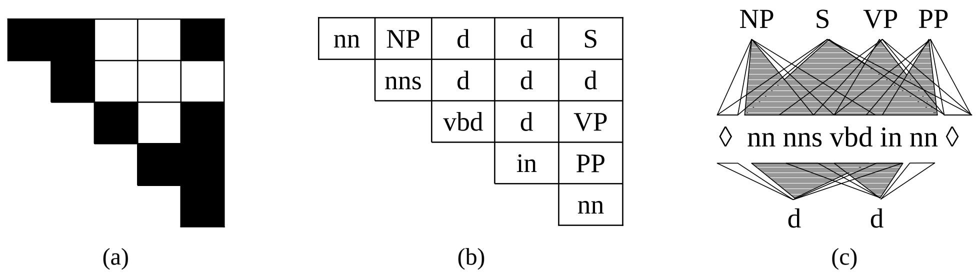


Figure 2: The generative process. (a) A binary tree-equivalent bracketing is chosen at random. (b) Constituent (black) nodes are labeled with a constituent label, distituent (white) nodes are labeled *d*. In the two-class case, there is only a single consituent label *c*, and so this step can be conceptually omitted. (c) *Each* span generates its yield and context (empty spans not shown here). Derivations which are not coherent are given mass zero.

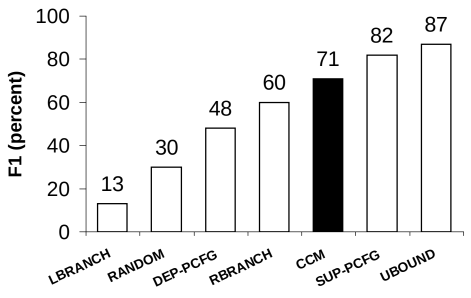


Figure 3: F_1 for various models on WSJ-10.

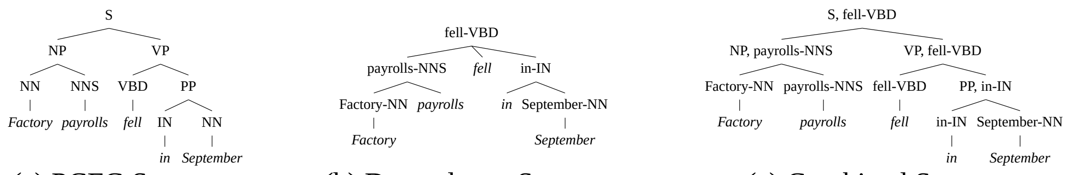
tional methods is precisely because they do *not* attempt to learn recursive facts about language. Learning recursion involves positing intermediate symbols in a grammar whose utility in improving a generative model is only indirect.[3] Our distributional model is intended as an illustration that when attempting to recover syntactic tree structures, it can be useful to model nesting without modeling recursion.

4 A Dependency Model

While the distributional constituent-context approach above is highly successful in recovering constituency (tree) structures which broadly agree with human-assigned bracketings, it is imperfect in several ways. First, most state-of-the-art parsers make use of specific lexical information in addition to word-class information – perhaps lexical information could be a useful source of information for unsupervised methods. Second, one of the main reasons tree structures are useful in computational language processing is because one can extract dependencies – function-argument and modification structures – from them. We address these two issues in this section with a dependency model. Finally, many true constituency errors remain, and we would like a better constituency induction system. In section 5, we use the depedency model presented in this section to augment the CCM.

The dependency induction task has received relatively less attention than the tree- or CFG-induction tasks. We only mention three previous systems below, but 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 pairs of words. For example, in the dependency structure of figure 4(b), the dependencies are {(ROOT, *fell*),

[3]As a concrete example, figure 3 includes a replication of the experiments of Carroll and Charniak (1992b) under DEP-PCFG.



(a) PCFG Structure (b) Dependency Structure (c) Combined Structure

Figure 4: Three kinds of parse structures.

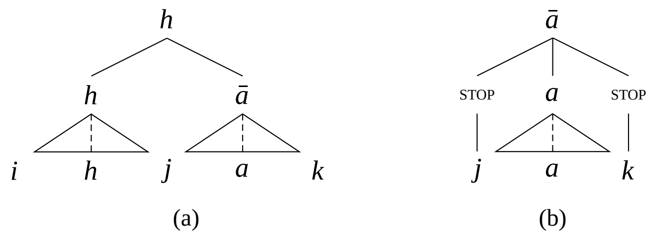


Figure 5: A dependency configuration in a lexicalized tree. h and a are words, while i, j, and k are positions between words.

(*fell, payrolls*), (*fell, in*), (*in, September*), (*payrolls, factory*)}[4] This is distinct from high-performance supervised dependency models, where 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, usually conditioning on at least the direction of the attachment (left arguments vs. right arguments), and often on some notion of distance or valence as well. Moreover, in a head-outward model, it is natural to model stop steps, where the final argument on each side of a head is alway the special symbol STOP.

Our dependency model therefore consists of the following steps. We begin at the ROOT. Each head generates a series of non-STOP arguments to the right, then a STOP argument to the right, then a series of non-STOP arguments to the left, then a STOP to the left. This is shown schematically in figure 5. Figure 5(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, as shown in figure 5(b), and can take no more attachments.[5]

In the 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 5, 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).

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

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

These two multinomials are estimated directly without further decomposition or smoothing for word-class dependency structures. For the case of lexical dependency structures, backoff to word-classes is required. A cubic dynamic program similar to those presented in Eisner (1996) was used to iteratively re-estimate this model using EM. At all times, we constrained ROOT to have a single dependent.

[4]We use a special ROOT symbol to mark the head of the sentence, which simplifies the math and the evaluation metric.

[5]The requirement of attaching left arguments before right arguments has no effect on the dependency structures induced, but does introduce an asymmetry of the model for the constituency structures learned. As a result, in the experiments below we removed this requirement, which leads to a bias towards structures where heads take arguments on both sides. This bias is, unsurprisingly, slightly negative, but it avoids the asymmetry, which could be considered a language-specific bias. We are currently experimenting with a formulation which avoids either bias.

Initialization is important to the success of any local search procedure. As before, 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. Current work is examining other intialization methods.

All dependency structures over *n* words contain *n* dependencies (including the sentence head as a dependent of ROOT). Therefore, precision and recall are equal, so we present accuracy numbers, both over directed and undirected dependencies.[6,7] We got best performance from the *non-lexicalized* dependency model; the lexicalization seemed to increase the model capacity far too much. The word-class dependency model recovers a substantial fraction of the broad dependency trends: 45.0% of guessed directed dependencies were correct (63.6% ignoring direction). 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 and is sufficiently frequent that we also report a second metric (right two columns in figure 6), which is the dependency accuracy when all NPs in the test set containing a determiner are changed to have the determiner as the correct head. By this metric, the score jumps to 55.7% directed (67.9% undirected).

We can usefully compare this formulation with several other related models. First, the model of (Carroll and Charniak, 1992b) is essentially a dependency model over word classes which models directionality, but not valence or distance. Their experiments included large intital perturbations, which turn out not to be required for symmetry-breaking in their model. With this initial perturbation removed, their model does perform better (see (Klein and Manning, 2002)), though even the minimal model of valence and stopping we use here greatly improves the quality of the learned trees and dependencies (compare our F_1 of 63.9 vs. 48 without a valence or distance model). Second, the models of (Yuret, 1998) and (Paskin, 2002), which are very closely related to each other, model lexical selection without modeling valence, distance, or even directionality, and contain no back-off to word classes. Even when trained on many millions of words of text, these models exhibit highly inconsistent treatment of function words and tend to link topically related (as opposed to functionally dependent) words. While the performance of these systems was not available for our exact data set, the undirected dependency accuracy for purely left-headed (right-branching) structures over *all* sentences in the Penn treebank is 52.3%, while the model in (Paskin, 2002) correctly identified only 39.7% of the dependencies in a 10% sample of the treebank. On the other hand, our model scored substantially over this baseline for the WSJ-10 subset (63.6% vs. 55.9%).

5 A Combined Model

Looking at the behavior of the CCM and the dependency models in figure 6, their strengths seem very 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, we have 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 our combined model, we score each tree with the product the probabilities from the individual models above. We use an $O(n^5)$ dynamic program (a variant of the inside-outside algorithm (Baker,

[6]Undirected values are reported by other authors, notably (Paskin, 2002), so we include them here.

[7]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.

	Unlabeled Brackets			Dep. Acc.		Dep. Acc.*	
Model	Prec.	Rec.	F_1	Dir.	Undir.	Dir.	Undir.
Left-Branching / Right-Headed	25.6	32.6	28.7	33.6	56.7	23.2	56.9
Random	40.0	50.8	44.8	25.6	46.7	24.6	47.1
Right-Branching / Left-Headed	55.1	70.0	61.7	24.0	55.9	34.8	56.3
Dependency	57.1	72.5	63.9	45.0	63.6	55.7	67.9
CCM	64.3	81.6	71.9	29.4	51.4	31.4	53.5
Combination (POS)	**67.4**	**85.6**	**75.4**	47.4	64.6	**58.7**	**69.7**
Combination (Semi-Distrib.)	66.8	84.9	74.8	**54.4**	**67.6**	50.6	65.6
Combination (Distributional)	65.2	82.8	72.9	42.3	60.4	50.6	64.8
Upper Bound	78.8	100.0	88.1	100.0	100.0	100.0	100.0

Figure 6: Performance measured by unlabeled bracketing precision, recall, F_1, and directed and undirected dependency accuracy. The reported numbers were at convergence; peak numbers were usually a little higher. The starred version of dependency accuracy considers determiners to be the heads of definite noun phrases. The upper bound on precision relfects that the guessed trees are always binary, while the gold trees are not.

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 5). 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).[8] On the same merge, the dependency model will assess two costs: first the cost of h taking a as a head at the distance $\langle j - h\rangle$, and, second, the two factors for sealing the argument, $(a : j, k)$ (see figure 5b). 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 when they were run individually, with $\mathrm{P}_{\mathrm{SPLIT}}(b)$ used as an initial completion for the CCM and harmonic initialization used for the dependency model. Sufficient statistics were taken off these completions, then the resulting models were used together from then on during re-estimation.

Figure 6 summarizes the results. The combined model beats the CCM at F_1: 75.4 vs. 71.9 (and in fact peaks at 77.0). It also beats the dependency model at the modified dependency accuracy: 58.7% directed vs. 55.7% directed. On the unmodified dependency score, the gap was smaller, but the combination model still outscored the dependency model alone. Figure 6 also shows the combination model's score when using a semi-distributional tagset, where words were automatically clustered, but with their contexts taken to be the distribution of preceding and following treebank parts-of-speech. The result shows improved dependency accuracy over the treebank part-of-speech classes by one metric, but not the other, suggested that perhaps distributional tags might be better suited for certain induction tasks. The final experiment shown used word classes which were induced entirely automatically. These classes show some further degradation, e.g. 72.9 F_1, but, though work in this area is still preliminary, these totally unsupervised numbers are still better than the original CCM with treebank word classes.

[8]For efficiency, we divide the entire probability expression by $\prod_{1 \le i \le j \le n} \mathrm{P}({}_is_j|d)\mathrm{P}(s_{i-1}, s_{j+1}|d)$ both here and with the CCM, so in fact we also assess the inverse of $\mathrm{P}({}_is_j|d)\mathrm{P}(s_{i-1}, s_{j+1}|d)$ at this time as well.

6 Conclusion

We have presented three models for the unsupervised induction of syntactic structure. To our knowledge, these systems are the first to beat the right-branching baseline on broad treebank parsing. One model is designed to induce constituency, while another is designed to induce dependencies. The third is a combination, which outperforms either individual model, by either metric. The key reason that these models are capable of recovering structure more accurately than previous models 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. For example, the CCM models nesting but not recursion. The dependency model is a recursive model, but each tree node is headed by one of the leaves it dominates, so no hidden categories must be posited. Moreover, in their basic forms presented here, neither model (nor their combination) requires any symmetry-breaking perturbations at initialization.

As closing comment, we would like to point out how we see this work in relation to other language learning work. We do not claim that this system reflects the psycholinguistic reality of human language acquisition, either in mechanism or in data. In particular, the system is provided with segmentation and possibly word class information that would be unavailable to a human learner, yet has no access to either the situational constraint or the indirect (or even direct) feedback that a human learner would have. Nonetheless, we see this investigation of what patterns can be recovered from corpora as important, both from a computational perspective and from a philosophical one. For example, we found it surprising that the broad constituent structure of a language could be recovered so well from so little training data.

References

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.

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.

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

Dan Klein and Christopher D. Manning. 2001. Natural language grammar induction using a constituent-context model. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, volume 1, pages 35–42. MIT Press.

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.

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

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.

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

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.