PCFGs, Topic Models, Adaptor Grammars and Learning Topical Collocations and the Structure of Proper Names

Mark Johnson
Department of Computing
Macquarie University
mjohnson@science.mq.edu.au

Abstract

This paper establishes a connection between two apparently very different kinds of probabilistic models. Latent Dirichlet Allocation (LDA) models are used as “topic models” to produce a low-dimensional representation of documents, while Probabilistic Context-Free Grammars (PCFGs) define distributions over trees. The paper begins by showing that LDA topic models can be viewed as a special kind of PCFG, so Bayesian inference for PCFGs can be used to infer Topic Models as well. Adaptor Grammars (AGs) are a hierarchical, non-parameteric Bayesian extension of PCFGs. Exploiting the close relationship between LDA and PCFGs just described, we propose two novel probabilistic models that combine insights from LDA and AG models. The first replaces the unigram component of LDA topic models with multi-word sequences or collocations generated by an AG. The second extension builds on the first one to learn aspects of the internal structure of proper names.

1 Introduction

Over the last few years there has been considerable interest in Bayesian inference for complex hierarchical models both in machine learning and in computational linguistics. This paper establishes a theoretical connection between two very different kinds of probabilistic models: Probabilistic Context-Free Grammars (PCFGs) and a class of models known as Latent Dirichlet Allocation (Blei et al., 2003; Griffiths and Steyvers, 2004) models that have been used for a variety of tasks in machine learning. Specifically, we show that an LDA model can be expressed as a certain kind of PCFG, so Bayesian inference for PCFGs can be used to learn LDA topic models as well. The importance of this observation is primarily theoretical, as current Bayesian inference algorithms for PCFGs are less efficient than those for LDA inference. However, once this link is established it suggests a variety of extensions to the LDA topic models, two of which we explore in this paper. The first involves extending the LDA topic model so that it generates collocations (sequences of words) rather than individual words. The second applies this idea to the problem of automatically learning internal structure of proper names (NPs), which is useful for definite NP coreference models and other applications.

The rest of this paper is structured as follows. The next section reviews Latent Dirichlet Allocation (LDA) topic models, and the following section reviews Probabilistic Context-Free Grammars (PCFGs). Section 4 shows how an LDA topic model can be expressed as a PCFG, which provides the fundamental connection between LDA and PCFGs that we exploit in the rest of the paper, and shows how it can be used to define a “sticky topic” version of LDA. The following section reviews Adaptor Grammars (AGs), a non-parametric extension of PCFGs introduced by Johnson et al. (2007b). Section 6 exploits the connection between LDA and PCFGs to propose an AG-based topic model that extends LDA by defining distributions over collocations rather than individual words, and section 7 applies this extension to the problem of finding the structure of proper names.

2 Latent Dirichlet Allocation Models

Latent Dirichlet Allocation (LDA) was introduced as an explicit probabilistic counterpart to Latent Semantic Indexing (LSI) (Blei et al., 2003). Like LSI, LDA is intended to produce a low-dimensional characterisation or summary of a doc-

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1148–1157,
Uppsala, Sweden, 11-16 July 2010. ©2010 Association for Computational Linguistics

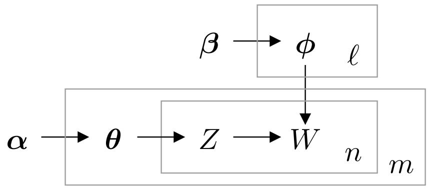

Figure 1: A graphical model "plate" representation of an LDA topic model. Here ℓ is the number of topics, m is the number of documents and n is the number of words per document.

ument in a collection of documents for information retrieval purposes. Both LSI and LDA do this by mapping documents to points in a relatively low-dimensional real-valued vector space; distance in this space is intended to correspond to document similarity.

An LDA model is an explicit generative probabilistic model of a collection of documents. We describe the "smoothed" LDA model here (see page 1006 of Blei et al. (2003)) as it corresponds precisely to the Bayesian PCFGs described in section 4. It generates a collection of documents by first generating multinomials $\boldsymbol{\phi}_i$ over the vocabulary V for each topic $i \in 1, \ldots, \ell$, where ℓ is the number of topics and $\phi_{i,w}$ is the probability of generating word w in topic i. Then it generates each document $D_j, j = 1, \ldots, m$ in turn by first generating a multinomial $\boldsymbol{\theta}_j$ over topics, where $\theta_{j,i}$ is the probability of topic i appearing in document j. ($\boldsymbol{\theta}_j$ serves as the low-dimensional representation of document D_j). Finally it generates each of the n words of document D_j by first selecting a topic z for the word according to $\boldsymbol{\theta}_j$, and then drawing a word from $\boldsymbol{\phi}_z$. Dirichlet priors with parameters $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ respectively are placed on the $\boldsymbol{\phi}_i$ and the $\boldsymbol{\theta}_j$ in order to avoid the zeros that can arise from maximum likelihood estimation (i.e., sparse data problems).

The LDA generative model can be compactly expressed as follows, where "$\sim$" should be read as "is distributed according to".

$$\begin{array}{rcll} \boldsymbol{\phi}_i & \sim & \mathrm{Dir}(\boldsymbol{\beta}) & i = 1, \ldots, \ell \\ \boldsymbol{\theta}_j & \sim & \mathrm{Dir}(\boldsymbol{\alpha}) & j = 1, \ldots, m \\ z_{j,k} & \sim & \boldsymbol{\theta}_j & j = 1, \ldots, m; k = 1, \ldots, n \\ w_{j,k} & \sim & \boldsymbol{\phi}_{z_{j,k}} & j = 1, \ldots, m; k = 1, \ldots, n \end{array}$$

In inference, the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ of the Dirichlet priors are either fixed (i.e., chosen by the model designer), or else themselves inferred, e.g., by Bayesian inference. (The adaptor grammar software we used in the experiments described below automatically does this kind of hyper-parameter inference).

The inference task is to find the topic probability vector $\boldsymbol{\theta}_j$ of each document D_j given the words $w_{j,k}$ of the documents; in general this also requires inferring the topic to word distributions $\boldsymbol{\phi}$ and the topic assigned to each word $z_{j,k}$. Blei et al. (2003) describe a Variational Bayes inference algorithm for LDA models based on a mean-field approximation, while Griffiths and Steyvers (2004) describe an Markov Chain Monte Carlo inference algorithm based on Gibbs sampling; both are quite effective in practice.

3 Probabilistic Context-Free Grammars

Context-Free Grammars are a simple model of hierarchical structure often used to describe natural language syntax. A *Context-Free Grammar* (CFG) is a quadruple (N, W, R, S) where N and W are disjoint finite sets of *nonterminal* and *terminal* symbols respectively, R is a finite set of productions or *rules* of the form $A \to \beta$ where $A \in N$ and $\beta \in (N \cup W)^\star$, and $S \in N$ is the *start symbol*.

In what follows, it will be useful to interpret a CFG as generating sets of finite, labelled, ordered trees $\mathcal{T}_A$ for each $X \in N \cup W$. Informally, $\mathcal{T}_X$ consists of all trees t rooted in X where for each *local tree* (B, β) in t (i.e., where B is a parent's label and β is the sequence of labels of its immediate children) there is a rule $B \to \beta \in R$.

Formally, the sets $\mathcal{T}_X$ are the smallest sets of trees that satisfy the following equations.

If $X \in W$ (i.e., if X is a terminal) then $\mathcal{T}_X = \{X\}$, i.e., $\mathcal{T}_X$ consists of a single tree, which in turn only consists of a single node labelled X.

If $X \in N$ (i.e., if X is a nonterminal) then

$$\mathcal{T}_X = \bigcup_{X \to B_1 \ldots B_n \in R_X} \mathrm{T}\textsc{ree}_X(\mathcal{T}_{B_1}, \ldots, \mathcal{T}_{B_n})$$

where $R_A = \{A \to \beta : A \to \beta \in R\}$ for each $A \in N$, and

$$\mathrm{T}\textsc{ree}_X(\mathcal{T}_{B_1}, \ldots, \mathcal{T}_{B_n}) = \left\{ \overset{X}{\overbrace{t_1 \ \ldots \ t_n}} : \begin{array}{l} t_i \in \mathcal{T}_{B_i}, \\ i = 1, \ldots, n \end{array} \right\}$$

That is, $\mathrm{T}\textsc{ree}_X(\mathcal{T}_{B_1}, \ldots, \mathcal{T}_{B_n})$ consists of the set of trees with whose root node is labelled X and whose ith child is a member of $\mathcal{T}_{B_i}$.

The set of trees generated by the CFG is $\mathcal{T}_S$, where S is the start symbol, and the set of strings generated by the CFG is the set of yields (i.e., terminal strings) of the trees in $\mathcal{T}_S$.

A *Probabilistic Context-Free Grammar* (PCFG) is a pair consisting of a CFG and set of multinomial probability vectors $\boldsymbol{\theta}_X$ indexed by nonterminals $X \in N$, where $\boldsymbol{\theta}_X$ is a distribution over the rules R_X (i.e., the rules expanding X). Informally, $\theta_{X \to \beta}$ is the probability of X expanding to β using the rule $X \to \beta \in R_X$. More formally, a PCFG associates each $X \in N \cup W$ with a distribution G_X over the trees $\mathcal{T}_X$ as follows.

If $X \in W$ (i.e., if X is a terminal) then G_X is the distribution that puts probability 1 on the single-node tree labelled X.

If $X \in N$ (i.e., if X is a nonterminal) then:

$$G_X = \sum_{X \to B_1 \ldots B_n \in R_X} \theta_{X \to B_1 \ldots B_n} \mathrm{TD}_X(G_{B_1}, \ldots, G_{B_n}) \quad (1)$$

where:

$$\mathrm{TD}_A(G_1, \ldots, G_n) \left(\begin{array}{c} X \\ \overbrace{t_1 \, \ldots \, t_n} \end{array} \right) = \prod_{i=1}^{n} G_i(t_i).$$

That is, $\mathrm{TD}_A(G_1, \ldots, G_n)$ is a distribution over $\mathcal{T}_A$ where each subtree t_i is generated independently from G_i. These equations have solutions (i.e., the PCFG is said to be "consistent") when the rule probabilities $\boldsymbol{\theta}_A$ obey certain conditions; see e.g., Wetherell (1980) for details.

The PCFG generates the distribution over trees G_S, where S is the start symbol. The distribution over the strings it generates is obtained by marginalising over the trees.

In a Bayesian PCFG one puts Dirichlet priors $\mathrm{Dir}(\boldsymbol{\alpha}_X)$ on each of the multinomial rule probability vectors $\boldsymbol{\theta}_X$ for each nonterminal $X \in N$. This means that there is one Dirichlet parameter $\alpha_{X \to \beta}$ for each rule $X \to \beta \in R$ in the CFG.

In the "unsupervised" inference problem for a PCFG one is given a CFG, parameters $\boldsymbol{\alpha}_X$ for the Dirichlet priors over the rule probabilities, and a corpus of strings. The task is to infer the corresponding posterior distribution over rule probabilities $\boldsymbol{\theta}_X$. Recently Bayesian inference algorithms for PCFGs have been described. Kurihara and Sato (2006) describe a Variational Bayes algorithm for inferring PCFGs using a mean-field approximation, while Johnson et al. (2007a) describe a Markov Chain Monte Carlo algorithm based on Gibbs sampling.

4 LDA topic models as PCFGs

This section explains how to construct a PCFG that generates the same distribution over a collection of documents as an LDA model, and where Bayesian inference for the PCFG's rule probabilities yields the corresponding distributions as Bayesian inference of the corresponding LDA models. (There are several different ways of encoding LDA models as PCFGs; the one presented here is not the most succinct — it is possible to collapse the Doc and Doc' nonterminals — but it has the advantage that the LDA distributions map straight-forwardly onto PCFG nonterminals).

The terminals W of the CFG consist of the vocabulary V of the LDA model plus a set of special "document identifier" terminals "$\_j$" for each document $j \in 1, \ldots, m$, where m is the number of documents. In the PCFG encoding strings from document j are prefixed with "$\_j$"; this indicates to the grammar which document the string comes from. The nonterminals consist of the start symbol $\mathrm{Sentence}$, Doc_j and Doc'_j for each $j \in 1, \ldots, m$, and Topic_i for each $i \in 1, \ldots, \ell$, where ℓ is the number of topics in the LDA model.

The rules of the CFG are all instances of the following schemata:

$$\begin{array}{ll}
\mathrm{Sentence} \to \mathrm{Doc}'_j & j \in 1, \ldots, m \\
\mathrm{Doc}'_j \to \_j & j \in 1, \ldots, m \\
\mathrm{Doc}'_j \to \mathrm{Doc}'_j \, \mathrm{Doc}_j & j \in 1, \ldots, m \\
\mathrm{Doc}_j \to \mathrm{Topic}_i & i \in 1, \ldots, \ell; j \in 1, \ldots, m \\
\mathrm{Topic}_i \to w & i \in 1, \ldots, \ell; w \in V
\end{array}$$

Figure 2 depicts a tree generated by such a CFG. The relationship between the LDA model and the PCFG can be understood by studying the trees generated by the CFG. In these trees the left-branching spine of nodes labelled Doc'_j propagate the document identifier throughout the whole tree. The nodes labelled Topic_i indicate the topics assigned to particular words, and the local trees expanding Doc_j to Topic_i (one per word in the document) indicate the distribution of topics in the document.

The corresponding Bayesian PCFG associates probabilities with each of the rules in the CFG. The probabilities $\boldsymbol{\theta}_{\mathrm{Topic}_i}$ associated with the rules expanding the Topic_i nonterminals indicate how words are distributed across topics; the $\boldsymbol{\theta}_{\mathrm{Topic}_i}$ probabilities correspond exactly to to the $\boldsymbol{\phi}_i$ probabilities in the LDA model. The probabilities

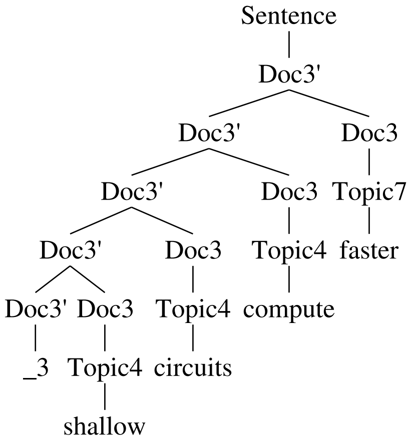

Figure 2: A tree generated by the CFG encoding an LDA topic model. The prefix "_3" indicates that this string belongs to document 3. The tree also indicates the assignment of words to topics.

$\boldsymbol{\theta}_{\mathrm{Doc}_j}$ associated with rules expanding Doc_j specify the distribution of topics in document j; they correspond exactly to the probabilities $\boldsymbol{\theta}_j$ of the LDA model. (The PCFG also specifies several other distributions that are suppressed in the LDA model. For example $\boldsymbol{\theta}_{\mathrm{Sentence}}$ specifies the distribution of documents in the corpus. However, it is easy to see that these distributions do not influence the topic distributions; indeed, the expansions of the $\mathrm{Sentence}$ nonterminal are completely determined by the document distribution in the corpus, and are not affected by $\boldsymbol{\theta}_{\mathrm{Sentence}}$).

A Bayesian PCFG places Dirichlet priors $\mathrm{Dir}(\boldsymbol{\alpha}_A)$ on the corresponding rule probabilities $\boldsymbol{\theta}_A$ for each $A \in N$. In the PCFG encoding an LDA model, the $\boldsymbol{\alpha}_{\mathrm{Topic}_i}$ parameters correspond exactly to the $\boldsymbol{\beta}$ parameters of the LDA model, and the $\boldsymbol{\alpha}_{\mathrm{Doc}_j}$ parameters correspond to the $\boldsymbol{\alpha}$ parameters of the LDA model.

As suggested above, each document D_j in the LDA model is mapped to a string in the corpus used to train the corresponding PCFG by prefixing it with a document identifier "$\_j$". Given this training data, the posterior distribution over rule probabilities $\theta_{\mathrm{Doc}_j \rightarrow \, \mathrm{Topic}_i}$ is the same as the posterior distribution over topics given documents $\theta_{j,i}$ in the original LDA model.

As we will see below, this connection between PCFGs and LDA topic models suggests a number of interesting variants of both PCFGs and topic models. Note that we are *not* suggesting that Bayesian inference for PCFGs is necessarily a good way of estimating LDA topic models. Current Bayesian PCFG inference algorithms require time proportional to the cube of the length of the longest string in the training corpus, and since these strings correspond to entire documents in our embedding, blindly applying a Bayesian PCFG inference algorithm is likely to be impractical.

A little reflection shows that the embedding still holds if the strings in the PCFG corpus correspond to sentences or even smaller units of the original document collection, so a single document would be mapped to multiple strings in the PCFG inference task. In this way the cubic time complexity of PCFG inference can be mitigated. Also, the trees generated by these CFGs have a very specialized left-branching structure, and it is straightforward to modify the general-purpose CFG inference procedures to avoid the cubic time complexity for such grammars: thus it may be practical to estimate topic models via grammatical inference.

However, we believe that the primary value of the embedding of LDA topic models into Bayesian PCFGs is theoretical: it suggests a number of novel extensions of both topic models and grammars that may be worth exploring. Our claim here is not that these models are the best algorithms for performing these tasks, but that the relationship we described between LDA models and PCFGs suggests a variety of interesting novel models.

We end this section with a simple example of such a modification to LDA. Inspired by the standard embedding of HMMs into PCFGs, we propose a "sticky topic" variant of LDA in which adjacent words are more likely to be assigned the same topic. Such an LDA extension is easy to describe as a PCFG (see Fox et al. (2008) for a similar model presented as an extended HMM). The nonterminals $\mathrm{Sentence}$ and Topic_i for $i = 1, \ldots, \ell$ have the same interpretation as before, but we introduce new nonterminals $\mathrm{Doc}_{j,i}$ that indicate we have just generated a nonterminal in document j belonging to topic i. Given a collection of m documents and ℓ topics, the rule schemata are as follows:

$$\begin{array}{ll}
\mathrm{Sentence} \rightarrow \ \mathrm{Doc}_{j,i} & i \in 1, \ldots, \ell; \\
 & j \in 1, \ldots, m \\
\mathrm{Doc}_{j,1} \rightarrow \ \_j & j \in 1, \ldots, m \\
\mathrm{Doc}_{j,i} \rightarrow \mathrm{Doc}_{j,i'} \ \mathrm{Topic}_i & i, i' \in 1, \ldots, \ell; \\
 & j \in 1, \ldots, m \\
\mathrm{Topic}_i \rightarrow \ w & i \in 1, \ldots, \ell; w \in V
\end{array}$$

A sample parse generated by a "sticky topic"

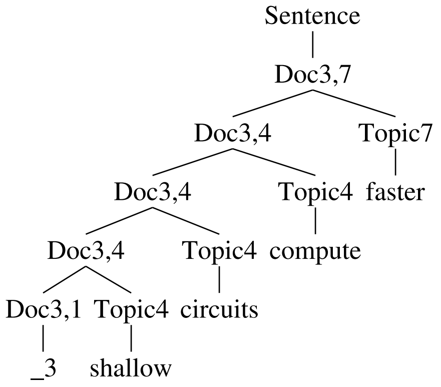

Figure 3: A tree generated by the "sticky topic" CFG. Here a nonterminal $\mathrm{Doc3}, 7$ indicates we have just generated a word in document 3 belonging to topic 7.

CFG is shown in Figure 3. The probabilities of the rules $\mathrm{Doc}_{j,i} \rightarrow \mathrm{Doc}_{j,i'}\ \mathrm{Topic}_i$ in this PCFG encode the probability of shifting from topic i to topic i' (this PCFG can be viewed as generating the string from right to left).

We can use non-uniform sparse Dirichlet priors on the probabilities of these rules to encourage "topic stickiness". Specifically, by setting the Dirichlet parameters for the "topic shift" rules $\mathrm{Doc}_{j,i'} \rightarrow \mathrm{Doc}_{j,i}\ \mathrm{Topic}_i$ where $i' \neq i$ much lower than the parameters for the "topic preservation" rules $\mathrm{Doc}_{j,i} \rightarrow \mathrm{Doc}_{j,i}\ \mathrm{Topic}_i$, Bayesian inference will be biased to find distributions in which adjacent words will tend to have the same topic.

5 Adaptor Grammars

Non-parametric Bayesian inference, where the inference task involves learning not just the values of a finite vector of parameters but which parameters are relevant, has been the focus of intense research in machine learning recently. In the topic-modelling community this has lead to work on Dirichlet Processes and Chinese Restaurant Processes, which can be used to estimate the number of topics as well as their distribution across documents (Teh et al., 2006).

There are two obvious non-parametric extensions to PCFGs. In the first we regard the set of nonterminals N as potentially unbounded, and try to learn the set of nonterminals required to describe the training corpus. This approach goes under the name of the "infinite HMM" or "infinite PCFG" (Beal et al., 2002; Liang et al., 2007; Liang et al., 2009). Informally, we are given a set of "basic categories", say NP, VP, etc., and a set of rules that use these basic categories, say $\mathrm{S} \rightarrow \mathrm{NP}\ \mathrm{VP}$. The inference task is to learn a set of refined categories and rules (e.g., $\mathrm{S}_7 \rightarrow \mathrm{NP}_2\ \mathrm{VP}_5$) as well as their probabilities; this approach can therefore be viewed as a Bayesian version of the "split-merge" approach to grammar induction (Petrov and Klein, 2007).

In the second approach, which we adopt here, we regard the set of rules R as potentially unbounded, and try to learn the rules required to describe a training corpus as well as their probabilities. Adaptor grammars are an example of this approach (Johnson et al., 2007b), where entire subtrees generated by a "base grammar" can be viewed as distinct rules (in that we learn a separate probability for each subtree). The inference task is non-parametric if there are an unbounded number of such subtrees.

We review the adaptor grammar generative process below; for an informal introduction see Johnson (2008) and for details of the adaptor grammar inference procedure see Johnson and Goldwater (2009).

An adaptor grammar $(N, W, R, S, \theta, A, C)$ consists of a PCFG (N, W, R, S, θ) in which a subset $A \subseteq N$ of the nonterminals are *adapted*, and where each adapted nonterminal $X \in A$ has an associated adaptor C_X. An *adaptor* C_X for X is a function that maps a distribution over trees $\mathcal{T}_X$ to a distribution over distributions over $\mathcal{T}_X$ (we give examples of adaptors below).

Just as for a PCFG, an adaptor grammar defines distributions G_X over trees $\mathcal{T}_X$ for each $X \in N \cup W$. If $X \in W$ or $X \notin A$ then G_X is defined just as for a PCFG above, i.e., using (1). However, if $X \in A$ then G_X is defined in terms of an additional distribution H_X as follows:

$$\begin{aligned} G_X &\sim C_X(H_X) \\ H_X &= \sum_{X \rightarrow Y_1 \dots Y_m \in R_X} \theta_{X \rightarrow Y_1 \dots Y_m} \mathrm{TD}_X(G_{Y_1}, \dots, G_{Y_m}) \end{aligned}$$

That is, the distribution G_X associated with an adapted nonterminal $X \in A$ is a sample from adapting (i.e., applying C_X to) its "ordinary" PCFG distribution H_X. In general adaptors are chosen for the specific properties they have. For example, with the adaptors used here G_X typically concentrates mass on a smaller subset of the trees $\mathcal{T}_X$ than H_X does.

Just as with the PCFG, an adaptor grammar generates the distribution over trees G_S, where $S \in N$

is the start symbol. However, while G_S in a PCFG is a fixed distribution (given the rule probabilities θ), in an adaptor grammar the distribution G_S is itself a random variable (because each G_X for $X \in A$ is random), i.e., an adaptor grammar generates a distribution over distributions over trees $\mathcal{T}_S$. However, the posterior joint distribution $\Pr(\boldsymbol{t})$ of a sequence $\boldsymbol{t} = (t_1, \ldots, t_n)$ of trees in $\mathcal{T}_S$ is well-defined:

$$\Pr(\boldsymbol{t}) \;=\; \int G_S(t_1) \ldots G_S(t_n)\, d\boldsymbol{G}$$

where the integral is over all of the random distributions $G_X, X \in A$. The adaptors we use in this paper are Dirichlet Processes or two-parameter Poisson-Dirichlet Processes, for which it is possible to compute this integral. One way to do this uses the predictive distributions:

$$\begin{aligned} &\Pr(t_{n+1} \mid \boldsymbol{t}, H_X) \\ &\quad \propto \int G_X(t_1) \ldots G_X(t_{n+1}) C_X(G_X \mid H_X)\, dG_X \end{aligned}$$

where $\boldsymbol{t} = (t_1, \ldots, t_n)$ and each $t_i \in \mathcal{T}_X$. The predictive distribution for the Dirichlet Process is the (labeled) *Chinese Restaurant Process* (CRP), and the predictive distribution for the two-parameter Poisson-Dirichlet process is the (labeled) *Pitman-Yor Process* (PYP).

In the context of adaptor grammars, the CRP is:

$$\mathrm{CRP}(t \mid \boldsymbol{t}, \alpha_X, H_X) \;\propto\; n_t(\boldsymbol{t}) + \alpha_X H_X(t)$$

where $n_t(\boldsymbol{t})$ is the number of times t appears in $\boldsymbol{t}$ and $\alpha_X > 0$ is a user-settable "concentration parameter". In order to generate the next tree t_{n+1} a CRP either reuses a tree t with probability proportional to number of times t has been previously generated, or else it "backs off" to the "base distribution" H_X and generates a fresh tree t with probability proportional to $\alpha_X H_X(t)$.

The PYP is a generalization of the CRP:

$$\begin{aligned} &\mathrm{PYP}(t \mid \boldsymbol{t}, a_X, b_X, H_X) \\ &\quad \propto \max(0, n_t(\boldsymbol{t}) - m_t\, a_X) + (m a_X + b_X) H_X(t) \end{aligned}$$

Here $a_X \in [0, 1]$ and $b_X > 0$ are user-settable parameters, and m_t is the number of times the PYP has generated t in $\boldsymbol{t}$ from the base distribution H_X, and $m = \sum_{t \in \mathcal{T}_X} m_t$ is the number of times any tree has been generated from H_X. (In the Chinese Restaurant metaphor, m_t is the number of tables labeled with t, and m is the number of occupied tables). If $a_X = 0$ then the PYP is equivalent to a CRP with $\alpha_X = b_X$, while if $a_X = 1$ then the PYP generates samples from H_X.

Informally, the CRP has a strong preference to regenerate trees that have been generated frequently before, leading to a "rich-get-richer" dynamics. The PYP can mitigate this somewhat by reducing the effective count of previously generated trees and redistributing that probability mass to new trees generated from H_X. As Goldwater et al. (2006) explain, Bayesian inference for H_X given samples from G_X is effectively performed from types if $a_X = 0$ and from tokens if $a_X = 1$, so varying a_X smoothly interpolates between type-based and token-based inference.

Adaptor grammars have previously been used primarily to study grammatical inference in the context of language acquisition. The *word segmentation task* involves segmenting a corpus of unsegmented phonemic utterance representations into words (Elman, 1990; Bernstein-Ratner, 1987). For example, the phoneme string corresponding to "you want to see the book" (with its correct segmentation indicated) is as follows:

y △ u ▲ w △ a △ n △ t ▲ t △ u ▲ s △ i ▲ D △ 6 ▲ b △ U △ k

We can represent any possible segmentation of any possible sentence as a tree generated by the following *unigram adaptor grammar*.

$$\begin{aligned} &\text{Sentence} \rightarrow \underline{\text{Word}} \\ &\text{Sentence} \rightarrow \underline{\text{Word}}\ \text{Sentence} \\ &\underline{\text{Word}} \rightarrow \text{Phonemes} \\ &\text{Phonemes} \rightarrow \text{Phoneme} \\ &\text{Phonemes} \rightarrow \text{Phoneme Phonemes} \end{aligned}$$

The trees generated by this adaptor grammar are the same as the trees generated by the CFG rules. For example, the following skeletal parse in which all but the Word nonterminals are suppressed (the others are deterministically inferrable) shows the parse that corresponds to the correct segmentation of the string above.

(Word y u) (Word w a n t) (Word t u)
(Word s i) (Word d 6) (Word b u k)

Because the Word nonterminal is *adapted* (indicated here by underlining) the adaptor grammar learns the probability of the entire Word subtrees (e.g., the probability that *b u k* is a Word); see Johnson (2008) for further details.

6 Topic models with collocations

Here we combine ideas from the unigram word segmentation adaptor grammar above and the PCFG encoding of LDA topic models to present a novel topic model that learns topical collocations. (For a non-grammar-based approach to this problem see Wang et al. (2007)). Specifically, we take the PCFG encoding of the LDA topic model described above, but modify it so that the Topic_i nodes generate sequences of words rather than single words. Then we adapt each of the Topic_i nonterminals, which means that we learn the probability of each of the sequences of words it can expand to.

$$\begin{array}{ll}
\mathrm{Sentence} \to \mathrm{Doc}_j & j \in 1, \ldots, m \\
\mathrm{Doc}_j \to \_j & j \in 1, \ldots, m \\
\mathrm{Doc}_j \to \mathrm{Doc}_j \ \mathrm{Topic}_i & i \in 1, \ldots, \ell; \\
 & j \in 1, \ldots, m \\
\underline{\mathrm{Topic}_i} \to \mathrm{Words} & i \in 1, \ldots, \ell \\
\mathrm{Words} \to \mathrm{Word} & \\
\mathrm{Words} \to \mathrm{Words\ Word} & \\
\mathrm{Word} \to w & w \in V
\end{array}$$

In order to demonstrate that this model works, we implemented this using the publically-available adaptor grammar inference software,[1] and ran it on the NIPS corpus (composed of published NIPS abstracts), which has previously been used for studying collocation-based topic models (Griffiths et al., 2007). Because there is no generally accepted evaluation for collocation-finding, we merely present some of the sample analyses found by our adaptor grammar. We ran our adaptor grammar with $\ell = 20$ topics (i.e., 20 distinct Topic_i nonterminals). Adaptor grammar inference on this corpus is actually relatively efficient because the corpus provided by Griffiths et al. (2007) is already segmented by punctuation, so the terminal strings are generally rather short. Rather than set the Dirichlet parameters by hand, we placed vague priors on them and estimated them as described in Johnson and Goldwater (2009).

The following are some examples of collocations found by our adaptor grammar:

Topic_0 → cost function
Topic_0 → fixed point
Topic_0 → gradient descent
Topic_0 → learning rates
Topic_1 → associative memory
Topic_1 → hamming distance
Topic_1 → randomly chosen
Topic_1 → standard deviation
Topic_3 → action potentials
Topic_3 → membrane potential
Topic_3 → primary visual cortex
Topic_3 → visual system
Topic_{10} → nervous system
Topic_{10} → action potential
Topic_{10} → ocular dominance
Topic_{10} → visual field

The following are skeletal sample parses, where we have elided all but the adapted nonterminals (i.e., all we show are the Topic nonterminals, since the other structure can be inferred deterministically). Note that because Griffiths et al. (2007) segmented the NIPS abstracts at punctuation symbols, the training corpus contains more than one string from each abstract.

_3 (Topic$_5$ polynomial size)
(Topic$_{15}$ threshold circuits)

_4 (Topic$_{11}$ studied)
(Topic$_{19}$ pattern recognition algorithms)

_4 (Topic$_2$ feedforward neural network)
(Topic$_1$ implementation)

_5 (Topic$_{11}$ single)
(Topic$_{10}$ ocular dominance stripe)
(Topic$_{12}$ low) (Topic$_3$ ocularity)
(Topic$_{12}$ drift rate)

7 Finding the structure of proper names

Grammars offer structural and positional sensitivity that is not exploited in the basic LDA topic models. Here we explore the potential for using Bayesian inference for learning linear ordering constraints that hold between elements within proper names.

The Penn WSJ treebank is a widely used resource within computational linguistics (Marcus et al., 1993), but one of its weaknesses is that it does not indicate any structure internal to base noun phrases (i.e., it presents "flat" analyses of the pre-head NP elements). For many applications it would be extremely useful to have a more elaborated analysis of this kind of NP structure. For example, in an NP coreference application, if we could determine that *Bill* and *Hillary* are both first

[1] http://web.science.mq.edu.au/ ˜mjohnson/Software.htm

names then we could infer that *Bill Clinton* and *Hillary Clinton* are likely to refer to distinct individuals. On the other hand, because *Mr* in *Mr Clinton* is not a first name, it is possible that *Mr Clinton* and *Bill Clinton* refer to the same individual (Elsner et al., 2009).

Here we present an adaptor grammar based on the insights of the PCFG encoding of LDA topic models that learns some of the structure of proper names. The key idea is that elements in proper names typically appear in a fixed order; we expect honorifics to appear before first names, which appear before middle names, which in turn appear before surnames, etc. Similarly, many company names end in fixed phrases such as *Inc.* Here we think of first names as a kind of topic, albeit one with a restricted positional location. One of the challenges is that some of these structural elements can be filled by multiword expressions; e.g., *de Groot* can be a surname. We deal with this by permitting multi-word collocations to fill the corresponding positions, and use the adaptor grammar machinery to learn these collocations.

Inspired by the grammar presented in Elsner et al. (2009), our adaptor grammar is as follows, where adapted nonterminals are indicated by underlining as before.

$$\begin{array}{l}
\mathrm{NP} \rightarrow (\mathrm{A0})\ (\mathrm{A1})\ \ldots\ (\mathrm{A6}) \\
\mathrm{NP} \rightarrow (\mathrm{B0})\ (\mathrm{B1})\ \ldots\ (\mathrm{B6}) \\
\mathrm{NP} \rightarrow \mathrm{Unordered}^{+} \\
\underline{\mathrm{A0}} \rightarrow \mathrm{Word}^{+} \\
\quad \ldots \\
\underline{\mathrm{A6}} \rightarrow \mathrm{Word}^{+} \\
\underline{\mathrm{B0}} \rightarrow \mathrm{Word}^{+} \\
\quad \ldots \\
\underline{\mathrm{B6}} \rightarrow \mathrm{Word}^{+} \\
\underline{\mathrm{Unordered}} \rightarrow \mathrm{Word}^{+}
\end{array}$$

In this grammar parentheses indicate optionality, and the Kleene plus indicates iteration (these were manually expanded into ordinary CFG rules in our experiments). The grammar provides three different expansions for proper names. The first expansion says that a proper name can consist of some subset of the six different collocation classes A0 through A6 in that order, while the second expansion says that a proper name can consist of some subset of the collocation classes B0 through B6, again in that order. Finally, the third expansion says that a proper name can consist of an arbitrary sequence of "unordered" collocations (this is intended as a "catch-all" expansion to provide analyses for proper names that don't fit either of the first two expansions).

We extracted all of the proper names (i.e., phrases of category NNP and NNPS) in the Penn WSJ treebank and used them as the training corpora for the adaptor grammar just described. The adaptor grammar inference procedure found skeletal sample parses such as the following:

(A0 barrett) (A3 smith)
(A0 albert) (A2 j.) (A3 smith) (A4 jr.)
(A0 robert) (A2 b.) (A3 van dover)
(B0 aim) (B1 prime rate) (B2 plus) (B5 fund) (B6 inc.)
(B0 balfour) (B1 maclaine) (B5 international) (B6 ltd.)
(B0 american express) (B1 information services) (B6 co)
(U abc) (U sports)
(U sports illustrated)
(U sports unlimited)

While a full evaluation will have to await further study, in general it seems to distinguish person names from company names reasonably reliably, and it seems to have discovered that person names consist of a first name (A0), a middle name or initial (A2), a surname (A3) and an optional suffix (A4). Similarly, it seems to have uncovered that company names typically end in a phrase such as *inc*, *ltd* or *co*.

8 Conclusion

This paper establishes a connection between two very different kinds of probabilistic models; LDA models of the kind used for topic modelling, and PCFGs, which are a standard model of hierarchical structure in language. The embedding we presented shows how to express an LDA model as a PCFG, and has the property that Bayesian inference of the parameters of that PCFG produces an equivalent model to that produced by Bayesian inference of the LDA model's parameters.

The primary value of this embedding is theoretical rather than practical; we are not advocating the use of PCFG estimation procedures to infer LDA models. Instead, we claim that the embedding suggests novel extensions to both the LDA topic models and PCFG-style grammars. We justified this claim by presenting several hybrid models that combine aspects of both topic models and grammars. We don't claim that these are necessarily the best models for performing any particular tasks; rather, we present them as examples of models inspired by a combination of PCFGs and LDA topic models. We showed how the LDA to PCFG embedding suggested a "sticky topic" model extension to LDA. We then discussed adaptor grammars, and inspired by the LDA topic models, presented a novel topic model whose primitive elements are multi-word collocations rather than words. We concluded with an adaptor grammar that learns aspects of the internal structure of proper names.

Acknowledgments

This research was funded by US NSF awards 0544127 and 0631667, as well as by a start-up award from Macquarie University. I'd like to thank the organisers and audience at the Topic Modeling workshop at NIPS 2009, my former colleagues at Brown University (especially Eugene Charniak, Micha Elsner, Sharon Goldwater, Tom Griffiths and Erik Sudderth), my new colleagues at Macquarie University and the ACL reviewers for their excellent suggestions and comments on this work. Naturally all errors remain my own.

References

M.J. Beal, Z. Ghahramani, and C.E. Rasmussen. 2002. The infinite Hidden Markov Model. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, pages 577–584. The MIT Press.

N. Bernstein-Ratner. 1987. The phonology of parent-child speech. In K. Nelson and A. van Kleeck, editors, *Children's Language*, volume 6. Erlbaum, Hillsdale, NJ.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jeffrey Elman. 1990. Finding structure in time. *Cognitive Science*, 14:197–211.

Micha Elsner, Eugene Charniak, and Mark Johnson. 2009. Structured generative models for unsupervised named-entity clustering. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 164–172, Boulder, Colorado, June. Association for Computational Linguistics.

E. Fox, E. Sudderth, M. Jordan, and A. Willsky. 2008. An HDP-HMM for systems with state persistence. In Andrew McCallum and Sam Roweis, editors, *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, pages 312–319. Omnipress.

Sharon Goldwater, Tom Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 459–466, Cambridge, MA. MIT Press.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:52285235.

Thomas L. Griffiths, Mark Steyvers, and Joshua B. Tenenbaum. 2007. Topics in semantic representation. *Psychological Review*, 114(2):211244.

Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325, Boulder, Colorado, June. Association for Computational Linguistics.

Mark Johnson, Thomas Griffiths, and Sharon Goldwater. 2007a. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 139–146, Rochester, New York, April. Association for Computational Linguistics.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007b. Adaptor Grammars: A framework for specifying compositional nonparametric Bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. MIT Press, Cambridge, MA.

Mark Johnson. 2008. Using adaptor grammars to identifying synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*, Columbus, Ohio. Association for Computational Linguistics.

Kenichi Kurihara and Taisuke Sato. 2006. Variational Bayesian grammar induction for natural language. In *8th International Colloquium on Grammatical Inference*.

Percy Liang, Slav Petrov, Michael Jordan, and Dan Klein. 2007. The infinite PCFG using hierarchical Dirichlet processes. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 688–697.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Probabilistic grammars and hierarchical Dirichlet processes. In *The Oxford Handbook of Applied Bayesian Analysis*. Oxford University Press.

Michell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York. Association for Computational Linguistics.

Y. W. Teh, M. Jordan, M. Beal, and D. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581.

Xuerui Wang, Andrew McCallum, and Xing Wei. 2007. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM)*, pages 697–702.

C.S. Wetherell. 1980. Probabilistic languages: A review and some open questions. *Computing Surveys*, 12:361–379.