Journal of Machine Learning Research ?? (2008) ??-?? Submitted 04/08; Published ??/??

# Pachinko Allocation: Scalable Mixture Models of Topic Correlations

**Wei Li** WEILI@CS.UMASS.EDU
*Department of Computer Science*
*University of Massachusetts*
*Amherst, MA 01003, USA*

**Andrew McCallum** MCCALLUM@CS.UMASS.EDU
*Department of Computer Science*
*University of Massachusetts*
*Amherst, MA 01003, USA*

**Editor:** ??

## Abstract

Statistical topic models are increasingly popular tools for summarization and manifold discovery in discrete data. However, the majority of existing approaches capture no or limited correlations between topics. In this paper, we propose the *pachinko allocation* model (PAM), which captures arbitrary topic correlations using a directed acyclic graph (DAG). The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). As we have observed, topic correlations are usually sparse. By taking advantage of this property, we develop a highly-scalable inference algorithm for PAM. In our experiments, we show improved performance of PAM in document classification, likelihood of held-out data, topical keyword coherence, and the ability to support a great number of fine-grained topics in very large datasets.

**Keywords:** pachinko allocation, topic models, Gibbs sampling

## 1. Introduction

A topic model defines a probabilistic procedure to generate documents as mixtures of a low-dimensional set of topics. Each topic is a multinomial distribution over words and the highest probability words briefly summarize the themes in the document collection. As an effective tool to dimensionality reduction and semantic information extraction, topic models have been used to analyze large amounts of textual information in many tasks, including language modeling, document classification, information retrieval, document summarization, data mining and social network analysis (Gildea and Hofmann (1999); Tam and Schultz (2005); Azzopardi et al. (2004); Wei and Croft (2006); Kim et al. (2005); Tuulos and Tirri (2004); Rosen-Zvi et al. (2004); McCallum et al. (2005)). In addition to textual data (including news articles, research papers and emails), they have also been applied to images, biological findings and other non-textual multi-dimensional discrete data (Sivic et al. (2005); Zheng et al. (2006)).

While topic models capture correlation patterns in words, the majority of existing approaches capture no or limited correlations among topics themselves. However, such correlations are common and complex in real-world data. The simplified assumptions of independent topics reduce the abilities of these models to discover large numbers of fine-grained, tightly coherent topics. For example, latent Dirichlet allocation (LDA) (Blei et al. (2003)) samples the per-document mixture proportions of topics from a single Dirichlet distribution and thus does not model topic correlations. The correlated topic model (CTM) (Blei and Lafferty (2006)) uses a logistic normal distribution instead of a Dirichlet. The parameters of this prior include a covariance matrix in which each entry specifies the covariance between a pair of topics. Therefore, topic occurrences are no longer independent from each other. However, CTM is limited to pairwise correlations only, and the number of parameters in the covariance matrix grows as the square of the number of topics.

In this paper, we present the *pachinko allocation* model (PAM), which uses a directed acyclic graph (DAG) structure to represent and learn arbitrary-arity, nested, and possibly sparse topic correlations. In PAM topics can be not only distributions over words, but also distributions over other topics. The model structure consists of an arbitrary DAG, in which each leaf node is associated with a word in the vocabulary, and each non-leaf "interior" node corresponds to a topic, having a distribution over its children. An interior node whose children are all leaves would correspond to a traditional topic. But some interior nodes may also have children that are other topics, thus representing a mixture over topics. With many such nodes, PAM therefore captures not only correlations among words (as in LDA), but also correlations among topics themselves. We also present recent new work in the construction of sparse variants on PAM that provide significantly faster parameter estimation times.

For example, consider a document collection that discusses four topics: *cooking*, *health*, *insurance* and *drugs*. The *cooking* topic co-occurs often with *health*, while *health*, *insurance* and *drugs* are often discussed together. A DAG can describe this kind of correlation. For each topic, we have one node that is directly connected to the words. There are two additional nodes at a higher level, where one is the parent of *cooking* and *health*, and the other is the parent of *health*, *insurance* and *drugs*.

In PAM each interior node's distribution over its children could be parameterized arbitrarily. We will investigate various options including multinomial distribution and Dirichlet compound multinomial (DCM). Given a DAG and a parameterization, the generative process samples a topic path for each word. It begins at the root of the DAG, sampling one of its children according to the corresponding distribution, and so on sampling children down the DAG until it reaches a leaf, which yields a word. The model is named for pachinko machines—a game popular in Japan, in which metal balls bounce down around a complex collection of pins until they land in various bins at the bottom.

Note that the DAG structure in PAM is extremely flexible. It could be a simple tree (hierarchy), or an arbitrary DAG, with cross-connected edges, and edges skipping levels. The nodes can be fully or sparsely connected. The structure could be fixed beforehand or learned from the data. PAM provides a general framework for which several existing models can be viewed as special cases such as LDA and CTM. We present a variety of model structures in Figure 1.

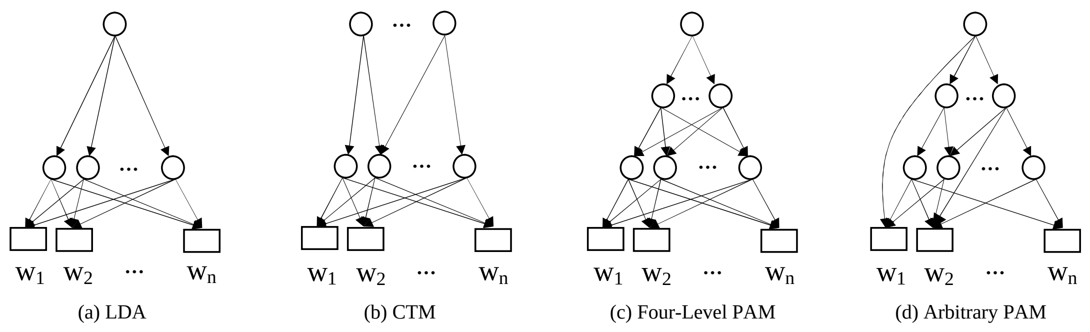

Figure 1: Model structures for four topic models. Each rectangle is associated with a word and each circle corresponds to a topic, where it has a distribution over its children specified by the arrows. (a) LDA: This model samples a multinomial over topics for each document, and then generates words from the topics. (b) CTM: Each topic at the lower level is a multinomial distribution over words and for each pair of them, there is one additional topic that has a distribution over them. (c) Four-Level PAM: A four-level hierarchy consisting of a root, a set of super-topics, a set of sub-topics and a word vocabulary. (d) PAM: An arbitrary DAG structure to encode the topic correlations. Each interior node is considered a topic and has a distribution over its children.

Since PAM captures complex topic correlations with a rich structure, its computational cost is increased accordingly. As we will explain later, the running time of a naive implementation of our inference algorithm is proportional to the product of the corpus size and the numbers of nodes at each level of the DAG. In addition, it requires more memory space since we need to store the distribution of every interior node over its children. This makes it challenging to use PAM in real-world applications such as information retrieval, where the datasets contain hundreds of thousands of documents and many topics. Below in Section 3, we address the efficiency issue and propose a highly-scalable approximation.

As we have observed in our experiments, the connectivity structure discovered by PAM is usually sparse. It is especially true for very large datasets with a lot of topics, most of which only contain small proportions of the word vocabulary. This property suggests more efficient inference and training algorithms by considering only a small subset of topics to sample a word in a document. Furthermore, we use a sparse representation to store the distributions of interior nodes over their children. In this way, we are able to dramatically reduce the time and space complexity.

Using text data from newsgroups and various research paper corpora, we show improved performance of PAM in three different tasks, including topical word coherence assessed by human judges, likelihood on held-out test data, and document classification accuracy. The approximation with sparsity is applied to large datasets with improved efficiency.

| | |
|---|---|
| $V$ | word vocabulary $\{w_1, w_2, ..., w_n\}$ |
| $T$ | a set of topics $\{t_1, t_2, ..., t_s\}$ |
| $r$ | the root, a special topic in $T$ |
| $g_i(\alpha_i)$ | Dirichlet distribution associated with topic $t_i$ |
| $d$ | a document |
| $\theta_{t_i}^{(d)}$ | multinomial distribution sampled from topic $t_i$ for document $d$ |
| $z_{wi}$ | the $i$th topic sampled for word $w$ |

Table 1: Notation in PAM.

## 2. Pachinko Allocation Model

In this section, we present the pachinko allocation model (PAM). In addition to the general framework, we will focus on one special setting, describing the generative process, inference algorithm and parameter estimation method.

### 2.1 General Framework

The notation for the pachinko allocation model is summarized in Table 2.1. PAM connects words in $V$ and topics in $T$ with a DAG structure, where topic nodes occupy the interior levels and the leaves are words. Several possible model structures are shown in Figure 1. Each topic $t_i$ is associated with a distribution $g_i$ over its children. In general, $g_i$ could be any distribution over discrete variables, such as Dirichlet compound multinomial and logistic normal.

First we describe the generative process for PAM with an arbitrary DAG, assuming that the distributions associated with topics are Dirichlet compound multinomials. Each distribution $g_i$ is parameterized with a vector $\alpha_i$, which has the same dimension as the number of children in $t_i$.

To generate a document $d$, we use the following two-step process:

1. Sample $\theta_{t_1}^{(d)}$, $\theta_{t_2}^{(d)}$, ..., $\theta_{t_s}^{(d)}$ from $g_1(\alpha_1)$, $g_2(\alpha_2)$, ..., $g_s(\alpha_s)$, where $\theta_{t_i}^{(d)}$ is a multinomial distribution of topic $t_i$ over its children.
2. For each word $w$ in the document,
   (a) Sample a topic path $\mathbf{z}_w$ of length $L_w$: $< z_{w1}, z_{w2}, ..., z_{wL_w} >$. $z_{w1}$ is always the root $r$ and $z_{w2}$ through $z_{wL_w}$ are topic nodes in $T$. $z_{wi}$ is a child of $z_{w(i-1)}$ and it is sampled according to the multinomial distribution $\theta_{z_{w(i-1)}}^{(d)}$.
   (b) Sample word $w$ from $\theta_{z_{wL_w}}^{(d)}$.

Following this process, the joint probability of generating a document $d$, the topic assignments $\mathbf{z}^{(d)}$ and the multinomial distributions $\theta^{(d)}$ is

$$P(d, \mathbf{z}^{(d)}, \theta^{(d)}|\alpha) = \prod_{i=1}^{s} P(\theta_{t_i}^{(d)}|\alpha_i) \times \prod_{w} (\prod_{i=2}^{L_w} P(z_{wi}|\theta_{z_{w(i-1)}}^{(d)}) P(w|\theta_{z_{wL_w}}^{(d)}))$$

Integrating out $\theta^{(d)}$ and summing over $\mathbf{z}^{(d)}$, we calculate the marginal probability of a document as:

$$P(d|\alpha) = \int \prod_{i=1}^{s} P(\theta_{t_i}^{(d)}|\alpha_i) \times \prod_{w} \sum_{\mathbf{z}_w} (\prod_{i=2}^{L_w} P(z_{wi}|\theta_{z_{w(i-1)}}^{(d)}) P(w|\theta_{z_{wL_w}}^{(d)})) \mathrm{d}\theta^{(d)}$$

Finally, the probability of generating a whole corpus is the product of the probability for every document:

$$P(\mathbf{D}|\alpha) = \prod_{d} P(d|\alpha)$$

### 2.2 Four-Level PAM

In PAM, both the DAG structure and parameterization can be very flexible. We will focus on one special case here. It is a four-level hierarchy consisting of one root topic $r$, $s$ topics at the second level $T = \{t_1, t_2, ..., t_s\}$, $s'$ topics at the third level $T' = \{t'_1, t'_2, ..., t'_{s'}\}$ and words at the bottom. We call the topics at the second level *super-topics* and the ones at the third level *sub-topics*. The root is connected to all super-topics, super-topics are fully connected to sub-topics and sub-topics are fully connected to words (Figure 1(c)).

We use two different distributions for the topics in this structure. In addition to a set of Dirichlet compound multinomials associated with the root $g_r(\alpha_r)$ and super-topics $\{g_i(\alpha_i)\}_{i=1}^{s}$, the sub-topics are modeled with fixed multinomial distributions $\{\phi_{t'_j}\}_{j=1}^{s'}$, sampled once for the whole corpus from a single Dirichlet distribution $g(\beta)$. The corresponding graphical model is shown in Figure 2. The generative process for a document $d$ is as follows:

1. Sample $\theta_r^{(d)}$ from the root $g_r(\alpha_r)$, where $\theta_r^{(d)}$ is a multinomial distribution over super-topics.
2. For each super-topic $t_i$, sample $\theta_{t_i}^{(d)}$ from $g_i(\alpha_i)$, where $\theta_{t_i}^{(d)}$ is a multinomial distribution over sub-topics.
3. For each word $w$ in the document,
   (a) Sample a super-topic $z_w$ from $\theta_r^{(d)}$.
   (b) Sample a sub-topic $z'_w$ from $\theta_{z_w}^{(d)}$.
   (c) Sample word $w$ from $\phi_{z'_w}$.

As we can see, both the model structure and generative process for this special setting are similar to LDA. The major difference is that it has one additional layer of super-topics modeled with Dirichlet compound multinomials, which is the key component capturing topic correlations here. Another way to interpret this structure is that given the sub-topics, each super-topic is essentially an individual LDA. Therefore, it can be viewed as a mixture over a set of LDAs.

Following this process, the joint probability of generating a document $d$, the super-topic assignments $\mathbf{z}^{(d)}$, the sub-topic assignments $\mathbf{z}'^{(d)}$ and the multinomial distributions $\theta^{(d)}$ is

$$P(d, \mathbf{z}^{(d)}, \mathbf{z}'^{(d)}, \theta^{(d)}|\alpha, \phi) = P(\theta_r^{(d)}|\alpha_r) \prod_{i=1}^{s} P(\theta_{t_i}^{(d)}|\alpha_i) \times \prod_{w} (P(z_w|\theta_r^{(d)}) P(z'_w|\theta_{z_w}^{(d)}) P(w|\phi_{z'_w}))$$

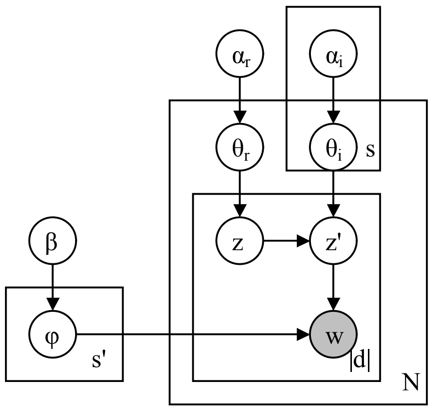

Figure 2: Graphical model for four-level PAM. For each document, PAM samples multinomials $\theta$ from the Dirichlet distributions at the root and the super-topics. Then for each word $w$, PAM samples a super-topic $z$ and a sub-topic $z'$ from $\theta$ and then the word from the multinomial distribution $\phi_{z'}$ associated with $z'$.

Integrating out $\theta^{(d)}$ and summing over $\mathbf{z}^{(d)}$ and $\mathbf{z}'^{(d)}$, we calculate the marginal probability of a document as:

$$P(d|\alpha,\phi) = \int P(\theta_r^{(d)}|\alpha_r)\prod_{i=1}^{s} P(\theta_{t_i}^{(d)}|\alpha_i) \times \prod_w \sum_{z_w, z'_w} (P(z_w|\theta_r^{(d)})P(z'_w|\theta_{z_w}^{(d)})P(w|\phi_{z'_w}))\mathrm{d}\theta^{(d)}$$

The probability of generating a whole corpus is the product of the probability for every document, integrating out the multinomial distributions for sub-topics $\phi$:

$$P(\mathbf{D}|\alpha,\beta) = \int \prod_{j=1}^{s'} P(\phi_{t'_j}|\beta)\prod_d P(d|\alpha,\phi)\mathrm{d}\phi$$

### 2.3 Inference

The hidden variables in the four-level PAM include the sampled multinomial distributions $\theta$, $\phi$ and topic assignments $\mathbf{z}^{(d)}$, $\mathbf{z}'^{(d)}$. Since Dirichlet priors are conjugate to the multinomial distributions, we can calculate the joint distribution of $P(\mathbf{D},\mathbf{z},\mathbf{z}')$ by integrating out $\theta$ and $\phi$. However, for the purpose of inference, we still need to calculate the conditional distribution

$$P(\mathbf{z},\mathbf{z}'|\mathbf{D}) = \frac{P(\mathbf{D},\mathbf{z},\mathbf{z}')}{P(\mathbf{D})}$$

Since it requires summing over all possible topic assignments to obtain the marginal distribution of $P(\mathbf{D})$, it is not feasible to perform exact inference in this model. One of the standard approximation techniques for models in the LDA family is Gibbs sampling. For an arbitrary DAG, we need to sample a topic path for each word given other variable assignments, enumerating all possible paths and calculating their conditional probabilities. In the special four-level PAM structure, each path contains the root, a super-topic and a sub-topic. Since the root is fixed, we only need to jointly sample the super-topic and sub-topic assignments for each word, based on their conditional distribution given observations and other assignments. In order to calculate this probability, we start with the joint distribution of the documents and topic assignments:

$$P(\mathbf{D}, \mathbf{z}, \mathbf{z}'|\alpha, \beta) = P(\mathbf{z}|\alpha) \times P(\mathbf{z}'|\mathbf{z}, \alpha) \times P(\mathbf{D}|\mathbf{z}', \beta)$$

By integrating out the sampled multinomials, we have

$$\begin{aligned}
P(\mathbf{z}|\alpha) &= \int \prod_d P(\theta_r^{(d)}|\alpha_r) \prod_w P(z_w|\theta_r^{(d)}) \mathrm{d}\theta \\
&= \left(\frac{\Gamma(\sum_{i=1}^s \alpha_{ri})}{\prod_{i=1}^s \Gamma(\alpha_{ri})}\right)^{|D|} \prod_d \frac{\prod_{i=1}^s \Gamma(n_i^{(d)} + \alpha_{ri})}{\Gamma(n_r^{(d)} + \sum_{i=1}^s \alpha_{ri})} \\
P(\mathbf{z}'|\mathbf{z}, \alpha) &= \int \prod_d (\prod_{i=1}^s P(\theta_{t_i}^{(d)}|\alpha_i) \prod_w P(z'_w|\theta_{z_w}^{(d)})) \mathrm{d}\theta \\
&= \prod_{i=1}^s \left\{ \left(\frac{\Gamma(\sum_{j=1}^{s'} \alpha_{ij})}{\prod_{j=1}^{s'} \Gamma(\alpha_{ij})}\right)^{|D|} \prod_d \frac{\prod_{j=1}^{s'} \Gamma(n_{ij}^{(d)} + \alpha_{ij})}{\Gamma(n_i^{(d)} + \sum_{j=1}^{s'} \alpha_{ij})} \right\} \\
P(\mathbf{D}|\mathbf{z}', \beta) &= \int \prod_{j=1}^{s'} P(\phi_{t'_j}|\beta) \prod_d (\prod_w P(w|\phi_{z'_w})) \mathrm{d}\phi \\
&= \left(\frac{\Gamma(\sum_{k=1}^n \beta_k)}{\prod_{k=1}^n \Gamma(\beta_k)}\right)^{s'} \prod_{j=1}^{s'} \frac{\prod_{k=1}^n \Gamma(n_{jk} + \beta_k)}{\Gamma(n_j + \sum_{k=1}^{s'} \beta_k)}
\end{aligned}$$

Here $n_r^{(d)}$ is the number of occurrences of the root $r$ in document $d$, which is equivalent to the number of tokens in the document; $n_i^{(d)}$ is the number of occurrences of super-topic $t_i$ in $d$; $n_{ij}^{(d)}$ is the number of times that sub-topic $t'_j$ is sampled from the super-topic $t_i$ in $d$; $n_j$ is the total number of occurrences of sub-topic $t'_j$ in the whole corpus and $n_{jk}$ is the number of occurrences of word $w_k$ in sub-topic $t'_j$. There are three types of Dirichlet parameters: $\alpha_r$ is an $s$-dimensional vector associated with the root, $\alpha_i$ is an $s'$-dimensional vector associated with super-topic $t_i$ and $\beta$ is the prior for all sub-topics. Finally, we obtain the Gibbs sampling distribution for word $w = w_k$ in document $d$ as

$$\begin{aligned}
P(z_w = t_i, z'_w = t'_j|\mathbf{D}, \mathbf{z}_{-w}, \mathbf{z}'_{-w}, \alpha, \beta) &\propto P(w, z_w, z'_w|\mathbf{D}_{-w}, \mathbf{z}_{-w}, \mathbf{z}'{}_{-w}, \alpha, \beta) \\
&= \frac{P(\mathbf{D}, \mathbf{z}, \mathbf{z}'|\alpha, \beta)}{P(\mathbf{D}_{-w}, \mathbf{z}_{-w}, \mathbf{z}'{}_{-w}|\alpha, \beta)} \\
&= \frac{n_i^{(d)} + \alpha_{ri}}{n_r^{(d)} + \sum_{i=1}^s \alpha_{ri}} \frac{n_{ij}^{(d)} + \alpha_{ij}}{n_i^{(d)} + \sum_{j=1}^{s'} \alpha_{ij}} \frac{n_{jk} + \beta_k}{n_j + \sum_{k=1}^n \beta_k}.
\end{aligned}$$

The notation $-w$ indicates all observations or topic assignments except word $w$. Also the numbers of occurrences do not include $w$ or its assignments. With this distribution, we jointly sample a super-topic and sub-topic pair for every word in every document. As we can see, the time complexity of each Gibbs sampling iteration is linear with the total number of tokens in the training corpus and the size of the sample space for each token, *i.e.* the product of the numbers of super-topics and sub-topics.

### 2.4 Parameter Estimation

Note that in the Gibbs sampling equation, we assume that the Dirichlet parameters $\alpha$ are given. While LDA can produce reasonable results with a simple uniform Dirichlet, we have to learn these parameters for the super-topics in PAM since they capture different correlations among sub-topics. As for the root, we assume a fixed Dirichlet parameter. To learn $\alpha$, we could use maximum likelihood or maximum a posteriori estimation. However, since there are no closed-form solutions for these methods and we wish to avoid iterative methods for the sake of simplicity and speed, we approximate it by moment matching (Casella and Berger (2001)). In each iteration of Gibbs sampling, the parameters are updated according to the following rules:

$$\begin{aligned}
mean_{ij} &= \frac{1}{N_i} \times \sum_d \frac{n_{ij}^{(d)}}{n_i^{(d)}} \\
var_{ij} &= \frac{1}{N_i} \times \sum_d \left(\frac{n_{ij}^{(d)}}{n_i^{(d)}} - mean_{ij}\right)^2 \\
m_{ij} &= \frac{mean_{ij} \times (1 - mean_{ij})}{var_{ij}} - 1 \\
\alpha_{ij} &= \frac{mean_{ij}}{\exp\left(\frac{\sum_j log(m_{ij})}{s'-1}\right)}
\end{aligned}$$

For each super-topic $t_i$ and sub-topic $t'_j$, we first calculate the sample mean $mean_{ij}$ and sample variance $var_{ij}$. $n_{ij}^{(d)}$ and $n_i^{(d)}$ are the same as defined before. If $n_i^{(d)} = 0$ for a document $d$, it will be ignored. $N_i$ is the total number of documents with non-0 counts of super-topic $t_i$. Then we estimate $\alpha_{ij}$, the $j$th component in $\alpha_i$ from sample means and variances.

Smoothing is important when estimating the Dirichlet parameters with moment matching. From the equations above, we can see that when one sub-topic $t'_j$ does not get sampled from super-topic $t_i$ in one iteration, $\alpha_{ij}$ will become 0. Furthermore from the Gibbs sampling equation, we know that this sub-topic will never have the chance to be sampled again by this super-topic. In order to avoid this situation, we introduce a smoothing factor by assuming a pseudo-document where each pair of super-topic and sub-topic is sampled exactly once:

$$\begin{aligned}
mean_{ij} &= \frac{1}{(N_i + 1)} \times \left( \sum_d \frac{n_{ij}^{(d)}}{n_i^{(d)}} + \frac{1}{s'} \right) \\
var_{ij} &= \frac{1}{(N_i + 1)} \times \left( \sum_d \left(\frac{n_{ij}^{(d)}}{n_i^{(d)}} - mean_{ij}\right)^2 + \left(\frac{1}{s'} - mean_{ij}\right)^2 \right)
\end{aligned}$$

## 3. Improving Efficiency in Topic Models

Many of the real-world applications for which topic models can be useful involve very large document collections. One example is information retrieval (IR). An IR system aims to understand a user's information request in the form of a query and extracts a set of relevant documents from its corpus. As queries are usually short lists of keywords, their relevant documents may not contain the exact terms. Therefore, it is more desirable to compare them in a low-dimensional space than direct word matching. Word clustering techniques have been used in the past to organize words into different groups based on co-occurrence patterns (Liu and Croft (2004)). Similarly, topic models can also be applied to discover latent concepts in document collections. They provide not only a low-dimensional representation for the words, but also a probabilistic framework to evaluate relevance between queries and documents. In the recent work by Wei and Croft (Wei and Croft (2006)), LDA has been used for ad-hoc retrieval and achieved improved performance over standard query likelihood model and a cluster-based approach.

While LDA can be applied to IR collections with reasonable efficiency, it is not practical to directly apply PAM with richer structures to very large datasets that consist of hundreds of thousands of documents. In this section, we will analyze the time and space complexity in Gibbs sampling and propose a more efficient training algorithm. With this approximation, we can reduce the training time of LDA by more than 50% without decreasing its performance in IR tasks. It also allows us to discover topic correlations in large collections with the sparse PAM.

### 3.1 Complexity of Gibbs Sampling

In general, the running time of one Gibbs sampling iteration is determined by $\sum_i V(x_i)$, where $x_i$ is an unobserved variable and $V(x_i)$ is the size of its value space. In the case of PAM, $V(x_i)$ equals the number of topic paths for each token. For the four-level DAG structure, we assume that the super-topics are fully connected with sub-topics and sub-topics are fully connected with words. Therefore, the number of topic paths is the product of the numbers of super-topics ($s$) and sub-topics ($s'$). The total running time for Gibbs sampling is then linear with $s$, $s'$, $L$ and $I$, where $L$ is the number of tokens in the corpus and $I$ is the number of iterations. Compared to an LDA with $s'$ topics, the four-level PAM will be $s'$ times slower.

Another source of increased complexity in PAM is memory usage. In addition to the per-corpus distribution of every sub-topic over words, we also need to store the per-document distributions of the root over super-topics and super-topics over sub-topics. So the space complexity is $O(s'n + sN + ss'N)$, where $n$ is the vocabulary size and $N$ is the number of documents.

### 3.2 Sparse PAM

We now present several techniques to build a sparse PAM.

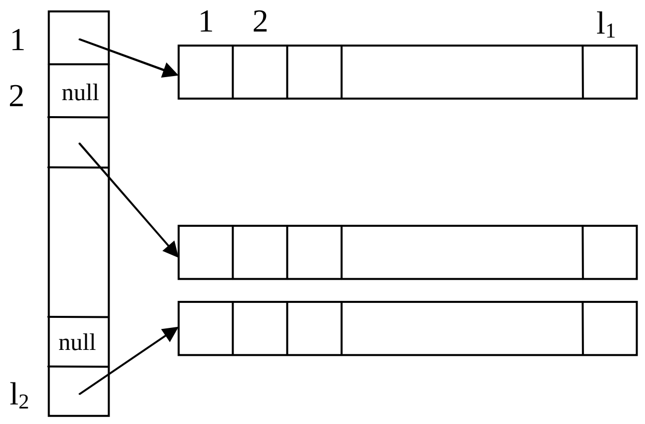

Figure 3: An example of the sparse array representation. Each block contains either 0 or $l_1$ items.

### 3.2.1 Sparse Representation

The main usage of memory is to store two tables: $n_{ij}^{(d)}$, the number of times sub-topic $t_j$ is sampled from super-topic $t_i$ in document $d$; and $n_{jk}$, the number of occurrences of word $w_k$ in sub-topic $t_j$. As we have observed, a large proportion of the table entries are 0s. For the IR collections in our experiments, the average number of tokens in each document is around 200, which is much less than the number of sub-topics we usually use. Therefore, most documents only contain a small number of sub-topics, and the table $n_{ij}^{(d)}$ is extremely sparse. Similarly, when we have a large vocabulary and a lot of sub-topics, the topic-word table $n_{jk}$ is also sparse. To take advantage of this property and reduce memory complexity, we use a sparse representation to store these tables.

The basic idea is to divide an array into different blocks. If one block contains all 0's, we do not allocate memory for it. One example is shown in Figure 3. There are two parameters in this representation: $l_1$ is the size of the blocks and $l_2$ is the number of blocks. Therefore it can store $l_1 \times l_2$ numbers. When accessing the $i$th number in the original array, we calculate the block index as $i/l_1$ and the within-block index as $i\%l_1$.

### 3.2.2 Edge Pruning

In addition to sparsity in these tables, we have also observed another kind of sparsity in the sample space. When sampling a topic path for a token, we need to consider all (super-topic, sub-topic) pairs. Because of the Dirichlet priors, every pair has a non-0 probability. However, some pairs are much less likely to be sampled than others. In our experiments, after several iterations of Gibbs sampling, we can obtain an average of 98% of probability mass by considering only topic pairs that consist of

- super-topic $t_i$, if $n_i^{(d)} > 0$;

- sub-topic $t'_j$, if there exists $t_i$ such that $n_{ij}^{(d)} > 0$;
- sub-topic $t'_j$, if $n_{jk} > 0$ for the current token $w_k$.

This provides us with a simple way to dramatically reduce the sample space without losing too much probability. However, there are some disadvantages with this pruning strategy. For example, if a super-topic is not assigned to any word in a document, it will never be considered for that document again. This is not desirable because our initialization is usually random. Our solution to this problem is to alternate accurate and approximate samplings for each document. The prunning algorithm is described below.

For every document $d$,

1. Let $C = \emptyset$, $C' = \emptyset$;
2. For every super-topic $t_i$,
   If $n_i^{(d)} > 0$, $C = C \cup \{t_i\}$;
3. For every sub-topic $t'_j$,
   If there exists $t_i$ such that $n_{ij}^{(d)} > 0$, $C' = C' \cup \{t'_j\}$;
4. For every word $w = w_k$ in the document,
   (a) Remove the current topic assignments for $w$;
   (b) Let $C'_w = \emptyset$;
   (c) For every sub-topic $t'_j$,
   If $n_{jk} > 0$, $C'_w = C'_w \cup \{t'_j\}$;
   (d) $C'_w = C'_w \cup C'$;
   (e) Let $X = C \times C'_w$;
   (f) Sample the new topic assignments $< t_i, t'_j >$ from $X$;
   (g) Update $C$, $C'$ and the tables.

#### 3.2.3 Sparse Initialization

So far we have presented two methods to reduce complexity in running time and memory, both of which depending on the sparsity in topic connections. However, when we initialize Gibbs samplers in a random way, the sparsity is not very obvious at the beginning. Therefore to further accelerate the training procedure, we use a different initialization method. We start with a very small number of documents and run Gibbs sampling for a few iterations until the topic connections become sparse. Then we gradually add more and more documents. Note that we do not randomly sample the topics for a newly added document. They are treated the same way as other documents except that the pruning strategy is a little different.

For a newly added document $d$,

1. Let $C$ = all super-topics;
2. For every word $w = w_k$ in the document,

```
(a) If $w_k$ is a new word, let $C'_w =$ all sub-topics;
(b) Else
    i. Let $C'_w = \emptyset$;
    ii. For every sub-topic $t'_j$,
        If $n_{jk} > 0$, $C'_w = C'_w \cup \{t'_j\}$;
(c) Let $X = C \times C'_w$;
(d) Sample the new topic assignments $< t_i, t'_j >$ from $X$;
(e) Update the tables.
```

#### 3.2.4 Multiple Markov Chains

The pruning algorithm significantly decreases one factor in the training time of PAM, *i.e.* the number of topic paths for each token. It is also possible to reduce the number of iterations by using multiple Markov chains (Wei and Croft (2006)). In our experiments, we initialize several Gibbs samplers with different randomization seeds. The average probabilities from multiple chains provide better performance than individual ones.

### 3.3 PAM-based Retrieval

With the sparse approximations, now we are able to apply PAM to ad-hoc retrieval. The basic framework we use here is the query likelihood (QL) model. For each document $d$, we first estimate a language model—a distribution over words $\{P(w|d)\}$. Then the documents are ranked according to their likelihoods of generating a query $q =< q_1, q_2, ..., q_l >$:

$$P(q|d) = \prod_{i=1}^{l} P(q_i|d)$$

The query terms are assumed to be conditionally independent from each other given the document model. One simple way of estimating $P(w|d)$ is maximum likelihood estimation (MLE). In other words,

$$P_{MLE}(w|d) = n_w^{(d)}/n^{(d)},$$

where $n_w^{(d)}$ is the number of occurrences of $w$ in $d$ and $n^{(d)}$ is the total number of tokens in $d$. As the distribution learned by MLE is usually sparse, we use the Dirichlet smoothing (Zhai and Lafferty (2001)) to obtain

$$P_{QL}(w|d) = \frac{n^{(d)}}{n^{(d)} + \mu} P_{MLE}(w|d) + \frac{\mu}{n^{(d)} + \mu} P_{MLE}(w)$$

Here $\mu$ is a smoothing parameter and $P_{MLE}(w)$ is the maximum likelihood estimation from the whole corpus.

Topic models provide another way to estimate the word distribution for each document. In the case of four-level PAM with $s$ super-topics and $s'$ sub-topics, we can estimate $P_{PAM}(w|d)$ as the probability of predicting a new word $w$ given the posterior distributions

$\hat{\theta}$ and $\hat{\phi}$. Therefore, we have

$$\begin{aligned}
P_{PAM}(w = w_k|d, \hat{\theta}, \hat{\phi}) &= \sum_{i=1}^{s}\sum_{j=1}^{s'} P(w = w_k, z_w = t_i, z'_w = t'_j|d, \hat{\theta}, \hat{\phi}) \\
&= \sum_{i=1}^{s}\sum_{j=1}^{s'} P(z_w = t_i|\hat{\theta}^{(d)})P(z'_w = t'_j|z_w, \hat{\theta}^{(d)})P(w = w_k|z'_w, \hat{\phi}) \\
&= \sum_{i=1}^{s}\sum_{j=1}^{s'} \frac{n_i^{(d)} + \alpha_{ri}}{n_r^{(d)} + \sum_{i=1}^{s}\alpha_{ri}} \frac{n_{ij}^{(d)} + \alpha_{ij}}{n_i^{(d)} + \sum_{j=1}^{s'}\alpha_{ij}} \frac{n_{jk} + \beta_k}{n_j + \sum_{k=1}^{n}\beta_k}.
\end{aligned}$$

Here $n_r^{(d)}$ is the number of occurrences of the root $r$ in document $d$, which is equivalent to the number of tokens in the document; $n_i^{(d)}$ is the number of occurrences of super-topic $t_i$ in $d$; $n_{ij}^{(d)}$ is the number of times that sub-topic $t'_j$ is sampled from the super-topic $t_i$ in $d$; $n_j$ is the total number of occurrences of sub-topic $t'_j$ in the whole corpus and $n_{jk}$ is the number of occurrences of word $w_k$ in sub-topic $t'_j$. $\alpha$ and $\beta$ are parameters in the Dirichlet distributions. Note that when we have $M$ Markov chains, we use their average to calculate

$$P_{PAM}(w|d) = \frac{1}{M}\sum_{m} P_{PAM(m)}(w|d).$$

As we can see, PAM estimates the probability for a document to generate a word via underlying topics. Unlike MLE, a word that does not occur in a document may still have a high probability if it occurs often in a topic discussed in the document. The corresponding word distributions are generally more smooth than MLE. Therefore, they may not be precise enough to distinguish between relevant and non-relevant documents. As previous work has shown, LDA-based representation alone does not perform as well as the query likelihood model while a combination of them can improve the retrieval performance (Wei and Croft (2006)). In our experiments, we also use a linear combination with a parameter $\lambda$:

$$P(w|d) = \lambda P_{QL}(w|d) + (1 - \lambda)P_{PAM}(w|d)$$

## 4. Experimental Results

### 4.1 Four-Level PAM

In this section, we present example topics that PAM discovers from real-world text data and evaluate against LDA using three measures: topic clarity by human judgement, likelihood of held-out test data, and document classification accuracy. We also compare held-out data likelihood with CTM and HDP.

In the experiments described below, we use a fixed four-level hierarchical structure for PAM, which includes a root, a set of super-topics, a set of sub-topics and a word vocabulary. For the root, we always assume a fixed symmetric Dirichlet distribution, where each component in the parameter vector is 0.01. This parameter can be changed to adjust the variance in the sampled multinomial distributions. We choose a small value so that the variance is high and each document contains only a small number of super-topics,

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| speech | 0.0694 | agents | 0.0909 | market | 0.0281 | students | 0.0619 |
| recognition | 0.0562 | agent | 0.0810 | price | 0.0218 | education | 0.0445 |
| text | 0.0441 | plan | 0.0364 | risk | 0.0191 | learning | 0.0332 |
| word | 0.0315 | actions | 0.0336 | find | 0.0145 | training | 0.0309 |
| words | 0.0289 | planning | 0.0260 | markets | 0.0138 | children | 0.0281 |
| system | 0.0194 | communication | 0.0246 | information | 0.0126 | teaching | 0.0197 |
| algorithm | 0.0194 | world | 0.0198 | prices | 0.0123 | school | 0.0185 |
| task | 0.0183 | decisions | 0.0194 | equilibrium | 0.0116 | student | 0.0180 |
| acoustic | 0.0183 | situation | 0.0165 | financial | 0.0116 | educational | 0.0146 |
| training | 0.0173 | decision | 0.0151 | evidence | 0.0111 | quality | 0.0129 |

Table 2: Example sub-topics discovered by PAM with 50 super-topics and 100 sub-topics from the Rexa dataset. On the left side of each column are the top 10 words in a sub-topic and on the right side are their corresponding probabilities.

which tends to make the super-topics more interpretable. The multinomial distributions for sub-topics are sampled once for the whole corpus from a given Dirichlet with parameter 0.01. Therefore the only parameters we need to learn are the Dirichlet parameters for the super-topics.

We use the standard setting for LDA, where the prior for the topic mixture proportions is a symmetric Dirichlet with parameter 1.0 and the prior for the topic distributions over words is 0.01. There are three parameters in the HDP. Following the same procedure described in (Teh et al. (2005)), we assume Gamma priors for the Dirichlet process parameters: $\alpha_0 \sim$ Gamma(1.0, 1.0) and $\gamma \sim$ Gamma(1.0, 0.1). Similar to PAM and LDA, the Dirichlet distribution over topics has a parameter of 0.01.

In Gibbs sampling for PAM, LDA and HDP, we use 2,000 burn-in iterations, and then draw 10 samples in the following 1,000 iterations. Each Gibbs sampler is initialized randomly. For PAM and LDA, we uniformly sample the topic assignment for each token. For HDP, we start with no topic at all and generate new topics when adding documents successively. In each iteration, we update the Dirichlet parameters in PAM and HDP.

CTM is trained with variational expectation-maximization (EM). In the E-step, we perform variational inference for each document. In the M-step, we optimize the parameters in each topic and the logistic normal distribution by maximum likelihood estimation. This algorithm is run until convergence. The implementation was provided by David Blei. The training speed is approximately 4 times faster than PAM with 50 super-topics.

### 4.1.1 Topic Examples

Our first dataset comes from Rexa, a search engine over research papers (http://Rexa.info). We randomly choose a subset of titles and abstracts from its large collection. In this dataset, there are 4,000 documents, 278,438 word tokens and 25,597 unique words. We use 50 super-topics and 100 sub-topics for PAM. The total training time is approximately 4 hours on a 2.4 GHz Opteron machine with 2GB memory. In Table 2, we show some of the topic examples. Each column corresponds to one sub-topic and lists the top 10 words and their probabilities.

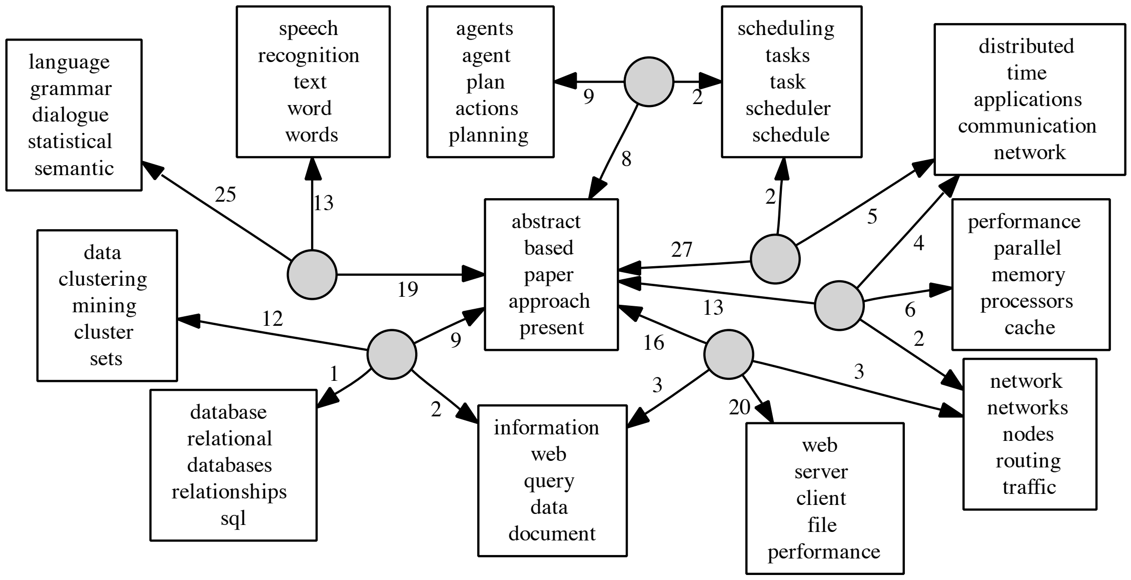
language grammar dialogue statistical semantic
speech recognition text word words
agents agent plan actions planning
scheduling tasks task scheduler schedule
distributed time applications communication network
data clustering mining cluster sets
abstract based paper approach present
performance parallel memory processors cache
database relational databases relationships sql
information web query data document
web server client file performance
network networks nodes routing traffic
25
13
19
9
2
8
2
27
5
4
6
2
13
16
3
20
3
12
9
1
2

Figure 4: Topic correlation in PAM. This is a small proportion of the topic structure in the Rexa dataset. We use 50 super-topics and 100 sub-topics for PAM. Each circle corresponds to a super-topic and each box corresponds to a sub-topic. One super-topic can connect to several sub-topics and capture their correlation. The numbers on the edges are the corresponding $\alpha$ values for the (super-topic, sub-topic) pair.

Figure 4 shows a subset of super-topics in the data, and how they capture correlations among sub-topics. For each super-topic $t_i$, we rank the sub-topics $\{t'_j\}$ based on the learned Dirichlet parameter $\alpha_{ij}$. In this graph, each circle corresponds to one super-topic and links to a set of sub-topics as shown by the boxes. The numbers on the edges are the corresponding $\alpha$ values. As we can see, all the super-topics here share the same sub-topic in the middle, which is a subset of stopwords in this corpus. Some super-topics also share the same content sub-topics. For example, the topic about *scheduling* and *tasks* co-occur with the topic about *agents* and also the topic about *distributed systems*. Another example is *information retrieval*. It is discussed along with both the *data mining* topic and the *web, network* topics.

### 4.1.2 Human Judgement

As a preliminary comparison of topic clarity between PAM and LDA, we conduct blind topic evaluation. Each of five human evaluators is provided with a set of topic pairs generated from the two models, anonymized and in random order. Evaluators are asked to choose which one has stronger sense of semantic coherence and specificity.

For this experiment, we use the NIPS abstract dataset (NIPS00-12), which includes 1,647 documents, a vocabulary of 11,708 words and 114,142 word tokens. We have 100

| PAM | LDA | PAM | LDA |
|---|---|---|---|
| control | control | motion | image |
| systems | systems | image | motion |
| robot | based | detection | images |
| adaptive | adaptive | images | multiple |
| environment | direct | scene | local |
| goal | con | vision | generated |
| state | controller | texture | noisy |
| controller | change | segmentation | optical |
| 5 votes | 0 vote | 4 votes | 1 vote |
| PAM | LDA | PAM | LDA |
| signals | signal | algorithm | algorithm |
| source | signals | learning | algorithms |
| separation | single | algorithms | gradient |
| eeg | time | gradient | convergence |
| sources | low | convergence | stochastic |
| blind | source | function | line |
| single | temporal | stochastic | descent |
| event | processing | weight | converge |
| 4 votes | 1 vote | 1 vote | 4 votes |

Table 3: Four topic pair examples provided to the human evaluators. Each pair consists of one sub-topic in PAM and one topic in LDA. They are generated from the NIPS dataset with 100 topics for LDA, and 50 super-topics and 100 sub-topics for PAM.

topics for LDA, and 50 super-topics and 100 sub-topics for PAM. The topics are generated from the final sample in Gibbs sampling and the pairs are created based on similarity. For each sub-topic in PAM, we find its most similar topic in LDA and consider them as a pair. We also find the most similar sub-topic in PAM for each LDA topic. Similarity is measured by the KL-divergence between topic distributions over words. After removing redundant pairs and dissimilar pairs that share less than 5 out of their top 20 words, we provide the evaluators with a total of 25 pairs. Several examples are shown in Table 3. There are 5 PAM topics that every evaluator agrees to be the better ones in their pairs, while LDA has none. Out of 25 pairs, 19 topics from PAM are chosen by the majority ($\geq$ 3 votes). We present the full evaluation results in Table 4.

### 4.1.3 Likelihood Comparison

In addition to human evaluation of topics, we also provide quantitative measurements to compare PAM with LDA, CTM and HDP. In this experiment, we use the same NIPS dataset and split it into two subsets with 75% and 25% of the data respectively. Then we learn the models from the larger set and calculate likelihood for the smaller set. The performance of PAM is more sensitive to the number of sub-topics than the number of super-topics. We

| | LDA | PAM |
|---|---|---|
| 5 votes | 0 | 5 |
| $\geq$ 4 votes | 3 | 8 |
| $\geq$ 3 votes | 9 | 16 |

Table 4: Human judgement results for the NIPS dataset. For all the categories: 5 votes, $\geq$ 4 votes and $\geq$ 3 votes, PAM topics are favored over LDA.

have experimented with 10, 20, 50 and 100 super-topics. While the best result is obtained with 50 super-topics, it is not significantly better than using 20 super-topics. The other two settings produce slightly worse results. Therefore we fix the number of super-topics to be 50, and the number of sub-topics varies from 20 to 180.

In order to calculate the likelihood of held-out data, we must integrate out the sampled multinomials and sum over all possible topic assignments. This problem has no closed-form solution. Previous work that uses Gibbs sampling for inference approximates the likelihood of a document $d$ by the harmonic mean of a set of conditional probabilities $P(d|\mathbf{z}^{(d)})$, where the samples are generated using Gibbs sampling (Griffiths and Steyvers (2004)). However, this approach has been shown to be unstable because the inverse likelihood does not have finite variance (Chib (1995)) and has been widely criticized (e.g. (Newton and Raftery (1994)) discussion).

In our experiments, we employ a more robust alternative in the family of non-parametric likelihood estimates—specifically an approach based on empirical likelihood (EL), *e.g.* (Diggle and Gratton (1984)). In these methods one samples data from the model, and calculates the empirical distribution from the samples. In cases where the samples are sparse, a kernel may be employed. For each topic model, we use the following algorithm to estimate data likelihood:

1. Randomly sample $M$ documents from the trained model, based on its own generative process.

2. For each sample $d_s$, estimate its distribution over words $P(w|d_s)$ as the frequency of word $w$ in $d_s$.

3. For each test document $d_t$,

   (a) $P(d_t|d_s) = \prod_w P(w|d_s)$;

   (b) $P(d_t) = \sum_s P(d_t|d_s)P(d_s) = \frac{1}{M}\sum_s P(d_t|d_s)$.

One thing to note is that the estimation of $P(w|d_s)$ requires smoothing since a sample document usually cannot cover every word in the vocabulary. However, in many cases, we can avoid that by combining the first two steps together. In other words, we don't need to sample the actual words in the documents but directly calculate their probabilities. For example, in the case of PAM, we only sample the multinomial distributions in each

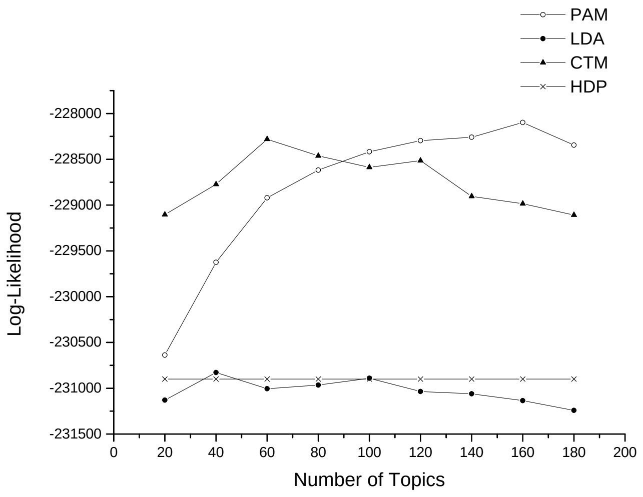

Figure 5: Likelihood comparison on the NIPS dataset with different numbers of topics. The results for PAM, LDA and HDP are averages over 10 different samples and the maximum standard error is 113.75. PAM uses a fixed number of 50 super-topics. HDP is a straight line because it automatically determines the number of topics.

document. Then for each word $w$, we have

$$P(w|d_s) = \sum_i \sum_j P(t_i|d_s) P(t'_j|t_i, d_s) P(w|t'_j)$$

Here $t_i$ is a super-topic and $t'_j$ is a sub-topic. $P(t_i|d_s)$ and $P(t'_j|t_i, d_s)$ are probabilities from the sampled multinomials in $d_s$. $P(w|t'_j)$ is the learned posterior distribution of $t'_j$ over words. This technique has also been applied to other models.

The number of samples $M$ is set to be 1,000 for each model. Unlike in Gibbs sampling, the samples are unconditionally generated; therefore, they are not restricted to the topic co-occurrences observed in the held-out data, as they are in the harmonic mean method.

We show the log-likelihoods on the test data for different numbers of topics in Figure 5. CTM is evaluated after variational EM converges, while the results of PAM, LDA and HDP are averages over 10 different samples. Compared to LDA, PAM always produces higher likelihoods for different numbers of sub-topics. The advantage is especially obvious with more topics. LDA's performance peaks at 40 topics and decreases as the number of topics increases. On the other hand, PAM supports larger numbers of topics and has its best performance at 160 sub-topics. When the number of topics is small, CTM exhibits better performance than both LDA and PAM. However, as we use more and more topics, its likelihood starts to decrease. The peak value for CTM is at 60 topics and it is lower than the best performance of PAM. We also apply HDP to this dataset. Since there is no pre-defined data structure, HDP does not model any topic correlations but automatically learns the number of topics. Therefore, the result of HDP does not change with the number of topics and it is similar to the best result of LDA. According to paired t-test results, the improvements of PAM over other topic models are statistically significant.

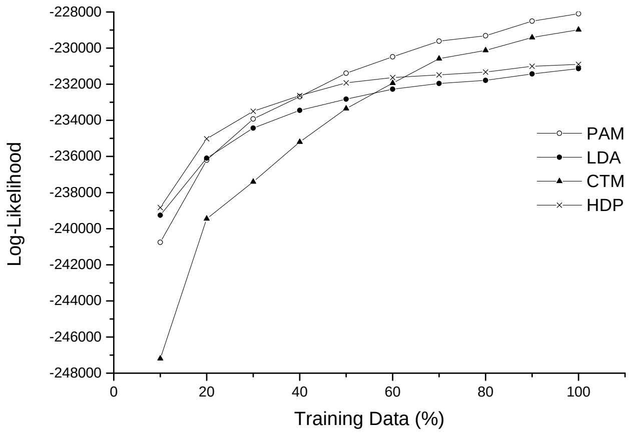

Figure 6: Likelihood comparison on the NIPS dataset with different amounts of training data. The results for PAM, LDA and HDP are averages over 10 different samples and the maximum standard error is 171.72.

| | LDA | HDP | CTM | PAM |
|---|---|---|---|---|
| bits-per-word | 11.43 | 11.43 | 11.31 | 11.29 |

Table 5: Average numbers of bits to represent a word in the NIPS dataset. The results are based on the the best settings in each model.

Based on the data likelihood, we can calculate the average number of bits needed to represent a word as

$$-\frac{\sum_d \log_2 P(d)}{\sum_d n^{(d)}},$$

where $d$ is a test document and $n^{(d)}$ is its length. The corresponding numbers for the best results in each model are shown in Table 5.

We also present the log-likelihoods for different numbers of training documents in Figure 6. The results are all based on 160 topics except for HDP. As we can see, the performance of CTM is noticeably worse than the other three models when there is limited amount of training data. One possible reason is that CTM has a large number of parameters to learn especially when the number of topics is large. Therefore it suffers from overfitting with insufficient training documents.

In Figure 7, we show how the empirical likelihood changes over 5,000 Gibbs sampling iterations. The model is PAM with 50 super-topics and 160 sub-topics. As we can see, it increases rapidly before 1,000 iterations and gradually stabilizes after that.

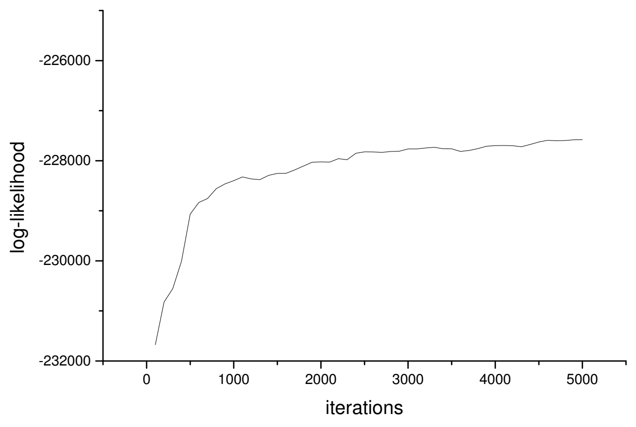

Figure 7: Log-likelihood for PAM over 5,000 iterations.

### 4.1.4 Document Classification

Another quantitative evaluation to compare PAM with LDA is document classification. The discovered topics can be utilized for this task in various ways. For example, the document-topic frequencies provide a low-dimensional set of features in addition to word frequencies. In our experiment, we choose a simpler method to use the topics for document classification, which demonstrates topic quality without much influence from other factors such as the choice of classifiers or feature engineering.

The data used in this evaluation is the comp5 subset of the 20 newsgroups dataset, which contains 4,836 documents with a vocabulary of 35,567 words. We conduct a 5-way classification, where the documents in every class are randomly divided into 75% training and 25% test datasets. For each class $c$, we use its own training set to learn a PAM model $M_c$. Then for a test document $d$, the predicted class label $L(d)$ is the one that assigns the highest probability to it:

$$L(d) = \arg\max_c P(d|M_c)$$

A similar approach is taken for LDA. The document likelihood is calculated in the same way as described in Section 3.5.3. We have tried 50, 100 and 200 topics. The best settings for PAM and LDA are 100 and 50 topics respectively. Again, PAM's performance is relatively insensitive to the number of super-topics and we still use 50 in this case. The classification accuracies for both PAM and LDA are presented in Table 6. Each row corresponds to one class and the last one is for all documents. As we can see, PAM outperforms LDA in every class. According to the sign test, the overall improvement of PAM over LDA is statistically significant with a $p$-value $< 0.05$.

## 4.2 Sparse Approximations

In this section, we present experimental results with the sparse approximations for both LDA and PAM.

| class | # docs | LDA | PAM |
|---|---|---|---|
| graphics | 243 | 83.95 | 86.83 |
| os | 239 | 81.59 | 84.10 |
| pc | 245 | 83.67 | 88.16 |
| mac | 239 | 86.61 | 89.54 |
| windows.x | 243 | 88.07 | 92.20 |
| total | 1209 | 84.70 | 87.34 |

Table 6: Document classification accuracies (%) for the 20 newsgroups comp5 dataset. We use 50 topics for LDA, and 50 super-topics and 100 sub-topics for PAM. Each row corresponds to one class and the last one shows the overall performance.

| Collection | Num. of Docs | Vocabulary Size | Data Size | Queries |
|---|---|---|---|---|
| AP | 242,918 | 255,928 | 0.73Gb | TREC 51-150 |
| SJMN | 90,257 | 150,890 | 0.29Gb | TREC 51-150 |

Table 7: Statistics of the two IR collections.

#### 4.2.1 Sparse LDA

Since LDA is a special case of PAM, the techniques we described in the previous section can also be applied to it. We will first evaluate the sparse LDA on information retrieval, focusing on the efficiency improvement compared to the ordinary LDA.

We use two TREC collections in our experiments: the Associated Press Newswire (AP) 1988-90 with queries 51-150 and San Jose Mercury News (SJMN) 1991 with queries 51-150. Statistics of the datasets are summarized in Table 7. Relevance evaluation comes from a judged pool of top documents retrieved by previous TREC participants. We only use queries from the "title" field of TREC topics, excluding the ones that do not have any relevant documents in the judged pool. This setting is exactly the same as used in (Wei and Croft (2006)).

There are four parameters we need to determine: $s'$, $M$, $\mu$ and $\lambda$. There is a thorough discussion about the effects of using different values for them in (Wei and Croft (2006)). Since the focus here is about training efficiency, we simply use their best settings: $s' = 800$, $M = 3$, $\mu = 1000$ and $\lambda = 0.7$. For the sparse LDA, we experiment with both random (sLDA model 1) and sparse (sLDA model 2) initializations for Gibbs sampling. The sparse initialization starts with 10000 documents. Then we double the number of documents every 5 iterations until the whole collection is processed.

We show the average retrieval precisions for LDA, sLDA models 1 and 2 in Tables 8 and 9. For each model, we draw a total of 5 samples with 20 iterations apart. For sLDA model 2 on AP collection, we ignored the first sample because it has not processed all training documents yet. Avg1 corresponds to the average result from 3 individual Markov chains and Avg2 is the result of combining them together. According to t-test results, there is no significant difference between the best results of LDA and sLDA.

| Iterations | LDA | | sLDA 1 | | sLDA 2 | |
|---|---|---|---|---|---|---|
| | Avg1 | Avg2 | Avg1 | Avg2 | Avg1 | Avg2 |
| 20 | 23.54 | 24.71 | 23.88 | 24.97 | | |
| 40 | 24.48 | 25.73 | 24.65 | 25.92 | 24.00 | 25.39 |
| 60 | 24.71 | 25.97 | 25.12 | 26.32 | 24.35 | 25.57 |
| 80 | 24.95 | 26.06 | 25.17 | 26.36 | 24.71 | 26.04 |
| 100 | 25.17 | 26.32 | 25.24 | 26.47 | 24.94 | 26.25 |

Table 8: Average retrieval precisions (%) of LDA, sLDA models 1 and 2 on the AP collection. For Avg1, we use individual word distributions from 3 Markov chains and then calculate their average performance. For Avg2, the Markov chains are combined first to calculate the average word distributions, which are then used for retrieval.

| Iterations | LDA | | sLDA 1 | | sLDA 2 | |
|---|---|---|---|---|---|---|
| | Avg1 | Avg2 | Avg1 | Avg2 | Avg1 | Avg2 |
| 20 | 20.71 | 21.58 | 20.68 | 21.70 | 20.38 | 21.34 |
| 40 | 21.19 | 22.33 | 21.35 | 22.20 | 21.04 | 21.96 |
| 60 | 21.55 | 22.60 | 22.02 | 22.87 | 21.53 | 22.36 |
| 80 | 21.82 | 22.83 | 22.11 | 22.94 | 21.76 | 22.65 |
| 100 | 21.84 | 22.83 | 22.14 | 22.96 | 21.91 | 22.78 |

Table 9: Average retrieval precisions (%) of LDA, sLDA models 1 and 2 on the SJMN collection.

We also show the training time spent for each model in Table 10. Compared to LDA, sLDA model 1 requires 22% less time for the AP dataset and 31% less time for the SJMN dataset. We improve the efficiency even more with sLDA model 2, which reduces the training time by 54% and 51% for the two collections. With the sparse initialization, we can reduce the sample space more rapidly than the random initialization. For example, at the 30th iteration for the AP dataset, sLDA model 1 still considers an average of 308 topics for each token while sLDA model 2 only considers 133 topics.

### 4.2.2 Sparse PAM

With the sparse approximation, we are now able to apply PAM to very large datasets and discover a lot of topics and their correlations. For this experiment, we use a subset of the Rexa corpus, which contains 1,339,137 documents and 383,082 words in the vocabulary. For a four-level DAG structure with 100 super-topics and 800 sub-topics, the sample space for each token is 80,000 topic paths. It takes almost 1 day for an ordinary PAM to finish one Gibbs sampling iteration. However, when we use sparse PAM, the average number of topic paths for each token stabilizes around 120 after all documents are processed. Some

| | LDA | sLDA 1 | sLDA2 |
|---|---|---|---|
| AP | 27h59m | 21h46m | 12h50m |
| SJMN | 6h9m | 4h14m | 3h1m |

Table 10: Total running time for 100 iterations of Gibbs sampling in LDA and sLDA models 1 and 2.

of the largest sub-topics are listed in Table 11. As we can see, they cover a wide range of topics in computer science, biology, economy, social studies, etc. In addition to these large topics, sparse PAM is also able to discover some very specific ones with only several hundreds or thousands of words. This is one advantage of being able to include a large number of topics. We show some examples in Table 12. The first topic talks about foraging techniques inspired from ant colonies; the second one is about word sense disambiguation; the third topic is a university in the city of Zurich, Switzerland and the fourth one is about ancient culture.

We have also discovered some interesting patterns in topic correlations. Table 13 shows super-topic examples with some of their top 5 sub-topics. The first example consists of topics about *speech recognition*, *neural networks* and generic words. This a typical combination of two independent but related topics. We observe a similar pattern in the second example. On the other hand, the third super-topic is only about *economy* with different sub-topics emphasizing different aspects.

We also apply sparse PAM to information retrieval, using 100 super-topics and 800 sub-topics. Parameters $M$, $\mu$ and $\lambda$ have the same values as used for LDA. The Gibbs samplers are initialized sparsely. While the training time of sparse PAM has been dramatically reduced compared to normal PAM, it is still slower than LDA. The retrieval precisions on the AP collection are shown in Table 14. The first sample from sPAM at iteration 20 is ignored because it does not include all documents. According to t-test results, there is no significant difference between the two models.

## 5. Related Work

The problem of discovering a low-dimensional representation for large text collections has been widely studied in the machine learning and information retrieval (IR) community. In this section, we will review related work in this area, including probabilistic latent semantic indexing, latent Dirichlet allocation and its variants, nonparametric approaches to structure learning and topic models with dynamic properties.

### 5.1 Probabilistic Latent Semantic Indexing

Latent semantic indexing (LSI) (Deerwester et al. (1990)) is one approach to dimensionality reduction for text collections. By applying singular value decomposition (SVD) to the high-dimensional matrix representation of document-word frequencies, LSI produces a low-dimensional latent-semantic space. Similarities between documents can then be evaluated abstract model time paper based

research software development design paper

image motion based object images

network networks communication routing mobile

protein dna structure proteins cell

information web user retrieval query

neural control networks network systems

speech recognition word language system

ray emission galaxies abstract observations

flow fluid model flows numerical

model ocean sea ice surface

robot control mobile robots system

performance cache data scheduling memory

knowledge based learning reasoning representation

web services service based server

user interaction interface computer human

computer science report university technical

wave scattering elastic waves boundary

models bayesian carlo monte markov

market price pricing prices model

growth business economic analysis paper

economic states united policy social

video coding compression image mpeg

logic programs program semantics programming

text semantic language word lexical

complexity bounds lower bound log

education students school study children

graph graphs problem number vertices

language xml semantics programming languages

agents agent social communication information

data query database queries databases

object oriented objects programming model

estimation model distribution regression models

health care clinical medical patient

planning plan plans problem decision

social study black health effects

matrix linear method methods matrices

neurons muscle cells activity rat

water ozone atmospheric measurements air

Table 11: Topics discovered by sparse PAM from the Rexa dataset. Each line lists the top five words in one sub-topic. They are sorted according to the number of word occurrences.

| |
|---|
| ant ants colony pheromone foraging |
| sense word disambiguation wordnet senses |
| technische hochschule zrich zurich eidgenssische |
| tribes mizoram harappan jewish bc |

Table 12: Some very specific topics discovered by sparse PAM from the Rexa dataset. Each line lists the top five words in one sub-topic.

| | |
|---|---|
| super-topic #1 | |
| | neural control networks network systems |
| | speech recognition word language system |
| | algorithm problem algorithms time problems |
| super-topic #2 | |
| | language xml semantics programming languages |
| | network traffic control networks performance |
| | 0.02083 design software system architecture hardware |
| | web services service based server |
| super-topic #3 | |
| | economic states united policy social |
| | growth business economic analysis paper |
| super-topic #4 | |
| | protein dna structure proteins cell |
| | brain human cortex activity cerebral |
| | brain patients disease study heart |
| super-topic #5 | |
| | model ocean climate water sea |
| | phase dynamics liquid molecular particle |
| | temperature high surface growth films |
| | wave scattering elastic waves boundary |

Table 13: Super-topic examples discovered by sparse PAM from the Rexa dataset.

in the new space, which captures word co-occurrence information and provides more robust estimation than simple word matching.

As an alternative to LSI, Hofmann introduced probabilistic latent semantic indexing (pLSI) (Hofmann (1999)), a generative model for latent semantic analysis. In pLSI, each document has a multinomial distribution over a set of latent classes, where each of them has a multinomial distribution over words. To generate a document, pLSI repeatedly samples a class based on the per-document multinomial and then a word from this class. As a probabilistic model, pLSI has the advantage of using statistical techniques for model estimation and other related problems. However, the multinomial distributions associated with the training documents are treated as parameters in the model instead of being generated from

| Iterations | LDA | | sPAM | |
|---|---|---|---|---|
| | Avg1 | Avg2 | Avg1 | Avg2 |
| 20 | 23.54 | 24.71 | | |
| 40 | 24.48 | 25.73 | 24.35 | 25.57 |
| 60 | 24.71 | 25.97 | 24.80 | 25.96 |
| 80 | 24.95 | 26.06 | 24.81 | 25.90 |
| 100 | 25.17 | 26.32 | 24.97 | 26.03 |

Table 14: Average retrieval precisions (%) of LDA and sPAM on the AP collection.

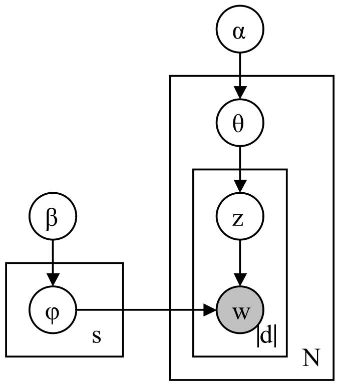

Figure 8: The graphical model for LDA. $\alpha$ is the parameter of the Dirichlet distribution, from which the per-document mixture proportions $\theta$ are sampled. $\beta$ is the parameter for the Dirichlet prior on the topic distributions $\phi$. For each word $w$, LDA samples one topic $z$ from $\theta$, and then samples the word from the topic according to its multinomial distribution $\phi_z$.

a higher-level process. Therefore, it leaves some open questions such as how to generate a new document that is not in the training set.

## 5.2 Latent Dirichlet allocation

Latent Dirichlet allocation (LDA) (Blei et al. (2003)) takes a further step to model document generation. It is a widely-used topic model, often applied to textual data, and the basis for many variants. LDA represents each document as a mixture of topics, where each topic is a multinomial distribution over words in a vocabulary. The generative process in LDA is similar to pLSI, except that the per-document multinomial distributions over topics are sampled from a Dirichlet distribution. The corresponding graphical model is shown in Figure 8. By introducing the additional Dirichlet distribution, LDA not only reduces the number of parameters in the model, but also addresses the problem to generate documents outside the training set.

The topics discovered by LDA capture correlations among words, but LDA does not explicitly model correlations among topics. This limitation arises because the topic proportions in each document are sampled from a single Dirichlet distribution. As a result, LDA has difficulty modeling data in which some topics co-occur more frequently than others. However, topic correlations are common in real-world text data, and ignoring these correlations limits LDA's ability to predict new data with high likelihood. Ignoring topic correlations also hampers LDA's ability to discover a large number of fine-grained, tightly-coherent topics. Because LDA can combine arbitrary sets of topics, it is reluctant to form highly specific topics, for which some combinations would be "nonsensical".

It is easy to see that LDA can be viewed as a special case of PAM: the DAG corresponding to LDA is a three-level hierarchy consisting of one root at the top, a set of topics in the middle and a word vocabulary at the bottom. The root is fully connected to all the topics, and each topic is fully connected to all the words. The model structure is shown in Figure 1(a). Each topic in LDA has a multinomial distribution over words, and the root has a Dirichlet compound multinomial distribution over topics.

## 5.3 Correlated Topic Model

An alternative model that not only discovers topics from data, but also learns their correlations, is the correlated topic model (CTM) (Blei and Lafferty (2006)). It is similar to LDA, except that rather than drawing topic mixture proportions from a Dirichlet, it does so from a logistic normal distribution, whose parameters include a covariance matrix in which each entry specifies the correlation between a pair of topics. Therefore topics in CTM are not independent from each other. The corresponding graphical model is shown in Figure 9. In a comparison against LDA using a collection of *Science* articles, CTM demonstrates better performance on log-likelihood of held-out test data and also supports larger numbers of topics.

Pairwise covariance matrix is one way to represent topic correlations. Another possibility is to use mixture models. The model structure of CTM can be described by a special case of PAM, as shown in Figure 1(b). The nodes that are directly connected to words correspond to CTM topics, and for each pair of them, there is one additional node that captures their correlation. One advantage of a mixture model is that it can have fewer parameters. Consider a simple example shown in Figure 10. There are 7 topics $\{A, B, C, D, E, F, G\}$, where $A$ through $E$ are correlated and $C$ through $G$ are also correlated. We can describe this kind of correlation with two different representations. The one on the left is the covariance matrix, where the color of each entry specifies the degree of correlation between a pair of topics. The one on the right is a mixture model with two clusters. The solid lines correspond to strong correlations and dashed lines correspond to weak correlations. In this example, we need 21 parameters in the covariance matrix while we only need 14 parameters for the mixture model. The advantage is especially obvious when we use a large number of topics because the number of parameters in the covariance matrix grows as the square of the number of topics.

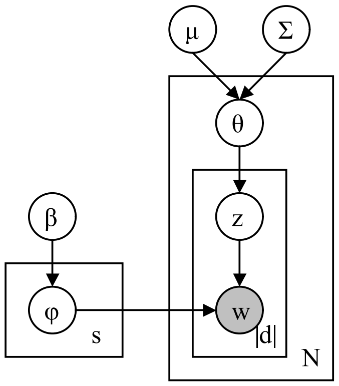

Figure 9: The graphical model for CTM. Instead of a Dirichlet, CTM uses a logistic normal distribution parameterized by mean $\mu$ and covariance matrix $\Sigma$ to sample the multinomial distribution over topics in every document.

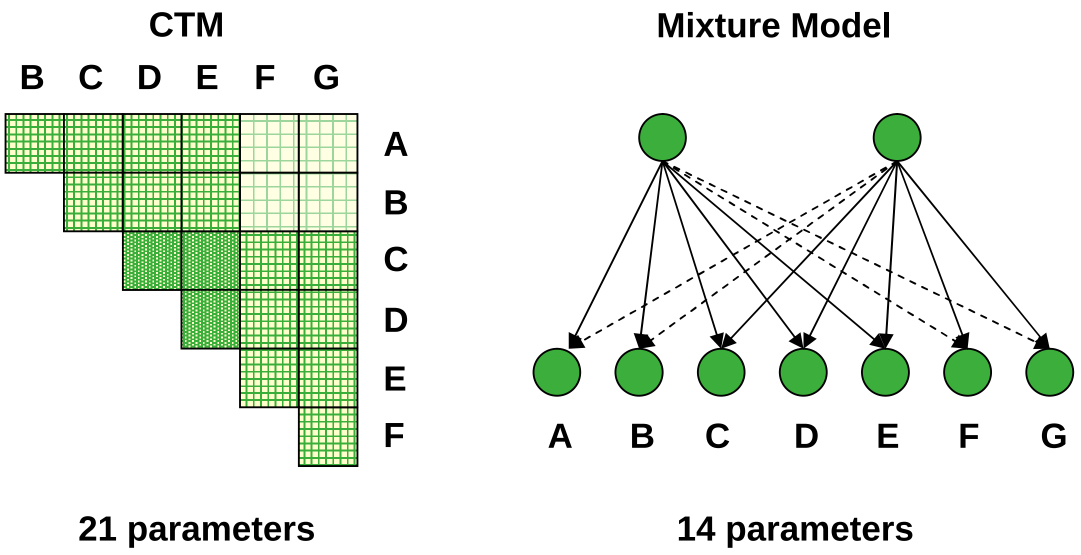

Figure 10: An example of topic correlations, which can be represented by both a symmetric covariance matrix (on the left) and a mixture model (on the right). One of the advantages of a mixture model is that it may include fewer parameters.

## 5.4 Hierarchical Dirichlet Processes

One important issue for mixture models is choosing an appropriate number of mixture components. Model selection methods such as cross-validation and Bayesian model testing are usually inefficient. A nonparametric solution with the Dirichlet process (DP) (Ferguson (1973)) is more desirable because it does not require specifying the number of mixture components in advance. Dirichlet process mixture models have been widely studied in many problems (Kim et al. (2006); Daume-III and Marcu (2005); Xing et al. (2004); Sudderth et al. (2005)).

In order to solve problems where a set of mixture models share the same mixture components, Teh et al. propose the hierarchical Dirichlet process (HDP) (Teh et al. (2005)). It is intended to model data that is pre-organized into nested groups. Each group is associated with a Dirichlet process, whose base measure is sampled from a higher-level Dirichlet process. Unlike PAM, HDP does not automatically discover topic correlations from unstructured data. One example of using this model is to learn the number of topics in LDA, in which each document is associated with a Dirichlet process.

### 5.5 Hierarchical LDA

Another closely related model that also represents and learns topic correlations is hierarchical LDA (hLDA) (Blei et al. (2004)). It is a variation of LDA that assumes a hierarchical structure among topics. Each topic has a distribution over words and can be reached by a unique path from the root. Topics at higher levels are more general, such as stopwords, while the more specific words are organized into topics at lower levels. To generate a document, hLDA first samples a leaf in the hierarchy. Then for each word in the document, it samples a node on the path from the leaf to the root, and this node generates the word. Thus hLDA can well explain a document that discusses a mixture of *computer science*, *artificial intelligence* and *robotics*. However, for example, the document cannot cover both *robotics* and *natural language processing* under the more general topic *artificial intelligence.* This is because a document is sampled from only one topic path in the hierarchy.

Compared to hLDA, PAM provides more flexibility for document generation because it samples a topic path for each word instead of each document. Note that it is possible to create a DAG structure in PAM that would capture hierarchically nested word distributions and obtain the advantages of both models.

## 6. Conclusion and Future Work

In this paper, we have presented pachinko allocation, a mixture model that uses a DAG structure to capture arbitrary topic correlations. Each leaf in the DAG is associated with a word in the vocabulary and each interior node corresponds to a topic that models the correlation among its children, where topics can be not only parents of words, but also other topics. The DAG structure is completely general, and some topic models like LDA can be represented as special cases of PAM. Compared to other approaches that capture topic correlations such as hierarchical LDA and correlated topic model, PAM provides more expressive power to support complicated topic structures and adopts more realistic assumptions for document generation.

One of our quantitative evaluation metrics is the likelihood of held-out test data. There is no closed-form solution to this problem for topic models like LDA and PAM. Therefore we propose an estimation technique based on empirical likelihood. After the model is trained, we unconditionally generate document samples. Unlike previous work that has used a harmonic mean method, these samples are not restricted to the topic co-occurrences observed in the held-out data. With this technique, PAM is compared against other related topic models and demonstrates significant improvement.

Complexity has been an obstacle for us to apply PAM to very large datasets. While we assume a fully-connected structure for the four-level PAM, many connections are indeed very sparse. By capturing such sparsity, we are able to dramatically reduce the sample space for Gibbs sampling. We describe several techniques to develop a scalable approximation. It allows us to apply LDA to information retrieval with improved efficiency and use PAM to discover topic correlations in large collections.

The four-level DAG structure is only a simple example of pachinko allocation. This model offers far more flexibility to describe topic correlations. One direction of our future work is to explore more complicated DAG structures. The first step would be introducing edges that skip layers in arbitrary ways. For large text collections, we are also interested in using more layers of topics.

Since topic models usually adopt a bag-of-words representation for the documents, they capture long-term dependencies within one document but ignore local dependencies between nearby words. Previous work has studied various ways to combine these two types of dependencies. For example, the HMM-LDA model (Griffiths et al. (2005)) integrates hidden Markov model (HMM) with LDA by designating one special state to generate topic words only. The non-topic states operate the same way as ordinary HMM states, thus capturing linear chain dependencies in word sequences. Another approach with a similar goal uses a hierarchical Bayesian framework to combine bigram language models and topic models (Wallach (2006)). In the future, we plan to study possible ways to incorporate $n$-gram statistics into PAM.

Topic models can be helpful for a wide range of applications including social network analysis, data mining and semi-supervised learning. We believe that with the great expressive power, PAM is a promising new technique for such tasks.

## Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010, and under contract number HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily re ect those of the sponsor. We also thank Charles Sutton and Matthew Beal for helpful discussions, David Blei and Yee Whye Teh for advice about a Dirichlet process version, Sam Roweis for discussions about alternate structure-learning methods, and Michael Jordan for help naming the model.

## References

L. Azzopardi, M. Girolami, and C. van Rijsbergen. Topic based language models for ad hoc information retrieval. In *International Joint Conference on Neural Networks*, 2004.

D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems 16*. 2004.

D. Blei and J. Lafferty. Correlated topic models. In *Advances in Neural Information Processing Systems 18*. 2006.

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

G. Casella and R. Berger. *Statistical Inference*. Duxbury Press, 2001.

S. Chib. Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, 1995.

H. Daume-III and D. Marcu. A Bayesian model for supervised clustering with the Dirichlet process prior. *Journal of Machine Learning Research 6*, pages 1551–1577, 2005.

S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman. Indexing by latent semantic analysis. *American Society of Information Science, 41(6)*, pages 391–407, 1990.

P. Diggle and R. Gratton. Monte Carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society*, 1984.

T. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics, 1(2)*, pages 209–230, 1973.

D. Gildea and T. Hofmann. Topic-based language models using EM. In *6th European Conference on Speech Communication and Technology*, 1999.

T. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl. 1):5228–5235, 2004.

T. Griffiths, M. Steyvers, D. Blei, and J. Tenenbaum. Integrating topics and syntax. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 537–544. MIT Press, Cambridge, MA, 2005.

T. Hofmann. Probabilistic latent semantic analysis. In *Uncertainty in Artificial Intelligence, UAI'99*, 1999.

S. Kim, M. Tadesse, and M. Vannucci. Variable selection in clustering via Dirichlet process mixture models. *Biometrika 93, 4*, pages 877–893, 2006.

S.-B. Kim, H.-C. Rim, and J.-D. Kim. Topic document model approach for naive Bayes text classification. *IEICE - Trans. Inf. Syst.*, E88-D(5):1091–1094, 2005.

X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *SIGIR '04*, pages 186–193, 2004.

A. McCallum, A. Corrada-Emanuel, and X. Wang. Topic and role discovery in social networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

M. Newton and A. Raftery. Approximate Bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society*, 1994.

M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *AUAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494, 2004.

J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. Technical report, MIT Technical Report MIT-CSAIL-TR-2005-012, 2005.

E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Describing visual scenes using transformed Dirichlet processes. In *Advances in Neural Information Processing Systems 17*, 2005.

Y. Tam and T. Schultz. Dynamic language model adaptation using variational bayes inference. In *INTERSPEECH*, pages 5–8, 2005.

Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 2005.

V. Tuulos and H. Tirri. Combining topic models and social networks for chat data mining. In *2004 IEEE/WIC/ACM International Conference on Web Intelligence*, page 206C213, 2004.

H. Wallach. Topic modeling: Beyond bag-of-words. In *International Conference on Machine Learning (ICML)*, 2006.

X. Wei and W. B. Croft. LDA-based document models for ad-hoc retrieval. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, 2006.

E. Xing, R. Sharan, and M. Jordan. Bayesian haplotype inference via the Dirichlet process. In *International Conference on Machine Learning (ICML)*, 2004.

C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01*, pages 334–342, 2001.

B. Zheng, D. C. McLean, and X. Lu. Identifying biological concepts from a protein-related corpus with a probabilistic topic model. *BMC Bioinformatics*, 7:58, 2006.