Guiding Unsupervised Grammar Induction Using Contrastive Estimation*

Noah A. Smith and **Jason Eisner**
Department of Computer Science / Center for Language and Speech Processing
Johns Hopkins University
3400 North Charles Street, Baltimore, MD 21218 USA
{nasmith,jason}@cs.jhu.edu

Abstract

We describe a novel training criterion for probabilistic grammar induction models, *contrastive estimation* [Smith and Eisner, 2005], which can be interpreted as exploiting *implicit negative evidence* and includes a wide class of likelihood-based objective functions. This criterion is a generalization of the function maximized by the Expectation-Maximization algorithm [Dempster *et al.*, 1977]. CE is a natural fit for *log-linear* models, which can include arbitrary features but for which EM is computationally difficult. We show that, using the same features, log-linear dependency grammar models trained using CE can drastically outperform EM-trained generative models on the task of matching human linguistic annotations (the MATCHLINGUIST task). The selection of an implicit negative evidence class—a "neighborhood"—appropriate to a given task has strong implications, but a good neighborhood one can target the objective of grammar induction to a specific application.

1 Introduction

Grammars are formal objects with many applications. They become particularly interesting when they allow ambiguity (cf. programming language grammars), introducing the notion that one grammar may be preferable to another for a particular use. Given an induced grammar, a researcher could try to apply it cleverly to her task and then measure its helpfulness on that task. This paper turns that scenario around.

Given a task, our question is how to induce a grammar—from unannotated data—that is especially appropriate for the task. Different grammars are likely to be better for different tasks. In natural language engineering, for example, applications like automatic essay grading, punctuation correction, spelling correction, machine translation, and language modeling pose different challenges and are evaluated differently. We regard traditional natural language grammar induction evaluated against a treebank (also known as unsupervised parsing) as *just another task*; we call it MATCHLINGUIST. A grammar induced for punctuation restoration or language modeling for speech recognition might look strange to a linguist, yet do better on those tasks. By the same token, traditional treebank-style linguistic annotations may not be the best kind of syntax for language modeling.

But without fully-observed data, how might one *tell* a learner to focus on one task or another? We propose that this is conveyed in the choice of an objective function that guides a statistical learner toward the right kinds of grammars for the task at hand. We offer a flexible class of "contrastive" objective functions within which something appropriate may be designed for existing and novel tasks.

In this paper, we evaluate our learned models on MATCHLINGUIST, which is a crucial task for natural language engineering. Automatic natural language grammar induction would bridge the gap between resource limitations (annotated treebanks are expensive, domain-specific, and language-specific) and the promise of exploiting syntactic structure in many applications. We argue that MATCHLINGUIST, just like other tasks, requires guidance.

For example, MATCHLINGUIST is decidedly different from the task that is explicitly solved by the Expectation-Maximization algorithm [Dempster *et al.*, 1977]: MAXIMIZELIKELIHOOD. EM tries to fit the numerical parameters of a (fixed) statistical model of hidden structure to the training data. To recover traditional or useful syntactic structure, it is not enough to maximize training data likelihood [Carroll and Charniak, 1992, *inter alia*], and EM is notorious for mediocre results. Our results suggest that part of the reason EM performs badly is that it offers very little guidance to the learner. The alternative we propose is *contrastive estimation*. It is within the same statistical modeling paradigm as EM, but generalizes it by defining a notion of learner guidance.

Contrastive estimation makes use of a set of examples that are similar in some way to an observed example (its *neighborhood*), but mostly perturbed or damaged in a particular way. CE requires the learner to move probability mass to a given example, taking only from the example's neighborhood. The neighborhood of a particular example is defined by the *neighborhood function*; different neighborhood functions

*This work was supported by a Fannie and John Hertz Foundation Fellowship to the first author and NSF ITR grant IIS-0313193 to the second author. The views expressed are not necessarily endorsed by the sponsors. The authors also thank colleagues at CLSP and two anonymous reviewers for comments on this work.

are suitable for different tasks and the neighborhood should be designed for the task.

We note that our approach to this problem is couched in a parameter-centered approach to grammar induction. We assume the grammar to be learned is structurally fixed and allows all possible structures over the input sentences; our task is to learn the *weights* that let the grammar disambiguate among competing hypotheses. A different approach is to focus on the hypotheses themselves and perform search in that space and/or the space of grammars (see, e.g., Adriaans [1992], Clark [2001], and van Zaanen [2002]). Those systems also use statistical techniques and offer guidance to the learner, both in the form of search criteria and search methods (e.g., searching for substitutable subsequences). We will not attempt to broadly formalize "guidance" here, noting only that it is ubiquitous.

We begin by motivating contrastive estimation and describing it formally (§2). Central to CE is the choice of a contrastive neighborhood function. In §3, we describe some neighborhoods expected to be useful for MATCHLINGUIST and other tasks. We discuss the algorithms required for application of CE with these neighborhoods in §4. §5 describes how log-linear models are a natural fit for CE and demonstrates how CE avoids the mathematical and computational difficulties presented by unsupervised estimation of log-linear models. We describe state-of-the-art results in dependency grammar induction in §6, showing that a good neighborhood choice can obviate the need for a clever initializer and can drastically outperform EM on MATCHLINGUIST. We address future directions (§7) and conclude (§8).

2 Implicit Negative Evidence

Natural language is a delicate thing. For any plausible sentence, there are many slight perturbations of it that will make it implausible. Consider, for example, the first sentence of this section. Suppose we choose one of its six words at random and remove it; odds are two to one that the resulting sentence will be ungrammatical. Or, we could randomly choose two adjacent words and transpose them; none of the results are valid conversational English sentences.[1] The learner we describe here takes into account not only the observed positive example, but a also set of similar examples that are deprecated as perhaps negative (in that they could have been observed but weren't).

2.1 Learning setting

Let $\vec{x} = \langle x_1, x_2, ... \rangle$, be our observed example sentences, where each $x_i \in \mathcal{X}$, and let $y_i^* \in \mathcal{Y}$ be the unobserved correct parse for x_i. We seek a model, parameterized by $\vec{\theta}$, such that the (unknown) correct analysis y_i^* is the best analysis for x_i (under the model). If y_i^* were observed, a variety of optimization criteria would be available, including maximum (joint or conditional) likelihood estimation, maximum classification accuracy [Juang and Katagiri, 1992], maximum *expected* classification accuracy [Klein and Manning, 2002a; Altun *et al.*, 2003], minimum exponential (boosting) loss [Collins, 2000], and maximum margin [Crammer and Singer, 2001]. Yet y_i^* is unknown, so none of these supervised methods apply. Typically one turns to the EM algorithm [Dempster *et al.*, 1977], which locally maximizes

$$\prod_i p\left(X = x_i \mid \vec{\theta}\right) = \prod_i \sum_y p\left(X = x_i, Y = y \mid \vec{\theta}\right) \quad (1)$$

where X is a random variable over sentences and Y is a random variable over parse trees (notation is often abbreviated, eliminating the random variables). EM has figured heavily in probabilistic grammar induction [Pereira and Schabes, 1992; Carroll and Charniak, 1992; Klein and Manning, 2002b; 2004]. An often-used alternative to EM is a class of so-called Viterbi (or "winner-take-all") approximations, which iteratively find the most probable parse $\hat{y}$ (according to the current model) and then, on each iteration, solve a supervised learning problem, training on $\hat{y}$.

Despite its frequent use, EM is not hugely successful at recovering the linguistic hidden structure . Merialdo [1994] showed that EM was helpful to the performance of a trigram HMM part-of-speech tagger only when extremely small amounts of labeled data were available. The EM criterion (Equation 1) simply doesn't correspond to the real merit function. Further, even if the goal *is* to maximize likelihood (e.g., in language modeling), the surface upon which EM performs hillclimbing has many shallow local maxima [Charniak, 1993], making EM sensitive to initialization and therefore unreliable. This search problem is discussed in Smith and Eisner [2004].

We suggest that part of the reason EM performs poorly is that it does not sufficiently constrain the learner's task. EM tells the learner only to move probability mass toward the observed x_i, paired with any y; the source of this mass is not specified. We will consider a class of alternatives that make explicit the source of the probability mass to be pushed toward each x_i.

2.2 A new approach: contrastive estimation

Our approach instead maximizes

$$\prod_i p\left(X = x_i \mid X \in \mathcal{N}(x_i), \vec{\theta}\right) \quad (2)$$

where $\mathcal{N}(x_i) \subseteq \mathcal{X}$ is the class of negative example sentences plus the observed sentence x_i itself. Note that the $x' \in \mathcal{N}(x_i)$ are not treated as hard negative examples; we merely seek to move probability mass from them to the observed x. The probability mass $p(x_i \mid \vec{\theta})$ attached to a single example is found by marginalizing over hidden variables (Equation 1).

The negative example set $\mathcal{N}$ depends on x and is written $\mathcal{N}(x)$ to indicate that it is a function, $\mathcal{N} : \mathcal{X} \rightarrow 2^{\mathcal{X}}$. In this work, $\mathcal{N}(x)$ contains examples that are perturbations of x, and we call this set the *neighborhood* of x. We then refer to $\mathcal{N}$ as the neighborhood function and the optimization of Equation 2 as *contrastive estimation* (CE). The neighborhood may be viewed as a class of *implicit negative evidence* that is fully determined by the example and may help to highlight what about the example the model should try to predict.

[1] "Natural language is a thing delicate" might be valid in poetic speech.

CE seeks to move probability mass from the neighborhood of an observed sentence x to x itself. The learner hypothesizes that good models are those which discriminate an observed sentence from its neighborhood. Put another way, the learner assumes not only that x is good, but that x is locally optimal in example space ($\mathcal{X}$), and that alternative, similar examples (from the neighborhood) are inferior. Rather than explain all of the data, the model must only explain (using hidden variables) why the observed sentence is better than its neighbors. Of course, the validity of the neighborhood hypothesis will depend on the form of the neighborhood function. Further, different neighborhoods may be appropriate for different tasks.

Consider grammar induction as an example. We might view the neighborhood of x as a variety of alternative surface representations using the same lexemes in slightly-altered configurations, like the single-deletion or single-transposition perturbations described earlier. While degraded, the inferred meaning of any of these examples is typically close to the intended meaning, yet the speaker *chose* x and not one of the other $x' \in \mathcal{N}(x)$. Why? Deletions are likely to violate subcategorization requirements, and transpositions are likely to violate word order requirements—both of which have something to do with syntax. x was the most grammatical option that conveyed the speaker's meaning, hence (we hope) roughly the most grammatical option in the neighborhood $\mathcal{N}(x)$, and the syntactic model should make it so. EM, on the other hand, offers no such guidance: EM notes only that the speaker chose x from the *entire set* $\mathcal{X}$, and therefore requires only that the learner move mass to x, without specifying where it should come from. Latent variables that distinguish x from the rest of $\mathcal{X}$ may have more to do with what people talk about than how they arrange words syntactically.

3 Neighborhoods Old and New

We next show how neighborhoods generalize EM and describe some novel neighborhood functions for natural language data.

3.1 EM

It is not hard to see that EM (more precisely, the objective in Equation 1) is equivalent to CE where the neighborhood for every example is the entire set $\mathcal{X}$, and the denominator equals 1. The EM algorithm under-determines the learner's hypothesis, stating only that probability mass should be given to x, but not stating at whose expense.

An alternative proposed by Riezler *et al.* [2000] and inspired by computational limitations is to restrict the neighborhood to the training set. This gives the following objective function:

$$\prod_i \left[p\left(x_i \mid \vec{\theta}\right) \Big/ \sum_j p\left(x_j \mid \vec{\theta}\right) \right] \tag{3}$$

Viewed as a CE method, this approach (though effective when there are few hypotheses) seems misguided; the objective says to move mass to each example at the expense of all other training examples. Smith and Eisner [2005] describe how several other probabilistic learning criteria are examples of CE; see also Table 1.

3.2 Neighborhoods of sequences

We next consider some neighborhood functions for sequences (e.g., natural language sentences). When $\mathcal{X} = \Sigma^+$ for some symbol alphabet Σ, certain kinds of neighborhoods have natural, compact representations. Given an input string $x = x_1^m$, we write x_i^j for the substring $x_i x_{i+1} ... x_j$ and x_1^m for the whole string. Consider first the neighborhood consisting of all sequences generated by deleting a single symbol from the m-length sequence x_1^m:

$$\textsc{Del1Word}(x_1^m) \quad = \quad \left\{ x_1^{\ell-1} x_{\ell+1}^m \mid 1 \leq \ell \leq m \right\} \cup \left\{ x_1^m \right\}$$

This set consists of $m+1$ strings and can be compactly represented as a lattice (see Figure 1a). Another neighborhood involves transposing any pair of adjacent words:

$$\begin{aligned} &\textsc{Trans1}(x_1^m) \\ &\quad = \quad \left\{ x_1^{\ell-1} x_{\ell+1} x_\ell x_{\ell+2}^m \mid 1 \leq \ell \leq m-1 \right\} \cup \left\{ x_1^m \right\} \end{aligned}$$

This set can also be compactly represented as a lattice (Figure 1b). We can combine DEL1WORD and TRANS1 by taking their union; this gives a larger neighborhood, DELORTRANS1. In general, the lattices are obtained by composing the observed sequence with a small finite-state transducer and determinizing and minimizing the result; the relevant transducers are shown at the right of Figure 1.

Another neighborhood we might wish to consider is LENGTH, which consists of Σ^m for an m-length sentence (Figure 1c). CE with the LENGTH neighborhood is very similar to EM; it is equivalent to using EM to estimate the parameters of a model defined by

$$p'(x_1^m, y \mid \vec{\theta}) \stackrel{\text{def}}{=} q(m) \cdot p(x_1^m, y \mid m, \vec{\theta})$$

where q is any fixed (untrained) distribution over lengths.

Generally speaking, CE is equivalent to some kind of EM when $x' \in \mathcal{N}(x)$ is an equivalence relation on examples, so that the neighborhoods partition the space of examples. Then q is a fixed distribution over neighborhoods.

The vocabulary Σ is never fully known for a natural language; approximations include using only the observed Σ from the training set or adding a special OOV symbol. When estimating finite-state models, CE with the LENGTH neighborhood is possible using a dynamic program. When the model involves deeper, non-finite-state structure (e.g., one with context-free power), the LENGTH neighborhood may become too expensive. This was not the case for models explored in this paper.

3.3 Task-based neighborhoods

When considering a specific application of grammar induction, specific features of a sentence may be particularly relevant to the modeling task. Put another way, if we want to perform a specific task, appropriate neighborhoods may be apparent. Suppose we desire a probabilistic context-free grammar that can discriminate correctly spelled or punctuated sentences from incorrectly spelled or punctuated ones. With

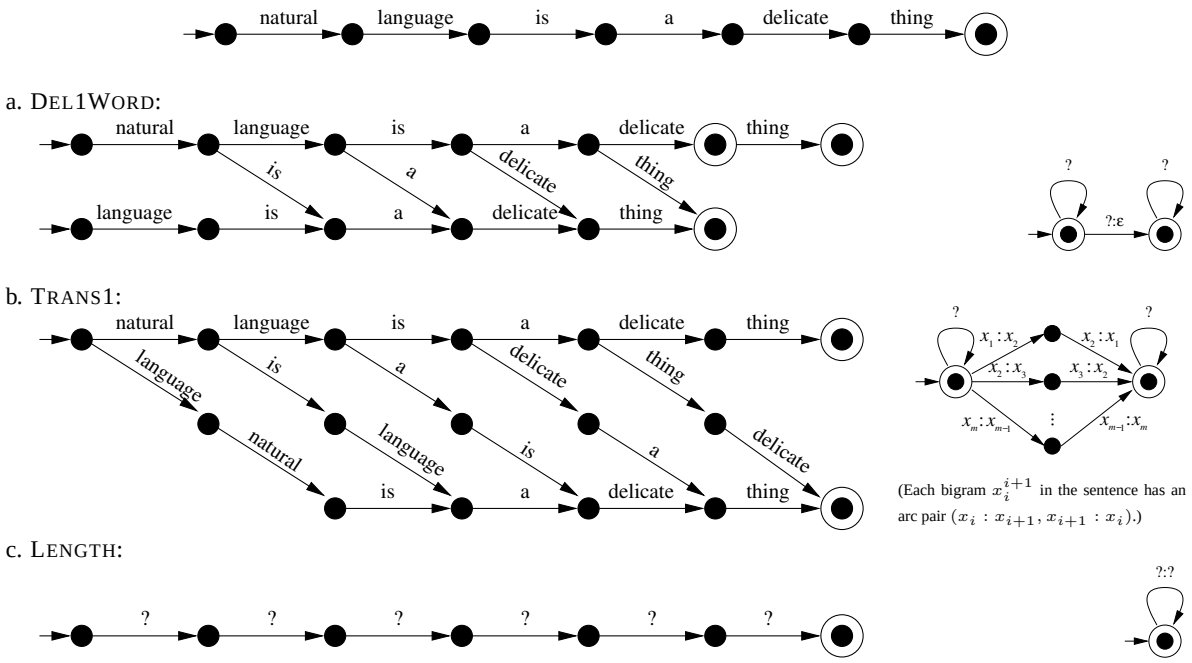

Figure 1: A sentence and three lattices representing some of its neighborhoods. The transducer used to generate each neighborhood lattice (via composition with the sentence followed by determinization and minimization) is shown to its right.

a large corpus of incorrectly punctuated sentences and their corrections, one could do supervised training of a translation model to distinguish the actual correction from other candidate corrections. However, sufficient training data would be hard to come by, especially if the model included latent syntactic variables.

Fortunately, manufacturing supervised data for this kind of task is easy: take real text, and mangle it. This is a classic strategy for training accent and capitalization restoration [Yarowsky, 1994]: just delete all accents from the good text.

In our case, we don't know the mangling process. The errors are not simply an omission of some part of the data; they are whatever mistakes humans make. Without a corpus of errors, this is difficult to model.

We suggest that it may be possible to get away with not knowing which mistakes a human would make; instead we try to distinguish each observed good sentence from *many* differently punctuated (presumably mispunctuated) versions. This is not as inefficient as it might sound, because lattices allow efficient training. (In CE terms, the set of all variants of the sentence with errors introduced is the neighborhood.)

For spelling correction, this neighborhood might be

$$\textsc{Spell}_{\leq k}(x_1^m) = \{\bar{x}_1^m : \forall i \in \{1,2,...,m\}, \mathbf{Lev}(x_i, \bar{x}_i) \leq k\} \quad (4)$$

where $\mathbf{Lev}(a, b)$ is the Levenshtein (edit) distance between words a and b [Levenshtein, 1965]. This neighborhood, like the others, can be represented as a lattice. This lattice will have a "sausage" shape.

A neighborhood for punctuation correction might be

$$\textsc{Punc}_{\leq k}(x_1^m) = \{\bar{x}_1^m : x \text{ and } \bar{x} \text{ differ only in punctuation and } \mathbf{Lev}(x, \bar{x}) \leq k)\} \quad (5)$$

which includes alternatively-punctuated sentences that differ in up to k edits from the observed sentence. In §5.4 we will discuss how to use these contrastively trained models.

4 Algorithms

We have described several neighborhoods that can be represented as lattices. Our major algorithmic tool will be the general technique known as lattice parsing. For any common grammar formalism that admits a polynomial-time dynamic programming algorithm for *string* parsing, there exists a straightforward generalization to a polynomial-time dynamic programming algorithm for *lattice* parsing. The probabilistic CKY algorithm for probabilistic CFGs [Baker, 1979; Lari and Young, 1990] and the Viterbi algorithm for HMMs [Baum *et al.*, 1970] are examples.

Contrastive estimation can then be applied using any such grammar formalism (finite-state, context-free, mildly context-sensitive, etc.). The reader may find it easiest to think about probabilistic context-free grammars and the CKY algorithm. In our experiments, however, we used a dependency parsing model (§6). We implemented our lattice parser using Dyna [Eisner *et al.*, 2004].

With probabilistic grammars, there are two versions of lattice parsing. One version finds the highest-probability parse of any string in the lattice (and the string it yields). The other finds the total probability of all strings in the lattice, summing over all of their parses. We refer to these throughout as BESTPARSE (sometimes called a "Viterbi" algorithm) and

SUMPARSES (sometimes called a generalized "inside" algorithm). Unfortunately we know of no efficient algorithm for finding the highest-weight *string* in the lattice, summed over all parses. We suspect that that problem is intractable, even for finite-state grammars.

We can generalize probabilistic grammars further by replacing probabilities (e.g., rewrite rule probabilities in PCFGs) with arbitrary *weights*; the resulting grammars are *weighted* grammars (e.g., WCFGs). If we define the probability of a (sentence, tree) pair as its total weight (its score) normalized by the sum of scores of all possible (sentence, tree) pairs allowed by the grammar, we have a *log-linear* CFG [Miyao and Tsujii, 2002]; log-linear models will be discussed further in §5. Importantly, BESTPARSE and SUMPARSES can be applied with weighted grammars with no modification. Log-linear CFGs are more flexible, in a probabilistic sense, than PCFGs (which are a subset of the former), because they can give arbitrary credit or penalties to any rewrite rules, without stealing from others.

The crucial difference between PCFGs and log-linear CFGs, from a computational point of view, is in the normalizing term required by the latter. A PCFG is defined as a generative process that assigns probabilities through the sequence of steps taken. Log-linear CFGs must normalize by the sum of scores of all allowed structures. The normalization term is called the *partition function*. For an arbitrary set of rewrite rule weights, this sum may not be finite.[2]

5 Log-Linear Models

Log-linear models, we will show, are a natural match for contrastive estimation. Log-linear models assign probability to a (sentence, parse tree) pair (x, y) according to

$$p\left(x, y \mid \vec{\theta}\right) \stackrel{\text{def}}{=} \frac{\exp\left[\vec{\theta} \cdot \vec{f}(x, y)\right]}{\sum_{(x', y') \in \mathcal{X} \times \mathcal{Y}} \exp\left[\vec{\theta} \cdot \vec{f}(x', y')\right]} \quad (6)$$

where $\vec{f} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^n_{\geq 0}$ is a nonnegative vector feature function and $\vec{\theta} \in \mathbb{R}^n$ are the corresponding feature weights. We will refer to the inner product of $\vec{\theta}$ and $\vec{f}(x, y)$ as the *score* $w(x, y)$. Because the features can take any form and even "overlap," log-linear models can capture arbitrary dependencies in the data and cleanly incorporate them into a model.

The relevant log-linear models here are log-linear CFGs. We emphasize that the contrastive estimation methods we describe are applicable to a wide class of sequence models, including chain-structured random fields [Smith and Eisner, 2005].

5.1 Supervised estimation

For log-linear models, both conditional likelihood estimation and joint likelihood estimation are available. CL is often preferred [Klein and Manning, 2002a, but see also Johnson, 2001]. The computational difficulty with supervised *joint* maximum likelihood estimation for log-linear models is the partition function (the denominator in Equation 6); as discussed earlier (§4), this sum may not be finite for all $\vec{\theta}$. Alternatives to exact computation of the partition function, like random sampling [Abney, 1997, for example] will not help to avoid this difficulty; in addition, convergence rates are in general unknown and bounds difficult to prove. An advantage of conditional likelihood estimation is that the full partition function need not be computed; it is replaced by a sum over $y' \in \mathcal{Y}$ of scores $w(x, y')$ for each x. Conditional random fields are log-linear models over sequences, estimated using conditional likelihood; typically they correspond to log-linear finite-state transducers [Lafferty *et al.*, 2001].

Log-linear models can also be trained contrastively using fully-annotated data; an example are the morphology models of Smith and Smith [2004] (see Table 1).

5.2 Unsupervised estimation

CE, which deals in conditional probabilities, restricts the denominators of the likelihood function, summing only over $x \in \mathcal{N}(x_i)$ and maximizing

$$\mathcal{L}_{\mathcal{N}}\left(\vec{\theta}\right) = \sum_i \log \frac{\sum_{y \in \mathcal{Y}} \exp\left[\vec{\theta} \cdot \vec{f}(x_i, y)\right]}{\sum_{(x,y) \in \mathcal{N}(x_i) \times \mathcal{Y}} \exp\left[\vec{\theta} \cdot \vec{f}(x, y)\right]} \quad (7)$$

The sums in the numerators, over $\{x_i\} \times \mathcal{Y}$, are computed using SUMPARSES; so are the denominators, since $\mathcal{N}(x_i)$ is represented as a lattice.

As discussed in §3.1, EM is a special case where the denominator is the sum of scores of all derivations of the entirety of Σ^*. This is the same partition function that joint likelihood training faces, and EM suffers from the same computational difficulty of a possibly divergent sum (§4). By making the sum finite—i.e., by defining finite neighborhoods—this problem disappears (a move analogous to the move from joint to conditional likelihood in supervised estimation).

5.3 Numerical optimization

To maximize the neighborhood likelihood (Equation 7), we apply a standard numerical optimization method (L-BFGS) that iteratively climbs the function using knowledge of its value and gradient [Liu and Nocedal, 1989]. The partial derivative of $\mathcal{L}_{\mathcal{N}}$ with respect to the jth feature weight θ_j is

$$\frac{\partial \mathcal{L}_{\mathcal{N}}}{\partial \theta_j} = \sum_i \mathbf{E}_{\vec{\theta}}[f_j \mid x_i] - \mathbf{E}_{\vec{\theta}}[f_j \mid \mathcal{N}(x_i)] \quad (8)$$

This looks similar to the gradient of log-linear likelihood functions on complete data, though the expectation on the left is in those cases replaced by an observed feature value $f_j(x_i, y_i^*)$. An alternative would be a doubly-looped algorithm that looks similar to EM. The E step would compute the two expectations in Equation 8 and the M step (the inner loop) would adjust the parameters to make them match (perhaps using an iterative algorithm). If the M step is not run to convergence, we have something resembling a Generalized EM algorithm, which avoids the double loop and may be

[2] For WCFGs in CNF with k nonterminal symbols, the problem is equivalent to solving a system of k multivariate quadratic equations.

	likelihood criterion	objective	sum in ith numerator	sum in ith denominator
supervised	joint	$\prod_i p\left(x_i, y_i^* \mid \vec{\theta}\right)$	$\{(x_i, y_i^*)\}$	$\mathcal{X} \times \mathcal{Y}$
	conditional	$\prod_i p\left(y_i^* \mid x_i, \vec{\theta}\right)$	$\{(x_i, y_i^*)\}$	$\{x_i\} \times \mathcal{Y}$
	contrastive	$\prod_i p\left(y_i^* \mid (X,Y) \in \mathcal{N}(x_i, y_i^*), \vec{\theta}\right)$	$\{(x_i, y_i^*)\}$	$\mathcal{N}(x_i, y_i^*)$
	contrastive (correction)	$\prod_i p\left(X = x_i \mid X \in \mathcal{N}(x_i), \vec{\theta}\right)$	$\{x_i\}$	$\mathcal{N}(x_i)$
unsupervised	marginal (*a là* EM)	$\prod_i \sum_y p\left(x_i, y \mid \vec{\theta}\right)$	$\{x_i\} \times \mathcal{Y}$	$\mathcal{X} \times \mathcal{Y}$
	contrastive	$\prod_i \sum_y p\left(X = x_i, y \mid X \in \mathcal{N}(x_i), \vec{\theta}\right)$	$\{x_i\} \times \mathcal{Y}$	$\mathcal{N}(x_i) \times \mathcal{Y}$

Table 1: Supervised and unsupervised estimation with log-linear models for classification. The supervised case marked "contrastive (correction)" is applicable to models for correcting possibly noisy input x_i, rather than classifying x_i.

faster; see, e.g., Riezler [1999]. The key difference between our approach and EM/GEM, of course, is that the probabilities in the objective function are conditioned on the neighborhood.

The expectations in Equation 8 are computed as a by-product of running SUMPARSES followed by an "outside" or "backward" pass dynamic program similar to back-propagation.

When there are no hidden variables, $\mathcal{L}_{\mathcal{N}}$ is globally concave (examples include supervised joint and conditional likelihood estimation). In general, with hidden variables, the function $\mathcal{L}_{\mathcal{N}}$ is *not* globally concave; our search will lead only to a local optimum. Therefore, as with EM, the initial bias in the initialization of $\vec{\theta}$ will affect the quality of the estimate and the performance of the method. In future work, we might wish to apply techniques for avoiding local optima, such as deterministic annealing [Smith and Eisner, 2004].

5.4 Inference in task-based neighborhoods

The choice of neighborhood affects training only in the construction of neighborhood lattices. The underlying probabilistic model and the algorithm for training it are unaffected by this choice. The *application* of these models to testing data is somewhat different for task-based neighborhoods.

Consider again the syntax induction problem: given a sentence x, we wish to recover the hidden syntactic structure. To do this, having trained a probabilistic model with hidden variables, we use BESTPARSE to infer (or decode) the most likely structure:

$$\hat{y} = \operatorname*{argmax}_y p\left(y \mid x, \vec{\theta}\right) \tag{9}$$

The spelling correction and punctuation restoration cases are slightly different. At test time, we observe a sentence that may contain errors (misspelled words or missing punctuation). Our goal is to select the sentence from its neighborhood that is most likely, according to our model. Note that the neighborhoods now are centered on the observed, possibly incorrect sentences, rather than correct training examples. They are still lattices, a fact we will exploit.

This approach is similar to certain noisy-channel spelling correction approaches [Kernighan *et al.*, 1990] in which, as for us, only correctly-spelled text is observed. Like them, we have no "channel" model of which errors are more or less likely to occur (only a set of possible errors that implies a set of candidate corrections), though the neighborhood could perhaps be weighted to incorporate a channel model (so that we consider not only the probability of each candidate correction but also its similarity to the typed string).[3] The model we propose is a language model—one that incorporates induced grammatical information—that might then be combined with an existing channel model. The other difference is that this approach would attempt to correct the entire sequence at once, making globally optimal decisions, rather than trying to correct each word individually.

A subtlety is that the quantity we wish to maximize is a sum:

$$\hat{x} = \operatorname*{argmax}_{x' \in \mathcal{N}(x)} p\left(x' \mid \vec{\theta}\right) = \operatorname*{argmax}_{x' \in \mathcal{N}(x)} \sum_{y \in \mathcal{Y}} p\left(x', y \mid \vec{\theta}\right) \tag{10}$$

where y ranges over possible parse trees. We noted in §4 that this problem is likely to be intractable.

A reasonable approximation to this decoding is to simply apply BESTPARSE, finding

$$(\hat{x}, \hat{y}) = \operatorname*{argmax}_{x' \in \mathcal{N}(x), y} p\left(x', y \mid \vec{\theta}\right) \tag{11}$$

This gives the best parse tree over any sequence in $\mathcal{N}(x)$, with the sequence, but not necessarily the best *sequence*. This is a familiar approximation in natural language engineering (e.g., machine translation often picks the most probable translation *and* alignment, given a source sentence, rather than marginalizing over all alignments).

6 Unsupervised Dependency Parsing

In prior work, we compared various neighborhoods for inducing a trigram part-of-speech tagger from unlabeled data [Smith and Eisner, 2005], given a (possibly incomplete) tagging dictionary. The best performing neighborhoods in those experiments were LENGTH, DELORTRANS1, and TRANS1. We found that DELORTRANS1 and TRANS1 were more

[3] The notion of training with weighted or probabilistic neighborhoods is an interesting one that we leave to future work.

robust than LENGTH when the tagging dictionary was degraded, and also more able to recover with the help of additional (spelling) features.

Here we explore a variety of contrastive neighborhoods on the MATCHLINGUIST task. Our starting point is essentially identical to the dependency model used by Klein and Manning [2004].[4] This model assigns probability to a sentence x_1^m and an unlabeled dependency tree as follows. The tree is defined by a pair of functions χ_{left} and χ_{right} (both $\{1, 2, ..., m\} \rightarrow 2^{\{1,2,...,m\}}$) which map each word to its dependents on the left and right, respectively. (The graph is constrained to be a projective tree, so that each word except the root has a single parent, and there are no cycles or crossing dependencies.) The probability of generating the subtree rooted at position i, given its head word, is:

$$P(i) = \prod_{d \in \{left, right\}} \left(\prod_{j \in \chi_d(i)} p_{\text{stop}}(\neg\text{stop} \mid x_i, d, f(x_j)) \cdot p_{\text{kid}}(x_j \mid x_i, d) \cdot P(j) \right) \cdot p_{\text{stop}}(\text{stop} \mid x_i, d, [\chi_d(i) = \emptyset]) \quad (12)$$

where the $f(x_j)$ is true iff x_j is the closest child (on either side) to its parent x_i. The probability of the entire tree is given by:

$$p(x_1^m, \chi_{left}, \chi_{right}) = p_{\text{root}}(x_r) \cdot P(r) \quad (13)$$

where r is the index of the root node.

In this model, $p_{\text{root}}, p_{\text{stop}}$, and p_{kid} are families of conditional probability distributions. A log-linear model that uses the same features replaces these by exponentials of feature weight functions ($\exp \theta_{\text{root}}(...), \exp \theta_{\text{stop}}(...)$, and $\exp \theta_{\text{kid}}(...)$, respectively), and includes a normalization factor (partition function) to make everything sum to one. As discussed in §5.2, the partition function may not converge, but we never need to compute it, because we only consider conditional probabilities. Note also that this is simply a log-linear (dependency) CFG—we have not incorporated any overlapping features.

We compared contrastive estimation with three different neighborhoods (LENGTH, TRANS1, and DELORTRANS1) to EM with the generative model. We varied the regularization in both cases; for the log-linear models, we used a single Gaussian prior with mean 0 and different variances ($\sigma^2 \in \{0.1, 1, 10, \infty\}$). Note that a lower variance imposes stronger smoothing [Chen and Rosenfeld, 2000]; variance of ∞ implies no smoothing at all. The generative model was smoothed using add-λ smoothing ($\lambda \in \{0, 0.1, 1, 10\}$).[5] Because all trials involved optimization of a non-concave objective function, we also tested two initializers. The first is very similar to the one proposed by Klein and Manning [2004]. For the generative model, this involves beginning with expected counts that bias against long-distance dependencies (but give some probability to any dependency), and normalizing to obtain initial probabilities. For the log-linear models, we simply set the corresponding weights to be the logs of those probabilities. The other initializer is a simple uniform model; for the generative model, each distribution is set to be uniform, and for the log-linear model, all weights start at 0. Note that our grammars are defined so that *any* dependency tree over any training example is possible.

The dataset is WSJ-10: sentences of ten words or fewer from the Penn Treebank, stripped of punctuation. Like Klein and Manning [2004], we parse sequences of part-of-speech tags. The complete model (over a vocabulary of 37 tags) has 3,071 parameters. Our experiments are ten-fold cross-validated, with eight folds for training and one for test.

Because the Penn Treebank does not include dependency annotations, accuracy was measured against the output of a supervised, rule-based system for adding heads to treebank trees [Hwa and Lopez, 2004]. (The choice of head rules accounts for the difference in performance we report for Klein and Manning's system and their results.) All trials were trained until the objective criterion converged to a relative tolerance of 10^{-5}. The average number of iterations of training required to converge to this tolerance is shown for each trial; note that in the non-EM trials, each iteration will require at least two passes of the dynamic program on the data (once for the numerator, once on the neighborhood lattice for the denominator)—potentially more during the line search.

Discussion Directed dependency attachment accuracy is reported in Table 2. The first thing to notice is that the LENGTH neighborhood—the closest we can reasonably get to EM on a log-linear variant of the original generative model, owing to the partition function difficulty (§4)—is consistently better than EM on the generative model. This should not be surprising. Log-linear models are (informally speaking) more probabilistically expressive than generative models, because the weights are unconstrained. (Recall that generative models are a subset of log-linear models, with nonnegativity and sum-to-one constraints on the exponentials of the weights $\vec{\theta}$.) This added expressivity allows the model to put a "bonus" (rather than a cost) on favorable configurations. For example, in the unsmoothed LENGTH trial, the attachment of a $ tag as the left child of a CD (cardinal number) had a learned weight of 3.75 and the attachment of a MD (modal) as the left child of a VB (base form verb) had a weight of 2.98. In a generative model, weights will never be greater than 0, because they are interpreted as log-probabilities.

The main result is that the best-performing parameter estimates were trained *contrastively* using the TRANS1 and DELORTRANS1 neighborhoods. Furthermore, they came from combining contrastive estimation with a uniform initializer. (Even the LENGTH neighborhood initialized *uniformly* performs nearly as well as the cleverly initialized EM-trained generative model.) That is a welcome change, as clever initializers are hard to design. There is a actually some reason to suppose that uniform initializers may provide a generically helpful implicit bias: Wang *et al.* [2002] have suggested that high-entropy models are to be favored in learning with latent

[4] Their best model was a combined constituent-context and dependency model; we explored only the dependency model.

[5] We note that prior work on unsupervised learning has not fully explored the effects of smoothing on learning and performance.

		Klein & Manning's initializer			Uniform initializer		
		training accuracy (%)	test accuracy (%)	iterations	training accuracy (%)	test accuracy (%)	iterations
untrained	$\lambda = 10$	21.7 $\pm$0.19	21.8 $\pm$0.82		*(this approximates random; smoothing has no effect on a uniform model)*		
(generative, sum-to-one)	1	23.5 $\pm$0.92	23.5 $\pm$1.32				
	0.1	23.3 $\pm$0.79	23.4 $\pm$1.18				
	no smoothing	23.3 $\pm$0.46	23.5 $\pm$1.06		22.3 $\pm$0.13	22.3 $\pm$0.72	
EM	$\lambda = 10$	30.5 $\pm$5.75	30.8 $\pm$5.57	33.1 $\pm$5.0	19.5 $\pm$0.35	19.5 $\pm$0.78	40.0 $\pm$7.5
(generative, sum-to-one)	1	34.5 $\pm$7.09	34.8 $\pm$6.43	55.8 $\pm$12.3	21.2 $\pm$0.29	21.1 $\pm$1.26	54.4 $\pm$1.8
	0.1	34.5 $\pm$7.13	34.7 $\pm$6.51	58.7 $\pm$8.4	22.1 $\pm$3.01	22.2 $\pm$3.38	63.8 $\pm$18.7
	no smoothing*	35.2 $\pm$6.59	35.2 $\pm$5.99	64.1 $\pm$11.1	23.6 $\pm$3.77	23.6 $\pm$4.31	63.3 $\pm$9.2
LENGTH	$\sigma^2 = 0.1$	**42.7** $\pm$7.58	**42.9** $\pm$7.57	150.5 $\pm$32.0	32.5 $\pm$3.54	32.4 $\pm$3.81	101.1 $\pm$17.0
(log-linear)	1	**42.6** $\pm$5.87	**42.9** $\pm$5.76	260.5 $\pm$121.1	33.5 $\pm$3.61	33.6 $\pm$3.75	177.0 $\pm$34.4
	10	**42.2** $\pm$5.76	**42.4** $\pm$5.73	259.2 $\pm$168.8	33.6 $\pm$3.80	33.7 $\pm$3.88	211.9 $\pm$49.4
	no smoothing	**42.1** $\pm$5.58	**42.3** $\pm$5.52	195.2 $\pm$56.4	33.8 $\pm$3.59	33.7 $\pm$5.86	173.1 $\pm$77.7
TRANS1	$\sigma^2 = 0.1$	32.7 $\pm$6.52	32.4 $\pm$6.03	54.9 $\pm$14.4	**41.4** $\pm$4.59	**41.5** $\pm$5.12	33.8 $\pm$6.7
(log-linear)	1	31.7 $\pm$9.41	31.5 $\pm$9.34	113.7 $\pm$28.3	**48.4** $\pm$0.71	**48.5** $\pm$1.15	82.5 $\pm$12.6
	10	37.4 $\pm$6.49	37.4 $\pm$6.06	215.5 $\pm$95.0	**48.8** $\pm$0.90	**49.0** $\pm$1.53	173.4 $\pm$71.0
	no smoothing	37.4 $\pm$6.29	37.4 $\pm$5.96	271.3 $\pm$66.8	**48.7** $\pm$0.92	**48.8** $\pm$1.40	286.6 $\pm$84.6
DELORTRANS1	$\sigma^2 = 0.1$	32.1 $\pm$4.86	32.0 $\pm$4.61	56.2 $\pm$11.8	**41.1** $\pm$4.16	**41.1** $\pm$4.77	38.6 $\pm$5.8
(log-linear)	1	**47.3** $\pm$5.96	**47.1** $\pm$5.88	132.2 $\pm$29.9	**46.5** $\pm$4.06	**46.7** $\pm$4.67	87.0 $\pm$12.1
	10	37.0 $\pm$4.35	37.1 $\pm$3.75	206.8 $\pm$59.5	**46.3** $\pm$5.07	**46.6** $\pm$5.63	201.7 $\pm$45.9
	no smoothing	36.3 $\pm$4.42	36.4 $\pm$3.99	287.9 $\pm$82.5	**46.0** $\pm$5.24	**46.2** $\pm$5.67	212.8 $\pm$119.4
		(initializer has no effect)					
supervised, JL	$\lambda = 10$	75.3 $\pm$0.31	75.0 $\pm$1.26				
(generative, sum-to-one)	1	75.9 $\pm$0.33	75.5 $\pm$1.06				
	0.1	76.0 $\pm$0.31	75.5 $\pm$1.15				
	no smoothing*	76.1 $\pm$0.34	75.3 $\pm$1.12				
supervised, CL†	$\sigma^2 = 0.1$	78.3 $\pm$0.22	77.8 $\pm$0.98	37.1 $\pm$1.9			
(log-linear)	1	79.5 $\pm$0.25	78.5 $\pm$0.72	99.6 $\pm$5.7			
	10	79.9 $\pm$0.24	78.6 $\pm$0.77	350.5 $\pm$54.4			

Table 2: MATCHLINGUIST results (directed attachment accuracy). The baseline (a reimplementation of Klein and Manning [2004]) is boxed. Trials that on average exceeded baseline performance are shown in bold face. Means across folds are shown, with standard deviation in small type. *Note that unsmoothed generative models can set some probabilities to zero which can result in no valid parses on some test examples; this counted toward errors. †Unsmoothed supervised CL training leads to weights that tend toward $\pm\infty$; such trials are omitted.

variables; the uniform model is of course *the* maximum entropy model. As for explicit task biases, it is better to incorporate these into the objective function than through clever initializers, which are hard to design and may interact unpredictably with a choice of numerical optimization method (after all, the initializer has influence only because the optimizer fails to escape local maxima).

Compared to Klein and Manning's clever initializer, the uniform initializer turned out empirically to port better to contrastive conditions, and tended to be more robust across cross-validation folds (see variances in small type in Table 2).

An important fact illustrated by our results is that smoothing can have a tremendous effect on the performance of a model. One well-performing model (DELORTRANS1 neighborhood, smoothed at $\sigma^2 = 1$, with Klein and Manning's initializer) is quite poor if the smoothing parameter is varied by an order of magnitude.

7 Future Work

The experiment described is circumstantial evidence—not a rigorous demonstration—of our claim that a contrastive objective is better correlated with performance on MATCHLINGUIST than EM's marginal likelihood criterion. Because both kinds of problems involve non-convex optimization, there is always a chance of good or bad luck with respect to local maxima. In future work, we hope to explore this question more rigorously, for a variety of problems, by comparing many solutions found by optimizing different criteria from a variety of starting points. A careful study of the non-convexity of these objective functions is also warranted.

In this work, we have not explored new features for grammar induction; however, by introducing a computationally tractable unsupervised estimation method for log-linear models, we have opened the door for such exploration. In particular, for natural language grammar induction to become widely useful, it will need to pay attention to words (rather than

parts-of-speech) and—for many languages—morphology. A morphology-based neighborhood might guide the learner to tree structures that enforce long-distance inflectional agreement. Other interesting models we hope to explore involve neighborhoods that treat function and content words differently. Novel uses of cross-lingual information are one exciting area where log-linear models are expected to be helpful [Kuhn, 2004; Smith and Smith, 2004], availing the learner of new information without requiring expensive synchronous grammar formalisms [Wu, 1997].

One may wonder about the relevance of word order-based neighborhoods (TRANS1, for instance) to languages that do not have strict word order. This is an open and important question, and we note that good probabilistic modeling of syntax for such languages may require a re-thinking of the models themselves [Hoffman, 1995] as well as good neighborhoods for learning (again, morphology may be helpful).

The neighborhoods we discussed are constructed by finite-state operations for tasks like MATCHLINGUIST, spelling correction, and punctuation restoration; we plan to explore neighborhoods for the latter two tasks. Another type of neighborhood can be defined for a specific *system*: define the neighborhood using mistakes made by the system and retrain it (or train a new component) to contrast the correct output with the system's own errors. Examples of this have been applied in acoustic modeling for speech recognition, where the neighborhood is a lattice containing acoustically-confusable words [Valtchev *et al.*, 1997]; the hidden variables are the alignments between speech segments and phones. Another example from speech recognition involves training a language model on lattices provided by an acoustic model [Vergyri, 2000; Roark *et al.*, 2004]; here the neighborhood is defined by the acoustic model's hypotheses and may be weighted. Neighborhood functions might also be iteratively modified to improve a system in a manner similar to bootstrapping [Yarowsky, 1995] and transformation-based learning [Brill, 1995].

Finally, we intend to address the "minimally" supervised paradigm in which a small amount of labeled data is available (see, e.g., Yarowsky [1995]). We envision a *mixed* objective function, with one term for fitting the labeled data and another for the unlabeled data—the latter could be a CE term.

8 Conclusion

We have described contrastive estimation, a novel generalization of parameter estimation methods that use unlabeled data. Contrastive estimation requires the choice of a neighborhood, which can be interpreted as a mapping from observations to classes of implicit negative evidence. CE moves probability mass from an example's deprecated neighborhood to the example itself. Many earlier approaches, including the EM algorithm, can be viewed as special cases of CE.

CE has several key advantages. First, it is particularly apt for log-linear models, which allow the incorporation of arbitrary features and dependencies into a probability model. Unsupervised estimation for log-linear models has, until now, been largely ignored due to the computational difficulties of the partition function. CE avoids those difficulties. Further, for models of sequence structure (such as WCFGs), marginalization over some kinds of neighborhoods (those expressible as lattices) is efficient using dynamic programming.

We introduced task-based neighborhoods. When estimating a model (with or without supervision), it is important to keep in mind its end use. This idea has been important in machine learning, inspiring conditional and discriminative approaches to parameter estimation. We have shown one way to apply the idea in unsupervised learning: choose a neighborhood that explicitly represents potential mistakes of the model, then train the model to avoid those mistakes.

We presented experimental results that show substantial improvement on the task of inducing dependency grammars to match human annotations. Our estimation methods performed far better than the EM algorithm (using the same features) and did not require clever initialization.

Finally, we have espoused a new view of grammar induction: hidden variables that are intended to model language in service of some end should be estimated with that end in mind. It may turn out that unsupervised learning is preferable to supervised learning, since the latent structure that is learned need not match anyone's intuition. Rather, the learned structure is learned precisely *because* it is helpful in service of that task.

References

[Abney, 1997] S. P. Abney. Stochastic attribute-value grammars. *Computational Linguistics*, 23(4):597–617, 1997.

[Adriaans, 1992] W. P. Adriaans. *Language Learning from a Categorial Perspective*. PhD thesis, Universiteit van Amsterdam, 1992.

[Altun *et al.*, 2003] Y. Altun, M. Johnson, and T. Hofmann. Investigating loss functions and optimization methods for discriminative learning of label sequences. In *Proc. of EMNLP*, 2003.

[Baker, 1979] J. K. Baker. Trainable grammars for speech recognition. In *Proc. of the Acoustical Society of America*, 1979.

[Baum *et al.*, 1970] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41:164–71, 1970.

[Brill, 1995] E. Brill. Unsupervised learning of disambiguation rules for part of speech tagging. In *Proc. of VLC*, 1995.

[Carroll and Charniak, 1992] G. Carroll and E. Charniak. Two experiments on learning probabilistic dependency grammars from corpora. Technical report, Department of Computer Science, Brown University, 1992.

[Charniak, 1993] E. Charniak. *Statistical Language Learning*. MIT Press, 1993.

[Chen and Rosenfeld, 2000] S. Chen and R. Rosenfeld. A survey of smoothing techniques for ME models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50, 2000.

[Clark, 2001] A. S. Clark. *Unsupervised Language Acquisition: Theory and Practice*. PhD thesis, University of Sussex, 2001.

[Collins, 2000] M. Collins. Discriminative reranking for natural language parsing. In *Proc. of ICML*, 2000.

[Crammer and Singer, 2001] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2(5):265–92, 2001.

[Dempster *et al.*, 1977] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood estimation from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[Eisner *et al.*, 2004] J. Eisner, E. Goldlust, and N. A. Smith. Dyna: A declarative language for implementing dynamic programs. In *Proc. of ACL* (companion volume), 2004.

[Hoffman, 1995] B. Hoffman. *The Computational Analysis of the Syntax and Interpretation of Free Word Order in Turkish.* PhD thesis, University of Pennsylvania, 1995.

[Hwa and Lopez, 2004] R. Hwa and A. Lopez. On the conversion of constituent parsers to dependency parsers. Technical Report TR-04-118, Department of Computer Science, University of Pittsburgh, 2004.

[Johnson, 2001] M. Johnson. Joint and conditional estimation of tagging and parsing models. In *Proc. of ACL*, 2001.

[Juang and Katagiri, 1992] B.-H. Juang and S. Katagiri. Discriminative learning for minimum error classification. *IEEE Trans. Signal Processing*, 40:3043–54, 1992.

[Kernighan *et al.*, 1990] M. D. Kernighan, K. W. Church, and W. A. Gale. A spelling correction program based on a noisy channel model. In *Proc. of COLING*, 1990.

[Klein and Manning, 2002a] D. Klein and C. D. Manning. Conditional structure vs. conditional estimation in NLP models. In *Proc. of EMNLP*, 2002.

[Klein and Manning, 2002b] D. Klein and C. D. Manning. A generative constituent-context model for improved grammar induction. In *Proc. of ACL*, 2002.

[Klein and Manning, 2004] D. Klein and C. D. Manning. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proc. of ACL*, 2004.

[Kuhn, 2004] J. Kuhn. Experiments in parallel-text based grammar induction. In *Proc. of ACL*, 2004.

[Lafferty *et al.*, 2001] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*, 2001.

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

[Levenshtein, 1965] V. Levenshtein. Binary codes capable of correcting spurious insertions and deletions of ones. *Problems of Information Transmission*, 1:8–17, 1965.

[Liu and Nocedal, 1989] D. C. Liu and J. Nocedal. On the limited memory method for large scale optimization. *Mathematical Programming B*, 45(3):503–28, 1989.

[Merialdo, 1994] B. Merialdo. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–72, 1994.

[Miyao and Tsujii, 2002] Y. Miyao and J. Tsujii. Maximum entropy estimation for feature forests. In *Proc. of HLT*, 2002.

[Pereira and Schabes, 1992] F. C. N. Pereira and Y. Schabes. Inside-outside reestimation from partially bracketed corpora. In *Proc. of ACL*, 1992.

[Riezler *et al.*, 2000] S. Riezler, D. Prescher, J. Kuhn, and M. Johnson. Lexicalized stochastic modeling of constraint-based grammars using log-linear measures and EM training. In *Proc. of ACL*, 2000.

[Riezler, 1999] S. Riezler. *Probabilistic Constraint Logic Programming*. PhD thesis, Universität Tübingen, 1999.

[Roark *et al.*, 2004] B. Roark, M. Saraclar, M. Collins, and M. Johnson. Discriminative language modeling with conditional random fields and the perceptron algorithm. In *Proc. of ACL*, 2004.

[Smith and Eisner, 2004] N. A. Smith and J. Eisner. Annealing techniques for unsupervised statistical language learning. In *Proc. of ACL*, 2004.

[Smith and Eisner, 2005] N. A. Smith and J. Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Proc. of ACL*, 2005.

[Smith and Smith, 2004] D. A. Smith and N. A. Smith. Bilingual parsing with factored estimation: Using English to parse Korean. In *Proc. of EMNLP*, 2004.

[Valtchev *et al.*, 1997] V. Valtchev, J. J. Odell, P. C. Woodland, and S. J. Young. MMIE training of large vocabulary speech recognition systems. *Speech Communication*, 22(4):303–14, 1997.

[van Zaanen, 2002] M. van Zaanen. *Bootstrapping Structure into Language: Alignment-Based Learning*. PhD thesis, University of Leeds, 2002.

[Vergyri, 2000] D. Vergyri. *Integration of Multiple Knowledge Sources in Speech Recognition using Minimum Error Training*. PhD thesis, Johns Hopkins University, 2000.

[Wang *et al.*, 2002] S. Wang, R. Rosenfeld, Y. Zhao, and D. Schuurmans. The latent maximum entropy principle. In *Proc. of ISIT*, 2002.

[Wu, 1997] D. Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404, 1997.

[Yarowsky, 1994] D. Yarowsky. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proc. of ACL*, 1994.

[Yarowsky, 1995] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. of ACL*, 1995.