Journal of Machine Learning Research 11 (2010) 3017-3051 Submitted 4/10; Revised 9/10; Published 11/10

Covariance in Unsupervised Learning of Probabilistic Grammars

Shay B. Cohen SCOHEN@CS.CMU.EDU
Noah A. Smith NASMITH@CS.CMU.EDU
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA

Editors: Alex Clark, Dorota Glowacka, Colin de la Higuera, Mark Johnson and John Shawe-Taylor

Abstract

Probabilistic grammars offer great flexibility in modeling discrete sequential data like natural language text. Their symbolic component is amenable to inspection by humans, while their probabilistic component helps resolve ambiguity. They also permit the use of well-understood, general-purpose learning algorithms. There has been an increased interest in using probabilistic grammars in the Bayesian setting. To date, most of the literature has focused on using a Dirichlet prior. The Dirichlet prior has several limitations, including that it cannot directly model covariance between the probabilistic grammar's parameters. Yet, various grammar parameters are expected to be correlated because the elements in language they represent share linguistic properties. In this paper, we suggest an alternative to the Dirichlet prior, a family of logistic normal distributions. We derive an inference algorithm for this family of distributions and experiment with the task of dependency grammar induction, demonstrating performance improvements with our priors on a set of six treebanks in different natural languages. Our covariance framework permits soft parameter tying within grammars and *across* grammars for text in different languages, and we show empirical gains in a novel learning setting using bilingual, non-parallel data.

Keywords: dependency grammar induction, variational inference, logistic normal distribution, Bayesian inference

1. Introduction

One of the motivating applications for grammar induction, or unsupervised grammatical structure discovery, is for the syntactic analysis of text data. Grammar induction, in that case, may lead to the automatic acquisition of linguistic knowledge and the automatic construction of linguistic analyzers for under-studied text domains and languages, without the costly construction of manually annotated corpora. Grammar induction may also shed light on the cognitive process of language acquisition in humans.

When it comes to the problem of grammar induction from natural language data, a fruitful research direction has built on the view of a grammar as a parameterized, generative process explaining the data (Pereira and Schabes, 1992; Carroll and Charniak, 1992; Chen, 1995; Klein and Manning, 2002, 2004, *inter alia*). If the grammar is a probability model, then learning a grammar means selecting a model from a prespecified model *family*. In much prior work, the family is defined as the set of probabilistic grammar for a fixed set of grammar rules, so that grammar learning amounts to *parameter estimation* from incomplete data: sentences in the language are yields of hid-

©2010 Shay B. Cohen and Noah A. Smith.

den derivations from the grammar. Baker (1979) and Lari and Young (1990) describe how dynamic programming (the "inside-outside" algorithm) can be used within an Expectation-Maximization algorithm (Dempster et al., 1977) to estimate the grammar's probabilities from a corpus of text, in the context-free case.

Probabilistic grammars are attractive for several reasons. Like symbolic grammars, they are amenable to inspection by humans, so that it is relatively easy to understand what tendencies the model has captured if the underlying rules are understandable. Unlike purely symbolic grammars, they model frequency and provide a mechanism for reasoning in the face of ambiguity, which is ubiquitous in natural language. Probabilistic grammars can be specialized (e.g., as hidden Markov models for sequential structures) and generalized (e.g., as lexicalized grammars, as synchronous models over tuples of strings, and as grammars in context-sensitive classes). Probabilistic grammars are widely used to build models in natural language processing from *annotated* data, thus allowing easy comparison between unsupervised and supervised techniques. NLP applications of probabilistic grammars and their generalizations include parsing (Collins, 2003; Klein and Manning, 2003; Charniak and Johnson, 2005), machine translation (Wu, 1997; Ding and Palmer, 2005; Chiang, 2005), and question answering (Wang et al., 2007). Probabilistic grammars are probabilistic models, so they permit the use of well-understood methods for learning.

Meanwhile, in machine learning, significant attention has recently been devoted to Bayesian models. The attraction of Bayesian models is that they manage uncertainty in the face of learning from incomplete data, while permitting the use of background knowledge, in the form of a *prior* over models. This prior can be used to inject bias into a model. Such bias can be especially important in cases where the sample size is not large or when the grammar is highly non-identifiable, two scenarios that hold with grammar induction (see Cohen and Smith, 2010, for a discussion of the size of sample required for estimation of probabilistic grammars).

Bayesian methods have been applied to probabilistic grammars in various ways: specifying priors over the parameters of a PCFG (Johnson et al., 2007; Headden et al., 2009) as well as over the *states* in a PCFG (Finkel et al., 2007; Liang et al., 2007), and even over grammatical derivation structures larger than context-free production rules (Johnson et al., 2006; Cohn et al., 2009). The challenge in Bayesian grammar learning is efficiently approximating probabilistic inference. Variational approximations (Johnson, 2007; Kurihara and Sato, 2006) and randomized sampling approximations (Johnson et al., 2006; Goldwater, 2006) are typically applied.

Much of the Bayesian literature and its application to probabilistic grammars has focused on *conjugate priors* in the form of Dirichlet distributions. Conjugate priors were introduced by Raiffa and Schaifer (1961), who gave a desiderata for prior families, including analytical tractability. We argue that the literature has focused on this desideratum only, ignoring expressive power and interpretability. We begin by motivating the modeling of *covariance* among the probabilities of grammar derivation events, and propose the use of logistic normal distributions (Aitchison, 1986; Blei and Lafferty, 2006) over multinomials to build priors over grammars (Section 3). Our motivation relies on the observation that various grammar parameters are expected to be correlated because of the elements in language they represent share linguistic properties. Noting that grammars are built out of a large collection of multinomials, we introduce *shared* logistic normal distributions to allow *arbitrary* covariance among any grammar probabilities. We then describe efficient inference techniques to support decoding and learning with (shared) logistic normal priors over grammars (Section 4), facing the challenge of non-conjugacy of the logistic normal prior to the multinomial family. We experiment with probabilistic dependency grammar induction from data in six lan-

guages, showing how the new approach performs compared to non-Bayesian alternatives as well as more traditional Dirichlet prior-based alternatives (Section 5.1 and Section 5.2). We then demonstrate that the approach can also be effective when learning from *multilingual*, non-parallel text, softly tying parameters across languages (Section 5.4).

The research results in this paper build on work previously reported by Cohen et al. (2008) and Cohen and Smith (2009). Here we provide a more extensive discussion of the techniques, connections to related work, a full derivation of the variational inference algorithms, and a larger set of experiments on more data sets.

2. Probabilistic Grammars

We begin by discussing the general family of probabilistic grammars to which our methods are applicable. A probabilistic grammar defines a probability distribution over a certain kind of structured object (a derivation of the underlying symbolic grammar) explained step-by-step as a stochastic process. HMMs, for example, can be understood as a random walk through a probabilistic finite-state network, with an output symbol sampled at each state. PCFGs generate phrase-structure trees by recursively rewriting nonterminal symbols as sequences of "child" symbols (each itself either a nonterminal symbol or a terminal symbol analogous to the emissions of an HMM). Our experiments will consider a particular family of PCFGs that represent dependency structure (see Section 2.2).

Each step or emission of an HMM and each rewriting operation of a PCFG is conditionally independent of the others given a single structural element (one HMM or PCFG state); this Markov property permits efficient inference over derivations given a string.

In general, a probabilistic grammar defines the joint probability of a string $\mathbf{x}$ and a grammatical derivation $\mathbf{y}$:[1]

$$p(\mathbf{x},\mathbf{y} \mid \theta) = \prod_{k=1}^{K}\prod_{i=1}^{N_k} \theta_{k,i}^{f_{k,i}(\mathbf{x},\mathbf{y})} = \exp \sum_{k=1}^{K}\sum_{i=1}^{N_k} f_{k,i}(\mathbf{x},\mathbf{y}) \log \theta_{k,i}, \tag{1}$$

where $f_{k,i}$ is a function that "counts" the number of times the kth distribution's ith event occurs in the derivation. The parameters θ are a collection of K multinomials $\langle \theta_1, \ldots, \theta_K \rangle$, the kth of which includes N_k competing events. Letting $\theta_k = \langle \theta_{k,1}, \ldots, \theta_{k,N_k} \rangle$, each $\theta_{k,i}$ is a probability, such that

$$\forall k, \forall i, \qquad \theta_{k,i} \geq 0, \tag{2}$$

$$\forall k, \qquad \sum_{i=1}^{N_k} \theta_{k,i} = 1. \tag{3}$$

As is often the case in probabilistic modeling, there are different ways to carve up the random variables. We can think of $\mathbf{x}$ and $\mathbf{y}$ as correlated structure variables (often $\mathbf{x}$ is known if $\mathbf{y}$ is known), or the derivation event counts $\mathbf{f}(\mathbf{x},\mathbf{y}) = \langle f_{k,i}(\mathbf{x},\mathbf{y}) \rangle_{1 \leq k \leq K, 1 \leq i \leq N_k}$ as an integer-vector random variable (useful for variational inference, in Section 4). In this paper, $\mathbf{x}$ is always observable and $\mathbf{y}$ is hidden until we use gold standard data for testing.

Note that there may be many derivations $\mathbf{y}$ for a given string $\mathbf{x}$—perhaps even infinitely many in some kinds of grammars. For HMMs, there are three kinds of multinomials: a starting state multinomial, a transition multinomial per state and an emission multinomial per state. In that case $K = 2s+1$, where s is the number of states. The value of N_k depends on whether the kth multinomial is the starting state multinomial (in which case $N_k = s$), transition multinomial ($N_k = s$) or

1. A table of notation can be found in Appendix A, Table 4, page 3042.

emission multinomial ($N_k = t$, with t being the number of symbols in the HMM). For PCFGs, each multinomial among the K multinomials correspond to a set of N_k context-free rules headed by the same nonterminal. $\theta_{k,i}$ is then the probability of the ith rule for the kth nonterminal.

The field of grammatical inference also includes algorithms and methods for learning the *structure* of a (formal) language generator or grammar (Angluin, 1988; de la Higuera, 2005; Clark and Thollard, 2004; Clark et al., 2008, *inter alia*). This paper is complementary, focusing on the estimation of the *weights* assigned to the grammar's rules. The choice of using a fixed model family corresponds to a choice to work in a statistical parametric setting; extensions to nonparametric settings are possible (Goldwater, 2006; Johnson et al., 2006; Cohen et al., 2010). We focus on grammars which generate dependency structures for derivations. Dependency syntax is a popular representation that has been found useful in a wide range of natural language applications, including machine translation (Lin, 2004; Gimpel and Smith, 2009), question answering (Wang et al., 2007), as well as deeper semantic processing tasks (Johansson and Nugues, 2007; Das et al., 2010). The grammars used in our experiments are extremely permissive, allowing every possible dependency structure for a sentence (see Section 2.2).

2.1 Simple Example: Class-Based Unigram Model

It is helpful to keep in mind a simple model with a relatively small number of parameters such as a class-based unigram model (Brown et al., 1990). Let the observed symbols in $\mathbf{x}$ range over words in some language's vocabulary Γ. Let each word token x_i have an associated word class from a finite set Λ, denoted y_i; the y_i are all hidden. The derivation in this model is the sequence $\langle y_1, \ldots, y_n \rangle$. The probabilistic model consists of two parts:

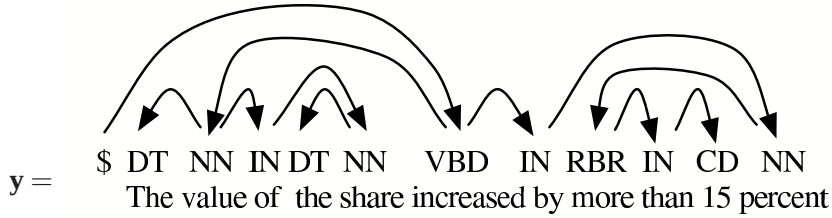
1. For all $y \in \Lambda \cup \{\text{stop}\}$, $\theta_{\text{c}}(y)$ is the probability that the next word will be generated by class y. $\theta_{\text{c}}(\text{stop})$ is the stopping probability.

2. For all $y \in \Lambda$ and all $x \in \Gamma$, $\theta_{\text{w}}(x \mid y)$ is the conditional probability that class y will generate word x.

In this simple model, $K = 1 + |\Lambda|$, $N_1 = |\Lambda|$, and for $k > 1$, $N_k = |\Gamma|$. This model can be thought of as a hidden Markov model with zero order, that is, it has no dependencies between the different hidden states. In addition, if we place a Dirichlet prior on the grammar parameters θ (Section 3.1) and treat θ as a latent variable sampled once per document, this model becomes equivalent to the latent Dirichlet allocation model (Blei et al., 2003). (Our θ_{w} is denoted β in their notation.) In this case, the derivation vector $\mathbf{y}$ corresponds to a set of topics selected for each word in the bag of words representing the document.

2.2 Dependency Model with Valence

Dependency grammar (Tesnière, 1959) refers to linguistic theories that posit graphical representations of sentences in which words are vertices and the syntax is a directed tree. Such grammars can be context-free or context-sensitive in power, and they can be made probabilistic (Gaifman, 1965). Dependency syntax is used in information extraction, machine translation, question answering, and other natural language processing applications. Our experiments perform unsupervised induction of probabilistic dependency grammars using a model known as "dependency model with valence" (Klein and Manning, 2004). The model is a probabilistic split head automaton grammar



$\mathbf{y} =$

$\mathbf{x} = \langle \$ \text{ DT NN IN DT NN VBD IN RBR IN CD NN} \rangle$

$$
\begin{aligned}
p(\mathbf{x}, \mathbf{y} \mid \theta) &= \theta_{\mathsf{c}}(\text{VBD} \mid \$, \mathsf{r}) \times p(\mathbf{y}^{(1)} \mid \text{VBD}, \theta) \\
p(\mathbf{y}^{(1)} \mid \text{VBD}, \theta) &= \theta_{\mathsf{s}}(\neg\text{stop} \mid \text{VBD}, \mathsf{l}, \mathsf{f}) \times \theta_{\mathsf{c}}(\text{NN} \mid \text{VBD}, \mathsf{l}) \times p(\mathbf{y}^{(2)} \mid \text{NN}, \theta) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{VBD}, \mathsf{l}, \mathsf{t}) \times \theta_{\mathsf{s}}(\neg\text{stop} \mid \text{VBD}, \mathsf{r}, \mathsf{f}) \times \theta_{\mathsf{c}}(\text{IN} \mid \text{VBD}, \mathsf{r}) \\
&\quad \times p(\mathbf{y}^{(4)} \mid \text{IN}, \theta) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{VBD}, \mathsf{r}, \mathsf{t}) \\
p(\mathbf{y}^{(2)} \mid \text{NN}, \theta) &= \theta_{\mathsf{s}}(\neg\text{stop} \mid \text{NN}, \mathsf{l}, \mathsf{f}) \times \theta_{\mathsf{c}}(\text{DT} \mid \text{NN}, \mathsf{l}) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{DT}, \mathsf{r}, \mathsf{f}) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{DT}, \mathsf{l}, \mathsf{f}) \theta_{\mathsf{c}}(\text{IN} \mid \text{NN}, \mathsf{r}) \times p(\mathbf{y}^{(3)} \mid \text{IN}, \theta) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{IN}, \mathsf{l}, \mathsf{f}) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{NN}, \mathsf{l}, \mathsf{t}) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{NN}, \mathsf{r}, \mathsf{t}) \\
p(\mathbf{y}^{(3)} \mid \text{IN}, \theta) &= \theta_{\mathsf{s}}(\neg\text{stop} \mid \text{IN}, \mathsf{r}, \mathsf{f}) \times \theta_{\mathsf{c}}(\text{NN} \mid \text{IN}, \mathsf{r}) \times \theta_{\mathsf{c}}(\text{DT} \mid \text{NN}, \mathsf{l}) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{DT}, \mathsf{r}, \mathsf{f}) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{DT}, \mathsf{l}, \mathsf{f}) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{NN}, \mathsf{r}, \mathsf{f}) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{NN}, \mathsf{l}, \mathsf{t}) \\
p(\mathbf{y}^{(4)} \mid \text{IN}, \theta) &= \theta_{\mathsf{s}}(\text{stop} \mid \text{IN}, \mathsf{l}, \mathsf{f}) \times \theta_{\mathsf{s}}(\neg\text{stop} \mid \text{IN}, \mathsf{r}, \mathsf{f}) \times \theta_{\mathsf{c}}(\text{NN} \mid \text{IN}, \mathsf{r}) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{NN}, \mathsf{r}, \mathsf{f}) \times \theta_{\mathsf{s}}(\neg\text{stop} \mid \text{NN}, \mathsf{l}, \mathsf{f}) \times \theta_{\mathsf{c}}(\text{RBR} \mid \text{NN}, \mathsf{r}) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{RBR}, \mathsf{l}, \mathsf{f}) \times p(\mathbf{y}^{(5)} \mid \text{RBR}, \theta) \\
p(\mathbf{y}^{(5)} \mid \text{RBR}, \theta) &= \theta_{\mathsf{s}}(\neg\text{stop} \mid \text{RBR}, \mathsf{r}, \mathsf{f}) \times \theta_{\mathsf{c}}(\text{IN} \mid \text{RBR}, \mathsf{r}) \times \theta_{\mathsf{c}}(\text{CD} \mid \text{IN}, \mathsf{r}) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{IN}, \mathsf{l}, \mathsf{f}) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{IN}, \mathsf{r}, \mathsf{t}) \times \theta_{\mathsf{s}}(\text{stop} \mid \text{CD}, \mathsf{r}, \mathsf{f}) \\
&\quad \times \theta_{\mathsf{s}}(\text{stop} \mid \text{CD}, \mathsf{l}, \mathsf{f})
\end{aligned}
$$

Figure 1: An example of a dependency tree (derivation $\mathbf{y}$). and its probability. The part-of-speech tags NN, VBD, DT, CD, RBR, and IN denote noun, past-tense verb, determiner, number, comparative adverb, and preposition, respectively, following Penn Treebank conventions. We break the probability of the tree down into recursive parts, one per head word, marked in blue (lighter). l, r, t, and f denote left, right, true, and false, respectively (see Equation 4).

(Alshawi and Buchsbaum, 1996) that renders inference cubic in the length of the sentence (Eisner, 1997). The language of the grammar is context-free, though our models are permissive and allow the derivation of any string in Γ^*. This is a major point of departure between theoretical work in grammatical inference and work on natural language text, particularly using probabilistic gram-

mars; our goal is to induce a distribution over derivations so that the most likely derivations under the model closely mimic those preferred by linguists (Smith and Eisner, 2005).

"Valence" here refers to the number of arguments controlled by the head of a phrase.[2] In the DMV, each word has a binomial distribution over whether it has at least one left child (similarly on the right), and a geometric distribution over the number of further children (for each side).

Let $\mathbf{x} = \langle x_1, x_2, ..., x_n \rangle$ be a sentence (here, as in prior work, represented as a sequence of part-of-speech tags). x_0 is a special "wall" symbol, \$, on the left of every sentence. A tree $\mathbf{y}$ is defined by a pair of functions $\mathbf{y}_{left}$ and $\mathbf{y}_{right}$ (both $\{0,1,2,...,n\} \to 2^{\{1,2,...,n\}}$) that map each word to its sets of left and right dependents, respectively. Here, the graph is constrained to be a *projective* tree rooted at $x_0 = \$$: each word except \$ has a single parent, and there are no cycles or crossing dependencies. $\mathbf{y}_{left}(0)$ is taken to be empty, and $\mathbf{y}_{right}(0)$ contains the sentence's single head. Let $\mathbf{y}^{(i)}$ denote the subtree rooted at position i (i.e., $\mathbf{y}^{(i)}$ is a tree consisting of all descendents of x_i in the tree $\mathbf{y}$). The probability $P(\mathbf{y}^{(i)} \mid x_i, \theta)$ of generating this subtree, given its head word x_i, is defined recursively:

$$p(\mathbf{y}^{(i)} \mid x_i, \theta) = \prod_{D \in \{left, right\}} \theta_{\mathrm{s}}(\mathrm{stop} \mid x_i, D, [\mathbf{y}_D(i) = \emptyset]) \times \prod_{j \in \mathbf{y}_D(i)} \theta_{\mathrm{s}}(\neg\mathrm{stop} \mid x_i, D, \mathrm{first}_{\mathbf{y}}(j)) \times \theta_{\mathrm{c}}(x_j \mid x_i, D) \times p(\mathbf{y}^{(j)} \mid x_j, \theta), \tag{4}$$

where $\mathrm{first}_{\mathbf{y}}(j)$ is a predicate defined to be 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 $p(\mathbf{x}, \mathbf{y} \mid \theta) = p(\mathbf{y}^{(0)} \mid \$, \theta)$. The parameters θ are the conditional multinomial distributions $\theta_{\mathrm{s}}(\cdot \mid \cdot, \cdot, \cdot)$ and $\theta_{\mathrm{c}}(\cdot \mid \cdot, \cdot)$. To follow the general setting of Equation 1, we index these distributions as $\theta_1, ..., \theta_K$. Figure 1 shows a dependency tree and its probability under this model (Equation 4).

Note that if all weights θ are greater than zero, the model permits *any* dependency tree over *any* sentence in Γ^*. Hence the goal of grammar induction is to model the *distribution* of derivations, not to separate grammatical strings or derivations from ungrammatical ones.

Klein and Manning's (2004) dependency model with valence is widely recognized as an effective probabilistic grammar for dependency grammar induction. Many recent studies on dependency grammar induction use it. For example, this model has been used to test estimation algorithms such as Viterbi EM (Spitkovsky et al., 2010b), contrastive estimation (Smith and Eisner, 2005), and algorithms which gradually introduce more data to the learning process (Spitkovsky et al., 2010a); it has been used to test the efficacy of multilingual learning through dependency grammar induction (Ganchev et al., 2009; Berg-Kirkpatrick and Klein, 2010); it has been used as a base model that has inspired more complex lexicalized models (Headden et al., 2009). The DMV has also been used as a base model within various estimation techniques with the goal of improving its performance by relying on other properties of language and text such as: dependencies between parameters in the model (Berg-Kirkpatrick et al., 2010), sparsity (Gillenwater et al., 2010), preference for short attachments (Smith and Eisner, 2006), and additional annotation offered by hypertext markup as found on the Web (Spitkovsky et al., 2010c). In addition, the DMV is related to the head-outward model used by Collins (2003) for supervised parsing; Collins' parser is one of the best performing parsers for English. In the rest of the paper, we assume we have a fixed grammar $\mathbf{G}$ for which we estimate the parameters.

2. Here, we refer to "head of a phrase" as in the linguistic sense—the word in a phrase that determines the syntactic category of this phrase.

2.3 Parameter Estimation by Maximum Likelihood

In the original framework, Klein and Manning (2004) treated the DMV as a model on its own, and also in combination with a model over bracketing structures called the "constituent-context model." Here we consider the DMV on its own as it is more capable of generalization and better exemplifies probabilistic grammars.

Klein and Manning learned the DMV using maximum likelihood estimation, carried out by the Expectation-Maximization (EM) algorithm. Because EM for probabilistic grammars has been well documented elsewhere (Lari and Young, 1990; Pereira and Schabes, 1992; Carroll and Charniak, 1992), we only briefly mention that it proceeds by alternating between two steps that update the model parameters. Let $\theta^{(t)}$ denote their values at time step t.

1. E-step: For each training example $\mathbf{x}$, infer the posterior distribution $p(\mathbf{y} \mid \mathbf{x}, \theta^{(t)}, \mathbf{G}) = p(\mathbf{x}, \mathbf{y} \mid \theta^{(t)}) / p(\mathbf{x} \mid \theta^{(t)}, \mathbf{G})$. This is accomplished by dynamic programming (for HMMs, the forward-backward algorithm; for PCFGs, the inside-outside algorithm; for the DMV, an algorithm due to Eisner, 1997), and the result is usually represented as a vector of derivation event expected frequencies, $\langle \mathbb{E}_{p(\cdot \mid \mathbf{x}, \theta^{(t)}, \mathbf{G})} f_{k,i}(\mathbf{x}, \cdot) \rangle_{k,i}$.

2. M-step: Estimate $\theta^{(t+1)}$ from the expected frequencies, as if they were observed frequencies. Since the model is built out of multinomials, there is a closed form solution obtained by normalizing the frequencies.

It is helpful to consider the problem EM iterations aim to solve in its declarative form, the problem of maximizing likelihood:

$$\max_{\theta} p(\mathbf{x} \mid \theta, \mathbf{G}) = \max_{\theta} \sum_{\mathbf{y}} p(\mathbf{x}, \mathbf{y} \mid \theta, \mathbf{G}).$$

(In fact, EM only locally maximizes this function.) In the above, we suppress the collection of sentences constituting the training data; to be precise, we should take a product of probabilities or a sum of log-probabilities for all training examples:

$$\max_{\theta} \prod_{m=1}^{M} \sum_{\mathbf{y}} p(\mathbf{x}_m, \mathbf{y} \mid \theta, \mathbf{G}).$$

In the Bayesian approach, we treat θ not as a set of parameters to be estimated, but rather as a random event. This contrasts with earlier research that aims to bias the grammar learner with prior information. Klein and Manning (2004), for example, biased the learner by initializing EM with a "harmonic" posterior over dependency attachments that preferred linking words that are closer together in the string to more distant words. Smith and Eisner (2006) more explicitly biased EM by manipulating the posterior calculated in the E-step with penalties for longer dependency attachments or, in an alternative model that permitted disconnected graphs, for contiguity.

3. Bayesian Models over Grammars

An attractive way to incorporate prior knowledge about grammars is through a prior distribution over the grammar's probabilities θ. Priors are often used to obtain *smooth* estimates; Smith (2006)

Model I:	Model II:
For $m \in \{1, \ldots, M\}$:	1. Draw θ from the prior $p(\theta \mid \mathbf{G}, \ldots)$.
1. Draw θ_m from the prior $p(\theta \mid \mathbf{G}, \ldots)$.	2. For $m \in \{1, \ldots, M\}$:
2. Draw $(\mathbf{x}_m, \mathbf{y}_m)$ from $p(\mathbf{x}_m, \mathbf{y}_m \mid \theta_m, \mathbf{G})$.	Draw $(\mathbf{x}_m, \mathbf{y}_m)$ from $p(\mathbf{x}_m, \mathbf{y}_m \mid \theta, \mathbf{G})$.

Figure 2: Two variations on Bayesian modeling of probabilistic grammars.

explored symmetric Dirichlet priors in the DMV in a maximum *a posteriori* framework for learning that can still be accomplished by EM:

$$\max_{\theta} p(\theta \mid \alpha, \mathbf{G}) \prod_{m=1}^{M} \sum_{\mathbf{y}} p(\mathbf{x}_m, \mathbf{y} \mid \theta, \mathbf{G}), \tag{5}$$

where α denotes the parameters of the prior over grammars. EM is efficient when $p(\theta \mid \alpha, \mathbf{G})$ is a collection of Dirichlet distributions with each $\alpha \geq 1$ (discussed below). For the moment, we leave aside the form of the prior, though it is a major focus of this article.

In this paper, we go farther. We treat θ as a hidden variable, not unlike $\mathbf{y}$. It will therefore be integrated out in defining the probability of the data:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_M \mid \alpha, \mathbf{G}) = \int p(\theta \mid \alpha, \mathbf{G}) \prod_{m=1}^{M} \sum_{\mathbf{y}} p(\mathbf{x}_m, \mathbf{y} \mid \theta, \mathbf{G}) \, d\theta. \tag{6}$$

In this setting, it is α, the distribution over grammar parameters, that encodes knowledge about the grammar, and it will be α that we estimate when we perform learning.

We consider two alternative variations on the Bayesian idea, illustrated in Figure 2. In the first, called "model I," the grammar's probabilities θ are drawn randomly once per sentence for the whole corpus $\mathbf{x}_1, \ldots, \mathbf{x}_M$. In "model II," the grammar parameters are drawn *once* for all of the sentences in the corpus.

Conceptually, both options have advantages and disadvantages when modeling natural language. Drawing θ for each derivation permits more flexibility across derivations, perhaps allowing the learner to capture variation across the corpus (even if not systematically, as the grammars are drawn IID), arising from different authors, for example. Generating θ only once suggests we need to do inference in a smaller space: we only need to find the posterior over a single θ, perhaps leading to better generalization. We will consider both forms in our experiments (Section 5.1).

The question of the choice of a prior distribution still remains. In their pioneering work about conjugate priors,[3] Raiffa and Schaifer (1961) set desiderata for prior distributions in parametric models. These desiderata, which serve as the foundation for *conjugate priors*, include: (i) analytical tractability—the posterior using a certain prior family should stay in the prior family, while it is reasonably easy to identify the posterior from a sample and a prior; (ii) richness—there should be a member in the prior family that is able to express the modeler's beliefs and prior information; (iii)

3. A prior family is conjugate for a family of distributions if the posterior over the family, after observing some data, is also in the prior family. See Raiffa and Schaifer (1961).

interpretability—the prior should be easily interpreted so the modeler can verify that the choice of prior matches prior judgments.

Unfortunately, much of the Bayesian literature for probabilistic grammars and even in general has diverged considerably from these desiderata, and focused only on the first requirement of analytical tractability. As a result, most of the Bayesian language learning literature has focused on Bayesian models with a Dirichlet prior (Johnson et al., 2007; Goldwater and Griffiths, 2007; Toutanova and Johnson, 2007; Kurihara and Sato, 2006, *inter alia*), which is conjugate to the multinomial family. We argue that the last two requirements are actually more important than the first one, which is motivated by mere mathematical and computational convenience. We suggest replacing the first requirement with "computational tractability"—it should be easy to represent the posterior (or an approximation of it) computationally. In that case, the modeler can focus on choosing *rich* priors that can more properly model different structural elements of a grammar. To solve the problem of inference, we can now use approximate inference algorithms such as the one we give in Section 4 and Appendix B. Indeed, approximations are sometimes required even for the conjugate case, and are always required when the data are incomplete.

We next give an overview of the Dirichlet prior that provides analytical tractability for probabilistic grammars, and then demonstrate the alternative which focuses on the second and third requirements, the logistic normal distribution. The logistic normal, we suggest, improves over the Dirichlet from the perspective of desideratum (ii), though we must take further steps to achieve sufficient "richness" to account for arbitrary covariance and for multilingual text data.

3.1 Dirichlet Distributions

From the computational perspective, the Dirichlet distribution is indeed a natural choice for a prior over the parameters of the grammar because of its analytical tractability, which makes inference more elegant and less computationally intensive in both the maximum *a posteriori* (Equation 5) and Bayesian (Equation 6) settings. In addition, a Dirichlet prior can encourage sparse solutions (i.e., many $\theta_{k,i} = 0$), a property which is desirable in natural language learning (Johnson et al., 2007), as it corresponds to eliminating unnecessary grammar rules. (Indeed, learning to exclude rules by setting their probabilities to zero is one way of going about symbolic grammar induction.)

If we use a Dirichlet distribution with a probabilistic grammar, then the hyperparameters for the grammar consist of K vectors with positive elements, the kth of which has length N_k. We denote these hyperparameters by α, in which case the prior over the grammar parameters θ has the form:

$$p(\theta \mid \alpha) = \prod_{k=1}^{K} \left(\frac{\prod_{i=1}^{N_k} \Gamma(\alpha_{k,i})}{\Gamma\left(\sum_{i=1}^{N_k} \alpha_{k,i}\right)} \prod_{i=1}^{N_k} \theta_{k,i}^{\alpha_{k,i}-1} \right) = B(\theta) \times \prod_{k=1}^{K} \prod_{i=1}^{N_k} \theta_{k,i}^{\alpha_{k,i}-1},$$

where $\Gamma(\cdot)$ is the Gamma function and $B(\theta)$ is a constant term with respect to θ.

Consider again the simple model of Section 2.1. If we embed it inside model I (Figure 2) we arrive exactly at the latent Dirichlet allocation model of Blei et al. (2003), where each example is a document (not a sentence).

The Dirichlet distribution can also be derived as a normalized set of variables of exponentiated *independent* Gamma-distributed variables. More precisely, for each multinomial θ_k ($k \in \{1, \ldots, K\}$), we can draw N_k independent random samples $v_{k,1}, \ldots, v_{k,N_k}$ from Gamma distributions with shapes

$\alpha_{k,1},\ldots,\alpha_{k,N_k}$, respectively, and scale 1 and then let:

$$\theta_{k,i} = \frac{v_{k,i}}{\sum_{i'=1}^{N_k} v_{k,i'}}.$$

This alternative representation of the Dirichlet distribution points to a weakness: there is no explicit covariance structure present when θ are drawn from a Dirichlet. The only way θ_k covary is through the normalization that maps $v_{k,i}$ to the probability simplex. In fact, the correlation between $\theta_{k,i}$ and $\theta_{k,i'}$ is always negative and equals $-\frac{(\alpha_{k,i}\alpha_{k,i'})^{1/2}}{((\alpha_{k,0}-\alpha_{k,i})(\alpha_{k,0}-\alpha_{k,i'}))^{1/2}}$ where $\alpha_{k,0} = \sum_{i=1}^{N_k}\alpha_{k,i}$. This relates back to the desiderata of Raiffa and Schaifer: the covariance (and in fact, variance) structure that the Dirichlet distribution offers is not rich. This is especially true when modeling language, as we explain in the section below.

3.2 Modeling Covariance with Logistic Normal Distributions

When we consider probabilistic grammars for natural languages, especially those over words or word classes like parts of speech, we *do* expect to see covariance structure. Intuitively, the probability of a particular word or word class having singular nouns as arguments is likely tied to the probability of the same word having *plural* nouns as arguments. Words that tend to attach to one type of parent are expected to tend to attach to similar parents. This follows because words and word classes tend to follow patterns. This is a large part of the empirical motivation for syntactic theories that make use of part of speech and phrase categories.

A natural candidate for a distribution that models covariance is the multivariate normal distribution. However, values drawn from the multivariate normal distribution can be both positive and negative, and they also do not necessarily normalize to 1, both are requirements from θ (see Equations 2–3). Aitchison (1986) suggested a logistic transformation on a multivariate normal variable to get values which correspond to points on the probabilistic simplex. He called it the "logistic normal" distribution.

The logistic normal (LN) distribution maps a $(d-1)$-dimensional multivariate Gaussian to a distribution on the d-dimensional probability simplex, $\{\langle z_1,\ldots,z_d\rangle \in \mathbb{R}^d : z_i \geq 0, \sum_{i=1}^d z_i = 1\}$, as follows:

1. Draw $\eta = \langle \eta_1,\ldots,\eta_{d-1}\rangle$ from a multivariate Gaussian with mean μ and covariance matrix Σ.
2. Let $\eta_d = 0$.
3. For $i \in \{1,\ldots,d\}$, let:
$$z_i = \frac{\exp \eta_i}{\sum_{j=1}^d \exp \eta_j}.$$

Drawing from a $(d-1)$-dimensional Gaussian preserves identifiability; a d-dimensional Gaussian would have an extra degree of freedom, allowing more than one outcome of η to lead to the same $\mathbf{z}$.

For probabilistic grammars, we define one LN distribution per multinomial. This gives a prior over each θ_k that permits covariance among $\langle \theta_{k,1},\ldots,\theta_{k,N_k}\rangle$.

Blei and Lafferty (2006) and Ahmed and Xing (2007) successfully used the LN distribution for topic models, extending the latent Dirichlet allocation model (Blei et al., 2003). In Cohen et al.

$$\left.\begin{array}{lclcl}
I_1 & = & \{1{:}2, 3{:}6, 7{:}9\} & = & \{\ I_{1,1},\ \ I_{1,2},\ \ I_{1,L_1}\ \} \\
I_2 & = & \{1{:}2, 3{:}6\} & = & \{\ I_{2,1},\ \ I_{2,L_2}\ \} \\
I_3 & = & \{1{:}4, 5{:}7\} & = & \{\ I_{3,1},\ \ I_{3,L_3}\ \} \\
I_N & = & \{1{:}2\} & = & \{\ I_{4,L_4}\ \} \\
 & & & & \quad J_1 \quad\ J_2 \quad\ J_K
\end{array}\right\} \text{prt. struct. } \mathcal{S}$$

$$\left.\begin{array}{lclcl}
\eta_1 & = & \langle \eta_{1,1}, \eta_{1,2},\ \eta_{1,3}, \eta_{1,4}, \eta_{1,5}, \eta_{1,6},\ \eta_{1,7}, \eta_{1,8}, \eta_{1,\ell_1} \rangle & \sim & \text{Normal}(\mu_1, \Sigma_1) \\
\eta_2 & = & \langle \eta_{2,1}, \eta_{2,2},\ \eta_{2,3}, \eta_{2,4}, \eta_{2,5}, \eta_{2,\ell_2} \rangle & \sim & \text{Normal}(\mu_2, \Sigma_2) \\
\eta_3 & = & \langle \eta_{3,1}, \eta_{3,2}, \eta_{3,3}, \eta_{3,4},\ \eta_{3,5}, \eta_{3,6}, \eta_{3,\ell_3} \rangle & \sim & \text{Normal}(\mu_3, \Sigma_3) \\
\eta_4 & = & \langle \eta_{4,1}, \eta_{4,\ell_4} \rangle & \sim & \text{Normal}(\mu_4, \Sigma_4)
\end{array}\right\} \text{sample } \eta$$

$$\left.\begin{array}{lcl}
\tilde{\eta}_1 & = & \frac{1}{3}\langle \eta_{1,1} + \eta_{2,1} + \eta_{4,1},\ \ \eta_{1,2} + \eta_{2,2} + \eta_{4,2} \rangle \\
\tilde{\eta}_2 & = & \frac{1}{3}\langle \eta_{1,3} + \eta_{2,3} + \eta_{3,1},\ \ \eta_{1,4} + \eta_{2,4} + \eta_{3,2},\ \ \eta_{1,5} + \eta_{2,5} + \eta_{3,3}, \\
 & & \eta_{1,6} + \eta_{2,6} + \eta_{3,4} \rangle \\
\tilde{\eta}_3 & = & \frac{1}{2}\langle \eta_{1,7} + \eta_{3,5},\ \ \eta_{1,8} + \eta_{3,6},\ \ \eta_{1,9} + \eta_{3,7} \rangle
\end{array}\right\} \text{combine } \eta$$

$$\left.\begin{array}{lcl}
\theta_1 & = & (\exp \tilde{\eta}_1) \Big/ \sum_{i'=1}^{N_1} \exp \tilde{\eta}_{1,i'} \\
\theta_2 & = & (\exp \tilde{\eta}_2) \Big/ \sum_{i'=1}^{N_2} \exp \tilde{\eta}_{2,i'} \\
\theta_3 & = & (\exp \tilde{\eta}_3) \Big/ \sum_{i'=1}^{N_3} \exp \tilde{\eta}_{3,i'}
\end{array}\right\} \text{softmax}$$

Figure 3: An example of a shared logistic normal distribution, illustrating Def. 1. $N = 4$ experts are used to sample $K = 3$ multinomials; $L_1 = 3$, $L_2 = 2$, $L_3 = 2$, $L_4 = 1$, $\ell_1 = 9$, $\ell_2 = 6$, $\ell_3 = 7$, $\ell_4 = 2$, $N_1 = 2$, $N_2 = 4$, and $N_3 = 3$. From top to bottom: the partition structure $\mathcal{S}$ describes I_j which tell how segment a normal expert into parts which are matched to multinomials ("prt. struct. $\mathcal{S}$"). Each normal expert is sampled from a multivariate normal ("sample η"), and then matched and averaged according to the partition strcture ("combine η"). The final step is exponentiating and normalizing η to get θ ("softmax"). This figure is best viewed in color.

(2008), we demonstrated how the LN distribution is an effective alternative to the Dirichlet for probabilistic dependency grammar induction in the Bayesian setting.

We note that the family of logistic normal distributions and the family of Dirichlet distributions are very different from each another. One cannot find two distributions from each class which are arbitrary close to each other in any meaningful sense. However, it can be shown (Aitchison, 1986) that given a Dirichlet distribution with very large α, we can find a logistic normal distribution such that the KL-divergence between the Dirichlet distribution and logistic normal distribution is small.

3.3 Sharing Across Multinomials

The LN distribution has an inherent limitation when we consider probabilistic models made up of more than one multinomial distribution, such as probabilistic grammars. Each multinomial is drawn separately from an independent Gaussian, so that covariance can only be imposed among events competing within one multinomial, not across multinomials. With the DMV, for example, the probability of a past-tense verb (VBD) having a noun as a right child might correlate with the

probability that other kinds of verbs (VBZ, VBN, etc.) have a noun as a right child. This correlation cannot be captured by the LN distribution, because the VBZ and VBN as parents are represented using their own multinomials over children, unrelated to that of VBD as a parent.

One way to mend this limitation is to define a single Gaussian over $N \triangleq \sum_{k=1}^{K} N_k$ variables with one $N \times N$ covariance matrix. Then, instead of applying the logistic transformation to the whole vector as a single multinomial, we can apply it to subvectors to get disjoint multinomials. When learning, the large covariance matrix captures correlations between all pairs of events in all multinomials. The induced distribution is called the *partitioned* logistic normal (PLN) distribution. It is a generalization of the LN distribution (see Aitchison, 1986).

In practice, creating a covariance matrix of size $N \times N$ is likely to be too expensive. DMV, for example, has $O(t^2)$ weights for a part-of-speech vocabulary of size t, requiring a very large multivariate normal distribution with $O(t^4)$ covariance parameters.

To solve this problem, we suggest a refinement of the class of PLN distributions. Instead of using a single normal vector for all of the multinomials, we use several normal vectors, partition each one and then *recombine* parts which correspond to the same multinomial, as an average. Next, we apply the logistic transformation on the mixed vectors (each of which is normally distributed as well). Figure 3 gives an example of a non-trivial case of using a SLN distribution, where three multinomials are generated from four normal experts.

We now formalize this notion. For a natural number N, we denote by $1{:}N$ the set $\{1,\ldots,N\}$. For a vector in $v \in \mathbb{R}^N$ and a set $I \subseteq 1{:}N$, we denote by v_I the vector created from v by using the coordinates in I. Recall that K is the number of multinomials in the probabilistic grammar, and N_k is the number of events in the kth multinomial. We define a shared logistic normal distribution with N "experts" over a collection of K multinomial distributions:

Definition 1 *Let $\eta_n \sim \mathrm{Normal}(\mu_n, \Sigma_n)$ be a set of multivariate normal variables for $n \in 1{:}N$, where the length of η_n is denoted ℓ_n. Let $I_n = \{I_{n,j}\}_{j=1}^{L_n}$ be a partition of $1{:}\ell_n$ into L_n sets, such that $\cup_{j=1}^{L_n} I_{n,j} = 1{:}\ell_n$ and $I_{n,j} \cap I_{n,j'} = \emptyset$ for $j \neq j'$. Let J_k for $k \in 1{:}K$ be a collection of (disjoint) subsets of $\{I_{n,j} \mid n \in 1{:}N, j \in 1{:}\ell_n, |I_{n,j}| = N_k\}$, such that all sets in J_k are of the same size, N_k. Let $\tilde{\eta}_k = \frac{1}{|J_k|}\sum_{I_{n,j} \in J_k} \eta_{n,I_{n,j}}$, and $\theta_{k,i} = \exp(\tilde{\eta}_{k,i}) \big/ \sum_{i'} \exp(\tilde{\eta}_{k,i'})$. We then say θ distributes according to the* shared logistic normal distribution *with* partition structure $\mathcal{S} = (\{I_n\}_{n=1}^{N}, \{J_k\}_{k=1}^{K})$ *and* normal experts $\{(\mu_n, \Sigma_n)\}_{n=1}^{N}$ *and denote it by* $\theta \sim \mathrm{SLN}(\mu, \Sigma, \mathcal{S})$.

The partitioned LN distribution in Aitchison (1986) can be formulated as a shared LN distribution where $N = 1$. The LN collection presented in Section 3.2 is the special case where $N = K$, each $L_n = 1$, each $\ell_k = N_k$, and each $J_k = \{I_{k,1}\}$.

We note that there is an issue with identifiability that we need to resolve with SLN distributions, as with the LN distribution. It is required that for all multinomials, we set the first value of the samples from the normal expert to 0. For simplicity, we did not include it explicitly in Definition 1, because this can be achieved by setting the normal expert's mean and variance values to 0 in the first index of each normal expert ($\eta_{n,1} = 0$ for all n).

The covariance among arbitrary $\theta_{k,i}$ is not defined directly; it is implied by the definition of the normal experts $\eta_{n,I_{n,j}}$, for each $I_{n,j} \in J_k$. We note that a SLN can be represented as a PLN by relying on the distributivity of the covariance operator, and merging all the partition structure into one (perhaps sparse) covariance matrix. SLNs, in that case, represent a subset of PLNs with a factored structure on the covariance matrices.

It is convenient to think of each $\eta_{i,j}$ as a weight associated with a unique event's probability, a certain outcome of a certain multinomial in the probabilistic grammar. By letting different $\eta_{i,j}$ covary with each other, we strengthen the relationships among $\theta_{k,j}$ and permit learning of the one to affect the learning of the other. Definition 1 also implies that we multiply several multinomials together in a product-of-experts style (Hinton, 1999), because the exponential of an average of normals becomes a product of (unnormalized) probabilities.

We note that the partition structure is a hyperparameter. In our experiments, it encodes domain knowledge about the languages we experiment with (Section 5.3). We believe this is a key advantage of SLN in this setting: marrying the notions of prior knowledge and a Bayesian prior. The beliefs of the model about a language can be encoded into a distribution over the parameters. We leave for future work the discovery of partition structure during the learning process.

3.4 Local Log-Linear Models over Parameters

We give now another interpretation of the shared logistic normal prior using a feature representation, which is related to recent work by Berg-Kirkpatrick et al. (2010). A probabilistic grammar with a shared logistic normal prior can be thought of as a probabilistic grammar where the grammar's parameters are themselves modeled using a local log-linear model with a Gaussian prior over the weights of this log-linear model. Let θ_k be a multinomial in the collection of multinomials for a probabilistic grammar. Then, according to Definition 1 we have:

$$\theta_{k,i} = \frac{\exp\left(g_k(i) \cdot \eta\right)}{Z_k(\eta)},$$

where η is a vector of length $\sum_{n=1}^{N} \ell_n$, a concatenation of all normal experts, and $g_k(i)$ is a feature vector, again of length $\sum_{n=1}^{N} \ell_n$, which is divided into subvectors $g_{k,n}(i)$ each of length ℓ_n. $g_{k,n,j}(i) = 1/|J_k|$ if the ith event in the kth multinomial uses the jth coordinate of the nth normal expert—that is, there exists an $I_{n,r} \in I_n \cap J_k$ such that $j \in I_{n,r}$ (according to Definition 1)—and 0 otherwise. The term $Z_k(\eta)$ is a normalization constant of the form:

$$Z_k(\eta) = \sum_{i'} \exp\left(g_k(i') \cdot \eta\right).$$

Note that the features in the local log-linear model refer to the *hyperparameters* of the SLN, more specifically, the partition structure. They do not refer to the observed data or the latent structural elements in the probabilistic grammar. These features have a Gaussian prior over them, represented by the normal experts' mean values and covariance matrices (μ and Σ). In that case, the Gaussian prior which we optimize during inference using empirical Bayes (Section 4) can be thought of as a quadratic penalty on the local log-linear weights. We note that in most cases in the literature, Gaussian priors (or L_2 regularizers) are used with mean value 0 and a uniform diagonal covariance matrix, in order to push feature weights to values close to 0. This is not the case with our model.

Berg-Kirkpatrick et al. (2010) used the idea of local log-linear models for several natural language processing tasks, including dependency grammar induction and part-of-speech tagging. Instead of using features that are based on a Gaussian prior, they used a set of ordinary binary features, which describe relationships between different parameters in a similar way to the ones presented in Section 5.3.

4. Inference and Learning

Having defined a family of probability models over grammars, we now consider the problem of inferring posterior distributions under this model. We first consider inference over $\mathbf{y}$, then over θ, then learning the parameters of the distribution over grammars in an empirical Bayesian framework.

4.1 Decoding: Inferring $\mathbf{y}$

Classical statistical approaches to language processing normally assume that inputs (here, sentences $\mathbf{x}$) are independently and identically distributed. Decoding is the problem of choosing an analysis (here, grammatical derivation $\mathbf{y}$) given the input. Most commonly this is accomplished by choosing the most probable analysis:

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}, \theta, \mathbf{G}) = \operatorname*{argmax}_{\mathbf{y}} p(\mathbf{x}, \mathbf{y} \mid \theta, \mathbf{G}). \tag{7}$$

This is commonly called "Viterbi" decoding, referring to the algorithm that accomplishes the maximization for hidden Markov models. An alternative is to choose the analysis that minimizes *risk*, or the expectation (under the model) of a cost function. Let $\text{cost}(\mathbf{y}, \mathbf{y}^*)$ denote the nonnegative cost of choosing analysis $\mathbf{y}$ when the correct analysis is $\mathbf{y}^*$.

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}} \mathbb{E}_{p(\cdot \mid \mathbf{x}, \theta, \mathbf{G})} \text{cost}(\mathbf{y}, \cdot) = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathbf{y}'} p(\mathbf{y}' \mid \mathbf{x}, \theta, \mathbf{G}) \ \text{cost}(\mathbf{y}, \mathbf{y}').$$

This is known as minimum Bayes risk (MBR) decoding.[4] For dependency parsing, the cost function counts the number of words attached to the wrong parent.

Decoding is a crucial step in evaluation of models of natural language. Typically for supervised and unsupervised models, decoding output is compared to expert human-annotated gold standard analyses, providing an objective measure of the quality of the learned model. Best practice measures quality on new test data unseen during training, to test the generalization ability of the learned model. This is an attractive approach to evaluating the quality of unsupervisedly induced grammars.

In the Bayesian setting, decoding might be accomplished using the posterior over derivations, marginalizing out the unknown grammar weights. For model I, Viterbi decoding would correspond to:

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}} p(\mathbf{y} \mid \alpha, \mathbf{G}) = \operatorname*{argmax}_{\mathbf{y}} \int \ p(\theta \mid \alpha, \mathbf{G}) p(\mathbf{x}, \mathbf{y} \mid \theta, \mathbf{G}) \ d\theta. \tag{8}$$

Unfortunately, there is no closed-form solution for the integral in Equation 8 and finding $\mathbf{y}^*$ is intractable. We therefore have to resort to approximate inference (Section 4.2). Model II creates dependence among the derivations of the different sentences in the training set, requiring a different inference procedure.

In this work, we consider three decoding techniques. The first takes a point estimate of θ and applies Viterbi decoding (Equation 7). The point estimate is derived using techniques discussed below. After estimating the μ (and the Σ), we use the logistic transformation on μ to obtain this point estimate for Viterbi decoding. Recall that for the DMV, decoding can be accomplished in cubic time using dynamic programming (Section 2.2).

4. In some cases, decoding selects only certain salient aspects of a derivation, such as the derived tree corresponding to a tree adjoining grammar's *derivation* tree. In such cases, Viterbi and/or MBR decoding may require approximations.

The second approach makes use of the same point estimate of θ, only with MBR decoding, as described above. The loss function we use is dependency attachment error, for the task of dependency grammar induction. MBR decoding in this case works as follows: using θ and the inside-outside algorithm, we compute the posterior probability of each dependency attachment (directed edge in the graph) being present in the grammatical derivation for the sentence. Then, we find the tree with the largest score, the score being the sum of the posterior probabilities of each edge present in the tree.

Neither Viterbi nor MBR decoding uses the entire distribution over grammar weights. In the LN case, for example, the covariance matrix Σ is ignored. We suggest "committee decoding," in which a set of randomly sampled grammar weights are drawn for each sentence to be parsed. The weights are drawn from the learned distribution over grammar weights, parameterized by μ and Σ in the LN case. Viterbi or MBR decoding can then be applied. Note that this decoding mechanism is randomized: we sample a grammar per sentence, and use it to decode. We apply this decoding mechanism ten times, and average performance. This decoding method is attractive because it has generalization error guarantees: in a PAC-Bayesian framework, it can be shown that the error of committee parsing on the sample given should be close to the expected error (see Seeger, 2002; McAllester, 2003; Banerjee, 2006).

4.2 Variational Inference with Logistic Normal Distributions

The lack of conjugacy of the logistic normal distribution to the multinomial family complicates the inference of distributions over θ and distributions over the hidden derivations $\mathbf{y}$ from the probabilistic grammar, given a sequence of observed sentences $x_1, \ldots, x_M$.

Mimno et al. (2008) explored inference with the logistic normal distribution using sampling with an auxiliary variable method. However, sampling is notoriously slow to converge, especially with complicated structures such as grammatical derivations. The algorithm Mimno et al. suggest is also rather complicated, while alternatives, such as mean-field variational inference (Wainwright and Jordan, 2008), offer faster convergence and a more intuitive solution to the problem of non-conjugacy of the logistic normal distribution.

Variational inference algorithms have been successfully applied to various grammar and syntax learning tasks (Kurihara and Sato, 2006; Liang et al., 2007; Headden et al., 2009; Boyd-Graber and Blei, 2010; Cohen et al., 2010, *inter alia*). We give the full technical details of mean-field variational inference for probabilistic grammars with logistic normal priors in Appendix B, and turn to give a brief overview of the main technical details next, under the simplifying assumption that we have a single observation $\mathbf{x}$.

Mean-field variational inference in the Bayesian setting relies on two principal approximations: the first approximation is done to the marginalized log-likelihood. Using Jensen's inequality and an auxiliary distribution $q(\theta, \mathbf{y})$, later to be used as our approximate posterior, we bound the log-likelihood, marginalizing out the parameters and the hidden derivations in the grammar:

$$\log \int \sum_{\mathbf{y}} p(\mathbf{x}, \mathbf{y}, \theta \mid \mu, \Sigma, \mathcal{S}, \mathbf{G})\, d\theta \geq \mathbb{E}_q[\log p(\mathbf{x}, \mathbf{y}, \theta \mid \mu, \Sigma, \mathcal{S}, \mathbf{G})]. \tag{9}$$

The goal of the approximation in Equation 9 is to derive a bound which is optimized with respect to q, instead of optimizing the marginalized log-likelihood, which is intractable. q serves as our approximate posterior.

The bound in Equation 9 requires further approximation, the mean-field approximation, to be tractable. This mean-field approximation states that $q(\theta, \mathbf{y})$ is factorized and has the following form:

$$q(\theta, \mathbf{y}) = q(\theta)q(\mathbf{y}).$$

The variational distributions, $q(\theta)$ and $q(\mathbf{y})$ can take an arbitrary form, as long as the bound in Equation 9 can be efficiently maximized with respect to these variational distributions. For the case of logistic normal priors, an additional approximation will be necessary (a first-order Taylor approximation to the log of the normalization of the logistic normal distribution), because of the lack of conjugacy of the logistic normal priors to the multinomial family (see Appendix B). We show in Appendix B that even though $q(\mathbf{y})$ can have an arbitrary form, in order to maximize the variational bound it needs to have the form of a probabilistic grammar, dominated by the grammar's variational parameters. This makes inference applicable through the use of an inside-outside algorithm with a weighted grammar of the same form as the original model. The mean-field approximation yields an elegant algorithm, which looks similar to the Expectation-Maximization algorithm (Section 2.3), alternating between optimizing the bound in Equation 9 with respect to $q(\theta)$ and with respect to $q(\mathbf{y})$.

4.3 Variational EM

The variational inference algorithm in Section 4.2 assumes that the μ and Σ are fixed. We are interested in obtaining an *estimate* for μ and Σ, so that we can fit the data and then use the learned model as described in Section 4.1 to decode new data (e.g., the test set in our experiments). To achieve this, we will use the above variational method within an EM algorithm that estimates μ and Σ in empirical Bayes fashion. (For Viterbi and MBR decoding, we then estimate θ as μ, the mean of the learned prior; see Section 4.1.) In the E-step, we maximize the bound with respect to the variational parameters using coordinate ascent as in Section 4.2. We optimize each of these separately in turn, cycling through them, using appropriate optimization algorithms for each. In the M-step, we apply maximum likelihood estimation with respect to μ and Σ given sufficient statistics gathered from the variational parameters in the E-step. Appendix C describes the algorithm in full.

5. Experiments

We applied our modeling framework to unsupervised learning of the dependency model discussed in Section 2.2. We consider four scenarios:

1. (Section 5.1) Experiments with dependency grammar induction for English text using the logistic normal distribution.
2. (Section 5.2) Experiments with text in five additional languages: Chinese, Portuguese, Turkish, Czech, and Japanese.
3. (Section 5.3) Experiments with the shared logistic normal distribution for tying parameters which correspond to the same coarse part-of-speech tag (English, Portuguese, and Turkish).
4. (Section 5.4) Experiments with the shared logistic normal distribution in *bilingual* settings (English, Portuguese, and Turkish).

	attachment accuracy (%)								
	Viterbi decoding			MBR decoding			Committee decoding		
	≤ 10	≤ 20	all	≤ 10	≤ 20	all	≤ 10	≤ 20	all
MLE	45.8	39.1	34.2	46.1	39.9	35.9		$*$	
Dirichlet-I	45.9	39.4	34.9	46.1	40.6	36.9		$*$	
LN-I, $\Sigma_k^{(0)} = \mathbf{I}$	56.5	42.9	36.6	58.4	45.2	39.5	**56.4**$_{\pm.001}$	**42.3**$_{\pm.001}$	**36.2**$_{\pm.001}$
LN-I, families	**59.3**	**45.1**	**39.0**	**59.4**	**45.9**	**40.5**	56.3$_{\pm.01}$	41.3$_{\pm.01}$	34.9$_{\pm.005}$
LN-II, $\Sigma_k^{(0)} = \mathbf{I}$	26.1	24.0	22.8	27.9	26.1	25.3	22.0$_{\pm.02}$	20.1$_{\pm.02}$	19.1$_{\pm.02}$
LN-II, families	24.9	21.0	19.2	26.3	22.8	21.5	26.6$_{\pm.003}$	22.7$_{\pm.003}$	20.8$_{\pm.0006}$

Table 1: Attachment accuracy of different learning methods on unseen test data from the Penn Treebank of varying levels of difficulty imposed through a length filter. MLE is a reproduction of an earlier result using EM (Klein and Manning, 2004). LN-I and LN-II denote using the logistic normal with model I and model II (Figure 2), respectively. Committee decoding includes ten averaged runs. Numbers in small font denote variance. Results in bold denote best results in a column. Training is done on sentences of length ≤ 10, though testing is done on longer sentences as well.

5.1 English Text

We begin our experiments with the *Wall Street Journal* Penn treebank (Marcus et al., 1993). Following standard practice, sentences were stripped of words and punctuation, leaving part-of-speech tags for the unsupervised induction of dependency structure. We note that, in this setting, using gold standard part-of-speech tags as the input to the learning algorithm is common (Klein and Manning, 2004; Smith and Eisner, 2006; Spitkovsky et al., 2010b,a; Gillenwater et al., 2010, *inter alia*).

We train on §2–21, tune on §22 (without using annotations), and report final results on §23. Details of this data set (and others) are found in Table 2. Unsupervised training for these data sets can be costly, and requires iteratively running a cubic-time inside-outside dynamic programming algorithm, so we follow Klein and Manning (2004) in restricting the training set to sentences of ten or fewer words in length. Short sentences are also less structurally ambiguous and may therefore be easier to learn from.

To evaluate the performance of our models, we report the fraction of words whose predicted parent matches the gold standard annotation in the treebank.[5] This performance measure is known as *attachment accuracy*. We will report attachment accuracy on three subsets of the test corpus: sentences of length ≤ 10 (typically reported in prior work and most similar to the training data set), length ≤ 20, and the full test corpus. We considered the three decoding methods mentioned in Section 4.1. For MBR decoding, we use the number of dependency attachment errors as the loss function. This means that at decoding time, we minimize the expected number of attachment errors according to the prediction of the estimated model. Because committee decoding is a randomized algorithm, we run it ten times on the unseen data, and then average the dependency attachment accuracy.

5. The Penn Treebank's phrase-structure annotations were converted to dependencies using the head rules of Yamada and Matsumoto, which are very similar to the ones by Collins (1999). See `http://www.jaist.ac.jp/~h-yamada`.

Initialization is important for all conditions, because likelihood and our variational bound are non-concave functions. For the values of the multinomials (θ), we use the harmonic initializer from Klein and Manning (2004). It estimates θ using soft counts on the training data where, in an n-length sentence, (i) each word is counted as the sentence's head $\frac{1}{n}$ times, and (ii) each word x_i attaches to x_j proportional to $|i-j|^{-1}$, normalized to a single attachment per word. This initializer is used with MLE and Dirichlet-I ("I" stands for model I from Figure 2). In the case of LN-I and LN-II, it is used as an initializer both for μ and inside the E-step.

For learning with the logistic normal prior, we consider two initializations of the covariance matrices Σ_k. The first is the $N_k \times N_k$ identity matrix. We then tried to bias the solution by injecting prior knowledge about the part-of-speech tags. To do that, we manually mapped the tag set (34 tags) to twelve disjoint tag "families." These are simply coarser tags: adjective, adverb, conjunction, foreign, interjection, noun, number, particle, preposition, pronoun, proper, verb. The coarse tags were chosen to loosely account for the part-of-speech tag sets of seven treebanks in different languages. The mapping from fine-grained tags to coarser tags are based on the annotation guidelines of the relevant treebank. This mapping into families provides the basis for an initialization of the covariance matrices for the dependency distributions: 1 on the diagonal, 0.5 between probabilities of possible child tags that belong to the same family, and 0 elsewhere. These results are denoted "families" and are compared to the identity matrix as an initializer.

We compared several models, where learning is accomplished using (variational) EM: MLE, standard maximum-likelihood estimation using EM; Dirichlet-I, a common baseline in the literature which uses a Dirichlet prior together with variational EM; and LN-I (LN-II), a model with the logistic normal distribution using model I (model II). In all cases, we either run the (variational) EM algorithm until convergence of the log-likelihood (or its bound) or until the log-likelihood on an unannotated development set of sentences does not increase.

We note that on the full test set, attaching each word to the word on its right ("Attach-Right") achieves about 30% accuracy, and attaching each word to the word on its left ("Attach-Left") achieves about 20% accuracy.

Table 1 shows the experimental results. Note that there are two variants which consistently get lower performance than their counterparts: using model II (versus using model I) and using committee decoding instead of Viterbi or MBR decoding. This suggests that the covariance matrices play a useful role during the learning process, but are not informative when performing decoding, since they are not used by Viterbi and MBR decoding. Interestingly, Smith and Eisner (2006) report a similar result for *structurally biased* DMV—a model that includes a parameter to control the length of the decoded dependencies. Their bias parameter is useful only during the learning process, but never during decoding. In general, the logistic normal distribution with model I outperforms substantially the baselines. It is interesting to note that LN-I outperforms Dirichlet-I and MLE even when using identity covariance matrices for initialization. The reason could be the fact that the logistic normal distribution, even when permitting only just diagonal covariance matrices (the case with identity covariance matrix initialization is weaker—we only initialize with diagonal matrices) allows to model the variance directly in the parameters. This is not possible with the Dirichlet distribution.

When we tested model II and committee decoding on other languages, the performance decrease was consistent. For the rest of the experiments, we report only MBR (and possibly Viterbi) decoding results using model I. The reason for the underperformance of model II could be the small number

language	tag set	training		development		test		baselines	
		tokens	sent.	tokens	sent.	tokens	sent.	A-R	A-L
English	34	55340	7179	35021	1700	49363	2416	30.2	20.4
Chinese	34	27357	4775	5824	350	7007	348	32.9	9.7
Portuguese	21	15976	2477	14558	907	5009	288	25.9	31.1
Turkish	29	18873	4497	7812	500	6288	623	68.2	4.3
Czech	47	67756	10674	32647	2535	33147	2535	24.4	28.3
Japanese	74	39121	10300	14666	1700	13648	1700	66.4	13.4

Table 2: Information about the data sets used in this paper. "Tag set" stands for the size of the part-of-speech tag set. Train, development and test columns show the number of tokens and number of sentences in each data set. The training set consists of sentences of length ten or less, as described in the text. The development set and the test set do not have any length restriction. The development set includes unannotated set of sentences from the respective language. A-R (A-L) stands for Attach-Right (Attach-Left), which are attachment accuracy baselines on the test set for all sentences. See text for details.

of parameters which is defined by the model. This small set of parameters cannot capture well the nuances across sentences in the data.

5.2 Additional Languages

Following Section 5.1, we experiment with other languages: Chinese, Portuguese, Turkish, Czech and Japanese.

- For Chinese, we used the Chinese treebank (Xue et al., 2004). We train on §1–270, use §301–1151 for development and test on §271–300.
- For Portuguese, we used the Bosque treebank (Afonso et al., 2002) from the CoNLL shared task in 2006 (Buchholz and Marsi, 2006).
- For Turkish, we used the METU-Sabancı treebank (Atalay et al., 2003; Oflazer et al., 2003) from the CoNLL shared task in 2006.
- For Czech, we used the Prague treebank (Hajič et al., 2000) from the CoNLL shared task in 2007 (Nivre et al., 2007).
- For Japanese, we used the VERBMOBIL Treebank for Japanese (Kawata and Bartels, 2000) from the CoNLL shared task in 2006.

Whenever using CoNLL shared task data, we used the first 80% of the data distributed in the shared task for training, and the rest was divided equally for development and testing. Table 2 gives statistics about the data sets used with the performance of the Attach-Right and Attach-Left baselines given for the whole test data. As in the case for English, sentences were stripped of words and punctuation, leaving part-of-speech tags for the unsupervised induction of dependency structure. All learning algorithms were run on sentences of length ten words or less. Note that strong performance is achieved for Turkish and Japanese by the Attach-Right baseline.

Results of running the different learning algorithms are given in Figure 4. Note that for Portuguese, the difference is much smaller between the EM baselines and logistic normal variational EM when only short sentences are considered, but there is a wider gap for longer sentences; the LN models appear to generalize better to longer sentences. For Turkish, no method outperforms Attach-Right, but there is still a big gap between variational EM with the logistic normal and the other EM baselines. The case is similar for Japanese, though logistic normal does outperform the Attach-Right baselines for shorter sentences. For Czech, it seems like Dirichlet and EM do somewhat better than the logistic normal prior, but performance of all four methods is close. It is conceivable that the approximation inherent in a projective syntax representation for the Czech sentences (whose gold-standard analyses have a relatively large fraction of nonprojective dependencies) interacts with different models in different ways.[6]

In general, the covariance matrices learned when initializing with the identity covariance matrix are rather sparse, but there is a high degree of variability across the diagonal (for the variance values learned). For the DMV, when using an identity initializer, diagonal matrices are the local optimum that is reached by the variational EM algorithm. When initializing the covariance matrices with the tag families initializer, the learned matrices are still rather sparse, but they have a larger number of significant correlations (for Portuguese, for example, using a t-test for testing the significance of the correlation, we found that 0.3% of the values in the covariance matrices had significant correlation).[7]

5.3 SLN with Nouns, Verbs, and Adjectives

We now turn to experiments where the partition structure lets parameters across multinomials covary, making use of the expressive power of the shared logistic normal distribution. We use a few simple heuristics to decide which partition structure $\mathcal{S}$ to use. Our heuristics rely mainly on the centrality of content words: nouns, verbs, and adjectives. For example, in the English treebank, the most common attachment errors (with the LN prior) happen with a noun (25.9%) or a verb (16.9%) parent. The fact that the most common errors happen with these attachments results from nouns and verbs being the most common parents in most of the data sets we experimented with.

Following this observation, we compare four different settings in our experiments (all SLN settings include one normal expert for each multinomial on its own, equivalent to the regular LN setting):

- TIEV: We add normal experts that tie all probabilities corresponding to a verbal parent (*any* verbal parent, using the coarse tags of Cohen et al., 2008). Let V be the set of part-of-speech tags that belong to the verb category. For each direction D (left or right), the set of multinomials of the form $\theta_c(\cdot \mid v, D)$, for $v \in V$, all share a normal expert. For each direction D and each boolean value B of the predicate $\text{first}_{\mathbf{y}}(\cdot)$, the set of multinomials $\theta_s(\cdot \mid v, D, B)$ for $v \in V$ share a normal expert.
- TIEN: This is the same as TIEV, only for nominal parents.

6. We note that we also experimented with other languages, including Hebrew and Arabic. We do not include these results, because in these cases all methods, including MLE, Dirichlet-I and LN-I performed badly (though Dirichlet-I and MLE could do better than LN-I). We believe that for these languages, the DMV is probably not the appropriate model. Developing better grammatical models for these languages is beyond the scope of this paper.
7. However, it is interesting to note that most of the elements of the covariance matrices were not exactly zero. For example, 90% of the values in the covariance matrices were larger (in absolute value) than 2.3×10^{-6}.

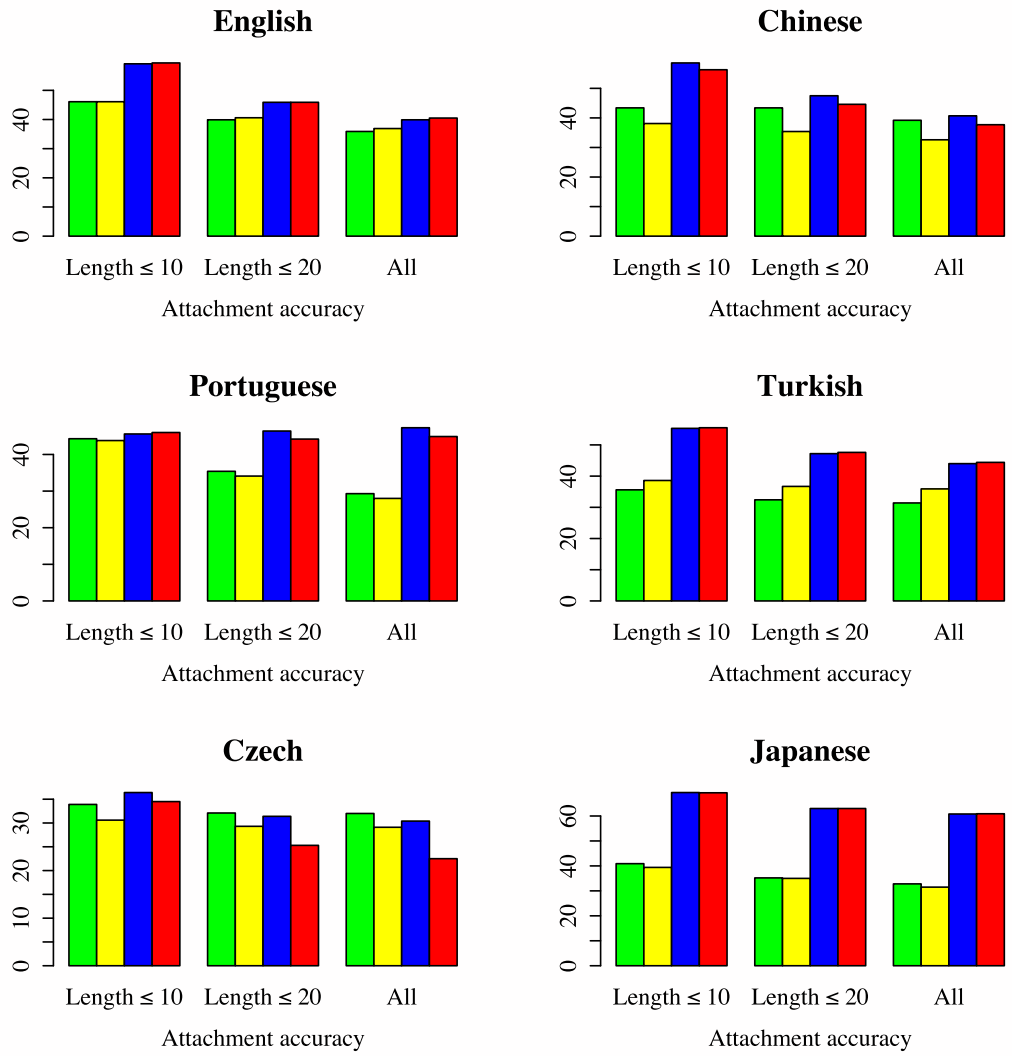


Figure 4: Attachment accuracy results for English (equivalent to Table 1), Chinese, Portuguese, Turkish, Czech and Japanese. The decoding mechanism used is MBR. Legend for the baselines: MLE (green, first column in each block); Dirichlet-I (yellow, second column); Legend for the methods in this paper: LN-I, $\Sigma_k^{(0)} = \mathbf{I}$ (blue, third column), and LN-I, families initializer (red, fourth column).

- TIEV&N: Tie both verbs and nouns (in separate partitions). This is equivalent to taking the union of the partition structures of the above two settings.

- TIEA: This is the same as TIEV, only for adjectival parents.

			English			Portuguese			Turkish		
			≤ 10	≤ 20	all	≤ 10	≤ 20	all	≤ 10	≤ 20	all
		MLE	46.1	39.9	35.9	44.3	35.4	29.3	35.6	32.4	31.4
		Dirichlet-I	46.1	40.6	36.9	43.8	34.1	28.0	38.6	36.7	35.9
		$\Sigma_k^{(0)} = \mathbf{I}$	59.1	45.9	40.5	45.6	**45.9**	**46.5**	55.3	47.2	44.0
		families	59.4	45.9	40.5	45.9	44.0	44.4	55.5	47.6	44.4
Trained with	English	TIEV	60.2	46.2	40.0	45.4	43.7	44.5	† 56.5	**48.7**	45.5
		TIEN	60.2	46.7	40.9	45.7	44.3	45.0	51.1	43.7	41.2
		TIEV&N	61.3	47.4	41.4	46.3	44.6	45.1	55.9	48.2	45.2
		TIEA	59.9	45.8	39.6	45.4	43.8	44.6	49.8	43.2	40.8
	Portuguese	TIEV	62.1	48.1	42.2	45.2	42.3	42.3	**56.7**	† 48.6	45.1
		TIEN	60.7	46.9	40.9	45.7	42.8	42.9	33.2	29.8	28.7
		TIEV&N	61.4	47.8	42.0	46.3	44.6	45.1	**56.7**	49.2	**46.0**
		TIEA	62.1	47.8	41.8	45.2	42.7	42.7	31.5	28.4	27.5
	Turkish	TIEV	**62.5**	**48.3**	**42.4**	45.4	43.2	43.7	55.2	47.3	44.0
		TIEN	61.0	47.2	41.2	45.9	43.9	44.4	45.1	39.8	37.8
		TIEV&N	† 62.3	**48.3**	† 42.3	**46.7**	44.3	44.6	55.7	**48.7**	45.5
		TIEA	† 62.3	48.0	42.1	45.1	43.2	43.7	38.6	34.0	32.5

Table 3: Attachment accuracy of different monolingual tying models and bilingual tying models in varying levels of difficulty imposed through a length filter (Sections 5.3 and 5.4). Monolingual results (Section 5.3) are described when the languages in both the column and the row are identical (blocks on the diagonal). Results for MLE and Dirichlet-I are identical to Figure 4. Results for $\Sigma_k^{(0)} = \mathbf{I}$ and families are identical to Table 1 and Figure 4. Each block contains the results of tying one language with the other, specifying performance for the column language. Results in bold denote best results in a column, and † marks figures that are not significantly worse (binomial sign test, $p < 0.05$).

Since learning a model with parameter tying can be computationally intensive, we first run the inference algorithm without parameter tying, and then add parameter tying to the rest of the inference algorithm's execution until convergence.

For the covariance matrices, we follow the setting described in Section 5.1. For each treebank, we divide the tags into twelve disjoint tag families. The covariance matrices for all dependency distributions were initialized with 1 on the diagonal, 0.5 between tags which belong to the same family, and 0 otherwise.

The results are given in the blocks on the diagonal of Table 3, where the languages in the columns and rows are identical. MBR decoding was used. For English, there are small improvements when adding the expressive power of SLN. The best results are achieved when tying both nouns and verbs together. Portuguese shows small benefits compared on shorter sentences, and when compared to the families-initialized LN-I model, but not the stronger identity-initialized LN-I model. For Turkish, tying across multinomials hurts performance.

5.4 Bilingual Experiments

Leveraging linguistic information from one language for the task of disambiguating another language has received considerable attention (Dagan, 1991; Yarowsky et al., 2001; Hwa et al., 2005; Smith and Smith, 2004; Snyder and Barzilay, 2008; Burkett and Klein, 2008). Usually such a setting requires a parallel corpus or other annotated data that ties between those two languages. One notable exception is Haghighi et al. (2008), where bilingual lexicons were learned from non-parallel monolingual corpora.

Our bilingual experiments use the data for English, Portuguese, and Turkish (two at a time), which are not parallel corpora, to train parsers for two languages at a time, jointly. Sharing information between two models is accomplished by softly tying grammar weights in the two hidden grammars.

For each pair of languages, we first merge the models for these two languages by taking a union of the multinomial families of each and the corresponding prior parameters. We then add a normal expert that ties between the parts of speech in the respective partition structures for both grammars together. Parts of speech are matched through the single coarse tag set. For example, with TIEV, let $V = V_{\text{Eng}} \cup V_{\text{Por}}$ be the set of part-of-speech tags which belong to the verb category for either the English or Portuguese treebank (to take an example). Then, we tie parameters for all part-of-speech tags in V. We tested this joint model for each of TIEV, TIEN, TIEV&N, and TIEA. After running the inference algorithm which learns the two models jointly, we use unseen data to test each learned model separately.

We repeat the generative story specifically for the bilingual setting, using the example of TIEV. For each language, there are normal experts for all part-of-speech tags, for the basic DMV. In addition, there are normal experts, for each language, that combine together all part-of-speech tags that belong to the verb category. Finally, there are normal experts, for the two languages *together*, that combine together all part-of-speech tags that belong to the verb category in either language. For each sentence in the corpus, the following two steps are conducted as before (model I): the normal experts are sampled from the SLN distribution and combined into multinomials to parameterize the DMV; a grammar derivation is sampled from the resulting DMV.

Table 3 presents the results for these experiments (blocks not on the diagonal). English grammar induction shows moderate gains when tied with Portuguese and strong gains with Turkish. Cohen and Smith (2009) reported qualitatively similar results when English was tied with Chinese. For Portuguese, there is not much gain from tying it with other languages, though it improves the performance of the other two languages. In general, the table shows that with the proper selection of pair of languages and multinomials to tie together, we can usually get improvement over the LN baselines and the technique is not harmful (cf. Turkish grammar induction with SLN, on its own). We note that selection of the multinomials to tie encodes prior knowledge about the languages. This knowledge simply requires being able to map fine-grained, treebank-specific part-of-speech tags to coarse categories. In addition, bilingual learning with SLN does not require bitext parsing at any point, which is an expensive operation. The runtime of the variational E-step for a sentence $\mathbf{x}$ is still cubic in the length of $\mathbf{x}$, as in EM, so that the runtime of the variational E-step scales in the multilingual case the same as it would be if we added an equivalent amount of data in the monolingual case.

Since the experiments reported here were conducted, others, notably Gillenwater et al. (2010) and Spitkovsky et al. (2010b), have reported performance surpassing ours, for some of the languages

in our experiments. Differences in the experimental settings prevent direct comparisons. Some of the improvements in dependency grammar induction are achieved because of techniques which are orthogonal to ours, such as improvements in the underlying grammar (instead of DMV; Headden et al., 2009; Gillenwater et al., 2010), and those techniques could be incorporated into the Bayesian model we described. Others are somewhat different (e.g., Viterbi training).

6. Discussion

We have shown that modeling covariance among grammar weights within a probabilistic grammar's multinomial distributions, *across* its distributions, and *across* grammars in two languages can have benefits for learning dependency structure in an unsupervised empirical Bayesian framework. This approach addresses one of the desiderata of Raiffa and Schaifer (1961) for prior distributions, "richness." The empirical benefits of modeling covariance, we have shown, are compelling.

We believe, however, that more remains to be done to incorporate prior linguistic knowledge into unsupervised grammar induction. Covariance structure is, perhaps, not the most *interpretable* kind of prior knowledge about grammars that might be brought to bear on learning. The empirical Bayesian paradigm explored here, and the use of variational approximations for coping with non-conjugacy, will be important tools in future work that brings together prior knowledge and unannotated data for grammar induction.

For natural language data, a direction for future work is to capture deeper linguistic phenomena. Here, background knowledge abounds: the entire field of theoretical linguistics has contributed both descriptive facts about the structure of specific natural languages and general theories about the way that structure is constrained. Viewing the logistic normal prior as local log-linear models (Section 3.4) is a first step towards encoding such prior knowledge. Similar to Berg-Kirkpatrick et al. (2010), it permits the use of arbitrary features in the parameterization of the grammar.

We note that our inference algorithm, described in detail in Appendix B, can be easily adapted to scenarios which do not necessarily use the multivariate normal distribution as the base distribution in the prior. The "softmax" can be applied to any multivariate sample to get a point in the probability simplex—perhaps capturing other tendencies in the data than covariance. The convenience of performing such an extension depends on the ability to effectively compute the moment generating function of the distribution replacing the multivariate Gaussian, in which case we can develop Equation 13 and proceed with optimizing the variational bound using this distribution.

7. Conclusion

In this paper we demonstrated the effectiveness of estimating probabilistic grammars in a Bayesian setting. We used the Bayesian setting to model covariance between the different parameters of probabilistic grammars. To model the covariance, we used the logistic normal distribution as a prior over the grammar parameters. In addition, we extended the logistic normal distribution to a new family of distributions, in order to model covariance across the multinomial family in a probabilistic grammar.

We proposed a variational inference algorithm for estimating the parameters of the probabilistic grammar, providing a fast, parallelizable,[8] and deterministic alternative to MCMC methods to approximate the posterior over derivations and grammar parameters.

8. We used a cluster running MapReduce (Dean and Ghemawat, 2004) to perform inference when training our models.

We experimented with grammar induction on six different languages, demonstrating the usefulness of our approach. Our experiments include a novel promising setting, in which syntactic trees are inferred in a bilingual setting that uses multilingual, non-parallel corpora. Notably, our approach tends to generalize better to longer sentences, despite learning (as in previous research) on short sentences. The focus of the experiments was on dependency grammar induction with the dependency model with valence. Our choice of the DMV is motivated by the fact that it is a widespread grammar for dependency grammar induction (Section 2.2), enabling us to tease apart the problem of estimation of the grammar from the problem of deciding on the grammar structure. Our inference algorithm, though, could be applied to any probabilistic grammar that has an efficient procedure, such as the inside-outside algorithm, for computing sufficient statistics in the form of expected counts of rule firing in grammar derivations.

Acknowledgments

The authors would like to thank the anonymous reviewers, Matthew Harrison, Mark Johnson, John Lafferty, and Eric Xing for their useful feedback and comments. The authors would also like to thank Yahoo! for making computational resources available. This research was supported by NSF grants IIS-0713265, IIS-0836431 and IIS-0915187.

Appendix A. Notation

Table 4 gives a table of notation for symbols used throughout this paper.

Appendix B. Variational Inference with Logistic Normal Priors

We give a derivation of a variational inference algorithm for model I, with the shared logistic normal distribution as a prior. The derivation is based on the one given in Blei and Lafferty (2006). The derivation for model II can be followed similarly, as explained below. For model I, we seek to find an approximation posterior function $q(\eta_1, ..., \eta_M, \mathbf{y}_1, ..., \mathbf{y}_M)$ that maximizes a lower bound (the negated variational free energy) on the log-likelihood, a bound which is achieved using Jensen's inequality (the following probability quantities should be understood as if we always condition on the grammar **G**):

$$\begin{aligned}&\sum_{m=1}^{M}\sum_{\mathbf{y}}\log p(\mathbf{x}_m, \mathbf{y} \mid \mu, \Sigma, \mathcal{S})\\&\geq \sum_{m=1}^{M}\left(\sum_{i=1}^{N}\mathbb{E}_q\left[\log p(\eta_{m,i} \mid \mu_i, \Sigma_i)\right] + \mathbb{E}_q\left[\log p(\mathbf{x}_m, \mathbf{y}_m \mid \eta_m, \mathcal{S})\right]\right) + H(q).\end{aligned} \tag{10}$$

$H(\cdot)$ denotes the Shannon entropy.

We make a mean-field assumption, and assume that the posterior has the following form:

$$q(\eta_1, ..., \eta_M, \mathbf{y}_1, ..., \mathbf{y}_M) = \prod_{m=1}^{M} q_m(\eta_m, \mathbf{y}_m), \tag{11}$$

	symbol	description
grammars and data	$\mathbf{G}$	grammar (for example, context-free grammar rules)
	M	number of observed sentences
	$\mathbf{x}_m$	mth observed sentence in the available data
	$\mathbf{y}_m$	inferred derivation grammatical structure for $\mathbf{x}_m$
	θ	parameters of a probabilistic grammar
	K	number of multinomials in the probabilistic grammar
	N_k	size of the kth multinomial of the probabilistic grammar
	$f_{k,i}(\mathbf{x},\mathbf{y})$	number of times the ith event fires in the kth multinomial in the derivations $\mathbf{x}$ and $\mathbf{y}$
priors	α	hyperparameters for the Dirichlet prior of a probabilistic grammar
	Σ	covariance matrices for the (shared) logistic normal prior of a probabilistic grammar
	μ	mean values for the (shared) logistic normal prior of a probabilistic grammar
	η	values drawn from the Gaussians for (S)LN, before the logistic transformation is applied
	$\mathcal{S}$	partition structure for the shared logistic normal distribution
	N	number of normal experts for the SLN
	l_n	length of nth normal expert (SLN)
	I_n	partition of the nth normal expert into segments mapping to multinomials in $\mathbf{G}$ (SLN)
	J_k	collection of segments of normal experts mapping to kth multinomial in $\mathbf{G}$ (SLN)
variational EM	$q_m(\theta,\mathbf{y})$	variational distribution which is used as an approximation posterior for the mth datum
	$\tilde{\mu}_{m,k,i}$	variational parameter for mean value of the ith event in the kth for the mth datum
	$\tilde{\sigma}_{m,k,i}$	variational parameter for variance of the ith event in the kth for the mth datum
	$\tilde{f}_{m,k,i}$	expected count of the ith event in the kth for the mth datum
	$\tilde{\psi}_{m,k,i}$	intermediate quantity aggregating variational parameters
	$\tilde{\zeta}_{m,k}$	variational parameter for the first-order Taylor approximation of LN's denominator

Table 4: Table of notation symbols used in this paper.

where

$$q_m(\eta_m,\mathbf{y}_m) = \left(\prod_{k=1}^{N}\prod_{i=1}^{L_k} q_m(\eta_{m,k,i} \mid \tilde{\mu}_{m,k,i},\tilde{\sigma}^2_{m,k,i})\right) \times q_m(\mathbf{y}_m),$$

and $q_m(\eta_{m,k,i} \mid \tilde{\mu}_{m,k,i},\tilde{\sigma}^2_{m,k,i})$ is a Gaussian with mean $\tilde{\mu}_{m,k,i}$ and variance $\tilde{\sigma}^2_{m,k,i}$. Note that this means that the *variational* distributions have a diagonal matrix for their covariance structure. The model covariance matrices (the hyperparameters Σ) can still have covariance structure. This selection of variational distributions makes inference much easier. The factorized form of Equation 11 implies the following identities:

$$\mathbb{E}_q\left[\log p(\eta_{m,i} \mid \mu_i, \Sigma_i)\right] = \mathbb{E}_{q_m}\left[\log p(\eta_{m,i} \mid \mu_i, \Sigma_i)\right],$$
$$\mathbb{E}_q\left[\log p(\mathbf{x}_m, \mathbf{y}_m \mid \eta_m, \mathcal{S})\right] = \mathbb{E}_{q_m}\left[\log p(\mathbf{x}_m, \mathbf{y}_m \mid \eta_m, \mathcal{S})\right],$$
$$H(q) = \sum_{m=1}^{N} H(q_m).$$

Let $\eta^C_{k,i}$ be an intermediate variable, denoting the average of the normal experts which appear in the partition structure and determine the value of the ith event in the kth multinomial of the grammar. More formally, we define the vector η^C_k of length N_k to be:

$$\eta^C_k \triangleq \frac{1}{|J_k|} \sum_{I_{r,j} \in J_k} \eta_{r,I_{r,j}}.$$

Unfolding the expectation with respect to $q_m(\mathbf{y}_m)$ in the second term in Equation 10, while recalling that θ_m is a deterministic function of η_m that averages different subvectors from the collection of multinomials η_m according to the partition structure $\mathcal{S}$, we have that:

$$\begin{aligned}
&\mathbb{E}_{q_m}\left[\log p(\mathbf{x}_m, \mathbf{y}_m \mid \eta_m, \mathcal{S})\right] \\
&= \mathbb{E}_{q_m(\eta_m)}\left[\sum_{k=1}^{K} \Sigma_{i=1}^{N_k} \underbrace{\sum_{\mathbf{y}} q_m(\mathbf{y}_m) f_{k,i}(\mathbf{x}_m, \mathbf{y}_m)}_{\tilde{f}_{m,k,i}} \log \theta_{m,k,i}\right] \\
&= \mathbb{E}_{q_m(\eta_m)}\left[\sum_{k=1}^{K} \sum_{i=1}^{N_k} \tilde{f}_{m,k,i} \left(\eta^C_{m,k,i} - \log \sum_{i'=1}^{N_k} \exp \eta^C_{m,k,i'}\right)\right],
\end{aligned} \tag{12}$$

where $\tilde{f}_{m,k,i}$ is the expected number of occurrences of the ith event in distribution k, under $q_m(\mathbf{y}_m)$. With many kinds of probabilistic grammars, this quantity can be computed using a dynamic programming algorithm like the forward-backward or inside-outside algorithm.

The logarithm term in Equation 12 is problematic because of the expectation with respect to $q_m(\eta_m)$. We approximate it with a first-order Taylor expansion, introducing $M \times K$ more variational parameters $\tilde{\zeta}_{m,k}$ for $m \in \{1, ..., M\}$ and $K \in \{1, ..., K\}$:

$$\log\left(\sum_{i'=1}^{N_k} \exp \eta^C_{m,k,i'}\right) \leq \log \tilde{\zeta}_{m,k} - 1 + \frac{1}{\tilde{\zeta}_{m,k}} \sum_{i'=1}^{N_k} \exp \eta^C_{m,k,i'}. \tag{13}$$

We note that the value $\mathbb{E}_{q_m(\eta_m)}\left[\exp(\eta^C_{m,k,i'})\right]$ can be calculated by evaluating the moment-generating function of the normal distribution $g(t) = \mathbb{E}_{q_m(\eta_m)}\left[\exp(t\eta^C_{m,k,i'})\right]$ at $t = 1$. We now have:

Algorithm 1: Variational EM for probabilistic grammars with LN prior

Input: initial parameters $\mu^{(0)}$, $\Sigma^{(0)}$, training data $\mathbf{x}$, and development data $\mathbf{x}'$
Output: learned parameters μ, Σ
$t \leftarrow 1$;
repeat
  Call E-Step for each training example $m = 1, ..., M$ *(Algorithm 2)*
  Call M-Step (Algorithm 3)
  $t \leftarrow t + 1$;
until *likelihood of held-out data,* $p(\mathbf{x}' \mid E[\mu^{(t)}])$, *decreases* ;
return $\mu^{(t)}$, $\Sigma^{(t)}$

$$
\begin{aligned}
&\mathbb{E}_{q_m}[\log p(\mathbf{x}_m, \mathbf{y}_m \mid \eta_m, \mathcal{S})] \\
&\geq \mathbb{E}_{q_m(\eta_m)}\left[\sum_{k=1}^{K}\sum_{i=1}^{N_k} \tilde{f}_{m,k,i}\left(\eta_{m,k,i} - \log\tilde{\zeta}_{m,k} + 1 - \frac{1}{\tilde{\zeta}_{m,k}}\sum_{i'=1}^{N_k}\exp\eta_{m,k,i'}\right)\right] \\
&= \sum_{k=1}^{K}\sum_{i=1}^{N_k} \tilde{f}_{m,k,i}\underbrace{\left(\tilde{\mu}_{m,k,i} - \log\tilde{\zeta}_{m,k} + 1 - \frac{1}{\tilde{\zeta}_{m,k}}\sum_{i'=1}^{N_k}\exp\left(\tilde{\mu}^C_{m,k,i} + \frac{(\tilde{\sigma}^C_{m,k,i})^2}{2}\right)\right)}_{\tilde{\psi}_{m,k,i}} \\
&= \sum_{k=1}^{K}\sum_{i=1}^{N_k} \tilde{f}_{m,k,i}\tilde{\psi}_{m,k,i}
\end{aligned}
$$

where we use again the properties of the shared logistic normal distribution and rely on the partition structure $\mathcal{S}$ to define:

$$
\begin{aligned}
\tilde{\mu}^C_{m,k} &\triangleq \frac{1}{|J_k|}\sum_{I_{r,j}\in J_k}\tilde{\mu}_{m,r,I_{r,j}}, \\
(\tilde{\sigma}^C_{m,k})^2 &\triangleq \frac{1}{|J_k|^2}\sum_{I_{r,j}\in J_k}\tilde{\sigma}^2_{m,r,I_{r,j}}.
\end{aligned}
$$

Note the shorthand $\tilde{\psi}_{k,i}$ to denote an expression involving $\tilde{\mu}^C$, $\tilde{\sigma}^C$, and $\tilde{\zeta}$.

The final form of our bound is:[9]

$$
\log p(\mathbf{x}, \mathbf{y} \mid \mu, \Sigma) \geq \left(\sum_{k=1}^{K}\mathbb{E}_q\left[\log p(\eta_k \mid \mu_k, \Sigma_k)\right]\right) + \left(\sum_{k=1}^{K}\sum_{i=1}^{N_k}\tilde{f}_{k,i}\tilde{\psi}_{k,i}\right) + H(q). \tag{14}
$$

Using an EM-style algorithm, we will alternate between finding the maximizing $q(\eta)$ and the maximizing $q(\mathbf{y})$. Maximization with respect to $q_m(\eta_m)$ is not hard, because $q(\eta)$ is parameterized. The following lemma shows that fortunately, finding the maximizing $q_m(\mathbf{y}_m)$, which we did not parameterize originally, is not hard either:

9. A tighter bound, based on a second-order approximation, was proposed in Ahmed and Xing (2007). We use a first-order approximation for simplicity, similar to Blei and Lafferty (2006).

Algorithm 2: E-Step (subroutine for Algorithm 1)

repeat

optimize for $\tilde{\mu}^{(t)}_{m,k}, k = 1, ..., K$: use conjugate gradient descent with

$$\frac{\partial B}{\partial \tilde{\mu}_{m,k,i}} = -\left((\Sigma_k^{(t-1)})^{-1})(\mu_k^{(t-1)} - \tilde{\mu}_{m,k}) \right)_i - \tilde{f}_{m,k,i} + \sum_{i'=1}^{N_k} \left(\tilde{f}_{m,k,i'} / \tilde{\zeta}_{m,k} \right) \exp\left(\frac{\tilde{\mu}_{k,i'} + \tilde{\sigma}^2_{k,i'}}{2} \right)$$

optimize $\tilde{\sigma}^{(t)}_{m,k}, k = 1, ..., K$: use Newton's method for each coordinate (with $\tilde{\sigma}_{m,k,i} > 0$) with

$$\frac{\partial B}{\partial \tilde{\sigma}^2_{m,k,i}} = -\frac{\Sigma^{(t-1)}_{k,ii}}{2} - \frac{\left(\sum_{i'=1}^{N_k} \tilde{f}_{m,k,i'} \right) \exp\left(\frac{\tilde{\mu}_{m,k,i} + \tilde{\sigma}^2_{m,k,i}}{2} \right)}{2\tilde{\zeta}_{m,k}} + \frac{1}{2\tilde{\sigma}^2_{m,k,i}}$$

update $\tilde{\zeta}^{(t)}_{m,k}, \forall k$:

$$\tilde{\zeta}^{(t)}_{m,k} \leftarrow \sum_{i=1}^{N_k} \exp\left(\tilde{\mu}^{(t)}_{m,k,i} + \frac{(\tilde{\sigma}^{(t)}_{m,k,i})^2}{2} \right)$$

update $\tilde{\psi}^{(t)}_{m,k}, \forall k$:

$$\tilde{\psi}^{(t)}_{m,k,i} \leftarrow \tilde{\mu}^{(t)}_{m,k,i} - \log \tilde{\zeta}^{(t)}_{m,k} + 1 - \frac{1}{\tilde{\zeta}^{(t)}_{m,k}} \sum_{i'=1}^{N_k} \exp\left(\tilde{\mu}^{(t)}_{m,k,i} + \frac{(\tilde{\sigma}^{(t)}_{m,k,i})^2}{2} \right)$$

compute expected counts $\tilde{\mathbf{f}}^{(t)}_{m,k}, k = 1, ..., K$: use an inside-outside algorithm to re-estimate expected counts $\tilde{f}^{(t)}_{m,k,i}$ in weighted grammar $q(y)$ with weights $e^{\tilde{\psi}_m}$;

until *B does not change* ;

Lemma 2 *Let $r(\mathbf{y}_m \mid \mathbf{x}_m, e^{\tilde{\psi}_m})$ denote the conditional distribution over $\mathbf{y}_m$ given $\mathbf{x}_m$ defined as:*

$$r_m(\mathbf{y}_m \mid \mathbf{x}_m, e^{\tilde{\psi}}) = \frac{1}{Z_m(\tilde{\psi}_m)} \prod_{k=1}^{K} \prod_{i=1}^{N_k} \exp\left(\tilde{\psi}_{m,k,i} f_{m,k,i}(\mathbf{x}_m, \mathbf{y}_m) \right)$$

where $Z_m(\tilde{\psi}_m)$ is a normalization constant. Then $q_m(\mathbf{y}_m) = r_m(\mathbf{y}_m \mid \mathbf{x}_m, e^{\tilde{\psi}_m})$ maximizes the bound in Equation 14.

Proof First note that $H(q_m) = H(q_m(\eta_m \mid \tilde{\mu}_m, \tilde{\sigma}_m)) + H(q_m(\mathbf{y}_m))$. This means that the terms we are interested in maximizing from Equation 14 are the following, after plugging in $\tilde{f}_{m,k,i}$ explicitly:

$$L = \operatorname*{argmax}_{q_m(\mathbf{y}_m)} \sum_{\mathbf{y}_m} q_m(\mathbf{y}_m) \left(\sum_{k=1}^{K} \sum_{i=1}^{N_k} f_{m,k,i}(\mathbf{x}_m, \mathbf{y}_m) \tilde{\psi}_{m,k,i} \right) + H(q_m(\mathbf{y}_m)).$$

Algorithm 3: M-Step (subroutine for Algorithm 1)

Estimate $\mu^{(t)}$ and $\Sigma^{(t)}$ using the following maximum likelihood closed-form solution:

$$\mu_{k,i}^{(t)} \leftarrow \frac{1}{M}\sum_{m=1}^{M}\tilde{\mu}_{m,k,i}^{(t)}$$

$$\left[\Sigma_k^{(t)}\right]_{i,j} \leftarrow \frac{1}{M}\left(\sum_{m=1}^{M}\tilde{\mu}_{m,k,i}^{(t)}\tilde{\mu}_{m,k,j}^{(t)} + (\tilde{\sigma}^{(t)})_{m,k,i}^{2}\delta_{i,j} + M\mu_{k,i}^{(t)}\mu_{k,j}^{(t)} - \mu_{k,j}^{(t)}\sum_{m=1}^{M}\tilde{\mu}_{m,k,i}^{(t)} - \mu_{k,i}^{(t)}\sum_{m=1}^{M}\tilde{\mu}_{m,k,j}^{(t)}\right),$$

where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise.

Then, note that:

$$L = \operatorname*{argmin}_{q_m(\mathbf{y}_m)} D_{\mathrm{KL}}\left(q_m(\mathbf{y}_m) \;\middle\|\; r_m(\mathbf{y}_m \mid \mathbf{x}_m, e^{\tilde{\psi}_m})\right), \tag{15}$$

where D_{KL} denotes the KL divergence. To see that, combine the definition of KL divergence with the fact that $\sum_{k=1}^{K}\sum_{i=1}^{N_k} f_{m,k,i}(\mathbf{x},\mathbf{y})\tilde{\psi}_{m,k,i} - \log Z_m(\tilde{\psi}_m) = \log r_m(\mathbf{y}_m \mid \mathbf{x}_m, e^{\tilde{\psi}_m})$ where $\log Z_m(\tilde{\psi})$ does not depend on $q_m(\mathbf{y}_m)$. Equation 15 is minimized when $q_m = r_m$. ■

The above lemma demonstrates that the minimizing $q_m(\mathbf{y}_m)$ has the same form as the probabilistic grammar **G**, only without having sum-to-one constraints on the weights (leading to the required normalization constant $Z_m(\tilde{\psi}_m)$). As in classic EM with probabilistic grammars, we never need to represent $q_m(\mathbf{y}_m)$ explicitly; we need only $\tilde{\mathbf{f}}_m$, which can be calculated as expected feature values under $r_m(\mathbf{y}_m \mid \mathbf{x}_m, e^{\tilde{\psi}_m})$ using dynamic programming.

Variational inference for model II is done similarly to model I. The main difference is that instead of having variational parameters for each $q_m(\eta_m)$, we have a single distribution $q(\eta)$, and the sufficient statistics from the inside-outside algorithm are used altogether to update it during variational inference.

Appendix C. Variational EM for Logistic-Normal Probabilistic Grammars

The algorithm for variational inference with probabilistic grammars using logistic normal prior is defined in Algorithms 1–3.[10] Since the updates for $\tilde{\zeta}_k^{(t)}$ are fast, we perform them after each optimization routine in the E-step (suppressed for clarity). There are variational parameters for each training example, indexed by m. We denote by B the variational bound in Equation 14. Our stopping criterion relies on the likelihood of a held-out set (Section 5) using a point estimate of the model.

10. An implementation of the algorithm is available at `http://www.ark.cs.cmu.edu/DAGEEM`. For simplicity, we give the vanilla logistic normal version of the algorithm in this appendix. The full version requires a more careful indexing and can be derived using the equations from Appendix B.

References

S. Afonso, E. Bick, R. Haber, and D. Santos. Floresta sinta(c)tica: a treebank for Portuguese. In *Proceedings of LREC*, 2002.

A. Ahmed and E. Xing. On tight approximate inference of the logistic normal topic admixture model. In *Proceedings of AISTATS*, 2007.

J. Aitchison. *The Statistical Analysis of Compositional Data*. Chapman and Hall, London, 1986.

H. Alshawi and A. L. Buchsbaum. Head automata and bilingual tiling: Translation with minimal representations. In *Proceedings of ACL*, 1996.

D. Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, pages 87–106, 1988.

N. B. Atalay, K. Oflazer, and B. Say. The annotation process in the Turkish treebank. In *Proceedings of LINC*, 2003.

J. Baker. Trainable grammars for speech recognition. In *The 97th meeting of the Acoustical Society of America*, 1979.

A. Banerjee. On Bayesian bounds. In *Proceedings of ICML*, 2006.

T. Berg-Kirkpatrick and D. Klein. Phylogenetic grammar induction. In *Proceedings of ACL*, 2010.

T. Berg-Kirkpatrick, A. Bouchard-Cote, J. DeNero, and D. Klein. Unsupervised learning with features. In *Proceedings of NAACL*, 2010.

D. M. Blei and J. D. Lafferty. Correlated topic models. In *Proceedings of NIPS*, 2006.

D. M. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

J. L. Boyd-Graber and D. M. Blei. Syntactic topic models. *CoRR*, abs/1002.4665, 2010.

P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. Class-based *n*-gram models of natural language. *Computational Linguistics*, 1990.

S. Buchholz and E. Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of CoNLL*, 2006.

D. Burkett and D. Klein. Two languages are better than one (for syntactic parsing). In *Proceedings of EMNLP*, 2008.

G. Carroll and E. Charniak. Two experiments on learning probabilistic dependency grammars from corpora. Technical report, Brown University, 1992.

E. Charniak and M. Johnson. Coarse-to-fine *n*-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, 2005.

S. F. Chen. Bayesian grammar induction for language modeling. In *Proceedings of ACL*, 1995.

D. Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*, 2005.

A. Clark and F. Thollard. PAC-learnability of probabilistic deterministic finite state automata. *Journal of Machine Learning Research*, 5:473–497, 2004.

A. Clark, R. Eyraud, and A. Habrard. A polynomial algorithm for the inference of context free languages. In *Proceedings of ICGI*, 2008.

S. B. Cohen and N. A. Smith. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of NAACL-HLT*, 2009.

S. B. Cohen and N. A. Smith. Empirical risk minimization with approximations of probabilistic grammars. In *NIPS*, 2010.

S. B. Cohen, K. Gimpel, and N. A. Smith. Logistic normal priors for unsupervised probabilistic grammar induction. In *NIPS*, 2008.

S. B. Cohen, D. M. Blei, and N. A. Smith. Variational inference for adaptor grammars. In *Proceedings of NAACL*, 2010.

T. Cohn, S. Goldwater, and P. Blunsom. Inducing compact but accurate tree-substitution grammars. In *Proceedings of HLT-NAACL*, 2009.

M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, U. Penn., 1999.

M. Collins. Head-driven statistical models for natural language processing. *Computational Linguistics*, 29:589–637, 2003.

I. Dagan. Two languages are more informative than one. In *Proceedings of ACL*, 1991.

D. Das, N. Schneider, D. Chen, and N. A. Smith. Probabilistic frame-semantic parsing. In *Proceedings of ACL*, 2010.

C. de la Higuera. A bibliographical study of grammatical inference. *Pattern Recognition*, 38:1332–1348, 2005.

J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. In *Proceedings of OSDI*, 2004.

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.

Y. Ding and M. Palmer. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of ACL*, 2005.

J. Eisner. Bilexical grammars and a cubic-time probabilistic parser. In *Proceedings of IWPT*, 1997.

J. R. Finkel, T. Grenager, and C. D. Manning. The infinite tree. In *Proceedings of ACL*, 2007.

H. Gaifman. Dependency systems and phrase-structure systems. *Information and Control*, 8, 1965.

K. Ganchev, J. Gillenwater, and B. Taskar. Dependency grammar induction via bitext projection constraints. In *Proceedings of ACL*, 2009.

J. Gillenwater, K. Ganchev, J. Graça, F. Pereira, and B. Taskar. Sparsity in dependency grammar induction. In *Proceedings of ACL*, 2010.

K. Gimpel and N. A. Smith. Feature-rich translation by quasi-synchronous lattice parsing. In *Proceedings of EMNLP*, 2009.

S. Goldwater. *Nonparametric Bayesian models of lexical acquisition*. PhD thesis, Brown University, 2006.

S. Goldwater and T. L. Griffiths. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of ACL*, 2007.

A. Haghighi, P. Liang, T. Berg-Kirkpatrick, and D. Klein. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL*, 2008.

J. Hajič, A. Böhmová, E. Hajičová, and B. Vidová Hldaká. The Prague dependency treebank: A three-level annotation scenario. *Treebanks: Building and Using Parsed Corpora*, pages 103–127, 2000.

W. P. Headden, M. Johnson, and D. McClosky. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of NAACL-HLT*, 2009.

G. E. Hinton. Products of experts. In *Proceedings of ICANN*, 1999.

R. Hwa, P. Resnik, A. Weinberg, C. Cabezas, and O. Kolak. Bootstrapping parsers via syntactic projection across parallel texts. *Journal of Natural Language Engineering*, 11(3):311–25, 2005.

R. Johansson and P. Nugues. LTH: Semantic structure extraction using nonprojective dependency trees. In *Proceedings of SemEval*, 2007.

M. Johnson. Why doesn't EM find good HMM POS-taggers? In *Proceedings EMNLP-CoNLL*, 2007.

M. Johnson, T. L. Griffiths, and S. Goldwater. Adaptor grammars: A framework for specifying compositional nonparameteric Bayesian models. In *NIPS*, 2006.

M. Johnson, T. L. Griffiths, and S. Goldwater. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proceedings of NAACL*, 2007.

Y. Kawata and J. Bartels. Stylebook for the Japanese treebank in VERBMOBIL. Technical Report Verbmobil-Report 240, Seminar für Sprachwissenschaft, Univerisität Tübingen, 2000.

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

D. Klein and C. D. Manning. Accurate unlexicalized parsing. In *Proceedings of ACL*, pages 423–430, 2003.

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

K. Kurihara and T. Sato. Variational Bayesian grammar induction for natural language. In *Proceedings of ICGI*, 2006.

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

P. Liang, S. Petrov, M. Jordan, and D. Klein. The infinite PCFG using hierarchical Dirichlet processes. In *Proceedings of EMNLP*, 2007.

D. Lin. A path-based transfer model for machine translation. In *Proceedings of COLING*, 2004.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19:313–330, 1993.

D. McAllester. PAC-Bayesian model averaging. *Machine Learning Journal*, 5:5–21, 2003.

D. Mimno, H. Wallach, and A. McCallum. Gibbs sampling for logistic normal topic models with graph-based priors. In *In Proceedings of NIPS Workshop on Analyzing Graphs*, 2008.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task, EMNLP-CoNLL*, 2007.

K. Oflazer, B. Say, D. Z. Hakkani-Tür, and G. Tür. Building a Turkish treebank. In A. Abeille, editor, *Building and Exploiting Syntactically-Annotated Corpora*. Kluwer, 2003.

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

H. Raiffa and R. Schaifer. *Applied Statistical Decision Theory*. Wiley-Interscience, 1961.

M. Seeger. PAC-Bayesian generalization bounds for Gaussian processes. *Journal of Machine Learning Research*, 3:233–269, 2002.

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

N. A. Smith. *Novel Estimation Methods for Unsupervised Discovery of Latent Structure in Natural Language Text*. PhD thesis, Johns Hopkins University, 2006.

N. A. Smith and J. Eisner. Guiding unsupervised grammar induction using contrastive estimation. In *Proceedings of IJCAI Workshop on Grammatical Inference Applications*, 2005.

N. A. Smith and J. Eisner. Annealing structural bias in multilingual weighted grammar induction. In *Proceedings of COLING-ACL*, 2006.

B. Snyder and R. Barzilay. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL*, 2008.

V. Spitkovsky, H. Alshawi, and D. Jurafsky. From baby steps to leapfrog: How "less is more" in unsupervised dependency parsing. In *Proceedings of NAACL*, 2010a.

V. I. Spitkovsky, H. Alshawi, D. Jurafsky, and C. D. Manning. Viterbi training improves unsupervised dependency parsing. In *Proceedings of CoNLL*, 2010b.

V. I. Spitkovsky, D. Jurafsky, and H. Alshawi. Profiting from mark-up: Hyper-text annotations for guided parsing. In *Proceedings of ACL*, 2010c.

L. Tesnière. *Élément de Syntaxe Structurale*. Klincksieck, 1959.

K. Toutanova and M. Johnson. A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In *Proceedings of NIPS*, 2007.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

M. Wang, N. A. Smith, and T. Mitamura. What is the Jeopardy model? a quasi-synchronous grammar for question answering. In *Proceedings of EMNLP*, 2007.

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

N. Xue, F. Xia, F.-D. Chiou, and M. Palmer. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 10(4):1–30, 2004.

D. Yarowsky, G. Ngai, and R. Wicentoswki. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of HLT*, 2001.