The Dissertation Committee for Austin Severn Waters
certifies that this is the approved version of the following dissertation:

Infinite-Word Topic Models for Digital Media

Committee:

Risto Miikkulainen, Supervisor

Joydeep Ghosh

Kristen Grauman

Peter Mueller

Pradeep Ravikumar

Infinite-Word Topic Models for Digital Media

by

Austin Severn Waters, M.S.C.S.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

May 2014

Contents

Chapter 1

Introduction

Over the past decade, rapid technological advances have lead to a surge in the amount of digital media produced across the world. Many consumer electronic devices today, including computers, cellular phones, and mobile devices, come equipped with built-in cameras and microphones that can readily capture high-quality digital photographs, videos, and audio recordings. Not only are these devices ubiquitous, they are far less costly to operate than their traditional, analog equivalents. Due to the low cost of digital storage, the cost of taking pictures on one's smart phone, for example, is near zero. In effect, modern technology provides us with both the ability and opportunity to take a seemingly-limitless stream of photos, videos, and other recordings. While past generations collected shoeboxes of old photos and home moves in closets and attics, the present generation accumulates its vast media collections on hard drives, cloud storage accounts, and social media networks. A recent estimate puts the number of online photos in the tens of billions [48, 1]. On YouTube, approximately one hour of new video is uploaded every *second* [2].

Such digital media collections hold an unprecedented source of knowledge and data about the world. Yet, even at current scales the data exceeds by many orders of magnitude the amount a single user could browse through in an entire lifetime.

Making use of such datasets requires computational tools that can index, search over, and organize media documents in ways that are meaningful to human users, based on the meaning of their content. Such systems do not exist yet, but someday might – they would be much faster than humans but nonetheless understand images and sounds in terms of human semantic concepts. Such systems could look through huge photo libraries to find photos to accompany news articles, listen to years of music to answer queries like *singers who sound like Tom Waits*, automatically discover events going on in a city by analyzing the images recently posted on social media, and any number of other tasks. This dissertation aims to make them a step closer to reality. It does so by introducing a new probabilistic *topic model* to automatically infer semantic descriptions of media documents, as well as efficient methods for training it on large, growing datasets.

1.1 Motivation

Currently, many systems for searching and browsing collections of digital media rely on the use of tags or other text annotations of content. These annotations are easily indexed to allow users to find documents quickly by making text-based searches. However, tagging must be done manually by human users, and can be tedious and time-consuming. Indeed, users tend to tag very sparsely or are unwilling to put forth the effort to tag at all: an analysis of 52 million images from Flickr found that 64% of photos had 3 or fewer tags [49]. A further problem is that many tags describe the circumstances under which a photo was taken (e.g. *2008*, *wedding*), rather than the image content itself. Clearly, text annotations offer at best a limited description of document content. Rather than relying on human annotations, it would be preferable to have intelligent systems that can understand a document's content directly.

The difficulty of designing such systems lies in bridging the so-called *se-*

mantic gap – that is, connecting a machine's low-level document representation to high-level semantic concepts. How might a machine represent the visual concept of *mountains*, for example, in terms of the pixel values of photographs? Individual pixels contain almost no information about the concept; rather, "mountainness" must be conferred through visual abstractions above the pixel level. An additional challenge is that there are many possible photographs of mountains, of different ranges, taken in different weather, with different perspectives or lighting conditions. A useful representation of *mountains* must account for such natural variability. Additionally, *mountains* is just one visual concept. How can systems be designed that understand many such concepts without a prohibitive amount of human engineering effort? Finally, images are only one type of digital media. Can similar systems be designed to understand concepts in sound and video?

This dissertation develops an approach to automated content analysis based on *topic models*. Topic models are a general class of Bayesian hierarchical models that automatically uncover latent structure in document collections. Originally developed for text analysis, they decompose the content of each document in terms of latent *topics*, or groups of words that frequently occur together. As their name suggests, topics tend to be semantically coherent. When topic models are trained on a set of news articles, for example, topics typically group together words related to government (*tax*, *program*, *Congress*, etc.), arts (*film*, *music*, *movie*, etc.), and education (*school*, *student*, *teacher*, etc.) [14]. Topic models express the content of each document as a weighted mixture over such topics. These *topic weights* serve as a quantitative representation of the document's semantics. They have been used successfully as features for document classification, collaborative filtering, exploratory analysis, and other tasks [14, 43].

Building on this prior work, this dissertation applies topic models to the automated analysis of content in non-text domains. Its primary contribution, the

Infinite-Word Topic Model (IWTM), helps extend topic modeling to such domains by removing model assumptions that do not make sense for them – in particular, the assumption that documents are composed of discrete, mutually-exclusive words from a fixed-size vocabulary. Among other benefits, IWTM achieves better performance than existing text-based models while simplifying the modeling process considerably. This dissertation also contributes fast, scalable variational inference [29] methods for IWTM, as well as tools for training efficiently on growing datasets. The focus of the dissertation is on the analysis of images; however, the model and methods it develops are also applicable to other types of digital media, such as videos and audio recordings. The next section describes existing methods for topic modeling on image collections.

1.2 Topic Models for Image Analysis

Even though topic models were originally developed for text documents, a number of studies have demonstrated their utility in image domains as well. In particular, topic models have been applied to object recognition [34, 50], segmentation [51, 63], and natural scene classification [36, 57, 22]. This dissertation mainly focuses on applications of the last type – natural scene classification.

One text-based topic model discussed throughout this dissertation is Latent Dirichlet Allocation (LDA) [14]. In order to use LDA on a collection of images, the images must first undergo preprocessing that converts them into a text-like *bag-of-words* (BOW) representation. Typically, this process is performed as follows. First, a set of local multivariate features is extracted from each image. Common choices are SIFT descriptors [39], densely-sampled Histograms of Oriented Gradients (HoG) [20], or local image patches. The features are then pooled across all images and used to train a clustering model, typically with the K-means algorithm. Finally, the image features are quantized against the clustering model, yielding a set of integer counts in $\mathbb{N}^K$ for each document. In this representation, the clusters of image features are termed *visual words*, the set of all K clusters comprises a *visual vocabulary*, and the parameter K is referred to as the vocabulary size.

The BOW representation allows text-based topic models like LDA to be used "out of the box" for image analysis tasks. However, in practice, BOW comes with a number of practical difficulties. One concern is how to select an optimal value for K, the visual vocabulary size. In practice, this quantity can have a significant impact on the subsequent performance of the topic model [22]. Unfortunately, because clustering is performed as an off-line preprocessing step, the vocabulary size cannot be learned as part of the topic model's training. Instead, optimizing the vocabulary size requires training and evaluating separate models with different values of K and picking the best according to some performance metric. Such a procedure is computationally expensive, since both a clustering model and topic model must be trained for each vocabulary size evaluated. Furthermore, when the dataset is large, the selection of the visual vocabulary size can be prohibitive.

Bayesian non-parametric clustering models, such as the Dirichlet Process Mixture Model (DPMM) [7], offer a potential solution. Rather than using a fixed model complexity K, DPMMs infer the number of cluster components based on the size and complexity of the data. In theory, bags of words could be formed by quantizing image features against a DPMM, rather than K-means. However, this approach, too, has undesirable properties. First, since the image representations going into the topic model are determined by an offline clustering step, any updates to the clustering require completely retraining the topic model. Second, although DPMMs define a full posterior distribution over each datum's cluster assignment, the quantization process assigns each datum entirely to its closest cluster. Doing so throws away information about the features that could be useful to the topic model.

A guiding principle of IWTM is to account for uncertainty in the topic modeling process that arises in non-text document domains. There are two such sources. First, media documents are not comprised of discrete, mutually-exclusive words; rather, they are groups of noisy, continuous, multivariate features. Second, it does not make sense to divide such features into a prescribed number of clusters. Instead, the visual vocabulary size should be inferred based on the data. IWTM accommodates the first issue by incorporating the clustering of features into the topic model itself. It addresses the second by using Bayesian nonparametrics to learn the visual vocabulary size along with the rest of the model.

1.3 Outline of the Dissertation

The next chapter discusses background relevant to the development of the Infinite-Word Topic Model and its inference procedures. It reviews a typical approach to topic modeling in image domains (Latent Dirichlet Allocation with bags of visual words) and provides background on Bayesian non-parametric clustering that motivates the design of IWTM.

Chapter 3 defines the Infinite-Word Topic Model, the primary contribution of this dissertation. It discusses the design of the model, motivated by limitations of existing text-based topic modeling approaches. Further, it compares IWTM to other models in machine learning and computer vision literature.

Chapters 4, 5, and 6 derive posterior inference methods for IWTM. Chapter 4 first develops a collapsed Gibbs sampler for the model. Using it, the advantages of IWTM over LDA are demonstrated on a natural scene classification task. Chapter 5 develops a variational inference procedure for IWTM that is far more efficient and scalable than the Gibbs sampler, as shown on the same dataset Chapter 6 scales IWTM even further by deriving a *stochastic variational inference* (SVI) procedure for it. Using SVI, IWTM is applied to a large-scale scene classification dataset with thousands of images and millions of image features.

Chapters 7 addresses a challenge presented by real-world applications – namely, how to handle the case where the datasets periodically grow, rather than being fixed for the lifetime of the model. This chapter introduces Incremental Variational Inference (IVI), a general method for training Bayesian non-parametric models incrementally with variational inference. Further, it develops an active learning method for topic models that allows new training data to be sampled intelligently. Such sampling allows making best use of the data, in order to learn faster and achieve higher final classification accuracy.

Chapters 8 and 9 consider the dissertation in hindsight. The former suggests directions for future work, and the latter summarizes the primary contributions of this dissertation: a new non-parametric topic model for digital media documents, efficient inference methods that scale it to large problems, and tools for efficiently training it and other models on growing datasets.

Chapter 2

Background

This chapter provides background material relevant to the development of the Infinite-Word Topic Model and its inference procedures. It reviews a typical approach to topic modeling in image domains (LDA with bags of visual words) and provides background on the Bayesian non-parametric clustering method that motivates the design of IWTM.

2.1 Images as Sets of Local Descriptors

A prerequisite for the automated analysis of image content is a scheme for extracting representations of visual appearance from images. A number of studies in computer vision have demonstrated that representations based on *local descriptors* are effective for many tasks, including object recognition, image classification, segmentation, and image retrieval [39, 22, 35, 23, 61, 63]. In this scheme, an image is divided into small patches, and the appearance of each patch is described by a vector of real numbers. Each image is then represented as a set of such multivariate descriptors.

A number of approaches, such as the scale-invariant feature transform (SIFT) [39] and histograms of oriented gradients (HOG) [20], describe patch-level appear-

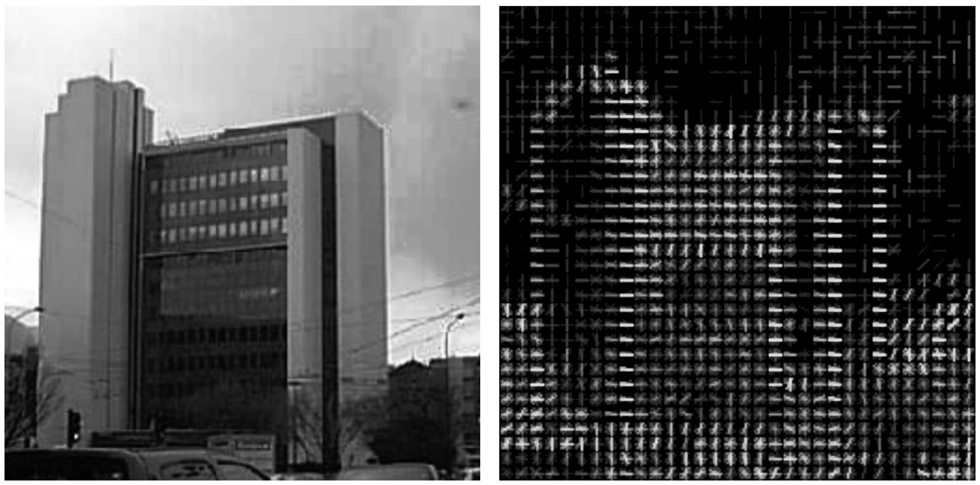

(a) Input image (b) Histogram of oriented gradients

Figure 2.1: Histograms of oriented gradients (HOG) representation of a sample image. Each 8×8 pixel patch of the image is represented as a histogram of the orientations of local illumination gradients. Such histograms describe local appearance, and are used as inputs to subsequent modeling.

ance in terms of local illumination gradients. Each patch is passed through filters that estimate the gradient magnitude in some number N regularly spaced directions, i.e. at angles $0, \frac{\pi}{N}, \ldots \frac{(N-1)}{N}\pi$. For each of C local cells that subdivide the patch, the gradients at each pixel are used to "vote" on the local dominant orientation. To form a patch-level descriptor, the statistics for the C cells are normalized and concatenated, yielding a real-valued descriptor of dimension $C \times N$. Altogether, the image is represented as a set of such descriptors, one for each patch. Figure 2.1 shows HOG descriptors computed over a regular grid of 8-by-8 pixel cells for a sample image.

Although both SIFT and HOG represent patches in the same manner, the approaches differ in how patches are selected. HOG simply uses a dense sampling of patches over a regular grid of the image (as in Figure 2.1), while SIFT uses a keypoint detector to extract patches of different sizes and orientations. Such a detector selects a sparse set of patches that can be identified in a manner invariant to changes in scale, rotation, and perspective. Such invariance provides robustness that can be beneficial in object recognition tasks [39]. However, this dissertation applies topic models to a different vision task – scene classification – in which sparse, invariant features are typically eschewed in favor of densely sampled ones like HOG. For the remainder of this dissertation, the set of local features for the dth image is denoted $\mathbf{x}_d = \{\mathbf{x}_{d1}, \ldots \mathbf{x}_{d,N_d}\}$, where each $\mathbf{x}_{di} \in \mathbb{R}^{N_{\text{dim}}}$ represents the appearance of a single patch, and N_d is the total number of local features (or, equivalently the number of local patches represented).

2.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) [14] is an unsupervised probabilistic model that was originally developed to analyze collections of *text* documents, where each document is represented as a discrete bag of words from a fixed vocabulary. To apply the model to non-text domains, the documents must be converted to conform to this representation.

2.2.1 Bag of Visual Words

The bag of visual words representation of a image corpus is formed by extracting a set of local features from each image and quantizing them into a text-like representation. First, a set of N_{dim}-dimensional local features $\mathbf{x}_d = \{\mathbf{x}_{d1}, \mathbf{x}_{d2}, \ldots \mathbf{x}_{dN_d}\}$ is extracted from each image $d \in 1 \ldots N_{\text{doc}}$ in the corpus. Second, the features from all images are pooled and used to train a clustering model with K components. Third, the clustering model is used to quantize the image features by mapping each feature $\mathbf{x}_{di}$ to the index of its nearest cluster. For each document d, quantization yields a set of visual word indicators $\boldsymbol{w}_d = \{w_{d1}, w_{d2}, \ldots, w_{dN_d}\}$, with each w_{di} an integer in $\{1, 2, \ldots K\}$.

2.2.2 Model Definition

LDA models the corpus using T topics $\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_T$, with each document maintaining a separate distribution $\boldsymbol{\theta}_d$ that describes the contribution of each topic. LDA assumes that each document is generated through the following procedure:

1. For each document d, draw a set of topic weights $\boldsymbol{\theta}_d$.
2. For each word i in document d,
   (a) Draw a topic $z_{d,i}$ by sampling from $\boldsymbol{\theta}_d$
   (b) Draw the word $w_{d,i}$ from the chosen topic $\boldsymbol{\phi}_{z_{d,i}}$

The full generative model is

$$\begin{aligned}
\boldsymbol{\phi}_t|\boldsymbol{\beta} &\sim \text{Dir}(\boldsymbol{\beta}), && t \in 1 \ldots T, && \text{(topics)} \\
\boldsymbol{\theta}_d|\boldsymbol{\alpha} &\sim \text{Dir}(\boldsymbol{\alpha}), && d \in 1 \ldots D, && \text{(topic weights)} \\
z_{d,i}|\boldsymbol{\theta}_d &\sim \boldsymbol{\theta}_d, && i \in 1 \ldots n_d, && \text{(topic indicators)} \\
c_{d,i}|\boldsymbol{\phi}_{z_{d,i}} &\sim \boldsymbol{\phi}_{z_{d,i}}, && i \in 1 \ldots n_d, && \text{(``words'')},
\end{aligned}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are hyperparameters smoothing the per-document weights and per-topic word distributions, respectively.

LDA can be thought of as a latent-factor model that attempts to explain the N_{doc} observed documents in terms of T topics, which are latent distributions over words. Because generally $T \ll N_{\text{doc}}$, the model is pressured to infer topics that place mass on groups of commonly co-occurring – and thus semantically related – words. In image analysis tasks, the inferred topics capture groups of related visual features. For example, in Fei-Fei and Perona's analysis of a corpus of natural scene images [22], the topics appear to group together patches containing the trunks and leaves of trees, different parts of the roofs of houses, and so on. The contribution of the topics to each document is described by the per-document topic weights

$\boldsymbol{\theta}_d$. Throughout this dissertation, the topic weights are used as a low-dimensional semantic representations of the documents.

Although LDA can learn semantic descriptions of documents that are *useful* for supervised learning tasks, LDA itself is an unsupervised model. Some additional mechanism must be used to, for example, predict documents' class labels. This dissertation explores two options for applying topic models to document classification tasks. Chapter 4 turns the models into *generative classifiers* by adding an extra layer for document labels to their graphical models. In chapters 5 and 6, the topic models are instead trained in an unsupervised way, and the document representations learned by them are fed into external support vector machine classifiers.

2.2.3 Issues

Although LDA has been used successfully for image classification and other tasks, representing images as bags of words has a number of shortcomings. Most importantly, the vocabulary size is not a parameter of the model itself, but rather one chosen in preprocessing that determines the image representations given to LDA. As a result, there is no way to tune this parameter during inference to maximize the likelihood of the observed data or the classification accuracy of the overall system. Instead, the best vocabulary size must be determined by empirically by evaluating LDA with many different vocabulary sizes. This process is computationally expensive, since it involves training separate K-means and LDA models in each case.

In addition to the issue of the vocabulary size, bag of words makes it difficult to train LDA effectively in an incremental manner. Because the image representations used by LDA are determined by an offline clustering method, an update to the clustering requires completely retraining the topic model. When new training data becomes available, one is forced to choose between keeping the visual vocabulary static and updating LDA, or updating the visual vocabulary and completely retraining LDA. Both of these choices are undesirable.

IWTM overcomes these issues by incorporating the clustering of document features *into the probabilistic model itself*, rather than clustering in a separate preprocessing step. As will be discussed in chapter 3, IWTM's clustering layer is a type of Gaussian Dirichlet Process Mixture Model (GDPMM). The following section motivates the use of GDPMM by drawing connections between it and the K-means algorithm used to form the bag-of-words representations for LDA. In particular, the GDPMM is shown to be a Bayesian non-parametric extension of the well-known mixture of Gaussians model, which in turn can be viewed as a fully probabilistic, soft-clustering version of K-means clustering.

2.3 Clustering models

In general, clustering attempts to partition a set of data $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^{N_{\text{dim}}}$ into a number of groups or clusters, where the data in each cluster are close together according to some distance measure. Given the data, the goal of clustering is to learn the parameters of the components and determine which data points belong to which clusters. This section traces the connections between three increasingly powerful clustering methods: K-means, Gaussian mixtures, and Gaussian Dirichlet process mixtures.

2.3.1 K-means

K-means seeks a partition of the data into exactly K clusters, where K is a fixed parameter of the algorithm. The goal of the algorithm is to determine the centers $\{\boldsymbol{\mu}_1 \ldots \boldsymbol{\mu}_K\}$ of the clusters as well as assignments $\{\mathbf{c}_1 \ldots \mathbf{c}_n\}$ so that the data in each cluster are close to the center in terms of Euclidean distance. More formally,

K-means seeks the values of these parameters that minimize the objective function

$$J = \sum_{i=1}^{n} \sum_{k=1}^{K} c_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2, \tag{2.1}$$

where $c_{i:}$ is a 1-of-K binary indicator vector that represents the cluster to which datum $\mathbf{x}_i$ belongs, i.e. $c_{ik} = 1$ if and only if $\mathbf{x}_i$ is a member of cluster k. K-means is said to be a *hard-clustering* algorithm because each data point belongs to exactly one cluster.

A global optimum of equation (2.1) is difficult to find because the parameters interact and the function is non-convex. Finding an optimal solution with respect to both $\mathbf{c}$ and $\boldsymbol{\mu}$ jointly is known to be NP-Hard [3]. K-means instead finds a local optimum by optimizing equation (2.1) iteratively with respect to the cluster indicators and centers individually. Given some initial values of the centers μ, the algorithm proceeds iteratively in two steps. First, the cluster indicators are updated by allocating each data point to its closest cluster in (squared) Euclidean distance:

$$c_i \leftarrow \arg\min_k \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2. \tag{2.2}$$

Second, the cluster centers are set to the mean of the data points assigned to it:

$$\boldsymbol{\mu}_k = \frac{1}{\#\{i : c_i = k\}} \sum_{i:c_i=k} \mathbf{x}_i. \tag{2.3}$$

Update (2.3) can be derived by differentiating the objective, setting the derivatives to zero, and solving for the parameter $\boldsymbol{\mu}_k$. Typically the algorithm is run until convergence (no cluster assignments are changed during the course of an iteration), until the change in the objective dips below a predefined threshold, or until some maximum number of iterations is exceeded. Given a sufficient number of iterations, K-means is guaranteed to converge because each step decreases or maintains the value of its objective [10].

Note that K-means is posed as an optimization problem; strictly speaking, it does not provide a *probabilistic* interpretation of the data. This presents a number of difficulties. First, it is unclear how many clusters K should be used to describe the data, or even what it objectively means to have a "correct" number of clusters. By contrast, probabilistic clustering methods come with principled metrics for choosing the model complexity, such as the likelihood of held-out data. Second, because K-means is not a generative model, it is not possible to combine it with or embed it inside of probabilistic models in order to capture more complex structure in the data. Thus, when used in conjunction with LDA, K-means must always be run as an offline preprocessing step.

2.3.2 Gaussian Mixtures

Although K-means itself is not a probabilistic model, it has a strong connection to the well-known Gaussian Mixture Model (GMM), which is a probabilistic clustering model. The GMM assumes each datum $\mathbf{x}_i \in \mathbb{R}^{N_{\text{dim}}}$ is generated by first selecting a cluster component $c_i \in \{1, \dots, K\}$ and then drawing $\mathbf{x}_i$ from its multivariate Gaussian distribution

$$c_i \sim \text{Discrete}(\boldsymbol{\pi}) \tag{2.4}$$

$$\mathbf{x}_i \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}, c_i \sim \mathcal{N}(\mathbf{x}_i \mid \boldsymbol{\mu}_{c_i}, \boldsymbol{\Sigma}_{c_i}), \tag{2.5}$$

where $\boldsymbol{\pi} = \{\pi_1, \dots, \pi_K\}$, $\sum_i \pi_i = 1$ defines a global set of weights with which each cluster is selected, and mean vectors $\boldsymbol{\mu}_k \in \mathbb{R}^{N_{\text{dim}}}$ and positive-definite covariance matrices $\boldsymbol{\Sigma}_k \in \mathbb{R}^{N_{\text{dim}} \times N_{\text{dim}}}$ parameterize the Gaussian components. As in K-means, the GMM assumes that there are a fixed number of clusters K in the data.

The goal of inference in GMMs is similar to that of K-means: given observed data $\mathbf{x}$, it is of interest to estimate the assignments c_i and Gaussian parameters $\boldsymbol{\mu}_k$

and $\mathbf{\Sigma}_k$ that explain the data. Whereas K-means makes hard assignments of data to clusters, GMMs define probabilistic or "soft" cluster assignments. Given settings of the cluster parameters, the posterior probability that datum $\mathbf{x}_i$ was generated from component k is given by Bayes' rule:

$$p(c_i = k \mid \mathbf{x}_i) = \frac{\pi_k \mathcal{N}(\mathbf{x}_i \mid \boldsymbol{\mu}_k, \mathbf{\Sigma}_k)}{\sum_{k'} \pi_{k'} \mathcal{N}(\mathbf{x}_i \mid \boldsymbol{\mu}_{k'}, \mathbf{\Sigma}_{k'})}. \tag{2.6}$$

The full posterior over cluster assignments defined by the GMM is always *at least as* informative as the hard assignments given to the data by K-means. If a datum lies between clusters or the clusters are overlapping, the posterior will split mass among many clusters, reflecting the ambiguity of the assignment. If the clusters are well separated, the posterior will concentrate mass at $\mathbf{x}_i$'s closest cluster, resembling a hard assignment. A hard cluster assignment can be thought of as a compressed version of the posterior: The former contains 0 bits of uncertainty about the cluster assignment, while the latter contains up to $\log_2 K$ bits about it.

The connection between the methods can also been seen in the procedures used to train them. Indeed, the K-means algorithm can be shown to be a degenerate version of the Expectation Maximization (EM) algorithm commonly used to train Gaussian mixtures [10]. The EM algorithm for GMMs is an iterative procedure that makes maximum-likelihood estimates of the parameters $\boldsymbol{\mu}$, $\mathbf{\Sigma}$, and $\boldsymbol{\pi}$. At each iteration, new values for the parameters are estimated by maximizing the expected complete data log likelihood

$$\mathbb{E}[\log p(X, \mathbf{c} \mid \boldsymbol{\mu}, \mathbf{\Sigma}, \boldsymbol{\pi})] = \sum_{i=1}^{N} \sum_{k=1}^{K} \tilde{p}(c_i = k) \left(\log \pi_k + \log \mathcal{N}(\mathbf{x}_i \mid \boldsymbol{\mu}_k, \mathbf{\Sigma}_k) \right), \tag{2.7}$$

where $\tilde{p}(c_i = k)$ is the posterior cluster assignment evaluated under the parameter settings of the *previous* iteration. The optimal updates to the parameters are:

$$\boldsymbol{\mu}_k \leftarrow \frac{1}{\sum_i \tilde{p}(c_i = k)} \sum_i \tilde{p}(c_i = k)\mathbf{x}_i \tag{2.8}$$

$$\boldsymbol{\Sigma}_k \leftarrow \frac{1}{\sum_i \tilde{p}(c_i = k)} \sum_i \tilde{p}(c_i = k)(\mathbf{x}_i - \boldsymbol{\mu}_k)^T(\mathbf{x}_i - \boldsymbol{\mu}_k). \tag{2.9}$$

K-means can be recovered from the EM procedure by making two simplifications to it. First, the clusters are assumed to have the same spherical covariance, i.e. $\boldsymbol{\Sigma}_k = \lambda\mathbb{I}$. By doing so, the Gaussian density reduces as

$$\mathcal{N}(\mathbf{x}_i \,|\, c_i = k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{|\boldsymbol{\Sigma}_k|^{-1/2}}{(2\pi)^{M/2}} \exp\left(-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_k)\right) \tag{2.10}$$

$$= \frac{1}{(2\pi\lambda)^{M/2}} \exp\left(-\frac{1}{2\lambda}\|\mathbf{x} - \boldsymbol{\mu}_k\|^2\right). \tag{2.11}$$

The second simplification is that the data are given hard cluster assignments, rather than probabilistic ones. Hard assignments can be achieved by taking the limit of the GMM as $\lambda \to 0$. As the cluster covariance shrinks, the posterior assignments become concentrated at the cluster closest in Euclidean distance to each datum, i.e.

$$p(c_i \,|\, \mathbf{x}_i) = \delta(c_i^*) \text{ where } c_i^* = \arg\min_k \|\mathbf{x}_i - \mu_k\|^2. \tag{2.12}$$

In the limit, the EM objective (2.7) reduces to

$$\mathbb{E}_c \left[\log p(\mathbf{X}, \mathbf{c} \,|\, \boldsymbol{\mu}, \lambda, \boldsymbol{\pi})\right] = -\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{I}[c_i = k]\|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 + C, \tag{2.13}$$

with C a constant [10]. The quantity (2.13) is equivalent to the K-means objective (2.1) with opposite sign. Thus, maximizing it with respect to the parameters is equivalent to minimizing the K-means objective.

2.3.3 Gaussian Dirichlet Process Mixtures

The previous section showed that the K-means algorithm can be viewed as a hard-clustering approximation of a GMM with restricted covariance structure. GMMs have several advantages over K-means: They provide probabilistic description of the data that is at least as informative as hard cluster assignments and, as proper probabilistic models, they can be embedded in other, more complex models. However, both methods still require that the number of clusters K be specified *a priori*, even though in general the optimal number of components is not known. Picking an appropriate value for K is important for the model to generalize well to unseen data. In general, using too few clusters will underfit the data, and using too many will overfit it.

The machine learning literature provides a number of methods for choosing the complexity of probabilistic models robustly – for example, measuring goodness-of-fit using held-out data or the Bayesian information criterion [45]. However, these methods only provide metrics to *compare* models of different complexities; generally, selecting the optimal complexity requires evaluating a number of models and choosing the best among them.

Bayesian non-parametric clustering methods, such as the Dirichlet Process Mixture Model (DPMM) [7], offer an appealing solution to this model-selection problem. Rather than using a fixed number of components K, DPMMs treat the number of clusters as a random quantity to be inferred based on the data. The DPMM actually supports an infinite number of clusters *a priori*; however, conditioned on a finite amount of observed data, the model concentrates mass on a small, finite number of clusters. The Gaussian DPMM (that is, the DPMM with Gaussian likelihood) is closely related to the Gaussian Mixture Model discussed in the previous subsection. In fact, the Gaussian DPMM (GDPMM) can be derived as a particularly type of GMM whose number of clusters K is taken to infinity.

The GDPMM is derived by making two modifications to the GMM. First, the GMM is made into a fully Bayesian model by putting priors on its parameters. The weights $\boldsymbol{\pi}$ and cluster parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are given priors as follows:

$$\begin{aligned}
\boldsymbol{\pi} \mid \alpha_0 &\sim \mathrm{Dir}(\gamma/K, \ldots, \gamma/K) \\
(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \mid H &\sim H \\
c_i \mid \boldsymbol{\pi} &\sim \mathrm{Discrete}(\boldsymbol{\pi}) \\
\mathbf{x}_i \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}, c_i &\sim \mathcal{N}(\boldsymbol{\mu}_{c_i}, \boldsymbol{\Sigma}_{c_i}).
\end{aligned}$$

Here, the cluster parameters come from a prior distribution H, and cluster weights $\boldsymbol{\pi}$ are drawn from a symmetric K-dimensional Dirichlet distribution with parameter γ/K. Note that as the number of clusters K increases, so does the sparsity of the Dirichlet prior on the cluster weights $\boldsymbol{\pi}$. The overall result is that the effective number of clusters used by the model remains small, even for large K, and grows instead as a function of the number of data points N and the parameter γ. In the limit $K \to \infty$, the Dirichlet prior becomes the *Dirichlet process.*

Using Dirichlet Process notation, the same mixture model (now with infinite capacity) is written as

$$\begin{aligned}
\boldsymbol{\phi} \mid \gamma &\sim \mathrm{DP}(\gamma \mathrm{H}) \\
\Omega_i \mid \phi &\sim \boldsymbol{\phi} \\
\mathbf{x}_i \mid \Omega_i &\sim \mathcal{N}(\Omega_i).
\end{aligned}$$

The latent variable $\boldsymbol{\phi}$, which is a sample from the Dirichlet Process, is itself a distribution. It can be expressed as a mixture over an infinite set of cluster parameters drawn from H; that is, $\boldsymbol{\phi} = \sum_{k=1}^{\infty} \phi_k \delta\left(\boldsymbol{\Psi}_k\right)$ with $\boldsymbol{\Psi}_k \sim$ H. The concentration parameter γ controls the prior over the component weights ϕ_k. Each datum in the mixture model is generated by selecting cluster parameters Ω_i from $\boldsymbol{\phi}$, and then drawing $\mathbf{x}_i$ from the corresponding Gaussian component.

It is important to note that the cluster parameters Ω_i, which come from the set $\{\boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2, \ldots\}$, will generally not be unique to a single datum. Indeed, a key property of the Dirichlet process is that a sequence of draws from it exhibits grouping. That is, for some k there will be many data indices S such that $\Omega_i = \boldsymbol{\Psi}_k$ for all $i \in S$. In other words, the DP prior prefers to use a number of clusters that is proportional to γ and scales slowly with the size of the dataset. The clustering properties of the Dirichlet process are made clearer in the next section, where the DP is given a more precise definition.

2.4 Dirichlet Processes

The previous section defined the Dirichlet process in terms of a K-dimensional Dirichlet *distribution* taken to the limit of $K \to \infty$. This mathematical definition is of little use in practice because, computationally, infinite-dimensional distributions cannot be represented explicitly. To perform posterior inference with Dirichlet processes, it is necessary to turn to other representations of them. This section details two different representations of the DP – the Chinese Restaurant Process (CRP) and the stick-breaking construction. Although they are equivalent, each describes the DP in a different way and lends itself to a different type of posterior inference method. Compared to the previous section's definition, these representations also make the clustering properties of the DP more explicit.

2.4.1 Chinese Restaurant Process

The CRP defines the Dirichlet Process indirectly in terms of a set of conditional draws from it. As in section 2.3.3, let $\boldsymbol{\phi} \sim \mathrm{DP}(\gamma H)$ be a Dirichlet process and let $\{\Omega_i\}_{i=1}^N$ denote draws from it. Consider the case where the Ω_i are drawn sequentially. Suppose that, in the first i draws, K unique values $\{\boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_K\}$ have been observed. Then, it can be shown that the conditional distribution of the $(i + 1)$st draw is (in mixture notation)

$$\Omega_{i+1} \,|\, \Omega_i, \ldots, \Omega_1, H, \gamma \sim \sum_{k=1}^{K} \frac{m_k}{i+\gamma} \delta\{\Psi_k\} + \frac{\gamma}{i+\gamma} H, \tag{2.14}$$

where m_k is the number of times the value $\boldsymbol{\Psi}_k$ has previously been drawn [11, 53]. This statement says that Ω_{i+1} is set to a previously sampled value with a probability proportional to the number of times that value has been drawn, or Ω_{i+1} is set to a *new* value from H with probability proportional to γ.

Equation (2.14) shows that DP follows a "rich get richer" scheme: the more times an atom has been sampled in the past, the more likely it is to be sampled again in the future. As a result, N consecutive draws are very unlikely to produce N unique values. Instead the draws will be grouped into $K(N)$ unique values (clusters), where $K(N) \ll N$ with high probability. Indeed, the expected number of clusters can be computed exactly using the definition of the CRP. Specifically, the expected number of clusters in N draws is the expected number of clusters in $(N-1)$ draws plus the probability that the Nth draw creates a new cluster, i.e.

$$\begin{aligned} \mathbb{E}[K(N)] &= \mathbb{E}[K(N-1)] + \frac{\gamma}{\gamma+N-1} \\ &= \gamma \sum_{i=1}^{N} \frac{1}{\gamma+i-1}. \end{aligned} \tag{2.15}$$

The sum can be written in closed form using the digamma function $\psi(\cdot)$, and approximated as:

$$\begin{aligned} \mathbb{E}[K(N)] &= \gamma(\psi(\gamma+N) - \psi(\gamma)) \\ &\approx \gamma \log(1 + N/\gamma). \end{aligned} \tag{2.16}$$

The first line uses the property that $\psi(x) = \psi(x-1) + 1/x$, and the second uses the fact that $\psi(x) \approx \log(x)$ for large arguments [16]. Thus, the Dirichlet process prior prefers to use a number of clusters that scales linearly with γ and approximately logarithmically in the number of data. Importantly, the number of clusters is a *random* quantity. The posterior of the DP may manifest more or fewer clusters than the mean, depending on the data being modeled.

Although the CRP only defines the conditional distributions of the Dirichlet process, the conditionals are sufficient to devise Markov Chain Monte Carlo (MCMC) methods for posterior inference. In chapter 4, a hierarchical version of the CRP called the Chinese Restaurant Franchise is used to develop a collapsed Gibbs sampler, an MCMC method, for IWTM.

2.4.2 Stick-Breaking Construction

The CRP defines the Dirichlet process implicitly, in terms of conditional draws from it. This subsection reviews Sethuraman's stick-breaking construction [46], which instead gives an explicit definition of the atoms and weights of the DP.

The stick-breaking construction states that a if $\boldsymbol{\phi} \sim \mathrm{DP}(\gamma H)$ is a Dirichlet-process distributed random measure, it can be written as the infinite mixture

$$\boldsymbol{\phi} = \sum_{k=1}^{\infty} \beta_k \delta \left\{ \boldsymbol{\Psi}_k \right\}, \tag{2.17}$$

where atoms $\boldsymbol{\Psi}_k \sim H$. Furthermore, it gives an explicit definition of the atom weights as

$$\beta_k = \beta'_k \prod_{j=1}^{k-1} (1 - \beta'_j) \tag{2.18}$$

in terms of random variables β'_k that are independent draws from the beta distribution $\mathrm{Beta}(1, \gamma)$. Note that $\sum_{j=1}^{\infty} \beta_j = 1$ almost surely. Thus, $\boldsymbol{\beta}$ is a random (discrete) probability distribution over the atoms $\boldsymbol{\Psi}$. The weights defined by (2.18) are said to follow a Griffiths-Engen-McCloskey distribution, and the generative process is sometimes written $\boldsymbol{\beta} \sim \mathrm{GEM}(\gamma)$ [53].

The stick-breaking construction is so-named because the process of generating the weights is analogous to breaking a unit-length stick into a set of progressively smaller pieces. The first stick piece, of size $\beta_1 = \beta'_1$, is formed by breaking off proportion β' from the whole stick; the second stick, of size $\beta_2 = \beta'_2(1 - \beta'_1)$, results from breaking off a proportion β'_2 of the remaining stick; and so on. Throughout this dissertation, the β random variables are referred to as the *stick weights* of the Dirichlet process.

Although the CRP and the stick-breaking construction are both representations of the Dirichlet process, they describe it in a subtly different ways: The CRP defines a distribution over *partitions* of data, while the stick-breaking prior (SBP) is a distribution over non-interchangeable *cluster labels* [42, 31]. The difference can be observed by noting that the prior probability of selecting a stick in the SBP depends on the stick's *index*. Indeed, the prior expectation of the stick weights are

$$\mathbb{E}[\beta_k] = \mathbb{E}[\beta'_k] \prod_{j=1}^{k-1} \mathbb{E}[1 - \beta'_j] = \frac{\gamma^{k-1}}{(1+\gamma)^k}, \tag{2.19}$$

which makes use of the independence of the β's and the formula for the beta mean, $\mathbb{E}[\beta'_k] = \frac{\gamma}{1+\gamma}$. Note that expected stick weight decreases *exponentially* with its index. The stick-breaking construction is said to give a *size-biased* ordering of the Dirichlet process: The largest cluster is most likely to occur at index one, the second-largest cluster is most likely to occur at index two, and so on [42, 31, 16]. In contrast, the definition of the CRP has no notion of cluster indices.

The difference between the CRP and SBP is further illustrated through a simple example. Suppose two draws from a DP, denoted z_1 and z_2, are assigned to the same cluster. The CRP defines the probability of such an event directly: it is

$1/(1+\gamma)$. The same probability can be computed from the SBP, but only indirectly. It is found by calculating the probability that both z_1 and z_2 are assigned to a *specific* cluster index k and then marginalizing over the value of k. Specifically, the joint probability of assigning both z_1 and z_2 to the kth cluster is

$$\begin{aligned} p(z_2 = z_1 = k) &= p(z_2 = k \mid z_1 = k) p(z_1 = k) \\ &= \frac{2(1+\gamma)^{k-1}}{(2+\gamma)^k} \frac{\gamma^{k-1}}{(1+\gamma)^k} \\ &= \frac{2\gamma^{k-1}}{(1+\gamma)(2+\gamma)^k}. \end{aligned} \tag{2.20}$$

Marginalizing over the cluster label k recovers the probability of interest:

$$\begin{aligned} \sum_{k=1}^{\infty} p(z_2 = z_1 = k) &= \frac{2}{\gamma(1+\gamma)} \left(\sum_{k=0}^{\infty} \left(\frac{\gamma}{2+\gamma} \right)^k - 1 \right) \\ &= \frac{1}{(1+\gamma)}. \end{aligned} \tag{2.21}$$

Thus, the CRP and SBP are equivalent descriptions of the Dirichlet process, but the former implicitly marginalizes out the labels and describes the partitions induced by the DP, while the latter describes a size-biased ordering of its clusters. The choice of which one to use depends on the inference method. With MCMC procedures where only the conditional distributions of the DP are needed, the collapsed space of the CRP is preferable. Variational inference (explored in later chapters) requires that the joint probability of all cluster weights be defined; it thus operates better on the stick-breaking construction.

2.5 Conclusions

When applied to non-text domains, text-based topic models like LDA require that document features be preprocessed into bags of words. This requirement leads to a number of issues. First, the vocabulary size K is a fixed parameter, and therefore selecting an optimal value for it requires training and evaluating many separate clustering and topic models. Doing so is computationally expensive and can be prohibitive for large datasets. Second, quantizing the document features throws away information about the data that may be useful for subsequent modeling steps. Third, K-means cannot be merged into a topic model like LDA because it is not actually a probabilistic model. Fourth, the fact that visual vocabulary is determined in preprocessing means that it must remain fixed for the lifetime of the topic model. This restriction makes incremental training difficult – that is, adapting or grow the visual vocabulary when new training documents become available.

All of these problems motivate the use of a more flexible and powerful method for clustering document-level features – namely, one based on Gaussian Dirichlet process mixtures. The analysis carried out in this chapter shows that the GDPMM can be thought of as a fully Bayesian, soft-clustering version of the K-means algorithm, equipped with a non-parametric prior that allows the visual vocabulary size to be inferred along with the rest of the model. Because the GDPMM is a proper probabilistic model, it can be integrated into, and trained jointly with, a topic model. The next chapter does just that, resulting in the Infinite-Word Topic Model.

Chapter 3

The Infinite-Word Topic Model

Chapter 2 described the conventional approach to topic modeling in non-text domains: document-level feature vectors are first converted to bags of words (BOW) using a offline clustering model like K-means, and then a text-based topic model like LDA is applied to them. This chapter defines the Infinite-Word Topic Model (IWTM), the primary contribution of this dissertation. IWTM integrates the clustering of the feature vectors into its generative process, which allows it to operate directly on such features without a separate preprocessing step. By building off of the hierarchical Dirichlet process, IWTM infers the number of clusters to use based on the size and complexity of the dataset being modeled.

3.1 Infinite-Word Topic Model

This section defines the Infinite-Word Topic Model. Section 3.1.1 first defines the model generally, in terms of an unspecified cluster likelihood and HDP base measure. Then, section 3.1.2 describes the likelihood and base measure used throughout this dissertation for image analysis tasks.

3.1.1 General Model

IWTM differs from LDA with bags of words (BOW) in two key ways. First, instead of quantizing image features in an offline preprocessing step, feature clustering is incorporated into the probabilistic model. In LDA, the observed documents come in the form of a set of discrete cluster indicators $\boldsymbol{w}_d$; in IWTM, the multivariate image features $\mathbf{x}_d$ are observed and the cluster indicators are latent variables. Second, whereas the BOW representation requires a fixed vocabulary size K, IWTM treats the number of visual words (clusters) as a random quantity to be inferred. A non-parametric prior is placed on the clustering parameters, allowing *a priori* an *infinite* number of visual words to be used. In practice, when conditioned on a finite dataset, the model uses a finite number of clusters that grows with the size and complexity of the data.

In LDA, topics are distributions over the same vocabulary, with each topic placing different weights on the words. Likewise, topics in IWTM are distributions over the same (countably infinite) set of cluster components, with each topic assigning different weights to the components. IWTM topics are modeled as a set of T dependent Dirichlet processes, with all DPs sharing the same set of atoms:

$$\boldsymbol{\phi}_0 \sim \mathrm{DP}(\gamma \mathrm{H}) \qquad \boldsymbol{\phi}_t \sim \mathrm{DP}(\alpha_0 \boldsymbol{\phi}_0) \quad \text{for } t \in \{1, \dots, T\}.$$

This construction is known as a Hierarchical Dirichlet Process (HDP) [53], where H is the base measure, a prior distribution over the cluster parameters. Both $\boldsymbol{\phi}_0$ and the topics $\boldsymbol{\phi}_t$ take the form of discrete mixtures over a shared, countably infinite set of cluster parameters $\{\boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2, \dots\}$, where $\boldsymbol{\Psi}_k \sim \mathrm{H}$. That is, each topic $\boldsymbol{\phi}_t$ can be written as a distribution

$$\boldsymbol{\phi}_t = \sum_{k=1}^{\infty} \phi_{tk} \delta\left(\boldsymbol{\Psi}_k\right), \tag{3.1}$$

where the parameters $\{\boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2, \dots\}$, $\boldsymbol{\Psi}_k \sim \mathrm{H}$ are *shared* among all topics. In effect, the first Dirichlet Process draw $\boldsymbol{\phi}_0$ generates the infinite set of cluster parameters, and each topic $\boldsymbol{\phi}_1, \dots \boldsymbol{\phi}_T$ is a reweighted mixture of the same set. The concentration parameter α_0 controls the variance of the weights around the mean topic $\boldsymbol{\phi}_0$, while γ controls the prior number of clusters. Like the Dirichlet process, the hierarchical Dirichlet process has multiple equivalent representations, and the representation used typically depends on the posterior inference method used. This chapter focuses on the remainder of IWTM, deferring a more precise definition of the hierarchical Dirichlet process until chapter 5.

Given the topics, a document corpus is generated using a procedure similar to that of LDA. The key difference is that IWTM generates a set of multivariate features $\mathbf{x}_d$ for each document, rather than a set of discrete "word" indicators:

- For each document d,
  1. Draw a set of topic weights $\boldsymbol{\theta}_d \sim \mathrm{Dir}(\boldsymbol{\alpha})$.
  2. For each datum index i in document d,
     (a) Draw a topic $z_{d,i}$ by sampling from $\boldsymbol{\theta}_d$
     (b) Draw cluster parameters Ω_{di} from the chosen topic $\boldsymbol{\phi}_{z_{d,i}}$
     (c) Draw datum $\mathbf{x}_{di}$ from cluster density $F(\Omega_{di})$.

The complete generative model for IWTM is

$$\begin{aligned}
\boldsymbol{\phi}_0 \mid \gamma &\sim \mathrm{DP}(\gamma \mathrm{H}) \\
\boldsymbol{\phi}_t \mid \alpha_0, \boldsymbol{\phi}_0 &\sim \mathrm{DP}(\alpha_0 \boldsymbol{\phi}_0), && t \in 1 \dots T, && \text{(topics)} \\
\boldsymbol{\theta}_d \mid \boldsymbol{\alpha} &\sim \mathrm{Dir}(\boldsymbol{\alpha}), && d \in 1 \dots D, && \text{(topic weights)} \\
z_{d,i} \mid \boldsymbol{\theta}_d &\sim \boldsymbol{\theta}_d, && i \in 1 \dots n_d, && \text{(topic indicators)} \\
\Omega_{d,i} \mid \boldsymbol{\phi}, z_{d,i} &\sim \boldsymbol{\phi}_{z_{d,i}}, && i \in 1 \dots n_d, && \text{(cluster parameters)} \\
\mathbf{x}_{d,i} \mid \Omega_{d,i} &\sim F(\Omega_{d,i}), && i \in 1 \dots n_d, && \text{(data).}
\end{aligned}$$

As in LDA, each document is described by a set of T topic weights $\boldsymbol{\theta}_d = (\theta_{d1}, \dots, \theta_{dT})$, where $\sum_t \theta_{dt} = 1$, and the topic for each datum $i = 1 \dots N_d$ is chosen randomly according to these weights. However, whereas LDA draws a discrete word indicator from the selected topic, IWTM draws cluster parameters $\boldsymbol{\Omega}_{di}$ from the topic and then generates the multivariate feature vector $\mathbf{x}_{di}$ from that cluster.

Note that the cluster parameters Ω_{di} are draws from $\boldsymbol{\phi}_{z_{di}}$, which is a Dirichlet process. As discussed in section 2.4, a key property of the Dirichlet process is that draws from it exhibit grouping. In IWTM, the Ω_{di} values will generally not be unique to a single datum. Rather, there will be a small number of unique values that are shared by many data across many different documents. Specifically, each Ω_{di} is one of the cluster parameters $\{\boldsymbol{\Psi}_k\}_{k=1}^{\infty}$ common to all topics.

3.1.2 A Likelihood and Base Measure for Image Analysis Tasks

Up to now, IWTM has been defined in terms of a general, unspecified likelihood function F and base measure H. For the domains considered in this dissertation, F is a multivariate normal likelihood with spherical covariance, with the base measure H putting a multivariate normal-gamma prior over the cluster means and precisions. That is, each set of cluster parameters $\boldsymbol{\Psi}_k$ is a pair consisting of a mean vector $\boldsymbol{\mu}_k$ and scalar precision λ_k generated as

$$\boldsymbol{\Psi}_k = (\boldsymbol{\mu}_k, \lambda_k) \qquad \boldsymbol{\mu}_k \,|\, \lambda_k \sim \mathcal{N}(\mathbf{m}, (\beta_0 \lambda_k)^{-1}\mathrm{I}) \qquad \lambda_k \sim \mathrm{Gam}(\chi_1, \chi_2), \tag{3.2}$$

where χ_1 and χ_2 are hyperparameters specifying the prior precision of the clusters, $\mathbf{m}$ is the prior mean of the cluster centers, and β_0 is the ratio of the precision of the centers to that of the clusters themselves.

Using a spherical covariance multivariate normal likelihood for F and a multivariate normal-gamma prior for H is appropriate for the following four reasons: First, making these choices for F and H turns the clustering portion of IWTM into a fully Bayesian, soft-clustering version of the K-means algorithm (see section 2.3).

Doing so enables comparisons between IWTM and the combination of LDA and K-means.

Second, using clusters with a simple, spherical covariance structure makes posterior inference in IWTM faster. Both the collapsed Gibbs sampler and variational inference procedures described in later chapters can be adapted to more complex covariances, such as diagonal-, semi-tied or full-covariance Gaussians, although at a greater computational cost.

Third, in the domains for which IWTM was designed, the data $\mathbf{x}_{di}$ being clustered are of high dimensionality, and complex covariance structures with many parameters may overfit the data. For example, in the image analysis experiments throughout this dissertation, the observed data are typically 128-dimensional dense SIFT descriptors. At this dimension, the full covariance matrix of a single cluster would have $128^2 = 16384$ parameters, which is greater than the number of data points that would be in many clusters. Therefore, using more complex covariances structures would most likely be detrimental to IWTM's performance.

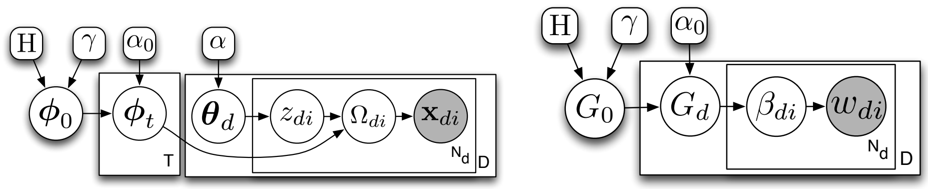
Fourth, the multivariate normal-gamma distribution is a conjugate prior to the multivariate normal likelihood [18], and conjugacy makes it possible to construct inference procedures for IWTM that actually perform better (in addition to being faster) than non-conjugate pairs. In the Gibbs sampling procedure defined in chapter 4, conjugacy allows the cluster parameters to be integrated out of the model in closed form, yielding a *collapsed* Gibbs sampler with better mixing properties. In chapter 5, conjugacy is used to derive a coordinate-ascent variational inference procedure with closed-form updates, and in chapter 6 it plays a vital role in the derivation of a stochastic variational inference for IWTM.

3.2 Related Work

IWTM is a novel contribution of this dissertation: it is the first topic model that integrates a non-parametric clustering layer into its graphical model. This section connects aspects of IWTM to existing work in the statistical machine learning and computer vision literature.

Several prior models have integrated the clustering of document-level features into a topic model, namely the Correspondence LDA model of Blei et al. [13], the Latent Mixture Vocabularies of Larlus and Jurie [34], and the Topic Random Field [63] of Zhao et al. [63]. These models differ from IWTM in three main ways. First, they are parametric models with a fixed vocabulary size. Picking the optimal vocabulary size thus requires training and evaluating multiple, separate models. Doing so is computationally expensive and becomes prohibitively costly for large datasets. In contrast, IWTM automatically infers the number of clusters to use based on the size and complexity of the data. Second, these existing models learn the parameters of their clusters using maximum likelihood, while IWTM makes Bayesian estimates of the cluster parameters. One benefit of doing so is that IWTM can make use of prior beliefs about the cluster means and variances to help prevent overfitting. Third, the existing models each assume that the topics are generated from a symmetric Dirichlet prior. In IWTM, the HDP induces *sharing* between topics, so that clusters likely to appear in one topic are pressured to appear in other topics.

Although there are a number of existing topics models that use non-parametric priors, IWTM uses the HDP orthogonally to such cases. For example, the HDP-LDA model [52] and SparseTM [56] use the HDP as a prior on the number of *topics* in the model, where each topic is a distribution over a fixed, finite vocabulary. The Focused Topic Model [60] uses the hierarchical beta process to similar effect. By contrast, the non-parametric prior in IWTM induces a *fixed* number of topics that



(a) IWTM

(b) HDP-LDA

Figure 3.1: Plate diagrams for IWTM and HDP-LDA. IWTM uses the HDP orthogonally to HDP-LDA. IWTM generates feature vectors $\mathbf{x}_{di}$ using T topics, each of which is an infinite mixture over cluster parameters. HDP-LDA generates discrete "word" indicators w_{di} using a countably infinite number of topics, each of which is a distribution over a fixed, finite vocabulary.

are distributions over an *infinitely-sized* vocabulary.[1] Graphical models for IWTM and HDP-LDA are shown in figure 3.1, which makes the distinction between the two methods clear.

More recently, Li et al. [36] developed a non-parametric model that, like IWTM, operates on document-level features directly. This work differs from IWTM in two key ways. First, the model itself is different: It is based on the hierarchical beta process [55] and is a combination of a topic model and dictionary learning method. That is, each observed image feature is generated as a weighted combination of dictionary atoms that are shared amongst all data, and the model's topics are corpus-wide patterns in the atom weights. In contrast, IWTM *clusters* the document features, and the topics have the usual interpretation as co-occurring sets of features. Second, inference on the dictionary learning topic model is limited to a Gibbs-slice sampler, which is slow and difficult to scale to large datasets: Inference on 1600 low-resolution images (250×250 pixels) takes approximately a week of CPU time [36]. In contrast, this dissertation develops efficient, parallelizable inference methods for IWTM that make it feasible to use it in place of existing modeling

[1]A possible extension of IWTM would place non-parametric priors on both the numbers of topics and clusters of the model. To limit the scope of this dissertation, such a model is reserved for future work (section 8.1).

tools such as K-means and Latent Dirichlet Allocation.

Another, more technical distinction of IWTM regards its use of the HDP. In general, the HDP defines a set of dependent DP mixtures where all mixtures share the same infinite set of atoms. In IWTM, the DP responsible for generating each datum is controlled by the latent variables z_{di}, while in most HDP-based models it is a fixed property of the data. For example, HDP-LDA [53] models a document corpus using N_{doc} dependent Dirichlet process mixtures – one for each document – and each document is generated from a mixture specific to it. Likewise, a HDP mixture model defines a set of mixtures over multiple, related groups of data, where group assignments are known *a priori*. For example, Xing et al. [62] used a HDP mixture to model gene haplotype data, where the data groups correspond to individuals' known ethnic groups. In contrast to such models, IWTM generates each datum from a random weighted mixture of dependent Dirichlet process mixtures.

3.3 Inference

In general, using a Bayesian hierarchical model to analyze a dataset requires *posterior inference*. That is, given a set of observed data and the model hyperparameters it is of interest to infer the posterior distribution over the latent variable settings that generated the data. IWTM's posterior is given by the expression

$$p(\boldsymbol{\phi}, \Omega, \boldsymbol{\theta}, \boldsymbol{z} \,|\, \mathbf{X}, a, b, \beta_0, \mathbf{m}, \gamma, \alpha_0) = \frac{p(\mathbf{X}, \Omega, \boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{z} \,|\, a, b, \beta_0, \mathbf{m}, \gamma, \alpha_0)}{p(\mathbf{X} \,|\, a, b, \beta_0, \mathbf{m}, \gamma, \alpha_0)}, \qquad (3.3)$$

where, as discussed in section 3.1, $\boldsymbol{\phi}$ are the topics, $\boldsymbol{\Omega}$ are the cluster parameters, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_d\}$ are the per-document topic weights, and $\boldsymbol{z}$ are the topic assignments for each document feature. The posterior topic weights are of particular interest: They describe each document in terms of the T topics in the model and provide a low-dimensional semantic description of the document's content. Throughout this dissertation, the topic weights are used to represent documents in classification tasks.

Unfortunately, it is not possible to compute the exact posterior directly, due to the complexity of the denominator in equation (3.3). The denominator is the normalization constant of the posterior or, equivalently, the marginal likelihood, i.e. the likelihood of the data after integrating out all possible settings of the latent variables:

$$\begin{aligned} &p(\mathbf{X} \mid \chi_1, \chi_2, \beta_0, \gamma, \alpha_0) = \\ &\qquad \iiint \sum_{\boldsymbol{z}} p(\mathbf{X} \mid \Omega) p(\Omega \mid \boldsymbol{z}, \boldsymbol{\phi}) p(\boldsymbol{z} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) p(\boldsymbol{\phi}) \; d\boldsymbol{\phi} \; d\boldsymbol{\theta} \; d\Omega. \end{aligned} \tag{3.4}$$

This quantity is intractable to compute; one reason is that the sum must be taken over an exponential number of topic assignment states $\boldsymbol{z}$, which is prohibitively expensive even for small datasets. Although it is not possible to compute the posterior directly, there are a number of established methods that can be used to approximate it. This dissertation applies several of them to IWTM, namely collapsed Gibbs sampling [37], coordinate ascent variational inference [29], and stochastic variational inference [26]. These are explained further in later chapters.

3.4 Conclusions

The primary contribution of this dissertation is IWTM, a new topic model specifically designed for modeling collections of digital media documents. It removes the assumption, common to text-based models like Latent Dirichlet Allocation, that the documents are sets of discrete, mutually exclusive words from a fixed vocabulary. To do so, it incorporates the clustering of document-level features into the topic model itself and places a non-parametric prior on the clustering parameters.

IWTM has four main advantages over existing text-based methods. First, it simplifies the process of modeling digital media collections. LDA requires that document-level features be quantized into a bag of words representation, while IWTM operates on such features directly. Second, IWTM obviates the need to tune the visual vocabulary size. In LDA, determining the optimal vocabulary size requires training and evaluating separate models. In contrast, IWTM treats the vocabulary size as a random quantity and infers it based on the size and complexity of the data. Third, IWTM uses more information about the document-level features than LDA. Its probabilistic cluster assignments contain more information about the features than bag of words, which assigns each feature entirely to a single cluster. Fourth, IWTM is better suited to online or incremental training. LDA's visual vocabulary is determined in preprocessing and must remain fixed for the lifetime of the model. In IWTM, the visual vocabulary is learned jointly with the topic structure and can adapt to changes in underlying dataset.

IWTM's strengths are discussed in detail and demonstrated experimentally throughout this dissertation. The next chapter derives a collapsed Gibbs sampler [37] for IWTM and, using it, demonstrates the model's key advantages over an LDA-based approach. Chapters 5 and 6 then focus on making inference in IWTM fast and scalable to large datasets. Specifically, chapter 5 develops a mean-field variational inference procedure [29] for IWTM that is an order of magnitude faster than Gibbs sampling. Chapter 6 then develops a stochastic variational inference procedure [26] for IWTM that leverages ideas from stochastic optimization to scale variational inference to much larger datasets. Chapter 7 develops a method for training IWTM efficiently on growing datasets, showcasing its ability to adapt the vocabulary based on the data.

Chapter 4

Gibbs Sampling

Like that of a vast majority of Bayesian hierarchical models, IWTM's posterior is intractable to compute directly and must be approximated instead (as was discussed in section 3.3). This chapter derives a posterior inference method for IWTM using collapsed Gibbs sampling [37]. Later, chapter 5 presents an alternative inference method based on mean-field variational inference [29] that allows IWTM to be scaled to large datasets.

4.1 Background

In general, Gibbs sampling is a Markov chain Monte-Carlo method that approximates a probability distribution by generating a sequence of samples from it. The algorithm is useful because it can be applied to distributions that are too complex for direct sampling methods to be possible. In the context of Bayesian probabilistic modeling, Gibbs sampling is popular as a method for posterior inference, that is, when the distribution being approximated is the posterior over the model's latent variables given a set of observed data.

In general, for a joint distribution $p(\mathbf{Z})$ over a set of random variables $\mathbf{Z} =$

Algorithm 1 Gibbs sampling on joint distribution $p(\mathbf{Z})$

```
Initialize the random variables to some state $\mathbf{Z}^{(0)}$.
for each sample $t = 1, 2 \ldots$ do
    for each random variable $j = 1 \rightarrow n$ do
        Sample $z_j^{(t)} \sim p(z_j^{(t)} \mid z_1, \ldots z_{(j-1)}^{(t)}, z_{(j+1)}^{(t-1)}, \ldots z_n^{(t-1)})$
    end for
end for
```

$\{z_1, \ldots z_n\}$, Gibbs sampling generates a sequence of samples $\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}, \ldots$ through the procedure described in algorithm 1. In the innermost loop, each individual random variable is resampled from its conditional distribution given the current state of all other variables. In many cases, it is simple to sample from these distributions, even though sampling directly from the joint distribution $p(\mathbf{Z})$ is not possible. Under mild assumptions, it can be shown that the samples $\{\mathbf{Z}^{(t)}\}$ form a Markov chain that converges to the joint distribution of interest, $p(\mathbf{Z})$ [6].

It should be noted that Gibbs sampling does not generate *independent* samples from $p(\mathbf{Z})$; rather, consecutive samples are correlated with one another. This fact is clear from the definition of the procedure: in any given iteration, the next selected value of a random variable depends on the current values of the other random variables. A corollary of this fact is that samples generated in the early iterations of the algorithm are affected by the choice of the initial state $\mathbf{Z}^{(0)}$ and do not reflect the underlying distribution $p(\mathbf{Z})$. In practice, this problem is remedied by running the sampler for an initial *burn-in period*, which allows the Markov chain to move closer to the target distribution, before using any of the samples it produces.

In some situations, such as the experiments later in this chapter, it is not strictly necessary to estimate the full distribution $p(\mathbf{Z})$, but rather to find the mode of the distribution, i.e. the most probable configuration of the random variables. The desired mode can be estimated simply by running the sampler for T iterations and retaining the sample with the largest joint probability, i.e. $\mathbf{Z}^* = \arg\max_{t=1}^{T} p(\mathbf{Z}^{(t)})$.

A heuristic that works nearly as well in practice is to use the last sample $\mathbf{Z}^{(T)}$ as the mode estimate. This method works because the samples reflect the underlying density $p(\mathbf{Z})$, and as the Markov chain converges to that distribution it is naturally drawn to high-probability configurations.

The remainder of this chapter applies Gibbs sampling to the task of posterior inference in IWTM, which is a novel contribution of this dissertation. The procedure for Gibbs sampling in IWTM is identical to that outlined above except that, rather than a generic distribution $p(\mathbf{Z})$, the sampled distribution is the joint posterior over the model's latent variables given a set of documents (equation (3.3). First, section 4.2 shows that several of IWTM's latent variables can be integrated out exactly, producing a collapsed model on which Gibbs sampling is simplified. Section 4.3 describes the Chinese Restaurant Franchise (CRF), the representation of the hierarchical Dirichlet process that is amenable to Gibbs sampling. Section 4.4 derives the actual Gibbs sampling procedure for IWTM based on the CRF. Then, using that procedure, section 4.5 evaluates IWTM on two scene classification tasks.

4.2 Model Modifications

The Gibbs sampling method for IWTM makes two modifications to the standard model presented in chapter 3. First, following prior work applying topic models to image classification tasks, an additional layer is added to IWTM to use and make predictions for documents' class labels. Second, some latent variables are integrated out of the model exactly, so that they do not need to be estimated by the Gibbs sampling procedure. The following subsections describe these modifications in detail.

4.2.1 Incorporating Supervision

IWTM was described in chapter 3 as a purely unsupervised model. Using the model for image classification tasks requires an additional mechanism for using the class labels of training documents and predicting the labels of test documents. There have been a number of supervised topic models proposed in the literature such as Supervised LDA [12] and Discriminative LDA [33]. The Gibbs sampler in this dissertation adopts the approach of Fei-Fei and Perona [22], which requires only a simple change to IWTM's generative model. Rather than generating topic weights $\boldsymbol{\theta}_d$ from a global Dirichlet distribution, they are drawn from a Dirichlet with a class-specific set of parameters. That is, given a document label y_d, the topic weights are generated as $\boldsymbol{\theta}_d \,|\, \boldsymbol{\alpha}, y_d \sim \mathrm{Dir}(\boldsymbol{\alpha}_{y_d})$. Essentially, this approach makes the assumption that each class of documents can be described as a unique distribution over topic weights. Labels for test documents are treated as unobserved parameters to be optimized by the Gibbs sampling procedure.

4.2.2 Collapsing

A benefit of using Gibbs sampling for inference in IWTM is that several parts of the generative model – namely, the per-document topic weights $\boldsymbol{\theta}_d$ and cluster parameters Ψ_k – actually do not need to be explicitly resampled. Instead, they can be *collapsed* or integrated out of the model when sampling other random variables. This particular approach is called *collapsed* Gibbs sampling. In general, collapsing out layers of a graphical model leads to sampling procedures that converge faster to the target distribution and are less susceptible to getting stuck in local modes [37].

Collapsing Topic Proportions

The topic proportions $\boldsymbol{\theta}_d$ can be integrated out of the model in closed form, so that the distribution over the topic assignments $\boldsymbol{z}_d$ can be written as a function of only the document's class label y and the hyperparameters $\boldsymbol{\alpha}$:

$$\begin{aligned}
p(\boldsymbol{z}_d \,|\, \boldsymbol{\alpha}, y_d = y) &= \int \prod_{i=1}^{n_d} p(z_{di} \,|\, \boldsymbol{\theta}_d) p(\boldsymbol{\theta}_d \,|\, \boldsymbol{\alpha}, y_d = y) \ \partial \boldsymbol{\theta}_d \\
&= \int \left(\prod_{i=1}^{n_d} \theta_{d,z_{di}} \right) \left(\frac{\Gamma(\sum_t \alpha_{yt})}{\prod_t \Gamma(\alpha_{yt})} \prod_{t=1}^{T} \theta_{dt}^{\alpha_{yt}-1} \right) \ \partial \boldsymbol{\theta}_d \\
&= \frac{\Gamma(\sum_t \alpha_{yt})}{\prod_t \Gamma(\alpha_{yt})} \int \prod_{t=1}^{T} \theta_{dt}^{\alpha_{yt}+N_{dt}-1} \ \partial \boldsymbol{\theta}_d
\end{aligned}$$

In the last line, $N_{dt} = \#\{i : \boldsymbol{z}_{di} = t\}$ denotes the number of features in the document assigned to topic t. Although the expression inside the integral appears quite complicated, the trick is to observe that it is actually the normalization constant of the distribution $\mathrm{Dir}(\boldsymbol{\theta}_d \,|\, \alpha_y + N_{d:})$. That is,

$$\int \prod_{t=1}^{T} \theta_{dt}^{\alpha_{yt}+N_{dt}-1} \ \partial \boldsymbol{\theta}_d = \frac{\prod_t \Gamma(\alpha_{yt} + N_{dt})}{\Gamma(\sum_t \alpha_{yt} + N_{dt})}.$$

Altogether, the marginal joint probability of the topic assignments for a document d in class y is:

$$p(\boldsymbol{z}_d \,|\, \boldsymbol{\alpha}, y_d = y) = \frac{\Gamma(\sum_t \alpha_{yt})}{\prod_t \Gamma(\alpha_{yt})} \frac{\prod_t \Gamma(\alpha_{yt} + N_{dt})}{\Gamma(\sum_t \alpha_{yt} + N_{dt})}. \tag{4.1}$$

Bayes' rule can be used to derive the prior conditional distributions as well. The conditional of a topic assignment $\boldsymbol{z}_{di}$, given the state of all others (denoted $\boldsymbol{z}^{-di}$, with topic counts N^{-di}) is:

$$\begin{aligned}
p(\boldsymbol{z}_{di} = z \,|\, \boldsymbol{z}^{-di}, \boldsymbol{\alpha}, y_d = y) &= \frac{p(\boldsymbol{z}_{di}, \boldsymbol{z}^{-di} \,|\, \boldsymbol{\alpha}, y_d = y)}{p(\boldsymbol{z}^{-di} \,|\, \boldsymbol{\alpha}, y_d = y)} \\
&= \frac{\alpha_{yz} + N_{dz}^{-di}}{\sum_t (\alpha_{yt} + N_{dt}^{-di})} \qquad (4.2)
\end{aligned}$$

This simply makes use of equation 4.1, along with the property $\Gamma(x) = (x-1)\,\Gamma\,(x-1)$.

For test documents where y_d is unobserved, the conditional can be computed by marginalizing out the class label in the above formulae:

$$p(\boldsymbol{z}_{di} = z \mid \boldsymbol{z}^{-di}, \boldsymbol{\alpha}) \propto \sum_{y} p(\boldsymbol{z}_{di} = z, \boldsymbol{z}^{-di} \mid \boldsymbol{\alpha}, y_d = y) p(y) \tag{4.3}$$

The other sampling steps, i.e. for the table and dish assignments, are identical to those for labeled documents.

Collapsing Cluster Parameters

As discussed in section 3.1.2, this dissertation gives IWTM a multivariate normal likelihood and sets H to a normal-gamma prior on the cluster means and precisions. That is, for $k = 1, 2, \ldots$ each set of cluster parameters $\boldsymbol{\Psi}_k$ is a pair $(\boldsymbol{\mu}_k, \lambda_k)$ generated as

$$\lambda_k \sim \mathrm{Gam}(a, b) \tag{4.4}$$

$$\boldsymbol{\mu}_k \mid \lambda_k \sim \mathcal{N}(\mathbf{m}, (\beta\lambda_k)^{-1}\mathrm{I}), \tag{4.5}$$

where a and b are hyperparameters specifying the prior precision of the clusters, m is the prior mean of the cluster centers, and β is the ratio of the precision of the centers to the precision of the clusters themselves. This subsection derives a collapsed cluster likelihood for IWTM where the means and precisions are integrated out in closed form. In the collapsed likelihood, the density of each cluster is a function only of its data and the hyperparameters $\mathbf{m}$, β_0, a and b. The integration over cluster parameters is derived first, followed by the integration over the cluster precisions.

Collapsing Cluster Means

The starting point in deriving the collapsed likelihood is to write down the joint density of drawing n data points $\mathbf{X} = \mathbf{x}_1 \ldots \mathbf{x}_n$, $\mathbf{x}_i \in \mathbb{R}^d$ from the same cluster, given its mean vector μ and its precision matrix Σ. The joint density can be written as:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_n \,|\, \boldsymbol{\mu}, \Sigma) = \frac{|\Sigma|^{n/2}}{(2\pi)^{nd/2}} \exp\left(-\frac{n}{2}(\bar{\mathbf{x}} - \boldsymbol{\mu})^\top \Sigma (\bar{\mathbf{x}} - \boldsymbol{\mu}) - \frac{n}{2}\mathrm{tr}(S\Sigma)\right), \tag{4.6}$$

where $S = \frac{1}{n}\sum(\mathbf{x}_i\mathbf{x}_i^\top) - \bar{\mathbf{x}}\bar{\mathbf{x}}^\top$ is the scatter matrix and $\mathrm{tr}(\cdot)$ denotes matrix trace. The mean vector $\boldsymbol{\mu}$ can be integrated out as:

$$\begin{aligned} p(\mathbf{X} \,|\, \Sigma) &= \int p(\mathbf{X} \,|\, \boldsymbol{\mu}, \Sigma) p(\boldsymbol{\mu} \,|\, \Sigma) d\boldsymbol{\mu} \\ &= \frac{\beta^{d/2}|\Sigma|^{(n+1)/2}}{(2\pi)^{(n+1)d/2}} \exp\left(-\frac{n}{2}\bar{x}^\top \Sigma \bar{x} - \frac{\beta}{2} m^\top \Sigma m - \frac{n}{2}\mathrm{tr}(S\Sigma)\right) \\ &\quad \times \int \exp\left(-\frac{\beta + n}{2}\mu^\top \Sigma \mu + \mu^\top \Sigma (n\bar{x} + \beta m)\right) d\boldsymbol{\mu}. \end{aligned} \tag{4.7}$$

Here, the last factor is the only one that depends on $\boldsymbol{\mu}$. Let $y = \frac{(n\bar{x}+\beta m)}{n+\beta}$ and $\Sigma' = (\beta + n)\Sigma$. Completing the square on the last factor yields the following:

$$\begin{aligned} &\int \exp\left\{-\frac{\beta + n}{2}\mu^\top \Sigma \mu + \mu^\top \Sigma (n\bar{x} + \beta m)\right\} d\boldsymbol{\mu} \\ &\quad = \exp\left\{\frac{y^\top \Sigma' y}{2}\right\} \int \exp\left\{-\frac{(\mu - y)^\top \Sigma' (\mu - y)}{2} d\boldsymbol{\mu}\right\} \\ &\quad = \exp\left\{\frac{y^\top \Sigma' y}{2}\right\} \frac{(2\pi)^{d/2}}{|\Sigma'|^{1/2}}. \end{aligned} \tag{4.8}$$

The last two lines are equivalent because the expression in the integral is the normalization constant of the the distribution $N(\boldsymbol{\mu} \,|\, y, \Sigma')$. Plugging (4.8) into (4.7) yields a closed-form for the marginal joint density of the sample

$$p(\mathbf{X} \,|\, \Sigma) = \frac{|\Sigma|^{n/2}}{(2\pi)^{nd/2}} \left(\frac{\beta}{(\beta + n)}\right)^{d/2} \exp\left\{-\frac{1}{2}\mathrm{tr}(\Sigma M)\right\} \tag{4.9}$$

where $M = \left(\frac{\beta n}{n+\beta}\right)(\bar{\mathbf{x}} - \mathbf{m})(\bar{\mathbf{x}} - \mathbf{m})^\top + nS$ is a modified scatter matrix.

Collapsing Cluster Precisions

In this dissertation, IWTM is assumed to have spherical covariance clusters. Thus, the precision matrix has the form $\Sigma = \lambda I$, where $\lambda \in \mathbb{R}^+$ is a scalar and I is the identity matrix in d dimensions. As discussed previously, the cluster precisions are given a prior $\lambda \sim \mathrm{Gam}(a, b)$. Under these assumptions, the joint density of the sample and the cluster precision λ is

$$\begin{aligned} p(\mathbf{X}) &= \int p(\mathbf{X} \,|\, \lambda) p(\lambda \,|\, a, b) d\lambda \\ &= \left(\frac{\beta}{(2\pi)^n(\beta+n)}\right)^{\frac{d}{2}} \frac{1}{b^a\Gamma(a)} \int \lambda^{\frac{nd}{2}+a-1} \exp\left\{-\lambda\left(\frac{\mathrm{tr}(M)}{2} + \frac{1}{b}\right)\right\} d\lambda \end{aligned} \tag{4.10}$$

The integral can be solved by observing that it is the normalization constant of a gamma density with parameters $a' = nd/2 + a$ and $b' = \frac{1}{\mathrm{tr}(M)/2+1/b}$. Therefore,

$$p(\mathbf{X}) = \left(\frac{\beta}{(2\pi)^n(\beta+n)}\right)^{d/2} \frac{1}{b^a\Gamma(a)} b'^{a'} \Gamma(a'). \tag{4.11}$$

Having integrated out both the cluster means and precisions, the collapsed likelihood is now only a function of the data and the model hyperparameters $\mathbf{m}$, β_0, a, and b. Implicitly, it also depends on the cluster assignments. (The analysis here has considered only a single cluster and assumed all data $\mathbf{X}$ are assigned to it.) In IWTM, the cluster assignments of the data are latent variables that are sampled by the inference procedure.

4.3 Chinese Restaurant Franchise

IWTM uses the Hierarchical Dirichlet Process to model topics as distributions over a shared, countably infinite set of clusters. One convenient aspect of using Gibbs sampling for inference is that the HDP can be defined simply in terms of its conditional distributions using a metaphor called the Chinese Restaurant Franchise (CRF) [53]. This section describes the CRF, which defines the *prior conditional* distributions of the HDP. In the next section, they are adapted to form the posterior conditional distributions needed by Gibbs sampling.

The Chinese Restaurant Franchise describes the HDP in terms of its conditional distributions, described by the metaphor of customers entering one of several Chinese restaurants to dine. When a new customer enters a restaurant, she is either randomly seated at a table with other people, or sits alone at a new unoccupied table. All customers seated at the same table share the same dish. The set of restaurants comprises a franchise in the sense that every restaurant serves the same set of dishes from a shared menu. To ground the metaphor in IWTM, each datum $\mathbf{x}_{di}$ is a customer; there are T "restaurants", each corresponding to a topic; a "dish" is a unique set of Gaussian cluster parameters; and a "table" defines a subset of data in a topic being drawn from the same cluster. The process of seating a customer in a restaurant corresponds to resampling the cluster assignment for a single datum.

In order to define the CRF conditional distributions, the following notation is introduced, borrowing from Teh et al. [53]. The latent variable z_{di} is the index of the restaurant (topic) in which datum i in document d is currently dining, and t_{di} is the index of the customer's table inside the restaurant. For any table t in restaurant r, k_{rt} is used to denote the dish being served at the table. Finally, n_{rt} represents the number of customers at table t in restaurant r, and m_{rk} denotes the number of tables in restaurant r serving dish k. Dots are used to indicate that a particular index should be marginalized (summed) over. For example, $m_{\cdot k}$ represents the total number of tables in all restaurants serving dish k. Note that because the assignment of customers to restaurants is determined by the latent variables $\boldsymbol{z}$, the summary statistics n and m should be considered functions of $\boldsymbol{z}$ as well.

The CRF specifies that when a new customer enters restaurant z_{di} she is either seated at an existing or new table with the following probabilities:

$$p(t_{di} = t \,|\, \boldsymbol{t}^{-di}, \boldsymbol{k}) \propto \begin{cases} n_{rt} & \text{if } t \text{ already exists} \\ \alpha_0 & \text{if } t = t^{\text{new}} \;. \end{cases} \tag{4.12}$$

If the table already exists, the customer eats the dish currently at the table. If a new table is created, a dish must be chosen to be served at the table. It is either selected from a global menu of dishes already being served at other tables, or a completely new dish k^{new} is created for the table. Specifically, the dish is selected by

$$p(k_{rt^{\text{new}}} = k \,|\, \boldsymbol{t}^{-rt^{\text{new}}}) \propto \begin{cases} m_{\cdot k} & \text{if } k \text{ already exists} \\ \gamma & \text{if } k = k^{\text{new}} \;. \end{cases} \tag{4.13}$$

The conditional distributions for the table and dish assignments define the Chinese Restaurant Franchise or, equivalently, the hierarchical Dirichlet process. In effect, the tables in each of the T restaurants, along with the weights with which customers are assigned to them, comprise the T topics $\{\boldsymbol{\phi}_t\}_{t=1}^{T}$. The global menu of dishes, along with the weights with which they are assigned to tables, comprise IWTM's master topic $\boldsymbol{\phi}_0$.

Note that the CRF makes the clustering properties of the HDP clear (section 2.4). The expected number of tables in each restaurant depends on the number of customers and the parameter α_0, while the number of unique dishes depends on the number of tables and the parameter γ. More specifically, if N customers enter a restaurant, the expected number of tables that will be created is approximately $\alpha_0 \log(1 + \frac{N}{\alpha_0})$. Likewise, if restaurants have a total of M tables between them, the expected number of unique dishes is approximately $\gamma \log(1 + \frac{M}{\gamma})$ [16].

Finally, given a dish assignment, the new datum $\mathbf{x}_{di}$ is generated by drawing from the cluster indicated by its dish index

$$p(\mathbf{x}_{di} \,|\, z_{di} = r, \mathbf{X}^{-di}, t_{di} = t, \boldsymbol{k}) = f_{k_{rt}}^{-di}(\mathbf{x}_{di}),$$

where the right side denotes the marginal conditional likelihood of drawing $\mathbf{x}_{di}$ from cluster k_{rt}, given all other data currently in the cluster, i.e.

$$f_k^{-di}(\mathbf{x}_{di}) = p\left(\mathbf{x}_{di} \,|\, \{\mathbf{x}_{uv} : k_{z_{uv},t_{uv}} = k \wedge (u,v) \neq (d,i)\}\right).$$

This expression is readily computed by applying Bayes' rule to the collapsed likelihood in Equation 4.11.

4.4 Collapsed Gibbs Sampling for IWTM

This section defines the collapsed Gibbs sampling procedure for IWTM. The sampler is derived for the collapsed, supervised variant of IWTM described in section 4.2, using the Chinese Restaruant Franchise representation of the HDP.

4.4.1 Resampling assignments

Because the cluster parameters $\boldsymbol{\Psi}$ and the topic weights $\boldsymbol{\theta}$ can be marginalized out in closed form, Gibbs sampling in IWTM consists of iteratively resampling the discrete latent variables $\boldsymbol{z}$, $\boldsymbol{t}$, and $\boldsymbol{k}$. In the implementation developed in this dissertation, all of the assignments of a single datum are resampled jointly. That is, for each datum index i in each document d, the following steps are performed. First, the datum is removed completely from the model by removing it from its dish, table and restaurant assignments and updating the sufficient statistics. Second, the restaurant assignment is resampled, marginalizing over table and dish assignments. Finally, the table and dish assignments for the datum are resampled, given the new restaurant.

The sampling updates are best explained in terms of the likelihood of $\mathbf{x}_{di}$, marginalizing over restaurant, table and dish assignments. Fixing the restaurant z, the likelihood of $\mathbf{x}_{di}$, marginalizing over *table* assignments is:

$$\begin{aligned} &p(\mathbf{x}_{di} \,|\, \mathbf{X}^{-di}, \boldsymbol{z}, z_{di} = z, \boldsymbol{t}^{-di}, \boldsymbol{k}^{-di}) \\ &\quad = \sum_{t=1}^{m_{z\cdot}} \frac{n_{zt}}{n_{z\cdot} + \alpha_0} f_{k_{zt}}^{-di}(\mathbf{x}_{di}) + \frac{\alpha_0}{n_{z\cdot} + \alpha_0} p(\mathbf{x}_{di} \,|\, \mathbf{X}^{-di}, \boldsymbol{z}, t_{di} = t^{\text{new}}, \boldsymbol{k}^{-di}) \end{aligned}$$

The first sum above corresponds to the case where the customer is assigned to one of the $m_{z\cdot}$ existing tables. The final term is the data likelihood given a *new* table, marginalizing over the choice of dish at the table. It is computed as:

$$p(\mathbf{x}_{di} \,|\, \mathbf{X}^{-di}, \boldsymbol{z}, t_{di} = t^{\text{new}}, \boldsymbol{k}^{-di}) = \sum_{k=1}^{K} \frac{m_{\cdot k}}{m_{\cdot\cdot} + \gamma} f_k^{-di}(\mathbf{x}_{di}) + \frac{\gamma}{m_{\cdot\cdot} + \gamma} f_{k^{\text{new}}}^{-di}(\mathbf{x}_{di}).$$

Here, the sum is taken over the $K+1$ possible choices for the dish at the new table: It is either chosen to be the K dishes currently being served, or it is a completely new dish.

Using these quantities, Gibbs sampling proceeds as follows. For datum $\mathbf{x}_{di}$, a new restaurant is first selected by sampling from the distribution:

$$\begin{aligned} &p(z_{di} = z \,|\, \boldsymbol{z}^{-di}, \mathbf{X}, \alpha_0, \boldsymbol{t}^{-di}, \boldsymbol{k}^{-di}) \\ &\quad \propto p(\mathbf{x}_{di} \,|\, z_{di} = z, \mathbf{X}^{-di}, \alpha_0, \boldsymbol{t}^{-di}, \boldsymbol{k}^{-di}) p(z_{di} = z \,|\, \boldsymbol{z}^{-di}). \end{aligned} \tag{4.14}$$

Given a new restaurant z, the customer is seated at either an existing table, or a new table is created. The table assignment is sampled from the following conditional:

$$p(t_{di} = t \,|\, z^{-di}, \mathbf{X}, \alpha_0, \boldsymbol{t}^{-di}, \boldsymbol{k}^{-di}) \propto \begin{cases} n_{zt} f_{k_{zt}}(\mathbf{x}_{di}) & \text{if } t \text{ exists} \\ \alpha_0 p(\mathbf{x}_{di} \,|\, \boldsymbol{z}, t_{di} = t^{\text{new}}) & \text{if } t = t^{\text{new}} \end{cases}.$$

If an existing table is selected, the customer eats the dish already at the table (that is, the datum is added to cluster k_{zt}). If a new table is created, the dish (cluster) for the new table is chosen according to

$$p(k_{rt^{\text{new}}} = k \mid \boldsymbol{t}, \mathbf{X}) \propto \begin{cases} m_{\cdot k} f_k^{-di}(\mathbf{x}_{di}) & \text{if } k \text{ exists} \\ \gamma f_k^{\text{new}}(\mathbf{x}_{di}) & \text{if } k = k^{\text{new}} \ . \end{cases}$$

In either case, the datum $\mathbf{x}_{di}$ is added to the specified cluster and its sufficient statistics are updated appropriately.

4.4.2 Making Class Predictions

In document classification tasks, it is of interest to use IWTM to infer the class labels of unlabeled test documents. For a test document with topic assignments $\boldsymbol{z}^{(\text{test})}$, the posterior distribution over the document's class labels is computed as:

$$\begin{aligned} p(y^{\text{test}} = y \mid \boldsymbol{z}^{\text{test}}, \boldsymbol{\alpha}) &= \frac{p(\boldsymbol{z}^{\text{test}} \mid y^{\text{test}} = y) p(y)}{\sum_{y'} p(\boldsymbol{z}^{\text{test}} \mid y^{\text{test}} = y') p(y')} \\ &\propto \frac{\Gamma(\sum_t \alpha_{yt})}{\prod_t \Gamma(\alpha_{yt})} \frac{\prod_t \Gamma(\alpha_{yt} + N_{dt})}{\Gamma(\sum_t \alpha_{yt} + N_{dt})} p(y). \end{aligned} \tag{4.15}$$

The prior $p(y)$ can be set to the empirical proportion of class y in the training corpus. An unlabeled document's class is predicted to be

$$y_d^* = \arg\max_y p(y_d = y \mid \boldsymbol{z}_d, \boldsymbol{\alpha}). \tag{4.16}$$

That is, the predicted class is the one with maximum probability given the current model state.

Hyperparameter updates

The decision rule in (4.16) depends only on the topic assignments $\boldsymbol{z}$ and the hyperparameter matrix $\boldsymbol{\alpha}$, whose rows $\boldsymbol{\alpha}_y$ describe each class y in terms of a distribution over topic weights. For IWTM to achieve high classification accuracy, it is important to fit $\boldsymbol{\alpha}$ to the training data.

To do so, $\boldsymbol{\alpha}$ is set to maximize the posterior probability of the current model state at regular intervals throughout the Gibbs sampling procedure. Maximizing the posterior with respect to $\boldsymbol{\alpha}$ is equivalent to maximizing the log joint probability with respect to each class's parameters $\boldsymbol{\alpha}_y$. Therefore, the posterior is maximized with respect to $\boldsymbol{\alpha}$ by setting $\boldsymbol{\alpha}_y \leftarrow \boldsymbol{\alpha}_y^*$, where

$$\boldsymbol{\alpha}_y^* = \arg\max_{\boldsymbol{\alpha}_y} \sum_{d:y_d=y} \log p(\boldsymbol{z}_d \,|\, \boldsymbol{\alpha}_y, y_d = y). \tag{4.17}$$

The solution to (4.17) can be found by computing the formulae for the gradients of this expression and plugging them into a standard numerical optimization method. The implementation developed for this dissertation uses the L-BFGS optimizer provided by the open-source SciPy library [28].

4.5 Experiments

In order to evaluate IWTM and compare it to other models, the model was run on two natural scene classification datasets used in the topic modeling literature. The first is the 13-scene dataset from [22]. The second is an eight-class subset of that data used by Li et al. [36] to evaluate a model that combines topic modeling with dictionary learning. For each dataset, IWTM's performance is compared against that of the other models.

4.5.1 Scene classification: 13-scene

In this task, IWTM is used to classify images by their *natural scene type*, using the 13-scene database of Fei-Fei and Perona [22]. The dataset consists of 3859 grayscale images over 13 image classes, both of indoor scenes (*bedroom*, *livingroom*, *kitchen*, *office*) and outdoor scenes (*coast*, *mountain*, *citystreet*, *open country*, *suburb*, *highway*, *forest*, *inside city*). As a baseline, the same task is performed using the supervised extension of LDA described in section 2.2. The LDA-based model is equivalent to Theme Model 1 in Fei-Fei and Perona's paper.

Following Fei-Fei and Perona [22], SIFT descriptors are densely sampled from each image over a regular grid with a step size of 10 pixels. This process yields a set of approximately 600 128-dimensional SIFT descriptors per image.

To convert the images into the appropriate representation for LDA, K-means is used to train a vector quantization model on the descriptors from the training set. Then, the data from all images are quantized by mapping each descriptor to the index $\{1 \ldots K\}$ of its closest cluster. The quantization process yields a set of visual word counts in $\mathbb{N}^K$ for each document. Because the optimal vocabulary size for LDA+K-means is not known *a priori*, it is necessary to train and evaluate LDA under multiple values of K. These experiments sweep over a variety of values in the range 25-2000, with a smaller step size at lower values where the effect is more sensitive. In IWTM, there is no need to sweep over the vocabulary size because number of feature clusters is a random quantity inferred by the model.

Each model is evaluated with different numbers of topics $T = \{25, 50\}$. To see the behavior of the models with different amounts of training data, models are trained with $n_{\text{train}} = \{25, 50, 100\}$ labeled images per class. The remaining 2856 images are used as the test set. In both IWTM and LDA, each test document is classified by its maximum a posteriori class label using the decision rule in 4.17 after undergoing 50 iterations of Gibbs sampling.

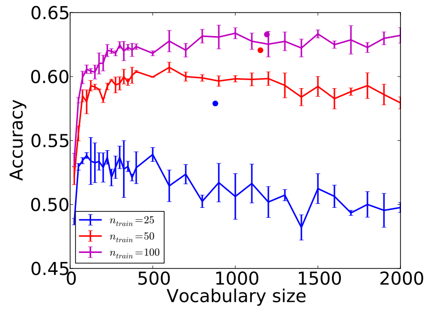
For each experimental trial, IWTM and LDA models are trained for 500 iterations of Gibbs sampling, with hyperparameters updated every 10 iterations after a burn-in period of 100 iterations. In both models, the $\boldsymbol{\alpha}$ matrix containing the parameters to the per-class Dirichlet prior on topic weights is optimized. In addition, the topic concentration hyperparameters β in LDA and α_0 in IWTM are both optimized. LDA is initialized with $\beta = 1$; IWTM is initialized with $\alpha_0 = 50$ and $\gamma = 1$. Hyperparameter $\mathbf{m}$ is set to an estimate of the corpus mean, and a, b, and β are set to induce a vague prior on the cluster precisions.

4.5.2 Results

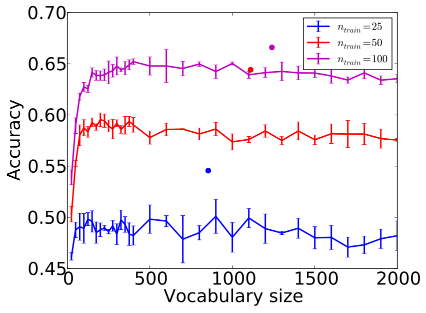
Both IWTM and LDA achieve their best results with a large amount of training data (100 documents per class) and large number of topics ($T = 50$). LDA has its highest mean accuracy, $65.15\% \pm 0.4$ with a vocabulary size of $K = 1000$. IWTM infers a vocabulary with mean size 1240, with slightly higher mean accuracy than LDA ($66.8\% \pm 0.3$). For comparison, Fei-Fei and Perona [22] report getting 65.2% accuracy with their LDA model on the same size dataset.

Figure 4.1 shows detailed comparisons of IWTM and LDA+BOW broken down over different numbers of topics (4.1(a), 4.1(b)) and different amounts of training data (4.1(c), 4.1(d)). IWTM accuracy exceeds the best LDA result in all but one scenario, showing larger improvements for small amounts of training data. The exception is ($n_{\text{train}} = 100$, $T = 25$), where IWTM performs approximately as well as LDA models with large vocabularies.

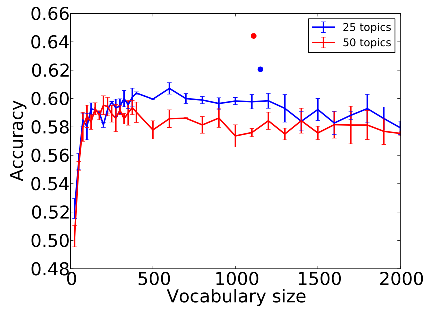
Note that these are comparisons between IWTM and the *best* LDA result in each setting, maximizing over the vocabulary size K. Empirically determining the best value of K can be a computationally expensive process, since it requires training both a clustering model and topic model for each choice of vocabulary size. To complicate matters further, the results suggest that the optimal vocabulary size



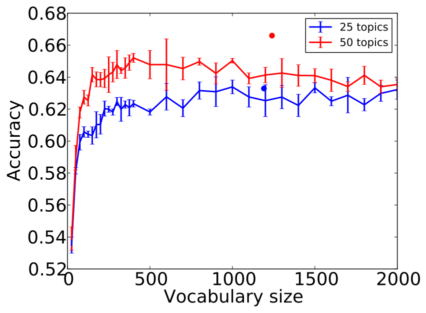
(a) $T = 25$



(b) $T = 50$



(c) $n_{\text{train}} = 50$

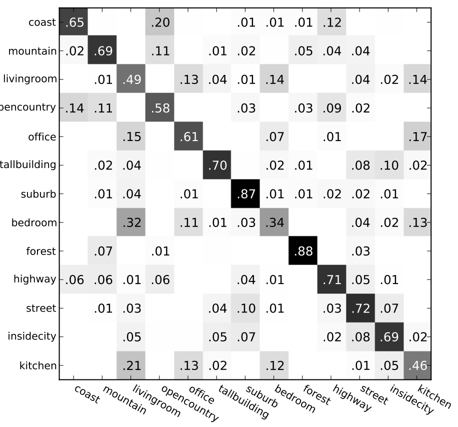


(d) $n_{\text{train}} = 100$

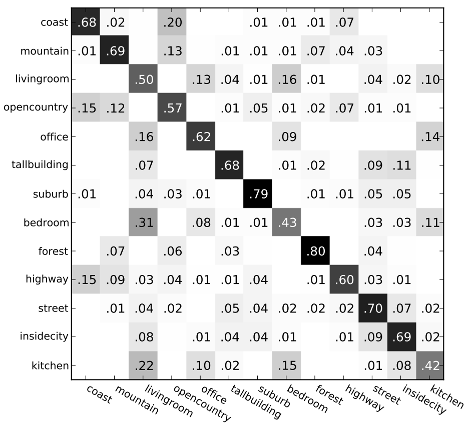
Figure 4.1: IWTM vs. LDA performance on the 13-scene classification task. Results are broken down over different numbers of topics (T) and the number of training images per class (n_{train}). Lines represent the results of LDA with bag of words, over a number of different vocabulary sizes; solid dots indicate the results of IWTM, which automatically infers the vocabulary size. In each setting, IWTM automatically infers an appropriate vocabulary size and outperforms LDA models with any vocabulary size.

for LDA can vary with the number of topics (see, e.g. figure 4.1(c)) and increase with the size of the training set (figure 4.1(a)). This observation supports the argument that it is best to train the vocabulary jointly with the rest of the model parameters, as is done in IWTM.

Interestingly, the results demonstrate that vocabulary *size* alone does not account for the performance difference between the models: in every experimental setting, IWTM accuracy meets or exceeds the accuracy of LDA models trained with approximately the same vocabulary size. This fact suggests that IWTM benefits



(a) IWTM



(b) LDA

Figure 4.2: Confusion matrices for the best-performing IWTM and LDA models on the 13-scene classification task.

from its soft clustering of the data, in contrast to the hard cluster assignments used to form LDA's bag of words representation. An alternate hypothesis is that, even when both models use the same number of clusters, IWTM is somehow able to find *better* clusters than the K-means algorithm. However, this explanation seems unlikely, since both methods assume the clusters are spherical-covariance Gaussians.

To investigate this further, a simple post-hoc experiment was performed in which LDA was allowed to use the clusters learned by IWTM. Specifically, rather than quantizing LDA's data against a K-means codebook, the data was quantized against clusters from IWTM by assigning each datum to the index of the highest likelihood cluster. LDA was then trained and tested on the same datasets as the source IWTM model.

The results are summarized in table 4.1. In each case, IWTM achieves a much higher accuracy than LDA, even though both models essentially use the same set of clusters. These results indicate that IWTM benefits significantly from the use of soft cluster assignments – in fact, IWTM performs approximately as well as a corresponding LDA model trained with *twice* the training data. For example, IWTM

n_{TRAIN}	VOCAB. SIZE	LDA+BOW	IWTM
325	867	51.0	**58.0**
650	1116	58.0	**61.3**
1300	1247	61.8	**64.0**

Table 4.1: Mean accuracy on the 13-scene task for IWTM and LDA with features quantized against IWTM's clusters (*LDA+BOW* column). Results are shown for $T = 25$ with three different training set sizes n_{train}; VOCAB. SIZE indicates the number of clusters used by IWTM in the model state used to quantize data for LDA. In each case, IWTM achieves much higher accuracy than LDA using the same clusters, indicating that IWTM benefits from the use of soft cluster assignments.

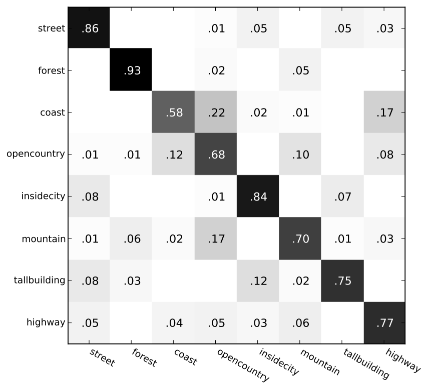
achieves 58% accuracy using 325 documents, while LDA requires 650 documents to attain this level of performance.

4.5.3 Eight-Scene Task

In a second set of experiments, IWTM is compared against a topic model recently proposed by Li et al. [36]. Their model, like IWTM, operates on image features directly, rather than a quantized bag of words representation. However, rather than *clustering* the image features, their model uses non-parametric dictionary learning to infer sparse code representations of the features, and learns a topic-like structure on top of the document-level sparse code patterns. This model is referred to as the Dictionary Learning Topic Model (DLTM).

To compare against DLTM, IWTM was trained on the eight-class subset of 13-scene. Following Li et al., 100 images per class were randomly selected for training and testing, and models were given $T = 100$ topics. Otherwise, the same experimental setup was used as in the 13-scene experiment: dense SIFT descriptors are computed on a regular grid, etc.

Figure 4.3 summarizes the results. Overall, IWTM achieves classification accuracy of $76.4\% \pm 0.2$, which is slightly higher than the 76.25% accuracy reported



Method	Accuracy
IWTM	76.4 ± 0.2
DLTM (reported in [36])	76.25

Figure 4.3: (left) Confusion matrix for IWTM on the eight-scene task. (right) IWTM's classification accuracy is comparable to the dictionary learning topic model [36], supporting the notion that modeling the image features directly, rather than through a quantized representation, is beneficial.

for the dictionary learning method. These results are encouraging. That the models perform comparably supports the hypothesis that it is beneficial to model the image features directly, rather than through a quantized representation.

4.6 Conclusions

The results on the 13-scene and eight-scene tasks show that IWTM performs as well or better than the baseline LDA model under a variety of parameter settings. Such results indicate that collapsed Gibbs sampling is an effective way of performing inference in IWTM. Nonetheless, one significant disadvantage to Gibbs sampling is that it is difficult to scale to large datasets, or to speed up inference significantly using parallel processing. In general, Gibbs sampling requires that groups of interdependent latent variables be updated sequentially, since changes to one variable will affect the way others are subsequently resampled. The latent variables in collapsed IWTM – the restaurant, table, and dish assignments $\boldsymbol{z}$, $\boldsymbol{t}$, and $\boldsymbol{k}$, respectively – are *all* interdependent, and as a result the entire sampling procedure must be performed sequentially. For large datasets, the time required to train a model with Gibbs sampling would be prohibitive.

Mean-field variational inference [29] is an alternative inference method that is appealing for its speed and scalability. In contrast to Gibbs sampling, variational inference operates by making a series of batch updates to the model state, with each iteration re-estimating large groups of latent variables independently. Generally, these updates only require computing closed-form expressions of the current model state. Moreover, because groups of parameters can be updated independently within each iteration, it is simple to speed up the algorithm by dividing work across multiple processors or machines. Variational inference is therefore a natural candidate for scaling inference to large datasets, as will be discussed in the next chapter.

Chapter 5

Variational Inference

In general, the task of inference in a Bayesian probabilistic model is to compute the posterior distribution of the model's latent variables, denoted $p(\mathbf{Z} \,|\, \mathbf{X})$, given a set of observed data $\mathbf{X}$. For most models of interest, including LDA and IWTM, $p(\mathbf{Z} \,|\, \mathbf{X})$ is intractable to calculate exactly, and instead it is of interest to approximate the posterior. The previous chapter described a Gibbs sampling method that approximates the posterior by constructing a Markov chain to sample from it. This chapter describes *mean-field variational inference* [29, 10], which takes a much different approach to approximating the posterior. In variational inference, the key idea is to estimate $p(\mathbf{Z} \,|\, \mathbf{X})$ by a simpler, tractable distribution $q(\mathbf{Z})$ and improve the approximation on each successive iteration.

5.1 Background

Variational inference poses posterior inference as optimization. Formally, for some family of approximating distributions $\mathcal{Q}$, it seeks the optimal distribution $q^*(\mathbf{Z}) \in \mathcal{Q}$ that is closest to the true model posterior in terms of KL divergence. That is, the problem is to find

$$q^*(\mathbf{Z}) = \underset{q \in \mathcal{Q}}{\arg\min}\, \mathrm{KL}(q \| p) \tag{5.1}$$

where

$$\mathrm{KL}(q \| p) = -\int q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z} \,|\, \mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}. \tag{5.2}$$

It is not feasible to solve (5.1) directly because the KL divergence is a function of the true model posterior, which is assumed to be intractable to compute. However, this minimization problem can be tackled indirectly by observing that it is equivalent to a simpler maximization problem. For any distribution $q(\mathbf{Z})$ over the latent variables, the following decomposition holds [10]:

$$\begin{aligned} \log p(\mathbf{X} \,|\, \boldsymbol{\theta}) &= \int \log p(\mathbf{X}, \mathbf{Z}) \mathrm{d}\mathbf{Z} \\ &= \mathcal{L}(q, \theta) + \mathrm{KL}(q \| p) \end{aligned} \tag{5.3}$$

where

$$\begin{aligned} \mathcal{L}(q, \theta) &= \int q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \\ &= \mathbb{E}_q \left[\log p(\mathbf{X}, \mathbf{Z}) \right] - \mathbb{E}_q \left[\log q(\mathbf{Z}) \right]. \end{aligned} \tag{5.4}$$

Here, the expectations are taken with respect to the distribution $q(\mathbf{Z})$. This decomposition expresses the marginal likelihood of the data as a sum of the function $\mathcal{L}(\cdot)$ and the KL divergence of interest. Note that the divergence term is strictly non-negative (with $\mathrm{KL}(q \| p) = 0$ if and only if $q \equiv p$), so $\mathcal{L}$ is in fact a lower bound on the marginal likelihood. Because the marginal likelihood, i.e. the sum of the two terms, is functionally independent of $q(\mathbf{Z})$, minimizing the KL divergence term is equivalent to *maximizing* the function $\mathcal{L}(q, \theta)$. The search for the optimal approximating distribution can then be recast as:

$$q^*(\mathbf{Z}) = \arg\max_{q \in \mathcal{Q}} \mathcal{L}(q, \theta). \tag{5.5}$$

Unlike the KL divergence, this expression is a function of the joint distribution $p(\mathbf{X}, \mathbf{Z})$, which is generally easy to compute. The function $\mathcal{L}(q, \theta)$ to be maximized is known as the Evidence Lower BOund (ELBO).

Because the decomposition above holds for any distribution $q(\mathbf{Z})$, the family of approximating distributions can be chosen in such a way that makes maximizing the ELBO tractable. In *mean-field variational inference* – the strategy adopted here — the joint distribution $q(\mathbf{Z})$ is assumed to factor into some J groups [29]:

$$q(\mathbf{Z}) = \prod_{j=1}^{J} q(\mathbf{Z}_j). \tag{5.6}$$

That is, the J groups of latent variables are assumed to be independent in q. Note that this is not an assumption of independence in the model posterior; we are merely choosing a factored distribution to *approximate* the posterior.

The task of variational inference is to find, from the set of factored distributions, the one that maximizes the ELBO objective. Although finding a global optimum is not generally possible, it can be shown [10] that the ELBO is locally maximized with respect to the jth factor when $q(\mathbf{Z}_j) = q^*(\mathbf{Z}_j)$, where

$$q^*(\mathbf{Z}_j) = \frac{1}{C} \exp\left\{\mathbb{E}_{i \neq j} \log p(\mathbf{X}, \mathbf{Z})\right\}. \tag{5.7}$$

Here, the expectation is taken with respect to the approximating distribution q over all latent variables *except* $\mathbf{Z}_j$, and C is a normalizing constant that does not depend on $\mathbf{Z}_j$. The right hand side of (5.7) depends on the other factors $\{q(\mathbf{Z}_i)\}_{i \neq j}$, meaning that it is the optimal distribution over $\mathbf{Z}_j$ given fixed distributions over the other latent variables. When the model contains conjugate exponential family distributions, the optimal parametric forms of the factors are also exponential family distributions, and updates implied by (5.7) can be carried out in closed form [26, 29]. Variational inference works by updating one factor at a time, keeping the others fixed, and cycling through these updates until a suitable convergence criterion is met. Each update is guaranteed to increase the ELBO (or, at worst, keep it the same). This procedure can be viewed as a form of block coordinate ascent [9] on the ELBO objective.

The remainder of this chapter applies the theory of variational inference to IWTM. Section 5.2 shows that, by using the stick-breaking construction of the Dirichlet Process, IWTM can be expressed as a model with fully conjugate exponential family distributions. As previously noted, conjugacy simplifies inference considerably because the factors of the approximating distribution are also exponential families and the update rules have simple closed forms. Section 5.3 describes the form of the approximating distribution, and section 5.4 contains the update rules for each factor. In section 5.5, variational inference for IWTM is evaluated on the 13-scene image classification task. Results show that IWTM trained with variational inference – just as with Gibbs sampling – achieves higher accuracy than an LDA-based method. However, variational inference is an order of magnitude faster than Gibbs sampling, and makes IWTM's computational requirements comparable to those of LDA and K-means.

5.2 Stick-Breaking Construction of IWTM

Recall that IWTM uses a two-level hierarchical Dirichlet process (HDP) to model topics. The Gibbs sampling procedure described in the previous chapter made use of the Chinese restaurant franchise representation of the HDP. The CRF is convenient for Gibbs sampling because it provides simple formulae for conditional draws from the HDP. However, the CRF is not amenable to efficient mean-field variational inference, which requires working with the model's log joint density $\log p(\mathbf{X}, \mathbf{Z})$. Under the CRF, the joint density of the HDP is a complex function of all table assignments $\boldsymbol{t}$ and dish assignments $\boldsymbol{k}$ in the model, and does not factor across the data or tables [59, 54]. As a result, it is not possible to derive exact, closed form update rules for a factorized approximation to the posterior. Fortunately, under a different representation of the HDP – a stick-breaking construction – the joint density has a more tractable form. Further, the resulting construction of IWTM contains only conjugate exponential family distributions, so that the update rules resulting from equation (5.7) have simple closed forms.

Recall that the topics in IWTM are T dependent Dirichlet processes generated using a common Dirichlet process as a base measure. That is,

$$\boldsymbol{\phi}_0 \sim \mathrm{DP}(\gamma \mathrm{H}) \qquad \boldsymbol{\phi}_t \sim \mathrm{DP}(\alpha_0 \boldsymbol{\phi}_0) \quad \text{for } t \in \{1, \dots, T\}$$

We have previously stated that both the master topic $\boldsymbol{\phi}_0$ and the topics $\boldsymbol{\phi}_t$ take the form of discrete mixtures over a shared, countably infinite set of cluster parameters $\{\boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2, \dots\}$, where $\boldsymbol{\Psi}_k \sim \mathrm{H}$. This relationship is made explicit by representing both levels of DPs using Sethuraman's stick-breaking construction (see section 2.4.2; [46]). The result is a stick-breaking construction of the *hierarchical* Dirichlet process [59], which is the representation on which variational inference for IWTM will operate.

The stick-breaking construction is first applied to $\boldsymbol{\phi}_0$. By its definition, the master topic $\boldsymbol{\phi}_0$ is a random measure over atoms (cluster parameters) $\{\boldsymbol{\Psi}_k\}$ drawn from the base measure H. The stick-breaking construction states that the atom weights, denoted $\boldsymbol{\beta}$, are generated as follows:

$$\begin{array}{rcll} \beta_k & = & \beta'_k \prod_{j=1}^{k-1}(1-\beta'_j) & \beta'_k \sim \text{Beta}(1,\gamma) \\ \boldsymbol{\Psi}_k & \sim & H & \\ \boldsymbol{\phi}_0 & = & \sum_{k=1}^{\infty} \beta_k \delta\left\{\boldsymbol{\Psi}_k\right\} & \end{array} \tag{5.8}$$

Likewise, each topic $\boldsymbol{\phi}_t \sim \text{DP}(\alpha_0 \boldsymbol{\phi}_0)$ is a random measure over atoms drawn from the master topic $\boldsymbol{\phi}_0$. Because $\boldsymbol{\phi}_0$ is already a discrete distribution, the topics have the same set of atoms as $\boldsymbol{\phi}_0$ and are effectively reweighted copies of it. Applying the stick-breaking construction to $\boldsymbol{\phi}_t$ shows that it has the following form:

$$\begin{array}{rcll} \pi_{tk} & = & \pi'_{tk} \prod_{j=1}^{k-1}(1-\pi'_{tj}) & \pi'_{tk} \sim \text{Beta}(1,\alpha_0) \\ c_{tk} & \sim & \beta_t & \\ \boldsymbol{\phi}_t & = & \sum_{k=1}^{\infty} \pi_{tk} \delta\left\{\boldsymbol{\Psi}_{c_{tk}}\right\} & \end{array} \tag{5.9}$$

Here, indicator variables $\mathbf{c}$ are introduced to make the mapping between the two levels of DPs explicit, where c_{tk} is the index (in the master topic) of the kth atom in topic t.

Using the stick-breaking construction of the HDP, the IWTM generative model becomes the following:

$$\begin{array}{rlll} \boldsymbol{\beta} \mid \gamma & \sim \text{GEM}(\gamma) & & \text{(master stick weights)} \\ \boldsymbol{\pi} \mid \alpha_0 & \sim \text{GEM}(\alpha_0) & & \text{(topic stick weights)} \\ \boldsymbol{\Psi}_k \mid H & \sim H, & k = 1, 2, \ldots & \text{(cluster parameters)} \\ c_{t,k} \mid \boldsymbol{\beta} & \sim \boldsymbol{\beta}, & t = 1 \ldots T, k = 1 \ldots, & \text{(topic atom-cluster map)} \\ \boldsymbol{\theta}_d \mid \boldsymbol{\alpha} & \sim \text{Dir}(\boldsymbol{\alpha}), & d \in 1 \ldots D, & \text{(topic weights)} \\ z_{d,i} \mid \boldsymbol{\theta}_d & \sim \boldsymbol{\theta}_d, & i \in 1 \ldots n_d, & \text{(topic indicators)} \\ k_{d,i} \mid \boldsymbol{\pi}, z_{d,i} & \sim \boldsymbol{\pi}_{z_{d,i}}, & i \in 1 \ldots n_d, & \text{(atom indicators)} \\ \mathbf{x}_{d,i} \mid \boldsymbol{\Psi}, z_{d,i}, k_{d,i} & \sim F(\boldsymbol{\Psi}_{c_{z_{di},k_{di}}}), & i \in 1 \ldots n_d, & \text{(data)} \end{array}$$

The benefit of the HDP construction above is that IWTM has full exponential-family conjugacy, which allows for a variational inference procedure with closed-form update rules. However, as discussed in section 2.4.2, the stick-breaking construction requires special care during posterior inference because it assumes a size-biased ordering of the DP atoms [30]. As is apparent from the form of $\boldsymbol{\beta}$ and $\boldsymbol{\pi}$ above, the stick weights in each DP decrease exponentially in expectation, so that "larger" atoms (ones drawn more times within the model) are more likely to appear at lower indices. During variational inference it is important to reorder the atoms of each DP according to their size. For IWTM, this means reordering topics' atoms according to the number of data each generates, and reordering the master topic by the number of *topic atoms* mapped to each cluster. The reordering process is described in greater detail in section 5.4.5.

5.2.1 Decoupled Classification

The experiments in this chapter, as in chapter 4, apply IWTM to the task of document classification. Note, however, that the construction of IWTM above is unsupervised: By itself, it contains no mechanism to use or predict document labels. In the Gibbs sampling experiments in chapter 4, document classification was performed by adding an additional layer to IWTM's graphical model. The approach taken in this chapter and the remainder of the dissertation is to *decouple* the topic model and classification mechanism. That is, the topic model is trained in a completely unsupervised manner, and classification is performed by training a separate classifier on document representations generated by the model. Decoupling the topic model and classifier has three main advantages:

Flexibility: A variety of document representations can be derived from the model's latent variable estimates, and a wide variety of classifiers can be trained on those representations, without modifying the graphical model and inference procedure. In general, different datasets have specific properties that make one representation or classification method more advantageous than another.

Performance: State-of-the art classifiers such as support vector machines (SVMs) [19] make assumptions about the mapping between features and labels that are difficult to encode in graphical models. SVMs are used in the scene classification experiments later in this chapter, where they achieve significantly better accuracy on the 13-scene task than the generative classifier used in chapter 4.

Extensibility: The scene classification experiments in section 5.5 use an SVM classifier trained on topic weights from IWTM, where all documents are assumed to have observed labels. In section 5.7, that method is extended to accommodate semi-supervised training. Later, in chapter 7, the method is further extended to active learning, where new unlabeled examples can be selected for labeling.

5.3 Approximating Distribution

Variational inference approximates the posterior over IWTM's latent variables with a simpler, factored distribution. This section describes the form and properties of the variational approximation.

5.3.1 Mean-Field Approximation

Under the stick-breaking construction, the posterior of the latent variables in IWTM is the distribution $p(\boldsymbol{\beta'}, \boldsymbol{\pi'}, \mathbf{c}, \boldsymbol{\mu}, \boldsymbol{\lambda}, \boldsymbol{z}, \boldsymbol{k}, \boldsymbol{\theta} \mid \mathbf{X})$. As discussed in section 3.3, the true posterior is intractable to work with. In particular, it is a complex, non-convex function of all of the latent variables, and computing its normalizing constant requires summing over an exponential number of states.

In the the variational inference method developed in this dissertation, IWTM's posterior is approximated by the distribution $q(\boldsymbol{\beta'}, \boldsymbol{\pi'}, \mathbf{c}, \boldsymbol{\mu}, \boldsymbol{\lambda}, \boldsymbol{z}, \boldsymbol{k}, \boldsymbol{\theta})$. To make variational inference tractable, it is assumed to have the following factored form:

$$q(\boldsymbol{\beta'}, \boldsymbol{\pi'}, \mathbf{c}, \boldsymbol{\mu}, \boldsymbol{\lambda}, \boldsymbol{z}, \boldsymbol{k}, \boldsymbol{\theta}) = \left(\prod_{c=1}^{C} q(\beta'_c) q(\boldsymbol{\mu}_c, \lambda_c)\right) \left(\prod_{t=1}^{T} \prod_{k=1}^{K_t} q(\pi'_{tk}) q(c_{tk})\right) \left(\prod_{d} q(\boldsymbol{\theta}_d) \prod_{i=1}^{N_d} q(z_{di}, k_{di})\right). \tag{5.10}$$

Each factor is parametrized by its own set of free parameters (denoted with tildes) that are optimized by the inference procedure. The factors are $q(\beta'_c \mid \tilde{u}_c, \tilde{v}_c)$ and $q(\pi'_{tk} \mid \tilde{a}_{tk}, \tilde{b}_{tk})$ (beta distributed); $q(\boldsymbol{\mu}_c, \lambda_c \mid \tilde{\mathbf{m}}_c, \tilde{\nu}_c, \tilde{\chi}_{1c}, \tilde{\chi}_{2c})$ (multivariate normal-gamma); $q(c_{tk} \mid \tilde{\boldsymbol{\beta}}_k)$ and $q(z_{di}, k_{di} \mid \tilde{\boldsymbol{\theta}}_{dk})$ (discrete). The parametric families of these distributions and the update rules for their parameters are derived from the general result in equation (5.7).

5.3.2 Truncation Limits

Following [30, 59, 26] the posterior approximation of the DPs are truncated to a finite number of atoms. Topics are allowed a maximum of K atoms by fixing $q(\pi'_{tK} = 1) = 1$ so that $q(\pi_{tk}) = 0$ for all $k > K$. Similarly, the master topic is truncated to a maximum of C clusters, with $q(\beta'_C = 1) = 1$ fixed so that $q(\beta_c) = 0$ for $c > C$. Note that C and K are finite limits on the approximating distribution q, not on the model itself. (The numbers of clusters and atoms in the true posterior remain bounded only by the number of observed data.) The truncation limits act as upper bounds on the numbers of clusters and atoms that can be manifested during inference. It suffices to set them to large values that are greater than the number of clusters and atoms that would feasibly be needed to model the dataset. Although it is possible to set the truncation limits to values that are too small, this condition can easily be detected. Only a small subset of the available clusters and atoms should actually be used by the model (i.e. assigned non-zero posterior mass). When the number of clusters or topic atoms in use is close to the truncation limit, the truncation limits should be increased.

5.3.3 Shared Topic Atom to Cluster Mappings

An important property of the approximating distribution is that the factors $q(c_{tk} \mid \tilde{\boldsymbol{\beta}}_k)$, by design, have no dependence on the topic index t. In other words, the atoms at corresponding indices in each topic are assumed to map to an identical distribution over clusters in the posterior. This assumption makes the inference procedure much more efficient because the atom-to-cluster mappings for all topics can be accessed through a single $K \times C$ matrix $\tilde{\boldsymbol{\beta}}$, rather than T such matrices.

As we will see in the following sections, this assumption puts only a minor restriction on the form of the approximate posterior. Although the topics must share a common set of atom mappings, each topic still maintains a separate set of weights over its atoms. Furthermore, while this assumption would appear to prevent reordering the topics independently, we show in section 5.4.5 that it is possible to *simulate* reordering the atoms without actually permuting their representations in memory.

5.4 Update Rules

Variational inference for IWTM proceeds by repeatedly maximizing each set of parameters in (5.10) one at a time in a coordinate-ascent procedure. The per-datum and per-document latent variables are updated given the global model state, and then these local estimates are used to update the global parameters of the model (the topics, clusters, and topic atom mappings). In general, each update rule is derived by applying equation (5.7) to the corresponding factor in the approximating distribution (5.10).

5.4.1 Document-Level Updates

For each document d, local inference consists of estimating the posterior topic-atom assignments of each datum, $q(z_{di}, k_{di} \,|\, \tilde{\boldsymbol{\theta}}_{di}) = \text{Discrete}(\tilde{\boldsymbol{\theta}}_{di})$, and the topic weights of the document, $q(\boldsymbol{\theta}_d \,|\, \tilde{\boldsymbol{\alpha}}_d) = \text{Dir}(\tilde{\boldsymbol{\alpha}}_d)$. Both sets of parameters are first initialized to appropriate values. Then, for each datum $i = 1 \ldots N_d$, the topic-atom assignments are updated as

$$\tilde{\theta}_{ditk} \propto \exp\left\{\mathbb{E}[\log\theta_{dt}] + \mathbb{E}[\log\pi_{tk}] + \textstyle\sum_c \tilde{\beta}_{kc}\mathbb{E}[\log p(\mathbf{x}_{di} \,|\, \boldsymbol{\mu}_c, \lambda_c)]\right\} \tag{5.11}$$

for topic indices $t \in \{1, \ldots, T\}$ and atom indices $k \in \{1, \ldots, K\}$. (The $\tilde{\theta}$'s are normalized over the last two dimensions, so that $\sum_{t,k} \tilde{\theta}_{ditk} = 1$.) Using these quantities, the posterior topic weights for the document are updated by setting:

$$\tilde{\alpha}_{dt} = \alpha + \sum_{i=1}^{N_d}\sum_{k=1}^{K} \tilde{\boldsymbol{\theta}}_{ditk} \qquad \text{for } t \in \{1, \ldots, T\} \tag{5.12}$$

A change in $\tilde{\boldsymbol{\alpha}}$ causes a corresponding change in the value of the expectation $\mathbb{E}[\log\theta_{dt}] = \psi(\tilde{\alpha}_{dt}) - \psi(\sum_t \tilde{\alpha}_{dt})$ used in (5.11). Local inference proceeds by cycling between steps (5.11) and (5.12) repeatedly until a suitable convergence criterion is met.

In the expressions above, $\mathbb{E}[\log\theta_{dt}]$ are the log topic weights, $\mathbb{E}[\log\pi_{tk}]$ is the log weight of the kth atom in topic t, and $\mathbb{E}[\log p(\mathbf{x}_{di} \,|\, \boldsymbol{\mu}_c, \lambda_c)]$ are the expected data log likelihoods. All expectations are taken with respect to the approximating distribution q and are therefore functions of q's parameters and the observed data. These expectations can be expressed in single closed forms using well-known properties of exponential family distributions. Table 5.1 provides a list of the expectations used throughout the inference procedure.

After optimizing over the document-level parameters $\tilde{\boldsymbol{\theta}}$ and $\tilde{\boldsymbol{\alpha}}$ for each document in the corpus, these quantities are then used the update the model's "global" latent variables, i.e. the cluster parameters, the topic and master topic stick weights, and the assignments of topic atoms to clusters.

5.4.2 Cluster Parameter Updates

The posterior over cluster parameters is approximated by the factors

$$q(\boldsymbol{\mu}_c, \lambda_c) = \mathcal{N}(\boldsymbol{\mu}_c \,|\, \tilde{\mathbf{m}}_c, (\tilde{\nu}_c \lambda_c \mathbb{I})^{-1}) \mathrm{Gam}(\lambda_c \,|\, \tilde{\chi}_{1c}, \tilde{\chi}_{2c}),$$

which are multivariate normal-gamma distributions. For convenience, let $\rho_{dic} = \sum_t \sum_k \tilde{\beta}_{tkc} \tilde{\theta}_{ditk}$ denote the posterior responsibilities of cluster index c generating datum (d, i) (note $\sum_c \rho_{dic} = 1$) and let $n_c = \sum_{d,i} \rho_{dic}$ be the pseudo-count of data generated from cluster c. The cluster parameter posteriors are updated as follows:

$$\tilde{\chi}_{1c} = \frac{n_c D}{2} + \chi_1 \tag{5.13}$$

$$\tilde{\chi}_{2c} = \left(\frac{\beta_0}{2} \|\tilde{\mathbf{m}}_c - \mathbf{m}\|^2 + \frac{1}{2} \sum_{d,i} \rho_{dic} \|\mathbf{x}_{di} - \tilde{\mathbf{m}}_c\|^2 + \chi_2 \right)^{-1} \tag{5.14}$$

$$\tilde{\mathbf{m}}_c = \left(\beta_0 \mathbf{m} + \sum_{d,i} \rho_{dic} \mathbf{x}_{di} \right) / \, (\beta_0 + n_c) \tag{5.15}$$

$$\tilde{\nu}_c = \beta_0 + n_c \tag{5.16}$$

5.4.3 Topic Atom to Cluster Mapping Updates

The mapping between topic atoms and clusters is modeled by the factors $q(\mathbf{c}_k) = \mathrm{Discrete}(\tilde{\boldsymbol{\beta}}_k)$ for atom indices $k \in \{1, \ldots, K\}$. The mapping parameters are updated as

$$\tilde{\beta}_{kc} \propto \exp\left\{ \mathbb{E}[\log \beta_c] + \sum_{d,i} \sum_{t=1}^{T} \tilde{\theta}_{ditk} \mathbb{E}[\log p(\mathbf{x}_{di} \,|\, \boldsymbol{\mu}_c, \lambda_c)] \right\} \tag{5.17}$$

where the normalization is over the last index, so that $\sum_{c=1}^{C} \tilde{\beta}_{kc} = 1$. The parameter $\tilde{\beta}_{kc}$ can be interpreted as the posterior estimate that the atoms at index k are mapped to the cth cluster.

As (5.17) makes apparent, the mappings will concentrate probabilty mass on the clusters that best fit (in terms of expected log likelihood) the data assigned to each atom (as estimated by $\tilde{\boldsymbol{\theta}}$). In practice, there are few such clusters, and each row $\tilde{\boldsymbol{\beta}}_k$ contains very few entries that are numerically non-zero. This phenomenon is illustrated in figure 5.1, which shows the $\tilde{\boldsymbol{\beta}}$ matrix learned by IWTM on a small dataset of images. The sparsity of $\tilde{\boldsymbol{\beta}}$ is significant for two reasons. First, it serves to justify the sharing of mappings between topics. Since the topic atom mappings resemble point masses, this suggests there would be little benefit to modeling them on a per-topic basis. Instead, a single $\tilde{\boldsymbol{\beta}}$ matrix of sufficient size can play the functional role of mapping all topic atoms to their respective clusters. Second, the sparsity of $\tilde{\boldsymbol{\beta}}$ can be exploited to make the inference procedure more efficient. Although the full mapping matrix contains $K \times C$ parameters, its nonzero values can be stored in a sparse-matrix format of size $\mathcal{O}(K)$. Correspondingly, computations over $\tilde{\boldsymbol{\beta}}$ can be sped up using sparse matrix operations. The mapping matrix must be read during document-level inference, so these time savings are significant.

5.4.4 Stick Weight Updates

Recall that the stick weights of each topic $\boldsymbol{\phi}_t$ are defined in terms of the beta-distributed random variables π'_{tk}. The posterior weights for the topic sticks are captured by the factors $q(\pi'_{tk}) = \text{Beta}(\tilde{a}_{tk}, \tilde{b}_{tk})$, for topics $t \in \{1, \ldots, T\}$ and atom indices $k \in \{1, \ldots, K-1\}$. For convenience, let $\tilde{\zeta}_{tk} = \sum_{d,i} \tilde{\theta}_{ditk}$ denote the pseudo-count of data being mapped to the kth atom in topic t. (They are given a tilde to denote that they are functions of the variational parameters.) Then the topic stick parameters are updated as follows:

$$\tilde{a}_{tk} = 1 + \tilde{\zeta}_{tk} \tag{5.18}$$

$$\tilde{b}_{tk} = \alpha_0 + \textstyle\sum_{j=k+1}^{K} \tilde{\zeta}_{tj} \tag{5.19}$$

The master topic stick updates have a similar form. The stick weights for the master topic are defined in terms of the beta-distributed random variables β'_c, and their posteriors are captured by factors $q(\beta'_c \mid \tilde{u}_c, \tilde{v}_c) = \mathrm{Beta}(\tilde{u}_c, \tilde{v}_c)$. Let $\tilde{\xi}_c = \sum_{t=1}^{T} \sum_{k=1}^{K} \tilde{\beta}_{tkc}$ be the pseudo-count of topic atoms mapped to cluster c. The master topic stick weights are updated by setting:

$$\tilde{u}_c = 1 + \tilde{\xi}_c \tag{5.20}$$

$$\tilde{v}_c = \gamma + \textstyle\sum_{j=c+1}^{C} \tilde{\xi}_j. \tag{5.21}$$

5.4.5 Optimal Reordering

As discussed in section 2.4.2, the stick-breaking construction of the Dirichlet process assumes a *size-biased ordering* of the atoms. In rough terms, the "largest" atom is most likely to be at the first index, followed by the second "largest" at the second index, and so on. More formally, the joint probability of a DP's stick weights and a set of draws from those weights is maximized when the atoms are sorted in non-increasing order by size. In variational inference, this sorted ordering of the atoms is also the ordering that maximizes the ELBO objective [32, 31]. Therefore, variational inference should maintain each DP in sorted order to achieve the best approximation of the posterior.

IWTM contains two levels of Dirichlet process that must be reordered during

Variable	Type	Expectations
$\boldsymbol{\theta}$	Dirichlet	$\mathbb{E}[\log \theta_{dt}] = \psi(\tilde{\alpha}_{dt}) - \psi(\Sigma_{j=1}^{K} \tilde{\alpha}_{dj})$
$\boldsymbol{\mu}, \boldsymbol{\lambda}$	Normal-gamma	$\mathbb{E}[\log p(\mathbf{x}_{di} \mid \boldsymbol{\mu}_c, \lambda_c)] = \frac{D}{2}\left(\mathbb{E}[\log \lambda_c] - \log 2\pi - \frac{1}{\tilde{\nu}_c}\right)$ $-\frac{1}{2}\mathbb{E}[\lambda_c] \lVert \mathbf{x}_{di} - \tilde{\mathbf{m}}_c \rVert^2$ $\mathbb{E}[\lambda_c] = \tilde{\chi}_{1c}\tilde{\chi}_{2c}$ $\mathbb{E}[\log \lambda_c] = \psi(\tilde{\chi}_{1c}) + \log(\tilde{\chi}_{2c})$
$\mathbf{c}$	Discrete	$\mathbb{E}[c_{tkc}] = \tilde{\beta}_{tkc}$
$\boldsymbol{\beta}'$	Beta	$\mathbb{E}[\log \beta_c] = \mathbb{E}[\log \beta'_c] + \sum_{j=1}^{c-1} \mathbb{E}[\log(1 - \beta'_j)]$ $\mathbb{E}[\log \beta'_c] = \psi(\tilde{u}_c) - \psi(\tilde{u}_c + \tilde{v}_c)$ $\mathbb{E}[\log(1 - \beta'_c)] = \psi(\tilde{v}_c) - \psi(\tilde{u}_c + \tilde{v}_c)$
$\boldsymbol{\pi}'$	Beta	$\mathbb{E}[\log \pi_{tk}] = \mathbb{E}[\log \pi'_{tk}] + \sum_{j=1}^{k-1} \mathbb{E}[\log(1 - \pi'_{tj})]$ $\mathbb{E}[\log \pi'_{tk}] = \psi(\tilde{a}_{tk}) - \psi(\tilde{a}_{tk} + \tilde{b}_{tk})$ $\mathbb{E}[\log(1 - \pi'_{tk})] = \psi(\tilde{b}_{tk}) - \psi(\tilde{a}_{tk} + \tilde{b}_{tk})$

Table 5.1: Expectations of various latent variables used in variational inference. All expectations are taken with respect to the approximating distribution q. The stick-weight expectations $\mathbb{E}[\beta_c]$ and $\mathbb{E}[\pi_{tk}]$ assume that the sticks in each DP are sorted in non-increasing order by size.

variational inference. In the master topic, atoms should be sorted by decreasing w_c, the pseudo-count of topic atoms mapped to the cth cluster. Additionally, within each topic, the atoms should be sorted in decreasing order by v_{tk}, the pseudo-count of data generated from the atom. Due to the restriction on $\tilde{\boldsymbol{\beta}}$, the atoms in the topics cannot be independently reordered in memory. However, it is not necessary to change how the stick counts are *stored* as long as their relative orderings are known and the expectations over the stick weights are computed appropriately.

To put it another way, the purpose of "reordering" a Dirichlet process is not to permute how it is stored in memory *per se*, but rather to compute its stick weight parameters and expectations with respect to a size-ranked ordering of its atoms. For example, the update $\tilde{b}_{tk} = \alpha_0 + \sum_{j=k+1}^{K} v_{tj}$ involves a sum of pseudo-counts at atom indices greater than k. When the atoms are sorted by decreasing pseudo-count, the sum is effectively taken over atoms of *smaller size* than atom k. Likewise, the expected log topic stick weight for the kth atom,

$$\mathbb{E}[\log \pi_{tk}] = \mathbb{E}[\log \pi'_{tk}] + \textstyle\sum_{j=1}^{k-1} \mathbb{E}[\log(1-\pi'_{tj})] \tag{5.22}$$

contains a sum over over indices less than k. Under the optimal ordering of the atoms, the sum is taken over atoms *larger* than atom k. Sorting the atoms is a convenient way of performing these size-aware summations since it puts the atom indices in direct correspondence with the atom sizes. However, the equivalent calculations can be performed *without sorting* by incorporating the rank statistics of the atoms directly into the above formulae. We term this method *simulated reordering.*

Updates that use simulated reordering appear throughout remainder of this chapter and chapter 6. To write them succinctly, some new notation is introduced. Let $\mathbf{n}$ denote a K-length vector. Then, define $\mathrm{r}(\mathbf{n}, j)$ to be the rank of element n_j within $\mathbf{n}$, with ties broken arbitrarily – that is, $\mathrm{r}(\mathbf{n}, j)$ is the new index that n_j would have if $\mathbf{n}$ were sorted in decreasing order. [1] Let $1 \leq k \leq K$ be an index of an element in $\mathbf{n}$. Then, define $\{j >_{\mathbf{n}} k\} \triangleq \{j : r(\mathbf{n}, j) > r(\mathbf{n}, k)\}$ as the set of indices j such that n_j has higher rank than n_k.

As before, let $\tilde{\zeta}_{tk} = \sum_{d,i} \tilde{\theta}_{ditk}$ denote the pseudo-count of data being mapped to the kth atom in topic t. Then, using simulated reordering, the topic stick parameters are updated as

$$\tilde{a}_{tk} = 1 + \tilde{\zeta}_{tk} \tag{5.23}$$

$$\tilde{b}_{tk} = \alpha_0 + \sum_{j >_{\tilde{\zeta}_t} k} \tilde{\zeta}_{tj}. \tag{5.24}$$

The sum in (5.24) is taken over the atoms whose pseudo-counts are larger than $\tilde{\zeta}_{tk}$. This update is equivalent to (5.19) when the atoms in each topic are reordered by

[1]One such definition is: $\mathrm{rank}(\mathbf{n}, j) = \sum_i [n_i > n_j] + [n_i = n_j \text{ and } i > j]$.

decreasing pseudo-count. It could be written equivalently in a longer form as

$$\tilde{b}_{tk} = \alpha_0 + \sum_{j=1}^{K} [r(\tilde{\boldsymbol{\zeta}}_t, j) > r(\tilde{\boldsymbol{\zeta}}_t, k)] \cdot \tilde{\zeta}_{tj}$$

where $[\cdot]$ is the Iverson bracket ($[x] = 1$ if condition x is true and $[x] = 0$ otherwise). The stick-weight expectations must also be computed using simulated reordering. Using the new notation they are,

$$\mathbb{E}[\log \pi_{tk}] = \mathbb{E}[\log \pi'_{tk}] + \sum_{j <_{\tilde{\boldsymbol{\zeta}}_t} k} \mathbb{E}[\log(1 - \pi'_{tj})] \tag{5.25}$$

Here, the sum is effectively taken over the atoms in topic t with pseudo-counts *lower* than $\tilde{\zeta}_{tj}$. The resulting expectation is equal to that in table 5.1 when the topic's atoms are sorted in decreasing order by pseudo-count.

Unlike the topics, the master topic can be reordered in memory. (This process involves permuting both the master topic pseudo-counts and the columns of $\tilde{\boldsymbol{\beta}}$, and then recomputing expectations $\mathbb{E}[\log \beta_c]$.) Therefore, it is not necessary to formulate the master topic updates and expectations in an order-aware manner. Nonetheless, for completeness, order-aware updates are derived for the master topic.

As before, let $\tilde{\xi}_c = \sum_{t=1}^{T} \sum_{k=1}^{K} \tilde{\beta}_{kc}$ pseudo-count of topic atoms mapped to cluster c. The stick-weight parameters and expectations are updated as follows:

$$\tilde{u}_c = 1 + \tilde{\xi}_c \tag{5.26}$$

$$\tilde{v}_c = \gamma + \sum_{j >_{\tilde{\boldsymbol{\xi}}} c} \tilde{\xi}_j. \tag{5.27}$$

Simiarly, the order-aware expectations as computed as

$$\mathbb{E}[\log \beta_c] = \mathbb{E}[\log \beta'_c] + \sum_{j <_{\tilde{\boldsymbol{\xi}}} c} \mathbb{E}[\log(1 - \beta'_j)] \tag{5.28}$$

As in the formulae for the topic stick weights, the update for the cth stick in the master topic effectively takes a sum of clusters *smaller* than cluster c, and the expected log stick weights take a sum over clusters *larger* than cluster c.

5.4.6 Algorithmic Structure

Variational inference is a form of coordinate ascent: It cycles through each set of latent variables in IWTM and updates its variational parameters, holding the others fixed. Each iteration is divided into two phases. In the first, the document-level variational parameters are estimated for each document, holding the shared components of the model fixed. Within each document, this is accomplished by repeatedly updating the topic atom assignments and the topic weights until convergence. In the second phase, the document-level variational parameters are fixed and used to update the shared model components: the cluster parameters, the topic atom weights, the master topic, and the topic atom to cluster mappings. By construction, each update is guaranteed to increase (or maintain the current value of) the ELBO objective.

The first stage requires estimating the variational parameters of all N_{doc} documents, which can be computationally expensive. In general, the computational effort required to do so grows linearly with the dataset size. Fortunately, the updates for the document-level parameters only depend on the shared model components, not the parameters of other documents. Such updates can therefore be parallelized by distributing them over up to N_{doc} separate cores or machines. The ability to be parallelized gives variational inference a significant advantage the collapsed Gibbs sampler from chapter 4, which must resample variables one at a time in sequence. Although a single-core implementation of variational inference is already much faster than Gibbs sampling (as demonstrated in the next section), the former can be sped up further by giving it additional computational resourcess.

5.4.7 Initialization

Because the update for each set of variational parameters depends on the values of the others, all parameters must be set to some initial values at the start of inference. In all experiments in this dissertation, the variational parameters are initialized as follows:

Cluster parameters The cluster means are set to random data points taken from the training corpus, and the cluster precisions are set to those of the prior ($\tilde{\chi}_{1c} = \chi_1$, $\tilde{\chi}_{2c} = \chi_2, \tilde{\nu}_c = \beta_0$). The initial number of clusters in the model is C, the truncation limit of the master topic.

Stick weights The pseudo-counts for the topic sticks are set to small random values, $w_{tk} \sim \text{Unif}(0, 0.1)$. The master topic stick weights are set to the prior by setting $v_c = 0$ for all c. There are initially T sticks in each topic and C sticks in the master topic.

Topic atom to cluster mappings The topic atom to cluster mappings are initialized so each cluster has two atoms per topic primarily assigned to it. For each atom, the mapping parameters are smoothed so that there is a non-zero probability of being assigned to other, close sufficiently clusters.

5.5 Experiments: Scene Classification

Variational inference for IWTM is evaluated on 13-scene image classification task described in the previous chapter.

As before, IWTM is compared against LDA using bags of words formed through an offline clustering step. In the first baseline, LDA+KMEANS, bags of words are formed in the usual manner using K-means clustering. That is, K-means is

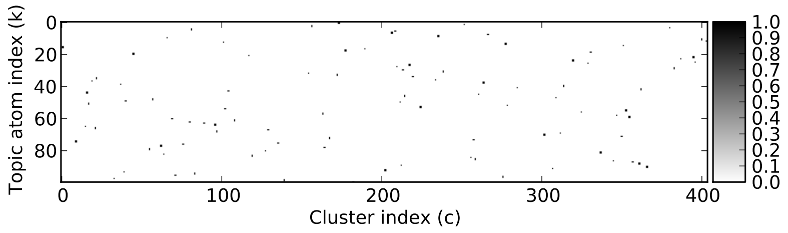


Figure 5.1: Topic atom to cluster mappings $\tilde{\boldsymbol{\beta}} = \{\tilde{\beta}_{kc}\}$ learned by IWTM on a small dataset of images. Each row contains the estimated posterior cluster assignment for a different topic atom (only the first 100 atoms are shown). Most atoms map to a single cluster with probability 1, with all other entries in the row numerically indistinguishable from zero. Such structure allows $\tilde{\boldsymbol{\beta}}$ to be stored in an efficient sparse-matrix format and motivates the sharing of atom-to-cluster mappings between topics (see section 5.4.3).

first trained on the descriptors from the training set, and then all images' descriptors are quantized by mapping each one to the index of its closest cluster. The vocabulary size K is a fixed parameter in this method. These experiments sweep over a range of values $K \in [125, 8000]$, training separate K-means and LDA models for each vocabulary size.

In the second baseline, LDA+GDPMM, bags of words are formed by quantizing descriptors against a Gaussian Dirichlet Process Mixture Model (GDPMM) with spherical covariance clusters. Like IWTM, GDPMM is a non-parametric model that infers the number of clusters to use based on the size and complexity of the data. However, GDPMM is merely a clustering model – it takes the place of K-means in the modeling pipeline. To form the bag of words representations for the images, a GDPMM is trained (using variational inference) on the descriptors from the training set. Then, the descriptors for all images are quantized by assigning each one to the cluster in the model with highest predictive density. Although GDPMM is a probabilistic model that provides "soft" cluster assignments for the data, the process of quantizing the data throws away this information.

Each topic model is trained on a set of $N_{\text{train}} = \{325, 650, 1300\}$ images and then used to infer the posterior mean topic weights for both the training set and $N_{\text{test}} = 2859$ test images. The training set weights are used to train an SVM classifier, which then predicts the labels of the test images. Out of several kernels tested, both LDA and IWTM perform best with a histogram intersection kernel [8]. The SVM soft-margin parameter C is selected using 10-fold cross validation.

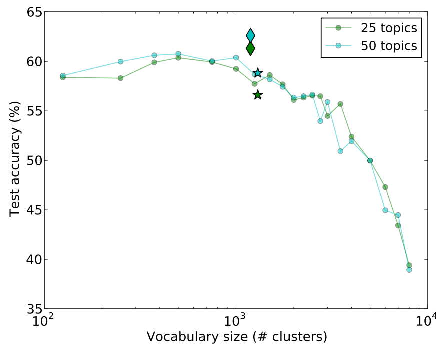
For each trial, IWTM and LDA models are trained for 50 iterations of variational inference.[2] LDA's smoothing parameters are set to $\beta = \alpha = 1/T$; for IWTM, DP concentrations are $\alpha_0 = 1$ and $\gamma = 10$. (Results do not change significantly when any of these parameters are varied within reasonable ranges.) The cluster hyperparameters are set so that $\chi_1 = 32N_{\text{dim}}$ and $\mathbb{E}[\lambda_c] \equiv \chi_1\chi_2 = \text{precision}(\mathbf{x})$, $\mathbf{m} = \bar{\mathbf{x}}$, and $\beta_0 = 1$. These settings are the same as those used in Gibbs sampling experiments in the previous chapter, with the exception of χ_1. The choice of χ_1 is motivated by the fact that spherical covariance clusters are being used to model high-dimensional data. A large value that scales with the data dimension pressures the model away from discovering many tiny clusters that do not generalize well.

5.5.1 Results

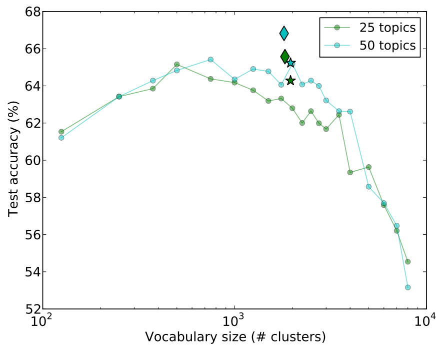
Figure 5.2 shows detailed comparisons of all three methods broken down over different numbers of topics and amounts of training data. Unsurprisingly, both models performed best with a large amount of training data (N_{train}=1300 images) and many topics ($T = 50$). IWTM had the highest accuracy, 71.28%, with a mean vocabulary size of 2784. LDA+Kmeans achieved slightly lower accuracy, 70.18%, at $K = 750$, with close peaks in performance around $K = \{1750, 3000\}$. (For comparison, Fei-Fei & Perona [22] report getting 65.2% accuracy with an LDA-like model using the same amount of training data.)

This pattern is repeated throughout the results. In each experimental setting,

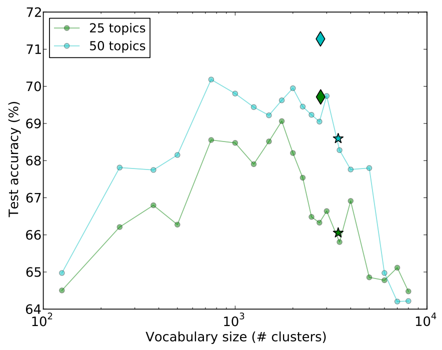
[2]These experiments use the LDA implementation in the open-source `gensim` package.



(a) $n_{\text{train}} = 325$



(b) $n_{\text{train}} = 650$



(c) $n_{\text{train}} = 1300$

Figure 5.2: Mean classification accuracy of IWTM (diamonds), LDA+KMEANS (line series), and LDA+GDPMM (stars) on the 13-scene task. Results are broken down over different numbers of topics and training images (n_{train}). IWTM outperforms all LDA models in each setting, regardless of the vocabulary size chosen (in LDA+KMEANS) or learned (in LDA+GDPMM). Note that LDA+KMEANS suffers when its vocabulary is too small or too large, following typical underfitting-overfitting curves. Overall, such results support the approach of IWTM: training the vocabulary jointly with the topic model and *inferring* its size based on the data.

IWTM met or exceeded the performance of the best LDA+KMEANS trained with any vocabulary size. In each case, IWTM's vocabulary size was near (although slightly larger than) the optimal size for LDA+KMEANS. The key difference is that IWTM automatically *learns* its vocabulary as part of inference, whereas the best value for K-means is determined by sweeping over different values and training many separate models.

Interestingly, LDA+GDPMM performed worst of all methods, achieving its best accuracy of 68.59% with inferred vocabulary size 3445. The GDPMMs consistently inferred vocabulary sizes that were close to those of IWTM. However, when LDA was trained on bags of words from the GDPMMs, it consistently performed worse than IWTM. This phenomenon suggests that converting image features into the bag of words representation throws away information about the data that is useful for the topic model.

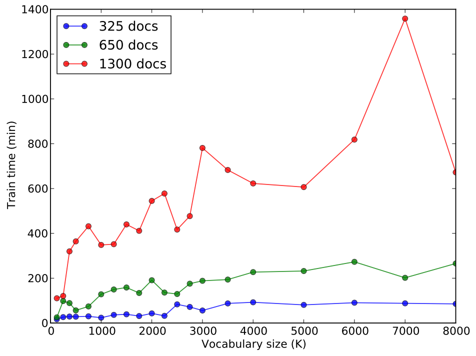
The full profile of LDA+KMEANS results in figure 5.2 suggests a larger trend: that the effect of vocabulary size in topic models follows a typical underfitting-overfitting curve. Performance suffers if too few or too many clusters are used to fit the data. This phenomenon demonstrates the necessity of the tuning vocabulary size through one method or another. However, the results also indicate that the optimal vocabulary size varies with respect to training set size and, to a lesser extent, the number of topics in the model. Furthermore, the performance of the system does not always vary as a smooth function of the vocabulary size. These facts indicate that simple heuristics to pick the vocabulary size – like always using a single value that seems "big enough" – may be ineffective. Instead, LDA+KMEANS will generally need to be evaluated with multiple values of K to maximize performance.

5.5.2 Runtime Speed

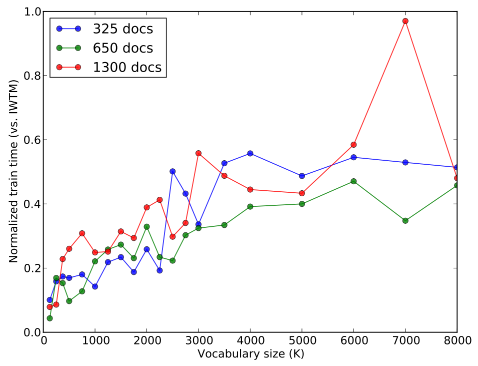
To evaluate the efficiency of IWTM's variational inference procedure, training times for both it and the LDA+KMEANS baseline were recorded on a standard reference machine. All software was run in a single thread on one processor; however, all three components – KMEANS, variational inference for LDA, and variational inference for IWTM – can easily be parallelized by splitting up the dataset over multiple cores or machines. The speed-ups of parallelization should be the same for all methods, so the single-processor runtimes are a fair comparison of the methods' relative efficiencies.

KMEANS is implemented using Lloyd's algorithm [38]. The implementation has been optimized to be as fast as possible. For example, distances are computed for blocks of data at a time using highly-tuned matrix multiplication operations. Although much work has been done to improve the efficiency of KMEANS over Lloyd's algorithm – for example, by storing the cluster centers in a kd-tree – these variants are most effective in low to moderate dimensionality (i.e. up to about 50 dimensions) and can actually be slower than Lloyd's algorithm in higher dimension [25].

Figure 5.3 shows the training times with different numbers of training documents and, for LDA+KMEANS different vocabulary sizes. In general, training times in both methods increase with the dataset size, and LDA+KMEANS generally takes longer with larger vocabulary sizes. Figure 5.3(b) shows the training times normalized against those of IWTM. A single run of LDA+KMEANS takes approximately one half to one fifth the time of IWTM on the same dataset, depending on the vocabulary size used in the former model. However, the former provides no mechanism for optimizing the vocabulary size. When LDA+KMEANS is trained multiple times with different vocabulary sizes, the total computation time is comparable to that of IWTM.



(a) Train times



(b) Normalized train times

Figure 5.3: (Left) Training times for LDA+KMEANS with different vocabulary sizes. (Right) The same training times normalized against those of IWTM on the same datasets. A single run of LDA+KMeans is 2-5 times faster than a single run of IWTM. However, after sweeping over multiple vocabulary sizes, both methods require approximately the same total computation time.

5.6 Discussion

Variational inference for IWTM is much faster than the Gibbs sampler: On the 13-scene task, it takes about 20 hours to train with 1300 document using variational inference, compared to about 14 days with Gibbs sampling. (Naturally, the training times depend on the number of training iterations. Both methods can be made faster by training for fewer iterations, with some corresponding loss in performance.)

The results show that vocabulary size does have a significant effect on the performance of topic models, and that there is a need to tune the vocabulary size through one manner or another. Indeed, in each experimental setting, the effect of vocabulary size on the performance of LDA+KMEANS was at least as large as that of the number of topics in the model. This is significant: Although the literature contains a number of non-parametric models that infer the number of topics in the model, IWTM is the first topic model that infers the vocabulary size. It is thus a valuable tool for topic modeling in digital media domains.

Finally, although IWTM is a more complex model than LDA, the variational inference procedure makes the required computation time competitive with the baseline LDA+KMEANS method. One reason variational inference is so fast is that the approximating distribution shares topic atom to cluster mappings between all topics. This technique – using a single mapping between the atoms of the first- and second-level DPs – may be useful in variational inference procedures for other HDP-based models. Sharing of mappings is made possible by *simulated reordering*, which enables the DPs to be operated over in optimally ordered forms without permuting their representations in memory.

5.6.1 Decoupled Topic Model Based Classification

The 13-scene experiments in the last section use *decoupled topic model based classification*, where a topic model is trained in a fully unsupervised manner and, using document representations derived from it, an external classifier is used to predict document labels. The experimental results implicitly validate such an approach. Although the topic models do not use the document labels in any way, they automatically infer structure in the corpus that is useful for image classification. In contrast, the Gibbs sampling implementations in chapter 4 incorporated supervision into the generative model and performed *worse* than the method in this chapter. With 100 training documents per class, the combination of IWTM and SVM had a mean accuracy of 71.5% on 13-scene, compared to 68.5% with the supervised Gibbs sampling model. The improved performance stems from the fact that SVM generalizes better than the generative classifier and, potentially, that variational inference achieves better fits of the data than the Gibbs sampler.

The power and flexibility of decoupled topic model based classification are explored further in the next section, which extends the method to semi-supervised learning.

5.7 Experiments: Semi-Supervised Classification

This section extends decoupled topic model based classification to semi-supervised learning, so that additional, unlabeled documents can be used to make labeled documents more effective. The proposed semi-supervised classification method is first described. Then, the method is evaluated by on the 13-scene task, where it is used to train IWTM on a combination of labeled and unlabeled documents.

5.7.1 Leveraging Unlabeled Documents

In general, semi-supervised classification is the practice of using *unlabeled* examples, together with labeled examples, to improve performance on a classification task [64]. Methods that can leverage unlabeled examples are beneficial because such examples can typically be acquired easily and at low cost. In contrast, getting new labeled examples requires effort by a human annotator, which can be time-consuming and expensive.

Decoupled topic model based classification can be extended to the semi-supervised case in a number of ways, depending on whether unlabeled documents are incorporated into the topic model, the classifier, or both. The method proposed here is to train the topic model on all available documents (both labeled and unlabeled) and then, using the document representations generated by the topic model, train a standard SVM on the labeled documents. This method is identical to that used in the 13-scene experiments in section 5.5, except that the topic model is trained on a combination of labeled and unlabeled documents.

Another semi-supervised approach to classification with topic models would be train a semi-supervised classifier, like transductive SVM [27] or label propagation [65], on both labeled and unlabeled documents. Such a method is not considered in this dissertation for two reasons. First, changing a single component of the document classification method makes it possible to isolate the effect of the topic model, which is the focus of this dissertation. Second, the standard SVM is simpler to use than semi-supervised classifiers, and its performance is actually difficult to beat. For example, early experiments showed that label propagation performed significantly worse than the standard SVM on the 13-scene task, even when the former was given a large number of unlabeled documents. Despite the simplicity of the proposed method, the experiments in the following subsection demonstrate that it is effective.

Why should the addition of unlabeled examples help in such a method? Although a variety of semi-supervised classification methods exist (see [64] for a survey), most share, at a high level, the same common idea: unlabeled examples contain information about the statistical structure of the data that can be exploited to make labeled examples more effective. The idea here is the same: Training the topic model on more documents allows it to discover topics that better capture the semantics of the documents. By improving the document representation going into the classifier, the classification accuracy on the supervised task should improve.

5.7.2 Experimental Method

The proposed semi-supervised learning method is evaluated on the 13-scene dataset. The experimental set-up is similar to that of section 5.5, except that IWTM is trained on a combination of labeled and unlabeled documents. Specifically, each experimental trial follows the following procedure. First, IWTM is trained on a combination of labeled and unlabeled documents for 50 iterations of variational inference. For simplicity, models are given $T = 25$ topics; the hyperparameter settings and initialization are the same as in section 5.5. Second, the model is used to infer the mean topic weights for the labeled documents, which are then used to train an SVM classifier with a histogram intersection kernel. Finally, at test time, the topic model is used to infer the mean topic weights for the new documents, and those representations are fed into the classifier, which predicts their class labels.

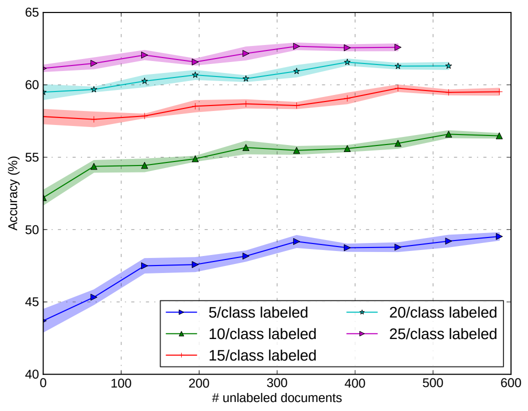


Figure 5.4: Semi-supervised classification results on 13-scene. Training the topic model on additional, unlabeled images increases classification accuracy. The classifier does not use the unlabeled examples directly; rather, it benefits from the improved document representations produced by the topic model. Unlabeled examples are most beneficial when labeled data are few, but provide some improvement regardless of the amount of labeled data.

To see the effect that unlabeled data has on the system's performance, models are trained with different proportions of labeled and unlabeled data. Specifically, the total number of image in the topic model's training set, N_{total} is varied between 65 and 650. Of these, $N_L = \{5, 10, 15, 20, 25\}$ image per class are selected to be the labeled examples that are used to train the SVM classifier. Classification accuracy is evaluated using the 2589 images in the 13-scene test set. Each experimental setting is repeated over a number of trials: Ten runs of IWTM are performed with each setting of N_{total}, and each set of N_L labeled examples is selected at random 10 times from each model.

5.7.3 Results

Figure 5.4 shows the test accuracy of IWTM with the SVM as the numbers of labeled and unlabeled documents are varied. Training the topic model on extra unlabeled images is clearly beneficial: Doing so almost always improves the classification accuracy of the system, and the benefit increases with the number of unlabeled examples used.

The unlabeled examples have the greatest benefit when labeled data are few, and have a diminishing effect as the labeled data increases. With five labeled images per class, adding 585 unlabeled images improves the mean classification accuracy by 5.8% absolute (from 43.7% to 49.5%), while with 10 labeled images per class, the accuracy is improved by 4.3% absolute (52.2% to 56.5%). The diminishing effect of unlabeled data is not surprising considering that the marginal benefit of more labeled data also decreases. For example, moving from 5 to 10 labeled images per class increases classification accuracy by approximately 7% absolute, while the difference between 10 and 15 images per class is on average about 3.5% absolute.

Overall, such results are significant for three reasons. First and foremost, they show that the proposed method is useful: In real-world settings where limited labeled data are available, unlabeled examples can be leveraged to achieve higher classification accuracy. Second, the results highlight the topic model's ability to learn semantic descriptions of documents in an unsupervised manner. The SVM, which makes the class predictions, does not use the unlabeled documents in any way. Unlabeled documents improve its performance only because they help the topic model infer better semantic descriptions of the labeled documents. Third, more broadly, the success of the method demonstrates the extensibility of decoupled topic model based classification. Semi-supervised learning is only one possible extension. Chapter 7 extends it to active learning, where the classifier actively selects new documents to be both labeled and added to the model's training corpus.

5.8 Conclusions

This chapter developed a mean-field variational inference procedure for IWTM, a novel contribution of this dissertation. In general, variational inference poses inference as an optimization problem and tries to find an approximating distribution that is as close as possible to the true posterior in KL divergence. Its main advantage over Gibbs sampling is its speed: Using it to train IWTM on the full 13-scene dataset takes only a matter of hours where it previously took *weeks*. Moreover, although not explored in this chapter, variational inference can easily be parallelized. Doing so only requires dividing the corpus among multiple cores or machines and distributing the document-level inference tasks among them.

Despite such qualities, variational inference is still difficult to scale to large datasets. Each iteration of the algorithm requires that local inference be run on all documents before the the global latent variables can be updated. In general, the computational effort required to do so scales linearly with the dataset size. When there are many training documents, the time required to perform even a single iteration will be prohibitive. The next chapter develops a *stochastic* variational inference (SVI) method for IWTM that addresses such an issue and allows variational inference to scale further. SVI applies ideas from stochastic optimization to the task of variational inference, using small mini-batches of the training corpus to make small, easily-computable updates to the model state.

Chapter 6

Stochastic Variational Inference

The previous chapter demonstrated that variational inference for IWTM is much faster than inference with a collapsed Gibbs sampler. Variational inference makes the training time of IWTM comparable to LDA with visual vocabularies trained with K-means, making it practical to use for the 13-scene task. However, variational inference is still too slow to scale to much larger datasets. In this chapter, a *stochastic variational inference* [26] algorithm is developed to scale IWTM further. Using the method, IWTM is trained on a subset of the SUN dataset [61] with about one and a half times the number of images and *seven times* the number of image descriptors as 13-scene. Applying IWTM to SUN showcases its ability to learn semantic descriptions of documents by automatically grouping semantically-related image patches into topics. In addition to scaling the model to larger data, stochastic variational inference is faster and produces better fits of the data than the coordinate-ascent method described in the previous chapter.

6.1 Approach

The primary impediment to scaling the variational inference procedure from chapter 5 is that it is a *batch-style* algorithm: Local inference must be performed on every document in the corpus before the global latent variables of the model can be updated. Such an algorithm is unappealing for two reasons. First, the computational cost of each iteration of inference scales linearly with the dataset size. When there are many training documents, it will take a prohibitively long time to perform even a single iteration. Second, batch-style inference is particularly inefficient at the beginning of training, when the shared components of the model have been initialized to random values that do not explain the data well. The local parameters of all documents must be estimated based on these initial values before they are ever updated. In IWTM, the early iterations of training are particularly expensive because the model is initialized with more clusters than it will eventually use.

Motivated by such issues, *stochastic variational inference* was recently developed [26, 59]. At a high level, stochastic variational inference (SVI) applies ideas from stochastic optimization to the framework of variational inference. In general, the goal of variational inference is to maximize the ELBO objective with respect to the parameters of the posterior approximation. Whereas "batch" variational inference maximizes the ELBO by coordinate ascent, SVI uses a variant of stochastic natural gradient ascent to maximize it.

SVI works by repeatedly sampling a "mini-batch" of documents, using them to estimate the natural gradient of the ELBO with respect to the variational parameters, and moving the parameters a small step in that direction. SVI has two key benefits over coordinate-ascent variational inference. First, the mini-batch is typically chosen to be a small fraction of the training corpus, and a single iteration of SVI is therefore much faster than an iteration of coordinate ascent. Correspondingly, SVI begins updating the model much sooner at the beginning of training.

Second, natural gradients are estimated on a mini-batch of documents, rather than computed exactly. The noise in the gradients allows SVI to escape poor local optima that coordinate ascent cannot. Both of these benefits are experimentally demonstrated later in the chapter.

Rather than following the standard gradient of the ELBO, SVI moves in the direction of the *natural gradient*. The natural gradient is analogous to the standard (Euclidean) gradient, except that it takes into account the information geometry of the parameter space [4]. Natural gradients have two benefits over Euclidean gradients. First, when performing stochastic optimization to fit parameters of probability distributions, using natural gradients leads to faster convergence rates [26, 5]. Second, in fully conjugate-exponential family models like IWTM, the natural gradients of the ELBO objective are far simpler and easier to compute than Euclidean gradients.

The remainder of this chapter is structured as follows. The next section reviews the theory underlying stochastic variational inference and derives the method for a general class of conjugate exponential-family models. Section 6.3 uses the general theory to derive a stochastic variational inference IWTM, which is a novel contribution of this dissertation. SVI for IWTM then evaluated in two sets of experiments. In section 6.4, the method is applied to the 13-scene dataset for the purposes of comparing it to coordinate-ascent variational inference. Then, in section 6.5, SVI is used to apply IWTM to a scene classification dataset almost seven times the size of 13-scene. In that task, the image representations learned by IWTM are found to perform as well or better than a number of features from the computer vision literature. Finally, section 6.6 concludes and discusses ideas for future work.

6.2 Background

This section briefly reviews the theory underlying stochastic variational inference. Some material presented here is largely a summary of seminal papers by Hoffman et al. [26] and Wang, Paisley, and Blei [59], and borrows some of their notation.

6.2.1 Natural Gradients

In general, the notion of a function's gradient is implicitly defined by the Riemannian metric used to measure distances between points in the function's domain. In the standard (Euclidean) gradient, this metric is defined by Euclidean distance. When the domain being considered is the parameter space of a probability distribution, the Euclidean metric is not appropriate, because the distance between two points does not reflect the difference in the distributions that they define. This fact can be observed by considering some simple examples. As pointed out by Hoffman et al. [26], the distributions $\mathcal{N}(10, 10000)$ and $\mathcal{N}(0, 10000)$ are almost indistinguishable, but the Euclidean distance between their parameters is 10. On the other hand, the distributions $\mathcal{N}(0, 0.01)$ and $\mathcal{N}(0.1, 0.01)$ have very little overlap, but the Euclidean distance between their parameter vectors is 0.1. A further deficiency of the Euclidean metric is that it is sensitive to the chosen parameterization of the distributions. If the distributions above were represented by the natural parameters of their exponential family, $\langle \mu/\sigma^2,\, -1/(2\sigma^2)\rangle$, the Euclidean distance between them would be different.

The *natural gradient* is the gradient that results when the distance between points in parameter space is given a more appropriate definition, namely, the symmetrized KL divergence between the distributions they define. Amari [5] showed that it can be computed by making an adjustment to the standard Euclidean gradient. In general, the gradient under an arbitrary Riemannian metric G can be computed by premultiplying by the metric's inverse,

$$\hat{\nabla}_\lambda f = G(\lambda)^{-1} \cdot \nabla_\lambda f(\lambda). \tag{6.1}$$

When the distance between λ and another point is considered to be the symmetrized KL divergence between the distributions they define, $G(\lambda)$ is the Fisher information matrix evaluated at λ [26]. The resulting Riemannian gradient is called the natural gradient. Unlike the Euclidean gradient, the natural gradient is invariant to parameterization. When used to fit parameters of probability distributions, natural gradients lead to faster convergence rates than Euclidean gradients [5].

6.2.2 A General Conjugate Exponential-Family Model

Deriving stochastic variational inference requires a more detailed definition of variational inference than that given in chapter 5. In particular, stochastic variational inference assumes that the model falls into a general class of conjugate exponential-family models. Doing so allows the ELBO objective to be written down in a more detailed manner, which facilitates the calculation of its natural gradients. SVI makes the following assumptions about the model:

Data There are N groups of data $\mathbf{x} = \{\mathbf{x}_d\}_{d=1}^{N}$, each containing N_d observations $\mathbf{x}_d = \{x_{di}\}_{i=1}^{N_d}$.

Global latent variables The model has "global" latent variables $\boldsymbol{\beta}$ that are used to model the entire dataset. The prior over the global latent variables is controlled by hyperparameters α.

Local latent variables The model contains "local" latent variables $\boldsymbol{z}$. Each data group has a corresponding group of such variables; that is, $\boldsymbol{z} = \{\boldsymbol{z}_d\}_{d=1}^{N}$.

Factorization The relationship between the global latent variables, local latent variables, and the data is formalized by the following factorization of the model:

$$p(\mathbf{x}, \boldsymbol{z}, \beta \,|\, \alpha) = p(\beta \,|\, \alpha) \prod_{d=1}^{N} p(\boldsymbol{z}_d \,|\, \beta) p(x_d \,|\, \boldsymbol{z}_d, \beta).$$

The global variables β play a role in generating the all data, while each local variable $\boldsymbol{z}_i$ is only responsible for generating the ith group of data.

Conjugate Exponential Families The factors $p(\beta \,|\, \alpha)$, $p(\boldsymbol{z}_i \,|\, \beta)$, and $p(x_i \,|\, \boldsymbol{z}_i, \beta)$ are assumed to be in the exponential family. Further, the distributions for the global and local variables are assumed to be conjugate [18], as are the distributions for the local variables and the data.

The assumptions above describe a variety of Bayesian hierarchical models, including but not limited to Bayesian mixture of Gaussians [10], Latent Dirichlet Allocation [14], and the Infinite-Word Topic Model. In IWTM, the global latent variables are the cluster parameters, the master topic stick weights, the topics sticks, and the mapping between topic atoms and clusters. The local latent variables, defined at the document level, are the topic atom assignments and topic weights. Although IWTM has multiple levels of global and local latent variables, the forthcoming analysis will lump all of the global variables together in β and all of the local variables for the ith document into z_i.

6.2.3 Complete Conditionals

Because the model contains conjugate exponential-family distributions, its complete conditionals are also in the exponential family [26]:

$$p(\beta \,|\, \mathbf{x}, \boldsymbol{z}, \alpha) = h(\beta) \exp\{\eta_g(\mathbf{x}, \boldsymbol{z}, \alpha)^\top t(\beta) - a_g(\eta_g(\mathbf{x}, \boldsymbol{z}, \alpha))\} \tag{6.2}$$

$$p(z_{di} \,|\, \mathbf{x}_d, \boldsymbol{z}_{d\neg i}, \beta) = h(z_{di}) \exp\{\eta_\ell(\mathbf{x}_d, \boldsymbol{z}_{d\neg i}, \beta)^\top t(z_{di}) - a_\ell(\eta_\ell(\mathbf{x}_d, \boldsymbol{z}_{d\neg i}, \beta))\} \tag{6.3}$$

In the terminology of exponential families, the $h(\cdot)$ is called the base measure, $a(\cdot)$ is the log normalizer, $\eta(\cdot)$ is the natural parameter, $t(\cdot)$ is the sufficient statistic [18]. (The first two are scalar functions, and the last two are vector-valued.) The natural parameters and log normalizer play a key role in stochastic variational inference. Note that the natural parameters are a function of the variables being conditioned on, and the log normalizer is a function of the natural parameters. The subscript on these functions is used to differentiate the global (g) and local (ℓ) latent variables.

6.2.4 Form of the ELBO

As discussed in chapter 5, variational inference approximates the true model posterior with a fully-factored approximating distribution. For the general model under consideration, it is

$$q(\beta, \boldsymbol{z}) = q(\beta \,|\, \lambda) \prod_{d=1}^{N} \prod_{i=1}^{N_d} q(z_{di} \,|\, \phi_{di}). \tag{6.4}$$

Strictly speaking, variational inference does not prescribe the parametric forms of q's factors. Here, they are assumed to be in the same exponential families as their complete conditionals in the model. For example, $q(z_{di} \,|\, \phi_{di})$ is in the same exponential family as $p(z_{di} \,|\, z_{d,\neg i}, \beta, \mathbf{x}_d)$. This is done for two reasons. First, these families form the optimal approximating distribution out of all distributions that factor as

(6.4) [10]. Second, endowing q with such structure leads to simpler variational inference procedures. In coordinate-ascent variational inference, it produces closed-form update rules. In stochastic variational inference, it enables natural gradients to be computed in closed form [26].

The goal of variational inference is to minimize the KL divergence between an approximating distribution q and the true model posterior – or, equivalently, to maximize the ELBO objective with respect to q's parameters. To do so, it is necessary to isolate the terms of the ELBO that depend on each set of variational parameters. The parts that depend on λ are

$$\begin{aligned}\mathcal{L}(\lambda) &= \mathbb{E}_q[\log p(\beta \,|\, \mathbf{x}, \boldsymbol{z})] - \mathbb{E}_q[\log q(\beta \,|\, \lambda) + \text{const} \\ &= \mathbb{E}_q[\eta_g(x, z, \alpha)]^\top \nabla_\lambda a_g(\lambda) - \lambda^\top \nabla_\lambda a_g(\lambda) + \text{const} \qquad (6.5)\end{aligned}$$

where const denotes a constant that does not depend on λ. Likewise, the parts of the ELBO that depend on each local variational parameter are

$$\begin{aligned}\mathcal{L}(\phi_{dj}) &= \mathbb{E}_q[\log p(z_{di} \,|\, \mathbf{x}_d, \boldsymbol{z}_{d\neg i}, \beta)] - \mathbb{E}_q[\log q(z_{nj} \,|\, \phi_{nj}) + \text{const} \\ &= \mathbb{E}_q[\eta_\ell(\mathbf{x}_d, \boldsymbol{z}_{d\neg i}, \beta)]^\top \nabla_{\phi_{nj}} a_\ell(\phi_{nj}) - \phi_{nj}^\top \nabla_{\phi_{nj}} a_g(\phi_{nj}) + \text{const}. \qquad (6.6)\end{aligned}$$

These equations use the property that, for exponential family distributions, the expectation of the sufficient statistic is the gradient of the log normalizer with respect to the natural parameter, $\mathbb{E}_q[t(\beta)] = \nabla_\lambda a_g(\lambda)$ and $\mathbb{E}_q[t(z_{nj})] = \nabla_{\phi_{nj}} a_\ell(\phi_{nj})$ [18].

The gradients of the ELBO are the starting point for both coordinate-ascent and stochastic variational inference. Taking gradients with respect to each variational parameters yields,

$$\nabla_\lambda \mathcal{L} = \nabla_\lambda^2 a_g(\lambda) \left(\mathbb{E}_q[\eta_g(x, z, \alpha)] - \lambda\right) \qquad (6.7)$$

$$\nabla_{\phi_{nj}} \mathcal{L} = \nabla_{\phi_{nj}}^2 a_\ell(\phi_{nj}) \left(\mathbb{E}_q[\eta_\ell(x_n, z_{n,\neg j}, \beta)] - \phi_{nj}\right). \qquad (6.8)$$

The next subsection shows how the Euclidean gradients of the ELBO are used to compute the natural gradients of it.

6.2.5 Natural Gradients of the ELBO

The natural gradient of the ELBO is formed by plugging the formula for the Euclidean gradient into the definition in (6.1),

$$\hat{\nabla}_{\lambda}\mathcal{L} = G(\lambda)^{-1} \cdot \nabla^2_{\lambda} a_g(\lambda) \left(\mathbb{E}_q[\eta_g(x, z, \alpha)] - \lambda\right). \tag{6.9}$$

As discussed in the section 6.2.1, $G(\lambda)$ is the Fisher information matrix, the Riemannian metric that defines the natural gradient. A benefit of having variational distributions in the exponential family is that the expression of the natural gradient reduces considerably. For such distributions, the Fisher information matrix is equivalent to the Hessian of the log normalizer, i.e. $G(\lambda) = \nabla^2_{\lambda} a_g(\lambda)$ [26, 18]. Therefore, the natural gradient simplifies to the following [26]:

$$\hat{\nabla}_{\lambda}\mathcal{L} = \mathbb{E}_q[\eta_g(x, z, \alpha)] - \lambda. \tag{6.10}$$

Compared to Euclidean gradients, the natural gradients have a simpler functional form and are far simpler to compute. However, in another sense, calculating the natural gradient is still inefficient: The expectation on the right side of (6.10) is a function of all the data in the corpus and local latent variables in the model. Stochastic variational inference addresses this last inefficiency by estimating the natural gradient using a small sample of the dataset. The following subsection completes the derivation of stochastic variational inference.

6.2.6 Stochastic Optimization with Natural Gradients

Stochastic variational inference is essentially a *stochastic natural gradient ascent* procedure. It is an iterative algorithm that maximizes the ELBO by repeatedly moving each global variational parameter a step in the direction of its (estimated) natural gradient. It is stochastic in the sense that the natural gradients are estimated noisily, using small, random samples or *mini-batches* of the training set. This subsection derives SVI by first giving a natural gradient ascent algorithm that uses exact natural gradients, and then showing how to estimate the natural gradients using a mini-batch.

The natural gradients are the direction of greatest change in the ELBO with respect to each variational parameter, and thus iteratively following the natural gradients locally maximizes the ELBO with respect to those parameters. This procedure is *natural gradient ascent*, analogous to standard gradient ascent. In each iteration, a new value for the global variational parameter is made by taking a step away from current value and in the direction of the natural gradient, as

$$\begin{aligned}\lambda^{(t+1)} &= \lambda^{(t)} + \rho_t \hat{\nabla}_\lambda \mathcal{L} \\ &= (1-\rho_t)\lambda^{(t)} + \rho_t \mathbb{E}_q[\eta_g(\mathbf{x}, \boldsymbol{z}, \alpha)] \qquad (6.11)\end{aligned}$$

where $\rho_t > 0$ is the learning rate, which controls the step size made in iteration t. Equation 6.11 says that each natural gradient ascent step is equivalent to setting the new value of the variational parameter to a weighted combination of its current value and the expected value of the natural parameter of the complete conditional.

As defined in (6.11), natural gradient ascent follows the exact natural gradient, which is expensive to compute. Stochastic variational inference instead approximates the natural gradients on a randomly selected mini-batch of data. Each update resembles (6.11), except the natural gradient is estimated on a mini-batch

M. First, $|M|$ data are sampled uniformly without replacement from the N training data. Second, local variational parameters are estimated the data in the mini-batch. Third, the expected natural parameter is estimated using only the data and latent variables from the mini-batch, as

$$\mathbb{E}_q[\eta_g(x, z, \alpha)] \approx \mathbb{E}_q[\eta_g(\mathbf{x}_M^{(N/|M|)}, \boldsymbol{z}_M^{(N/|M|)}, \alpha)], \tag{6.12}$$

where $\mathbf{x}_M^{(N/|M|)}$ and $\boldsymbol{z}_M^{(N/|M|)}$ denote the data and the local latent variables in the mini-batch replicated $N/|M|$ times [26]. Note that the right-hand side of (6.12) depends on M, which is a random quantity. Its expected value, averaging over the random mini-batch is

$$\mathbb{E}_M \left[\mathbb{E}_q[\eta_g(\mathbf{x}_M^{(N/|M|)}, \boldsymbol{z}_M^{(N/|M|)}, \alpha)]\right] = \mathbb{E}_q[\eta_g(x, z, \alpha)]. \tag{6.13}$$

Thus, stochastic variational inference moves the variational parameters along noisy natural gradients whose *expectations* are the true natural gradients. In general, the variance of the natural gradient estimates depends on the size of the mini-batch used: The fewer the data, the higher the variance. When the mini-batch is the entire dataset ($|M| = N$), the natural gradient estimates become exact.

6.2.7 Connection to Coordinate Ascent

Note that there is a strong connection between the updates of (exact) natural gradient ascent and those of coordinate-ascent variational inference: When $\rho_t \equiv 1$, the two methods are equivalent. Coordinate ascent sets the variational parameter λ directly to $\mathbb{E}_q[\eta_g(x, z, \alpha)]$, while natural gradient ascent sets it to a weighted combination of the current parameter and that expectation. *Stochastic* natural gradient ascent, a.k.a. stochastic variational inference, sets the variational parameter to a weighted combination of its current value and an *estimate* of the expectation computed on a mini-batch.

As discussed in chapter 5, coordinate ascent variational inference cycles through the variational parameters, setting each to the value that locally maximizes the ELBO. Coordinate ascent update rules can be derived from the formulae of the ELBO gradients because locally maximizing the ELBO with respect to a parameter is equivalent to setting the corresponding gradient to zero. A local parameter ϕ_{dj} is at a local optimum when the gradient (6.8) is zero. This condition can be achieved by setting

$$\phi_{dj} \leftarrow \mathbb{E}_q[\eta_\ell(x_n, z_{d,\neg j}, \beta)]. \tag{6.14}$$

Likewise, a global parameter λ is at a local optimum when (6.7) is zero, which is accomplished by setting

$$\lambda \leftarrow \mathbb{E}_q[\eta_g(x, z, \alpha)]. \tag{6.15}$$

Thus, coordinate-ascent consists of repeatedly applying updates of the form (6.14) and (6.15). First, using the current estimate of the global parameters, the local parameters within each group are optimized by cycling through updates of the form (6.14) until convergence. Then, using the local parameters from all groups, the global parameters are updated using equation (6.15). Each coordinate ascent update is guaranteed to increase the ELBO objective (or, at worst, maintain its current value).

The coordinate ascent update rules above are identical to that those from chapter 5, although they are presented in a different manner. To illustrate this point, consider the setting of λ that locally maximizes the ELBO, $\lambda = \mathbb{E}_q[\eta_g(x, z, \alpha)]$.

Under this parameter setting, the factor $q(\beta \mid \lambda)$ becomes

$$\begin{aligned} q(\beta \mid \lambda) &\propto \exp\{h(\beta) + \mathbb{E}_q[\eta_g(x, z, \alpha)]^\top t(\beta)\} \\ &\propto \exp\{\mathbb{E}_{\neg\beta}[\log p(\beta \mid \mathbf{x}, \boldsymbol{z})]\} \\ &\propto \exp\{\mathbb{E}_{\neg\beta}[\log p(\mathbf{x}, \boldsymbol{z}, \beta)]\}. \end{aligned} \tag{6.16}$$

The last two lines are equivalent because $p(\mathbf{x}, \boldsymbol{z}, \beta) = p(\beta \mid \mathbf{x}, \boldsymbol{z}) p(\mathbf{x}, \boldsymbol{z})$, and $p(\mathbf{x}, \boldsymbol{z})$ does not depend on β or λ. The final line shows that the setting of the natural parameters that locally maximizes the ELBO is equivalent to the solution implied by equation (5.7) in the previous chapter. Both say that the local optimum of an approximating factor is found by exponentiating the expected log joint density of the model. The analysis in this chapter is more precise; it expresses the ELBO and related quantities in terms of exponential families. This greater level of detail is necessary to derive stochastic variational inference.

The inefficiency of coordinate-ascent variational inference stems from the way global parameters are updated. The update 6.15 calls for setting λ to the expected natural parameter of the complete conditional, which is a function of all the data in the corpus and the posterior estimates of all local latent variables in the model. This quantity is expensive to compute, and the time required to do so scales linearly with the dataset size. Stochastic variational inference uses ideas from stochastic optimization to ameliorate this inefficiency. Rather than making exact updates to the global parameters using all of the data, it uses small samples of the data to make a series of noisy, partial updates.

6.3 Stochastic Variational Inference for IWTM

The theory discussed in the previous section was defined for a general class of exponential-family model. This section uses the general theory to derive stochastic variational inference for IWTM, which is a novel contribution of this dissertation. After giving an overview of the procedure, the update rule for each set of latent variables is derived.

6.3.1 Preliminaries

Three aspects of stochastic variational inference for IWTM are identical to the coordinate-ascent case and will only be reviewed briefly. First, SVI operates on the same stick-breaking construction of IWTM. This construction is amenable to stochastic variational inference because it defines the model using only conjugate exponential-family distributions. Second, the form of the approximate posterior q is the same. Specifically, it is assumed to have the form

$$q(\boldsymbol{\beta'}, \boldsymbol{\pi'}, \mathbf{c}, \boldsymbol{\mu}, \boldsymbol{\lambda}, \boldsymbol{z}, \boldsymbol{k}, \boldsymbol{\theta}) = \left(\prod_{c=1}^{C} q(\beta'_c) q(\boldsymbol{\mu}_c, \lambda_c)\right) \left(\prod_{t=1}^{T} \prod_{k=1}^{K_t} q(\pi'_{tk}) q(c_{tk})\right) \left(\prod_{d} q(\boldsymbol{\theta}_d) \prod_{i=1}^{N_d} q(z_{di}, k_{di})\right)$$

where factors $q(\beta'_c \,|\, \tilde{u}_c, \tilde{v}_c)$ and $q(\pi'_{tk} \,|\, \tilde{a}_{tk}, \tilde{b}_{tk})$ are beta distributed, $q(\boldsymbol{\mu}_c, \lambda_c \,|\, \tilde{\mathbf{m}}_c, \tilde{\nu}_c, \tilde{\chi}_{1c}, \tilde{\chi}_{2c})$ are multivariate normal-gamma, and $q(c_{tk} \,|\, \tilde{\boldsymbol{\beta}}_k)$ and $q(z_{di}, k_{di} \,|\, \tilde{\boldsymbol{\theta}}_{dk})$ are discrete. Finally, stochastic variational inference uses the exact same document-level inference procedure as coordinate ascent. The key difference between the methods is *how many* documents inference must be performed on to update a global parameter: SVI only needs a mini-batch of documents to update a global parameter, whereas coordinate-ascent must perform local inference on all documents to do so.

The following subsections derive the update rule for each set of global latent variables in IWTM. In general, each update is performed by setting the natural parameter of an approximating factor to a weighted combination of its current value and the expected natural parameter of the complete conditional. A potential source of confusion is that, for some families of distributions, the natural parameterization is not the one commonly used. For example, the natural parameters of a univariate Gaussian are μ/σ^2 and $-1/(2\sigma^2)$, not the mean and variance. It will be helpful to talk about such distributions using their typical parameterization most of the time and then switch to the natural parameterization when defining the update rules. In general, the two parameterizations are bijective functions of one another, and it is possible to switch back and forth between them as necessary.

Some new notation is required to distinguish between the natural parameters of the approximating distributions and the complete conditionals of the model. For a latent variable z, $\eta_{q(z)}$ is used to denote the natural parameters of the posterior approximation $q(z)$. Similarly, $\eta_{p(z)}$ is used to denote the natural parameter of the complete conditional of z in the model. (To simplify notation, the latent variables being conditioned on are dropped.)

6.3.2 Overview

Each iteration of SVI operates the same way on IWTM as was described in the previous section. First, a mini-batch of documents S of predetermined size is sampled from the corpus. Second, using the current estimate of the model's global components, local inference is performed on each document. This step computes the topic weights of the documents and topic-atom assignments of the individual data that locally maximize the ELBO. Third, the mini-batch is used to move the topic stick weights, topic atom to cluster mappings, and cluster parameters in the direction of their estimated natural gradients. The update for each parameter is performed by using the mini-batch to estimate its new value and then setting the new value to a weighted combination of the current value and the mini-batch estimate. Finally, the master topic stick weights are set to their optimum exactly, using an update identical to the one in coordinate ascent. The reason that the master topic sticks are not updated stochastically is explained in section 6.3.6.

SVI for IWTM deviates slightly from the method described in the last section because there are multiple sets of global latent variables in IWTM, rather than a single global variable β. This more complex scenario is accommodated simply by updating each global variable one at a time in sequence. In this algorithm, the cluster parameters are updated first, followed by the atom to cluster mappings, followed by the topic stick weights, and then finally the master topic sticks.

6.3.3 Cluster Parameters

In IWTM, the complete conditional of the cth set of cluster parameters is a multivariate-normal gamma distribution,

$$p(\mu_c, \lambda_c \,|\, \mathbf{x}, \boldsymbol{z}, \boldsymbol{k}, \mathbf{c}) = \mathrm{NG}\Bigg(\frac{\beta_0 \mathbf{m} + \sum_{d,i} r_{dic} \mathbf{x}_{di}}{\beta_0 + \sum_{d,i} r_{dic}},\ \beta_0 + \sum_{d,i} r_{dic}$$

$$\chi_1 + \sum_{d,i} r_{dic}/2,\ \chi_2 + \sum_{di} \frac{r_{dic}}{2} \|\mathbf{x}_{di} + \bar{\mathbf{x}}_c\|^2 - \frac{\beta_0 n_c}{\beta_0 + n_c} \frac{\|\bar{\mathbf{x}}_c - \mathbf{m}\|^2}{2} \Bigg). \tag{6.17}$$

For convenience, r_{dic} is defined to be an indicator variable that is one when datum i in document d is assigned to cluster c, n_c is the count of data assigned to cluster c, and $\bar{\mathbf{x}}_c$ is the average of the data assigned to cluster c:

$$r_{dic} = \sum_k \sum_t [\boldsymbol{z}_{di} = t \wedge \boldsymbol{k}_{di} = k \wedge \mathbf{c}_{tk} = c] \tag{6.18}$$

$$n_c = \sum_{d,i} r_{dic} \tag{6.19}$$

$$\bar{\mathbf{x}}_c = \frac{1}{n_c} \sum_{d,i} r_{dic} \mathbf{x}_{di}. \tag{6.20}$$

Stochastic variational inference calls for moving the natural parameters of $q(\boldsymbol{\mu}_c, \lambda_c)$ towards the expected natural parameters of the complete conditional. The parameterization in (6.17) is not the natural parameterization of the normal-gamma distribution. Its natural parameters are

$$\eta_{p(\boldsymbol{\mu}_c,\lambda_c)} = \Big\langle \beta_0\mathbf{m} + \sum_{d,i} r_{dic}\mathbf{x}_{di}, \quad \beta_0 + \sum_{d,i} r_{dic}, \quad \chi_1 + \sum_{d,i} r_{dic}/2 \\ -\chi_2 - \sum_{di} \frac{r_{dic}}{2}\|\mathbf{x}_{di} - \bar{\mathbf{x}}_c\|^2 - \frac{\beta_0 n_c}{\beta_0 + n_c}\frac{\|\bar{\mathbf{x}}_c - \mathbf{m}\|^2}{2} \Big\rangle \tag{6.21}$$

Note that the first natural parameter is the same dimension of the data (N_{dim}), while other three are scalar values.

The posterior for the cth set of cluster parameters is approximated by the multivariate-normal gamma factor $q(\boldsymbol{\mu}_c, \boldsymbol{\lambda}_c) = NG(\tilde{\mathbf{m}}_c, \tilde{\nu}_c, \tilde{\chi}_1, \tilde{\chi}_2)$. Its natural parameters are:

$$\eta_{q(\boldsymbol{\mu}_c,\lambda_c)} = \langle \tilde{\mathbf{m}}_c\tilde{\nu}_c,\ \tilde{\nu}_c,\ \tilde{\chi}_{1c} - 1,\ -1/\tilde{\chi}_{2c}\rangle. \tag{6.22}$$

The new natural parameters are estimated by taking the expectation of (6.21) on a replicated copy of the document mini-batch:

$$\begin{aligned}\hat{\eta}_{q(\boldsymbol{\mu}_c,\lambda_c)} = \Big\langle & \beta_0\mathbf{m} + \frac{N_{\text{doc}}}{|S|}\left(\mathbb{E}_S[\bar{\mathbf{x}}_c]\right), \\ & \beta_0 + \frac{N_{\text{doc}}}{|S|}\left(\mathbb{E}_S[n_c]\right), \\ & \frac{N_{\text{doc}}}{|S|}\left(\mathbb{E}_S[n_c]\frac{N_{\text{dim}}}{2}\right) + \chi_1 - 1, \\ & \frac{N_{\text{doc}}}{|S|}\left(-\frac{1}{2}\sum_{d\in S}\sum_{i=1}^{n_d}\mathbb{E}[r_{dic}]\|\mathbf{x}_{di} - \tilde{\mathbf{m}}_c\|^2\right) - \frac{\beta_0}{2}\|\tilde{\mathbf{m}}_c - \mathbf{m}\|^2 \Big\rangle. \end{aligned} \tag{6.23}$$

Here, $\mathbb{E}[r_{dic}]$ is the expectation of the cluster assignment, and $\mathbb{E}_S[n_c]$, $\mathbb{E}_S[\bar{\mathbf{x}}_c]$ are the expectations of the cluster sizes and data means computed on the mini-batch S:

$$\mathbb{E}[r_{dic}] = \sum_t \sum_k \tilde{\beta}_{kc}\tilde{\theta}_{ditk} \tag{6.24}$$

$$\mathbb{E}_S[n_c] = \sum_{d\in S} \sum_{i=1}^{N_d} \rho_{dic} \tag{6.25}$$

$$\mathbb{E}_S[\bar{\mathbf{x}}_c] = \frac{1}{\mathbb{E}_S[n_c]} \sum_{d\in S} \sum_{i=1}^{N_d} \rho_{dic}\mathbf{x}_{di} \tag{6.26}$$

The natural parameters of the approximating distribution are then updated as

$$\eta^{(t+1)}_{q(\boldsymbol{\mu}_c,\boldsymbol{\lambda}_c)} = (1-\rho)\eta^{(t)}_{q(\boldsymbol{\mu}_c,\lambda_c)} + \rho\cdot\hat{\eta}_{q(\boldsymbol{\mu}_c,\lambda_c)}. \tag{6.27}$$

The natural parameters of the clusters are related to their standard parameterization $(\tilde{\mathbf{m}}_c, \tilde{\nu}_c, \tilde{\chi}_1, \tilde{\chi}_2)$ through equation (6.22). Thus, updating the natural parameters induces an update on the standard parameters as well.

6.3.4 Topic Atom to Cluster Mappings

The complete conditional for the topic atom to cluster mappings is a discrete distribution,

$$p(\mathbf{c}_k = c \,|\, \boldsymbol{\beta'}, \boldsymbol{\mu}, \lambda, \boldsymbol{z}, \boldsymbol{k}) \propto \exp\left\{ \beta_c \prod_{d,i}\prod_{t=1}^{T} p(\mathbf{x}_{di}\,|\,\boldsymbol{\mu}_c, \lambda_c)^{[z_{di}=t \wedge k_{di}=k]} \right\} \tag{6.28}$$

In general, the natural parameters of discrete distributions are defined in terms of their log probabilities. To simplify notation, let ℓ_{kc} denote the unnormalized log probability of atom k being assigned to cluster c,

$$\ell_{kc} = \log(\beta_c) + \sum_{d,i} \sum_{t=1}^{T} [z_{di} = t \wedge k_{di} = k] \log p(\mathbf{x}_{di} \,|\, \boldsymbol{\mu}_c, \lambda_c). \tag{6.29}$$

The natural parameter is the vector of log probabilities with an arbitrarily chosen "pivot" index subtracted out. For convenience, the variational truncation limit C is chosen as the pivot,

$$\eta_{p(\mathbf{c}_k)} = \langle \ell_{kc} - \ell_{kC} \rangle_{c=1}^{\infty}. \tag{6.30}$$

The use of the pivot sets the Cth element to zero, so that the natural parameter effectively has one fewer dimension. Doing so corrects for the fact that a N-dimensional discrete distribution only has $N-1$ free parameters, and is necessary to express the discrete distribution as a canonical (non-curved) exponential family [18]. Note that although the complete conditional has infinite dimension, inference only has to estimate the parameter up to the truncation limit C.

In the approximating distribution, the natural parameters of kth atom assignment is the C-length vector

$$\eta_{q(\mathbf{c}_k)} = \langle \log \tilde{\beta}_{kl} - \log \tilde{\beta}_{kC} \rangle_{l=1}^{C}. \tag{6.31}$$

As in the other updates, the mini-batch S is used to estimate new parameters, which are the expectation of the complete conditional's natural parameters:

$$\hat{\ell}_{kc} = \mathbb{E}[\log \beta_c] + \frac{N_{\text{doc}}}{|S|} \sum_{d \in S} \sum_{i} \sum_{t=1}^{T} \tilde{\theta}_{ditk} \mathbb{E}[\log p(\mathbf{x}_{di} \,|\, \boldsymbol{\mu}_c, \lambda_c)] \tag{6.32}$$

$$\hat{\eta}_{q(\mathbf{c}_k)} = \langle \hat{\ell}_{kl} - \hat{\ell}_{kC} \rangle_{l=1}^{C}. \tag{6.33}$$

The natural parameters of the approximating factors are then updated by setting

$$\eta_{q(\mathbf{c}_k)}^{(t+1)} = (1-\rho)\eta_{q(\mathbf{c}_k)}^{(t)} + \rho \cdot \hat{\eta}_{q(\mathbf{c}_k)}. \tag{6.34}$$

The new $\tilde{\beta}$ values can then be computed by inverting equation (6.31), i.e. exponentiating and normalizing the new natural parameters.

6.3.5 Topic Stick Weights

The update for the topic weights is slightly different from that of the other parameters due to the reordering of topic atoms. Recall that the topic sticks weights are defined in terms of the count of data associated with each atom. The complete conditionals for the topic stick weights are beta distributions

$$p(\pi'_{tk} \mid \boldsymbol{z}, \boldsymbol{k}) = \text{Beta}\left(1 + n_{tk}, \alpha_0 + \sum_{j >_{\mathbf{n}_t} k} n_{tj}\right), \tag{6.35}$$

where $n_{tk} = \sum_{d,i}[z_{di} = t \wedge k_{di} = k]$ is the number of observed data generated from the kth atom in topic t, and the sum in the second parameter is over atoms in the topic whose counts are greater than n_{tk} (using notation defined in section 5.4.5). Equation (6.35) is the natural parameterization of the beta distribution, and both natural parameters are determined entirely by the counts (and, implicitly, the relative ranks of the counts). The natural gradient update can therefore be defined directly in terms of such counts.

The variational approximation of the topic stick weights is modeled by factors $q(\pi_{tk}) = \text{Beta}(\pi_{tk} \mid \tilde{a}_{tk}, \tilde{b}_{tk})$. The parameters $\tilde{a}$ and $\tilde{b}$ are determined entirely by the pseudo-counts $\tilde{\zeta}_{tk}$. Let $\tilde{\zeta}_{tk}^{(t)}$ denote the data count associated with the kth atom in topic t in the current iteration. Similar to other updates, a mini-batch of documents is used to estimate the new pseudo-counts as

$$\hat{\zeta}_{tk} = \frac{N_{\text{doc}}}{|S|} \sum_{d \in S} \sum_{i=1}^{N_d} \tilde{\theta}_{ditk}, \tag{6.36}$$

and the current counts are set to a weighted average of the old and new ones:

$$\tilde{\zeta}_{tk}^{(t+1)} = (1-\rho)\tilde{\zeta}_{tk}^{(t)} + \rho\hat{\zeta}_{tk}. \tag{6.37}$$

The topic weight parameters themselves are then set using the new count estimates via simulated reordering. That is,

$$\tilde{a}_{tk}^{(t+1)} = 1 + \tilde{\zeta}_{tk}^{(t+1)} \tag{6.38}$$

$$\tilde{b}_{tk}^{(t+1)} = \alpha_0 + \sum_{j >_{\tilde{\zeta}} k} \tilde{\zeta}_{tj}^{(t+1)}. \tag{6.39}$$

6.3.6 Master Topic Stick Weights

The updates for the master topic stick weights are the same as in coordinate-ascent variational inference. The reason is that their complete conditionals depend only on other global latent variables in the model, not on document-level latent variables. As a result, the natural gradient can be computed exactly with minimal effort – it does not have to be estimated from the data. Correspondingly, the stick weights can be set to a local optimum exactly.

The complete conditionals of the master sticks weights are beta distributed,

$$p(\beta'_c \,|\, \mathbf{c}) = \text{Beta}(1 + n_c, \sum_{c' >_{\mathbf{n}} c} n_{c'}), \tag{6.40}$$

where $n_c = \sum_{t,k}[c_{tk} = c]$ is the number of topic atoms associated with the cth cluster, and the sum in the second parameter is taken over clusters with counts greater than n_c. As with the topic sticks, the natural parameters of (6.40) are defined entirely by the counts and their order statistics.

The posterior approximation of the master topic stick weights is captured by factors $q(\beta'_c) = \mathrm{Beta}(\tilde{u}_c, \tilde{v}_c)$, where both variational parameters are defined by the pseudo-counts $\tilde{\boldsymbol{\xi}}_c$. The update sets the pseudo-counts directly to their expectation,

$$\tilde{\xi}_c^{(t+1)} = \mathbb{E}[n_c] = \sum_{t=1}^{T} \sum_{k=1}^{K} \tilde{\beta}_{kc}. \tag{6.41}$$

The parameters themselves are then computed from the pseudo-counts as

$$\tilde{u}_c^{(t)} \triangleq 1 + \tilde{\xi}_c^{(t)} \tag{6.42}$$

$$\tilde{v}_c^{(t)} \triangleq \gamma + \sum_{c' >_{\tilde{\boldsymbol{\xi}}} c} \tilde{\xi}_{c'}^{(t)} \tag{6.43}$$

where the sum is taken over cluster indices c' with counts larger than c. Note that the variational parameters and the pseudo-counts are bijections of each other; the pseudo-count estimates can be recovered from the natural parameters themselves and vice versa.

6.3.7 Learning Rate and Mini-Batch Size

The learning rate parameter ρ has not yet been specified. In general, it should satisfy $0 < \rho \leq 1$. When $\rho = 0$, the algorithm effectively does nothing. When $\rho = 1$, each iteration completely replaces the current variational parameters with their new values estimated from the document mini-batch. A special case occurs when $\rho = 1$ and $|S| = N_{\mathrm{doc}}$, i.e. when the mini-batch is the entire training corpus. In this case, the natural gradient estimates are exact, and each update sets the variational parameters exactly to their local maximum. The algorithm for stochastic variational inference then reduces to the coordinate-ascent procedure described in chapter 5. In this case, each update made by inference is guaranteed to increase the ELBO objective.

When the corpus is large and coordinate-ascent variational inference is prohibitively expensive, the mini-batch size will typically be a fraction of the training corpus ($|S| \ll N_{\text{doc}}$). The natural gradient estimates will therefore be noisy. In general, moving in the direction of the noisy natural gradient is not guaranteed to increase the ELBO objective, but will tend to do so when the estimate is close enough to the true natural gradient. Estimates taken on a small set of documents will tend to have high variance but be inexpensive to compute, while larger mini-batches will yield lower-variance estimates that are expensive to compute. To balance these concerns, the experiments in this dissertation set the learning rate proportional to the mini-batch size, $\rho = |S|/N_{\text{doc}}$. Under this scheme, updates made with noisier gradients have a smaller effect on the model. In addition, setting $|S| = N_{\text{doc}}$ leads to a learning rate of 1, so that stochastic variational inference reduces to coordinate ascent.

Despite the practical appeal of this learning rate scheme, the resultant algorithm is not formally guaranteed to converge to a local optimum. The convergence of stochastic optimization methods is typically achieved by using a learning rate that decreases over time. When the learning rate schedule satisfies the conditions $\sum_{t=1}^{\infty} \rho_t = \infty$ and $\sum_{t=1}^{\infty} \rho_t^2 < \infty$, convergence to a local maximum is guaranteed [15, 26].[1] Hoffman et al. [26] suggest using $\rho_t = (t + \tau)^{-\kappa}$, where $k \in (0.5, 1]$ and $\tau \geq 0$ are free parameters [26]. While appealing in theory, such learning rate schedules are not useful in practice for two reasons. First, they are computationally wasteful. Decreasing learning rates cause later iterations to have diminishing effect; however, the natural gradients in each iteration still have a fixed, non-trivial cost to compute. Second, convergence is an asymptotic property; when the algorithm is run for a finite amount of time, there is no guarantee that the final solution will be

[1]Additionally, that the function being optimized must satisfy some requirements, such as being thrice-differentiable. The ELBO objective satisfies such properties [26].

a local optimum. In light of these issues, a decreasing learning rate is eschewed in favor of the fixed learning rate scheme described above.

6.3.8 Escaping Local Optima

While stochastic variational inference has been motivated as being more efficient than coordinate ascent, it has another advantage as well: the ability to escape (some) poor local optima. Coordinate ascent's updates are exact and deterministic, and each one is guaranteed to increase (or maintain the value of) the ELBO. It can therefore never escape any local optimum; by definition, doing so would require making at least one update that lowers the ELBO. In contrast, SVI's natural gradient estimates are noisy, so its updates will not always increase the ELBO. Somewhat paradoxically, this property is beneficial because the objective is non-convex. Making updates that are locally suboptimal enables it to escape local optima and (sometimes) find better solutions. In general, the ability of SVI to do so depends on the mini-batch size, which determines the variance of the natural gradient estimates. Smaller mini-batches produce noisier natural gradients, making SVI less susceptible to getting stuck in poor solutions. Of course, such noise may also prevent SVI from converging on a good solution. The effect of noise in the natural gradients is evaluated experimentally in the next section.

6.4 Experiments on the 13-Scene Dataset

To understand stochastic variational inference for IWTM better, the method is first applied to the same 13-scene dataset used in chapters 4 and 5. By doing so, SVI can be compared to coordinate-ascent variational inference.

6.4.1 Method

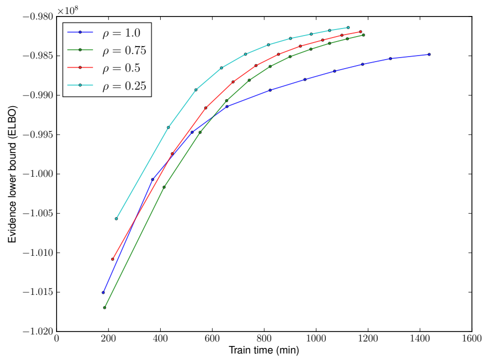
The purpose of this experiment is to compare SVI and coordinate-ascent variational inference in terms of their speed and ability to fit the dataset. To do so, IWTM is trained on 13-scene under a variety of learning rates and mini-batch sizes. Specifically, models are trained with $\rho = \{0.25, 0.5, 0.75, 1.0\}$ and, per section 6.3.7, mini-batch sizes are set as $|S| = \rho N_{\text{doc}}$. On one extreme, the $\rho = 1$ setting reduces stochastic variational inference to coordinate-ascent variational inference. In this case, all natural gradients are computed exactly and each update has a large effect on the model; however, computing the exact natural gradients is expensive because it requires performing local inference on every document in the corpus. On the other extreme, the $\rho = 0.25$ and $|S| = 0.25 N_{\text{doc}}$ causes SVI to make small updates based on noisy but efficiently-computable natural gradient estimates.

To account for the fact that different settings will use different numbers of documents per iteration, the number of iterations is varied appropriately. Specifically, all models are trained until they have seen a total of 65000 documents, or the equivalent of 50 iterations of coordinate-ascent variational inference. Therefore, models with $\rho = 1$ train for 50 iterations, models with $\rho = 0.5$ train for 100 iterations, and so on.

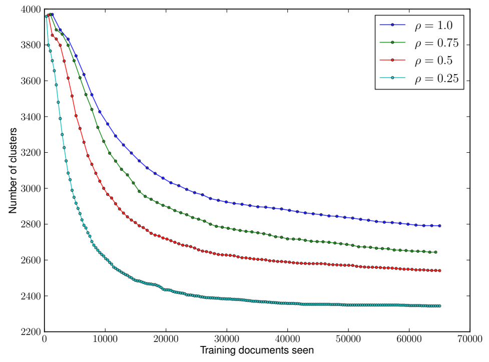
The evaluation metrics of interest are the model ELBOs, which measure how well each fits the training corpus, and the speed of inference itself. To measure the former, snapshots of the model state are saved at regular intervals (after every 10% of the training iterations have been completed). Regardless of the mini-batch size used to train the model, the ELBO is always measured on the entire training corpus. To make training times comparable, all models are trained on identical machines [2] using a single-threaded implementation.

For simplicity, all models are trained with $T = 25$ topics. The hyper-

[2]Specifically, `narsil-*.cs.utexas.edu`, a set of 13 machines with 3.06GHz processors and 96 GB of RAM each. (The inference procedure only needs a few hundred MB of RAM at most.)



(a) ELBO over time



(b) # clusters used

Figure 6.1: Effect of varying the learning rate and mini-batch of stochastic variational inference on the 13-scene task. (a) Stochastic variational inference ($\rho = 0.25, 0.5, 0.75$) results in models that train faster and fit the training data better than coordinate-ascent variational inference ($\rho = 1$). (b) The reason is that noise in the natural gradients allows stochastic variational inference to escape local optima. Coordinate ascent, and SVI with "cleaner" natural gradients, get stuck using an unnecessarily large number of clusters, leading to suboptimal solutions.

parameters are given the same settings as in chapter 5, specifically $\gamma = \alpha_0 = 10, \beta_0 = 1, \alpha = 1/T$. The truncation limits are also the same: The master topic is truncated at $C = 4000$ clusters and the topics are truncated at $T = 8000$ atoms. Finally, IWTM is also initialized in the same way as in coordinate ascent.

6.4.2 Results

Results are summarized in figure 6.1. Figure 6.1(a) shows the mean ELBO of each setting, averaged over five runs, as a function of training time. SVI not only produces models with higher ELBO than coordinate ascent, it also trains them faster. Coordinate ascent takes about 1400 minutes (23 hours) to train and performs the worst of all settings; meanwhile, the fastest SVI ($\rho = 0.25$) is also the setting that produces the highest ELBO. It surpasses the best ELBO of coordinate ascent after only 600 minutes (10 hours), less than half the training time of the latter. SVI fin-

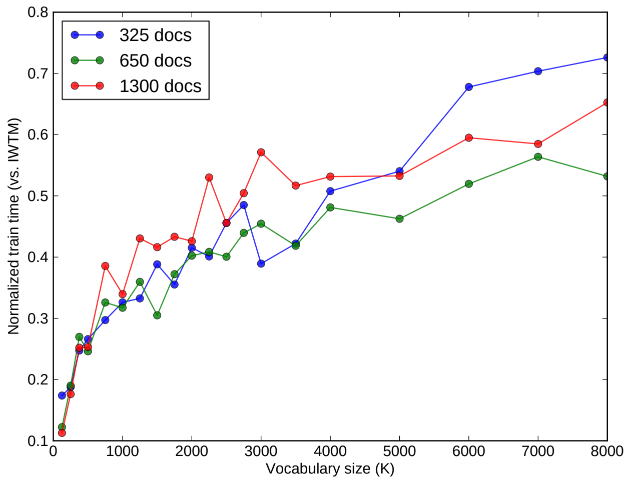


Figure 6.2: Training times of LDA+K-means on 13-scene, normalized against IWTM's training time on the same amount of data. Using stochastic variational inference, IWTM only takes about 2.5-3 times as long to train as a single run of LDA+K-means, while obviating the need to manually sweep over the vocabulary size. (Compare to figure 5.3.)

ishes training altogether in 1150 minutes (19.2 hours), or about 16.5% faster than coordinate ascent, at which point its ELBO is significantly higher.

Given that all methods see an equal number of training documents, what accounts for the difference in their speed and performance? Both stem from the fact that SVI makes noisy natural gradient updates that allow it to escape poor local optima. Specifically, variational inference has a tendency to get stuck in poor local optima where it uses too many clusters, which slows down the inference procedure. Such an explanation is supported by figure 6.1(b), which shows the number of clusters used by each method as training progresses. The slowest, worst-performing method, coordinate ascent, uses the most clusters; meanwhile the fastest, best-performing method, SVI at $\rho = 0.25$, uses the fewest clusters.[3] Furthermore, the lower ρ is, the faster clusters are pruned from the model. Thus, the noise in SVI's

[3]One should not conclude that IWTM's ELBO is always higher when fewer clusters are used. This is not so, as demonstrated in section 7.4.

natural gradients benefits IWTM doubly: It allows better solutions to be found and, as a side-effect of doing so, makes inference faster.

Such results show that SVI for IWTM is superior to coordinate-ascent variational inference even on modestly-sized datasets where the latter is a viable option. On datasets of sufficient scale, coordinate ascent is too expensive to be run at all. The next section uses SVI to train IWTM on one such dataset, SUN397, which is about seven times the size of 13-scene.

6.5 Experiments on the SUN Dataset

To demonstrate that stochastic variational inference allows IWTM to be scale to much larger datasets, the method was applied to the SUN397 dataset [61]. SUN397 contains over 39700 photographs of 397 different categories of indoor and outdoor scenes. In these experiments, a randomly-selected subset of 1985 images (5 images per class) is selected for the training set, and 19850 (50 images per class) are set aside for testing. Even after sampling, the training set is much larger than IWTM has been applied to previously. It contains over 50% more images than the largest training set used in the 13scene task. More significantly, the SUN images are higher resolution than those in 13scene, and produce over 6.7 times the total number of local descriptors. Running IWTM on a dataset of this size is made vastly more practical by stochastic variational inference.

As in 13-scene, IWTM is evaluated for its ability to generate image descriptions that are useful for scene classification. SUN397 is a challenging task. Due to the large number of classes, guessing randomly gets only 0.25% accuracy. Furthermore, some categories are differentiated by nuanced details – for example, *abbey*, *church*, and *cathedral*. IWTM's performance on the task hinges on its ability to generate image descriptions that capture semantic content.

6.5.1 Method

The methods considered in this experiment are based on the image features and type of classifier reported to work the best on the task by the dataset creators [61]. Specifically, the image features used are HOG2x2 descriptors, and the classifiers are support vector machines with spatial pyramid histograms as kernels (both defined in this subsection). The rest of this section is structured as follows. First, the base image features and the approach used by Xiao et al. [61] are described. Building on that method, a modification is then developed that uses features derived from IWTM. In the next subsection, both methods are evaluated on the SUN397 task.

HOG2x2 Features

Both methods considered in this chapter use HOG2x2 as the base image features, though the different methods do different things with them. HOG2x2 features are generated using the code of Xiao et al. [61], which performs the following steps. First, large images are downsized so that their maximum dimension is 640 pixels; small images are not upscaled. Second, histograms of oriented gradients (HOGs) are extracted at intervals of eight pixels over a regular grid. This step yields a 31-dimensional descriptor for each image patch. Finally, the HOGs for adjacent 2×2 patches are concatenated together, producing a 124-dimensional feature vector for each patch. The number of feature vectors used to represent each image depends on the image size. The median number is 3619, but some images have as many as 5929 and one image has as few as 40. The 1985 images in the training set comprise a total of 5,646,426 HOG2x2 descriptors.

Classification using Visual Words

In the method of Xiao et al. [61], the HOG2x2 features are clustered and quantized into $K = 300$ visual words using K-means. Then, using the visual words representation, one-versus-all support vector machines are trained using three-level spatial pyramid histograms as the kernel [35]. This approach is the best known single-feature method for the SUN397 task.[4] The topic model-based approaches are not necessarily expected to outperform it.

The three-level spatial pyramid is formed by dividing each image into three spatial grids of different resolutions: 1×1, 2×2, and 4×4. The appearance of the image within each spatial bin is represented as a histogram of the visual words occurring within that bin. The 1×1 grid produces a single K-length histogram of the entire image's visual words; the 2×2 grid produces four K-length histograms, one for each quadrant of the image; and the 4×4 grid produces 16 K-length histograms. Within each level, the kernel similarity between two images is defined by histogram intersection. The kernel matrices at the three levels are normalized by their respective means and combined using equal weights.

At a high level, the visual words and pyramid match kernel can be thought of as defining two similarity measures – one between the appearance of local image patches, and one between images as a whole. The similarity of local image patches is implicitly defined by the visual words. Specifically, two image patches are considered similar if they map to the same visual word and otherwise they are dissimilar. The pyramid match kernel then defines similarity between *images* in terms of the number of similar-looking patches they share. The use of multiple levels increases the similarity for cases where the corresponding patches lie in approximately the same image region.

6.5.2 Classification Using IWTM

Intuitively, visual words define a limited notion of patch-level similarity: Patches that are semantically related but have sufficiently different appearances will be di-

[4]Classifier ensembles combining HOG2x2 with other feature types perform better on the task [61]. To limit the scope of this chapter they are not considered here.

vided into separate visual words. For example, different parts of the same cloud or tree are expected to be divided into multiple, distinct visual words, even though they belong to the same texture or object. Motivated by this fact, a modification of the above method was developed in this dissertation wherein *topic features* from IWTM are used in place of visual words.

IWTM is trained directly on the HOG2x2 representation of the images IWTM is with $T = 100$ topics. Due to the large size of the dataset, stochastic variational inference with a learning rate $\rho = 0.1$ and a mini-batch size $|S| = N_{\text{doc}}/10$ are used. The features provided to the classifier are the maximum a posterior (MAP) topic assignment of each image patch,

$$\tilde{\theta}^{*}_{d,i} = \arg\max_{t} \sum_{k=1}^{K} \tilde{\theta}_{ditk} \tag{6.44}$$

where $\tilde{\theta}_{ditk}$ are the posterior topic atom assignments estimated by variational inference. As in the visual words method, classification is performed by support vector machines with a three-level pyramid match kernel. Using topic assignments in place of visual words changes the notion of patch-level similarity – specifically, two patches are considered similar if and only if their posterior topic assignments are the same. Intuitively, the topics should group together commonly co-occurring clusters of features, and the topic assignment of each patch should capture some higher-level semantic meaning of its content.

6.5.3 Results

Both the visual-words and IWTM-based method were evaluated on the test set of 10 images per class. Overall, the visual words method achieves a mean recognition rate of $11.7\% \pm 0.1$ on the test set, while IWTM gets $7.1\% \pm 0.2$. Both methods perform the best on the same three categories, *car front seat*, *car back seat*, and *cockpit*.

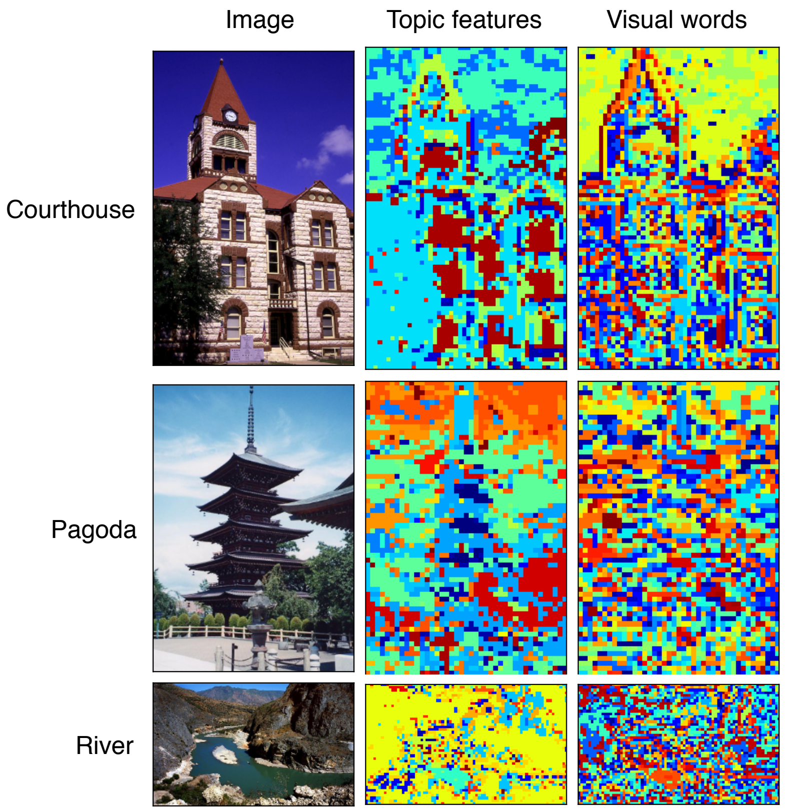


Figure 6.3: Images from the SUN dataset along with the topic assignments generated by IWTM (middle) and the visual word assignments (right). Colors are arbitrarily chosen and are used to visually differentiate the feature values. IWTM groups semantically-related patches together in the same topic, while visual words make a more fine-grained categorization of visual appearance.

However, each feature type also has some categories on which it outperforms the other. On average, visual words perform better (by at least a margin of 1% absolute) on 219 categories, topics features perform better on 91 categories, and the two methods tie on the remaining 87 categories. The categories on which visual words most outperforms topic features are *street* (50% vs. 5%), *skatepark* (58% vs. 16.7%), and football stadium (50% vs. 10%). On the other hand, topic features fare better on *carrousel* (16.7% vs. 0%), *snowfield* (20% vs. 8%), and *raft* (20% vs. 8%).

The fact that IWTM does not perform as well as the visual words method is not surprising, considering that the latter has been reported to be the best-performing single-feature method on SUN397. However, the results for IWTM are considered positive for following two reasons. First, the model performs comparably with a number of computer vision based approaches. Based on published results by Xiao et al. [61], IWTM's accuracy with 5 training documents per class is slightly better than GIST [40] and texton histograms, and is slightly worse than dense SIFT features. In the context of these methods, IWTM performs quite reasonably. Second, it has been demonstrated that multiple feature types can be combined to achieve a higher recognition rate on SUN397 than any single-feature method, including HOG2x2 visual words [61]. Topic assignments are yet another feature type that can added to the mix.

6.5.4 Discussion

To gain insight into why the methods perform as they do on, the image representations produced by each were more closely examined. Figure 6.3 shows some training images along with the representations produced by both methods. The left column shows the original image, the middle column shows the topic assignments produced by IWTM, and the right column shows the cluster indices produced by the visual word method; each row corresponds to a single image.

As expected, IWTM groups many semantically-related patches together into the same topic. For example, in the top image of figure 6.3, the courthouse windows all fall into a single topic (visualized by dark red), as does the white stone exterior (visualized by light blue). In the middle image, much of the pagoda exterior is grouped together (the light blue topic), as are the cloudy patches of the sky (the mint green topic). Finally, in the last image, most of the mountainous terrain surrounding the river is assigned to the same topic (visualized by bright yellow). In contrast, the visual-words representation makes a more fine-grained categorization of patch appearance. Each of the aforementioned image regions is broken into many distinct visual words.

In general, IWTM is able to group together disparate-looking but related patches because its topics capture co-occurring patterns of visual appearance. The topic assignments can be thought of as a mid-level semantic representation: They provide an abstraction over patch-level appearance that cannot be captured by the HOG2x2 features or visual words alone. Although the topic assignments arguably capture more meaningful semantics, the results indicate that they are not superior features for scene classification. Abstracting over patch-level appearance seems to wash out the fine visual details needed to discriminate between some similar classes. For example, 85% of the misclassified *car back seat* images are predicted to be *car front seat*, and 30% of the misclassified *car front seat* images are predicted to be *car back seat*. Another illustrative example is that *outdoor tennis court* images are often mistaken for *baseball field* or *outdoor basketball court*.

6.5.5 Future Work

IWTM's ability to group together semantically-related image patches suggests that its features may be useful for other image analysis tasks, in particular image segmentation. Such an idea is supported by existing work that has successfully applied related topic models to segmentation. Two examples are the spatially-coherent latent topic model (Spatial-LTM) [17] and the topic random field (TRF) model [63]. In both models, the most likely topic assignment of each image region is used as its segment label. Because these topic models were designed with segmentation in mind, they have additional complexity that enforces spatial coherence in the topic labeling – that is, adjacent image regions are pressured to be assigned to the same topic. In future work, spatial coherence could also be added to IWTM to make it more suitable for segmentation. Such a model would inherit the strengths of IWTM, including its ability to infer the feature vocabulary non-parametrically, and could be trained on large datasets with a slight modification to the existing stochastic variational inference procedure.

As demonstrated in the previous chapter, manually tuning the visual vocabulary size (number of clusters) used by a topic model is expensive, since doing so requires training and evaluating separate models with different vocabulary sizes. When the dataset is very large, as in the SUN task, manual tuning of the visual vocabulary becomes prohibitive. Bayesian non-parametric models like IWTM are particularly appealing in such settings because they infer the model complexity based on the data. Stochastic variational inference is the key to training such models on large datasets. Per the results on the SUN task, a good fit of the data can be achieved using as little as one tenth of the dataset in each iteration.

6.6 Conclusions

Stochastic variational inference applies ideas from stochastic optimization to the task of variational inference. This chapter developed a stochastic variational inference procedure for IWTM, which is a novel contribution of this dissertation. SVI allows IWTM to be scaled to large datasets. Furthermore, even on more modestly-sized datasets like 13-scene, it trains faster and produces objectively better models than coordinate ascent. This is in part because SVI can escape from poor local optima that coordinate ascent cannot.

The next chapter switches gears and addresses how to train IWTM (and other Bayesian nonparametric models like it) efficiently on growing datasets. There, SVI is used to develop an incremental training method for IWTM, where new documents can be added to the training set and the model's complexity grows commensurately. SVI is key in making incremental training effective: Its ability to escape poor local optima actually prevents the incremental training from growing the model complexity *too much* when new documents are added.

Chapter 7

Incremental Training and Active Learning

The previous chapters make important advances towards applying the Infinite-Word Topic Model in realistic settings. Variational inference makes the training time of IWTM comparable to that of LDA with K-Means clustering, and allows the model to be scaled up to larger datasets than Gibbs sampling. Semi-supervised training allows IWTM to train on unlabeled data, which is typically plentiful, to improve performance on supervised tasks. This chapter addresses another challenge presented by real-world applications – namely, how to handle the case where the datasets periodically grow, rather than being fixed. In the streaming data setting, new documents periodically become available for inclusion in the topic model's training set. For example, the new documents could be a daily batch of photographs posted by users of a social networking website during the last 24 hours.

7.1 Motivation

Datasets that grow over time pose challenges to machine learning systems. To maximize performance, it is generally desirable to train on datasets that are as large as possible and include data that are up-to-date as possible. These are, however, competing goals: training takes longer on large datasets, and such long training times make the models less "fresh" at the time of deployment. For example, if it takes 24 hours to train a model before it can be deployed, that model's training set cannot (conventionally) contain examples generated within 24 hours of the time of deployment.

In situations where datasets grow, training can be sped up significantly by using an existing model as a starting point when fitting a larger dataset. This approach is *incremental training.* Such an approach exploits the idea when the dataset grows gradually over time, the model should change gradually over time as well. To fit a dataset that is 10% larger, for example, it should suffice to adjust the current model rather than training a completely new one from scratch.

These issues have been partially addressed by work in stochastic variational inference and, more broadly, online learning. These methods use small batches of data to make a series of small updates to the model, and are naturally designed to handle streams of training data. In this sense, they are well suited to incremental training: New training data can be piped into the inference procedure as they become available.

Unfortunately, a key property of mean-field variational inference – including stochastic variational inference – makes it less useful when training *Bayesian non-parametric models* on growing datasets. Specifically, variational inference will not increase the number of components in a BNP model in response to new data being added to the training set. In general, BNP models display a "paring back" phenomenon throughout training: They must be initialized with a large number of components, and variational inference monotonically reduces the model complexity over time. This issue is discussed at length in the next section. Its consequence is that BNP models trained with variational inference on growing datasets have an artificially low complexity because variational inference fits the model's complexity to the initial training set and is then unable to move to regions of the posterior where new components are added. As the dataset continues to grow, the model achieves a progressively suboptimal fit of the data.

This chapter introduces a novel extension of variational inference, called *incremental variational inference* (IVI), to support incremental training of Bayesian non-parametric models better. Incremental variational inference is identical to standard variational inference, except that the former executes an operation to grow the model's complexity when new data is added to its training set. Section 7.3 first describes the method in general, and then applies it to incremental training in IWTM. Section 7.4 demonstrates the speed and efficacy of IVI for IWTM experimentally. Section 7.5 then uses IVI to develop an active learning method for IWTM, where new data added to the model are intelligently subsampled to get better scene classification accuracy with smaller amounts of labeled data.

7.2 Variational Inference in Non-Parametric Models

As discussed in the previous section, variational inference pares back the complexity of Bayesian non-parametric models as training progresses. This section discusses the phenomenon at greater length, and explains why it occurs and why it is an obstacle to incremental training.

7.2.1 The Paring-Back Phenomenon

The paring back phenomenon has been noted by several authors in the literature. Figure 7.1 demonstrates it occurring in two BNP models – a DP mixture and HDP-

LDA – in a study by Wang, Hoffman, and Blei [58]. In both cases, the models are initialized with a large number of components (clusters in the DP mixture; topics in HDP-LDA) and use fewer and fewer components as training progresses. (That is, fewer components are associated with any data.) Later in the chapter, we demonstrate the same phenomenon occurring in IWTM. The numbers of clusters and topic atoms used by the model start out large and are decrease monotonically as variational inference progresses.

Adding new components to a Bayesian non-parametric model involves going through configurations of the latent variables that have low posterior density. For example, consider adding a new cluster to a Dirichlet process mixture model – having an empty cluster begin to be associated with some data. Because the cluster is initially empty, its expected stick weight in the Dirichlet process is very small. Moreover, the cluster's parameters are initially set to those of the prior, so that it provides a poor fit for any of the data. This combination makes it extremely unlikely for any data to be assigned to the cluster initially. Even if instantiating the new cluster would yield a better fit of the dataset, the intermediate configurations of the latent variables are very unlikely. Therefore, it can be difficult for some inference methods to add new clusters.

Thinking of inference as an optimization procedure leads to further insight. Variational inference directly optimizes the ELBO objective, while Gibbs sampling indirectly optimizes the joint posterior $p(\mathbf{Z} \mid \mathbf{X})$ (because its Markov chain naturally moves towards high probability configurations of the latent variables). These objectives are known to be non-convex for a broad class of models, including Gaussian mixtures. Configurations where new clusters are added lie in different local optima. In this sense, the ability of an inference method to add components in a BNP model depends on how well it can escape local optima.

In Gibbs sampling, the ability to move between local optima (alternatively, to *mix* or *mode-switch*) is a common concern. Generally, mode-switching is possible in Gibbs sampling because the procedure may produce a new configuration of latent variables that has a lower probability than the current one. It has generally been shown that mixing is improved in collapsed Gibbs sampling, i.e. where some parameters of the model can be integrated out [37]. In this light, it is helpful to look back at the collapsed Gibbs sampler for IWTM discussed in chapter 4. In that sampler, the cluster parameters of the model were completely integrated out. Additionally, it operated on the Chinese Restaurant Process representation of the Dirichlet Process. The CRP is effectively a collapsed version of the stick-breaking construction, where the non-interchangable cluster *labels* in the latter are marginalized over to produce a distribution over *partitions* [42] (section 2.4). Both of these facts help IWTM's Gibbs sampler to move between local optima and add new clusters.

In contrast, variational inference does not have a natural mechanism for escaping local optima, as discussed in chapter 6. In coordinate-ascent variational inference, each update is guaranteed to locally maximize (or maintain the same value of) the ELBO objective. Thus, coordinate ascent will *never* escape *any* local optimum. On the other hand, stochastic variational inference moves in the direction of noisy natural gradient estimates. Noise allows the ELBO to decrease (or increase suboptimally) sometimes and escape its current local optimum. However, in practice doing so results in *fewer* clusters being used rather than new clusters being added. This phenomenon is demonstrated in section 6.4 and again later in this chapter.

The fact that stochastic variational inference will not add components in BNP models can be explained theoretically. As observed by Bishop [10], mean-field variational inference (whether stochastic or coordinate ascent) is "zero-avoiding". That is, it systematically underestimates regions of low density in the model posterior. Variational inference minimizes $\mathrm{KL}(q\|p)$, defined as

$$\mathrm{KL}(q\|p) = \int q(\mathbf{Z}) \log q(\mathbf{Z}) - q(\mathbf{Z}) \log p(\mathbf{Z} \mid \mathbf{X}) d\mathbf{Z},$$

where $\mathbf{Z}$ denotes the latent variables of the model, $p(\cdot)$ is the true posterior, and $q(\cdot)$ is the variational approximation. By this measure, the largest difference between the two distributions occurs in configurations where $p(\cdot)$ is very small ($-\log p$ is very large) and $q(\cdot)$ is not zero. Thus, minimizing the KL divergence causes q to be set to zero in such areas. To put it another way, variational inference is unlikely to move to low probability configurations because doing so would make the KL divergence much higher. Thus, the variational objective exacerbates the issue of escaping local minima and, in the context of BNP models, all but assures that new components will not be created.

Since standard variational inference will not venture on its own into parts of the parameter space where clusters are added, another method *external* to it must be used to grow the complexity of the model. The next subsection discusses the limitations of existing work that has addressed similar issues. Following that, incremental variational inference is developed.

7.2.2 Related Work

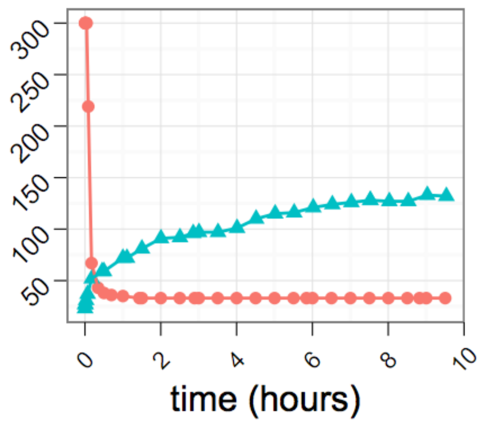
A limited amount of work has addressed issues *related to*, but not directly concerning, incremental training of Bayesian non-parametric models with variational inference. Kurihara, Welling, and Vlassis [32] devised a split-merge method to grow and shrink the truncation limit (and, effectively, the number of clusters) in a variational inference procedure for Gaussian Dirichlet process mixtures. This method has two main limitations. Primarily, the method is specific to Gaussian mixtures, and it is unclear how it could be applied to more complex models like IWTM, which contains two levels of DPs that must be dynamically resized. In contrast, incremental variational inference can be applied to a wide range of Bayesian non-parametric models. Secondarily, split-merge does not integrate with stochastic variational inference. The proposed method naturally allows a stochastic implementation, making it scalable to larger datasets.

More recently, Wang and Blei introduced a method called *truncation-free variational inference* [58] that is also capable of growing the model complexity throughout training. The key idea is to use collapsed Gibbs sampling instead of variational approximations to estimate document-level latent variables. Unlike variational inference, the Gibbs sampler can "explore" the parameter space and make moves to low-probability configurations where new components are instantiated. While general enough to apply to IWTM, the speed of truncation-free variational inference depends on the speed of the embedded Gibbs sampling steps. Wang and Blei [58] apply the method to inference in HDP-LDA, which has a very efficient collapsed Gibbs sampler. Unfortunately, this method is impractical for models like IWTM, whose Gibbs samplers are relatively slow. Indeed, the inefficiency of sampling is typically the motivation to use variational inference in the first place.

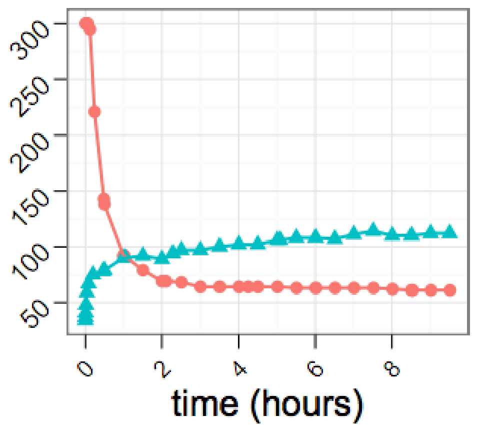
Unlike truncation-free variational inference and split-merge methods, the proposed method for growing the model complexity, *incremental variational inference* (IVI), is conceptually simple and only requires an implementation of variational inference. All inference steps are performed with variational inference, making the method fast and scalable to large datasets. The next section describes IVI in greater detail.

7.3 Incremental Variational Inference

IVI is a general method that can be used to train a wide range of Bayesian non-parametric model incrementally. This section first describes the method at a high level and gives a general recipe for implementing it. It then details how IVI is



(a) DP Mixture



(b) Hdp-Lda

Figure 7.1: Variational inference on a Bayesian non-parametric model shrinks the model's complexity over time. (a) Number of clusters used by a DP mixture model and (b) number of topics used by Hdp-Lda as training progresses with standard mean-field variational inference (dots) and truncation-free variational inference (triangles). Non-parametric models trained with standard variational inference are initialized with a large number of components and "shrink" over time, making them unsuited to incremental training. Truncation-free variational inference grows the model complexity as training progresses, but is too slow to apply to IWTM, which motivates incremental variational inference. (Figures taken from Wang, Hoffman, and Blei [58].)

implemented for IWTM specifically, and discusses its expected behavior.

7.3.1 General Method

IVI is equivalent to standard variational inference during periods where the training set size remains constant. However, when the training set grows, IVI triggers an operation called Extend that adds additional components to the model. The extended model is then trained for some number of iterations on the new, larger dataset.

The purpose of Extend is to circumvent the "paring back" phenomenon that makes standard variational inference unsuitable for growing datasets. Standard

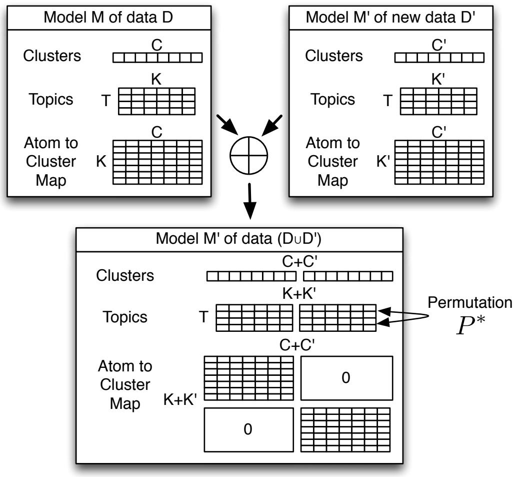


Figure 7.2: To overcome the paring back phenomenon of variational inference, IVI's EXTEND operation increases a model's complexity when new data is added to its training set. EXTEND does so by concatenating the variational parameters of the new data model $\mathcal{M}'$ with those of the existing model $\mathcal{M}$. The topics of the new model are permuted so that they best correspond to the topics of the existing model.

variational inference uses an ever-smaller number of clusters to fit a dataset as training progresses. As demonstrated later in the chapter, the number of clusters will not increase even if the dataset grows to several times its original size during training. By allowing the model complexity to grow as the dataset grows, IVI achieves vastly better fits of the data in such situations. IVI is also much faster than training a new model each time the dataset grows, since it simply extends an existing model to fit new data.

IVI is a general method for incremental training of Bayesian non-parametric models. While some details of the EXTEND operation depend on the type of model being used, a general recipe is provided for implementing it using variational inference as a subroutine. This chapter focuses on the application of IVI to IWTM. However, the method can be applied to Dirichlet process mixtures (or, more generally, Pittman-Yor process mixtures), infinite latent factor models [24], and other Bayesian non-parametric models. Implementing IVI only requires a variational inference procedure for the model of interest, and can be implemented using either batch or stochastic methods.

The recipe for Extend is conceptually simple. Given a model $\mathcal{M}$ trained on a dataset $\mathcal{D}$ and a set of new data $\mathcal{D}'$, the task is to produce an extended model $\mathcal{M}^{\text{Ext}}$ with additional components to fit the new data. To accomplish this, a separate model $\mathcal{M}'$ is first trained on the new data $\mathcal{D}'$ using variational inference. The extended model is then formed by taking the union of the components from $\mathcal{M}$ and $\mathcal{M}'$. This "union" is implemented by simply concatenating the corresponding variational parameters of $\mathcal{M}$ and $\mathcal{M}'$ and recomputing the expectations used by the inference procedure. Thus, rather than adding randomly-generated components that may not help fit the new data, the components added by Extend are specifically designed to fit such data.

The following subsection describes the implementation of Extend for IWTM.

7.3.2 Extend for IWTM

Recall that IWTM contains two levels of Dirichlet processes. Correspondingly, IVI grows IWTM's complexity in two directions – it adds additional clusters to the the master topic $\boldsymbol{\phi}_0$ as well as new atoms to the topics $\{\boldsymbol{\phi}_t\}_{t=1}^{T}$. In general, if the current model $\mathcal{M}$ has C clusters and a truncation of K atoms per topic, and the new data model $\mathcal{M}'$ has C' clusters and K' atoms per topic, Extend produces a model $\mathcal{M}^{\text{Ext}}$ with $C^{\text{Ext}} = C + C'$ clusters and $K^{\text{Ext}} = K + K'$ atoms per topic that contains the union of the clusters and atoms of the constituent models.

Because $\mathcal{M}$ and $\mathcal{M}'$ are trained with variational inference, each model state can be represented as a tuple of its variational parameters, i.e.

$$\mathcal{M} = (\tilde{\mathbf{m}}, \tilde{\boldsymbol{\nu}}, \tilde{\boldsymbol{\chi}}_1, \tilde{\boldsymbol{\chi}}_2, \tilde{\boldsymbol{\zeta}}, \tilde{\boldsymbol{\xi}}, \tilde{\boldsymbol{\beta}})$$
$$\mathcal{M}' = (\tilde{\mathbf{m}}', \tilde{\boldsymbol{\nu}}', \tilde{\boldsymbol{\chi}}_1', \tilde{\boldsymbol{\chi}}_2', \tilde{\boldsymbol{\zeta}}', \tilde{\boldsymbol{\xi}}', \tilde{\boldsymbol{\beta}}').$$

The first four sets of parameters in each model $(\tilde{\mathbf{m}}, \tilde{\boldsymbol{\nu}}, \tilde{\boldsymbol{\chi}}_1, \tilde{\boldsymbol{\chi}}_2)$ are the cluster parameters; the next two $(\tilde{\boldsymbol{\zeta}}, \tilde{\boldsymbol{\xi}})$ are pseudo-counts of the master topic and topic sticks; and the last is the topic atom to cluster mapping. For the most part, the "union" of the models can be formed simply by concatenating the corresponding sets of variational parameters together and then recomputing the expectations used by variational inference.

Cluster parameters The first four sets of parameters approximate the posterior over cluster parameters in each model. Parameters $\tilde{\mathbf{m}} \in \mathbb{R}^{N_{\text{dim}} \times C}$ and $\tilde{\mathbf{m}}' \in \mathbb{R}^{N_{\text{dim}} \times C'}$ are matrices of cluster centers, $\tilde{\boldsymbol{\nu}} \in \mathbb{R}^C$ and $\tilde{\boldsymbol{\nu}}' \in \mathbb{R}^{C'}$ are the degrees of freedoms (cluster pseudo-counts), and $\tilde{\boldsymbol{\chi}}_1, \tilde{\boldsymbol{\chi}}_2 \in \mathbb{R}^C$ and $\tilde{\boldsymbol{\chi}}'_1, \tilde{\boldsymbol{\chi}}'_2 \in \mathbb{R}^{C'}$ define the shape and scale of the cluster precisions. Concatenating each set of parameters along its last axis yields cluster means $\tilde{\mathbf{m}}^{\text{Ext}} = \tilde{\mathbf{m}} \oplus \tilde{\mathbf{m}}'$ (a $N_{\text{dim}} \times (C + C')$ matrix), degrees of freedom $\tilde{\nu}^{\text{EXT}} = \tilde{\nu} \oplus \tilde{\nu}'$ (a $(C + C')$-length vector), and precision parameters $\tilde{\boldsymbol{\chi}}_1^{\text{EXT}} = \tilde{\boldsymbol{\chi}}_1 \oplus \tilde{\boldsymbol{\chi}}'_1$ and $\tilde{\boldsymbol{\chi}}_2^{\text{EXT}} = \tilde{\boldsymbol{\chi}}_2 \oplus \tilde{\boldsymbol{\chi}}'_2$ (both $(C+C')$-length vectors). In the extended model, the original model's clusters are at indices $1 \dots C$ and the new model's clusters are at indices $(C + 1) \dots (C + C')$.

Master topic stick weights Pseudo-counts $\tilde{\boldsymbol{\xi}}$ and $\tilde{\boldsymbol{\xi}}'$ represent the posterior approximation of the master topic in each model. They define distributions over C sticks in $\mathcal{M}$ and $\mathcal{C}'$ sticks in $\mathcal{M}'$, respectively, and the goal of Extend is to combine them together to form a distribution over $C + C'$ sticks in the extended model. Doing so only requires concatenating the pseudo-count vectors together and using order-aware updates to generate the other variational parameters. Let $\tilde{\boldsymbol{\xi}}^{\text{Ext}} = \tilde{\boldsymbol{\xi}} \oplus \tilde{\boldsymbol{\xi}}'$ denote the concatenated pseudo-counts, a $(C + C')$-length vector. The stick weight parameters are then formed by applying the formulae for the order-aware updates in (5.27), i.e.

$$\tilde{u}_c^{\textsc{Ext}} = 1 + \tilde{\xi}_c^{\textsc{Ext}} \tag{7.1}$$

$$\tilde{v}_c^{\textsc{Ext}} = \gamma + \sum_{j >_{\xi} c} \tilde{\xi}_j^{\textsc{Ext}}. \tag{7.2}$$

Note that the way EXTEND combines the models' master topic stick weights is consistent with how it combines their cluster parameters: The original model's clusters are at indices $1 \ldots C$ and the new model's clusters are at indices $(C + 1) \ldots (C + C')$.

Topic stick weights Pseudo-counts $\tilde{\boldsymbol{\zeta}}$ and $\tilde{\boldsymbol{\zeta}}'$ represent the posterior approximation of the topic weights in each model. They define distributions over K atoms per topic in $\mathcal{M}$ and K' atoms per topic in $\mathcal{M}'$, respectively. EXTEND should combine them to define a distribution over $K + K'$ atoms per topic in the extended model. One challenge to doing so is that the correspondence between the two models' topics is not known. To address this issue, a correspondence between them is estimated in the following manner. First, a sample of N_{corr} documents is selected randomly from the corpus. Second, their mean topic weights are computed using both models, yielding two weight matrices W and W', both of size $N_{\text{corr}} \times T$. These matrices describe the documents in terms of each models' topics. Third, the topic correspondence is estimated using such matrices. Specifically, the optimal correspondence is defined as the $T \times T$ permutation matrix that minimizes the distortion

$$P^* = \operatorname*{arg\,min}_{P \in \pi_T} \|W - W \cdot P\|_F, \tag{7.3}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The solution P^* can be computed efficiently in $\mathcal{O}(T^3)$ time using Hungarian matching [21].

Given a correspondence between the constituent models' topics, the extended model's topics are formed in a two-step process. First, the extended model's pseudo-counts are formed by combining those of the constituent models under the optimal topic correspondence – that is, they are constructed as $\tilde{\boldsymbol{\zeta}}^{\textsc{Ext}} = \tilde{\boldsymbol{\zeta}} \oplus (\tilde{\boldsymbol{\zeta}}' \cdot P^*)$. Then, the variational topic weight parameters are computed using the order-aware update in equation (5.24). That is,

$$\tilde{a}_{tk}^{\textsc{Ext}} = 1 + \tilde{\zeta}_{tk} \tag{7.4}$$

$$\tilde{b}_{tk}^{\textsc{Ext}} = \alpha_0 + \sum_{j >_{\tilde{\zeta}_t^{\textsc{Ext}}} k} \tilde{\zeta}_{tj}^{\textsc{Ext}} \tag{7.5}$$

In the extended model, the topic atoms of the original model are at indices $1 \ldots K$ and the indices of the new model are at indices $(K+1) \ldots (K+K')$. The final step of Extend is to map the new topic atoms to their appropriate clusters, which is described next.

Topic-atom to cluster mapping The last parameters to set in the extended model are the topic-atom to cluster mappings. Parameters $\tilde{\boldsymbol{\beta}} \in \mathbb{R}_+^{K \times C}$ and $\tilde{\boldsymbol{\beta}}' \in \mathbb{R}_+^{K' \times C'}$ map the topic atoms to the clusters in the two constituent models. In the extended model, the mapping matrix $\tilde{\boldsymbol{\beta}}^{\textsc{Ext}}$ has size $(K+K') \times (C+C')$ and maps the total $(K+K')$ atom indices to all of the $(C+C')$ clusters. It is constructed by combining the mappings of the constituent model in the following block structure:

$$\tilde{\boldsymbol{\beta}}^{\textsc{Ext}} = \left[\begin{array}{c|c} \tilde{\boldsymbol{\beta}} & \mathbf{0}^{K \times C'} \\ \hline \mathbf{0}^{K' \times C} & \tilde{\boldsymbol{\beta}}' \end{array} \right]. \tag{7.6}$$

Thus, both the topic atoms from $\mathcal{M}$ and those from $\mathcal{M}'$ map to the same clusters that they did originally, but the extended model contains the union of all topic atoms and clusters.

7.3.3 Expected Behavior

As discussed previously, the purpose of IVI is to allow a Bayesian non-parametric model's complexity to grow when the training set grows, since standard variational inference will not grow it at all. However, properties of BNP priors predict that IVI will actually grow the model's complexity *too much* – that is, the DPs in the extended model will contain an artificially large number of components relative to the dataset size. The extended model contains the union of the components of the two constituent models, specifically, one that uses C clusters to fit $|\mathcal{D}|$ data and one that uses C' clusters to fit $|\mathcal{D}'|$ data. It thus uses $C + C'$ clusters to fit the combined $|\mathcal{D}| + |\mathcal{D}'|$ data. However, the expected number of components under the DP grows logarithmically, and thus *subadditively*, with the dataset size (section 2.4).[1] In other words, it pressures the model to use fewer than $C+C'$ clusters to fit $|\mathcal{D}|+|\mathcal{D}'|$ data.[2]

That EXTEND may produce models with artificially high complexity is not a major concern; when the extended model is trained subsequently, (non-incremental) variational inference can pare its complexity back. Because stochastic variational inference is particularly effective at removing extra components (section 6.4), it is used in IVI in the next section's experiments.

7.4 Incremental Variational Inference Experiments

To evaluate incremental variational inference for IWTM, the method was used to train models on the 13-scene corpus described in chapters 4-6. This experiment simulates a situation where new training documents periodically become available and are added to the model's training set.

[1]A function $f : \mathbb{R} \rightarrow \mathbb{R}$ is subadditive iff $\forall x, y,\ f(x + y) \leq f(x) + f(y)$.

[2]This argument is a little loose because IWTM is built on the HDP, not the standard DP. However, its conclusion is the same, since the expected number of clusters under the HDP is also subadditive in the dataset size.

7.4.1 Method

In an initial "burn-in" phase, each model initially has a training set of 65 documents (5 per class) on which it is trained for 100 iterations. Subsequently, additional training documents are added to the model 65 at a time until the size of the dataset reaches 650 documents. Each time new data are added, the model is extended in the manner described in the previous section. Specifically, a separate model of the new data is trained for 50 iterations and then used to extend the existing model. The combined model is then trained for 20 iterations on the augmented dataset to allow it to remove extra components.

To understand the behavior of incremental variational inference better, the method is compared to two baselines. First, to highlight the effect of extending the model, another set of models is trained using a version of IVI with the EXTEND operation ablated. We call this method IVI-NO-EXTEND. Each time new documents become available, they are simply added to the model's active training set; no new components are explicitly added to the model. As in IVI, the models are trained for additional 20 iterations each time the training set grows.

The second baseline compares IVI to *non-incremental* training. In this method, new models are trained "from scratch" for 100 iterations each time documents are added to the training set. Because training a new model does not leverage an existing model's solution to fit a larger dataset, it is expected to be more computationally expensive than IVI.

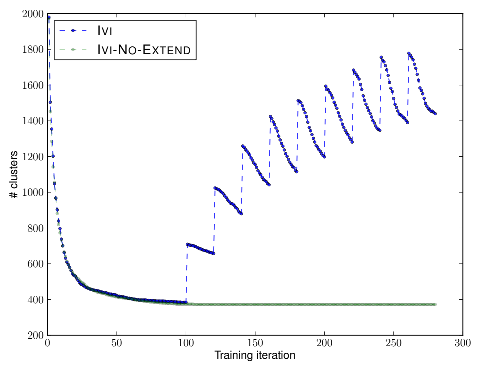
The training for all methods uses stochastic variational inference with a learning rate of 0.5 and a mini-batch size of half of the model's current document set. As discussed in the previous chapter, stochastic variational inference achieves a better fit of the data than batch variational inference, and it helps avoid getting stuck in local optima where the model uses an unnecessarily large number of components. IWTM is initialized in the same manner and run with the same hyperparameter settings as in chapters 5 and 6.

Model performance is evaluated by looking at the value of the ELBO, i.e. the objective being optimized by variational inference. It is a lower bound on the marginal log-likelihood of the data under the model. Larger values are better and indicate that the mean-field approximation is closer in KL-divergence to the true model posterior.

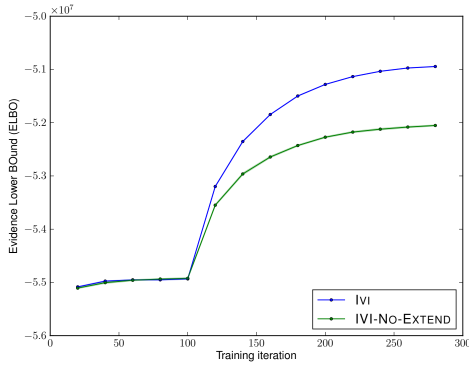
To assess training times for the two methods, all models are trained on the same reference machine (Dell Precision-T3400, dual 3.15 GHz Intel processors and 2 GB of RAM). In these experiments, variational inference runs in a single thread on only one processor. However, as discussed in section 5.4.6, document-level inference can be distributed easily over multiple cores or multiple machines to achieve significant speedups.

7.4.2 Results

Figure 7.3 shows the results for the two incremental training methods: Ivi and the Ivi-No-Extend baseline. The difference between the two methods can be seen in figure 7.3(a), which shows the complexity of the respective models as incremental training progresses. Both methods start with a large number of clusters and use progressively fewer over time during the burn-in period. However, the methods diverge when the training set begins to grow at iteration 100. In Ivi-No-Extend the number of clusters continues to decrease monotonically throughout training, even as the dataset grows to several times its original size. As discussed earlier in this chapter, this paring back behavior is standard in variational inference. Ivi avoids such behavior by forcing the model to increase in complexity whenever the training set grows. Although each Extend operation adds more components than are necessary to fit the larger dataset, the extra components are removed in subsequent iterations of training. The result is that the number of clusters used by Ivi over



(a)



(b)

Figure 7.3: Numbers of clusters (left) and mean ELBOs (right) of Ivi and the Ivi-No-Extend baseline as training progresses. (Error bars are present on the right, but too small to be seen.) Both methods first train for 100 iterations on a seed set of 65 documents; then 65 new documents are added every 20 iterations. Ivi gets progressively better ELBOs because the Extend operation grows the model complexity each time new documents are added. In the baseline, the number of clusters shrinks over time, which is the standard behavior of variational inference.

time exhibits a jigsaw pattern. At the end of training, Ivi uses around 1400 clusters to fit the final, 650-document dataset, whereas the baseline uses around 380.

Figure 7.3(b) shows the ELBOs of both methods, measured on the full dataset, as incremental training progresses. Ivi achieves a significantly better fit than the baseline. Although Ivi-No-Extend does improve as new documents are added to its training set, Ivi always improves more because its model complexity grows with the training set size. Note that the gap in the ELBOs of the two methods widens each time the training set grows. The baseline performs increasingly poorly because it has a limited number of clusters with which to fit a growing set of data.

Table 7.1 compares the speed and fit of IVI and retraining from scratch. Training incrementally with IVI takes a fraction of the time that training a new model does because the former only has to extend an existing model to fit a larger dataset. For example, training a new model on 130 documents takes over an hour,

	Train time (min.)		ELBO	
# training docs	From scratch	Ivi (delta)	From scratch	Ivi
65	15	–	-5.735×10^6	–
130	64	6.8	-1.122×10^7	-1.126×10^7
195	102	14	-1.640×10^7	-1.645×10^7
260	139	21.4	-2.151×10^7	-2.155×10^7
325	171	28.4	-2.644×10^7	-2.645×10^7
390	196	34.5	-3.148×10^7	-3.148×10^7
455	228	41.8	-3.644×10^7	-3.642×10^7
520	251	48.3	-4.138×10^7	-4.134×10^7
585	287	53.2	-4.626×10^7	-4.622×10^7
650	415	64	-5.100×10^7	-5.094×10^7

Table 7.1: Results of the incremental training experiment. Each row shows the training times and ELBOs achieved by incremental variational inference and by training new models "from scratch" each time the dataset grows. Ivi achieves nearly identical fits of the data while taking only a fraction of the time of training new models.

but training a model of 65 documents on an additional 65 documents with IVI takes only 6.8 minutes, which is almost *ten times* as fast. Likewise, training on the full 650 documents takes 415 minutes, but adding 65 documents to an existing 585-document model with IVI takes only 64 minutes, which is nearly six and a half times as fast. At the same time, the ELBOs of models trained with IVI and from scratch are very close. IVI actually performs slightly better as time goes on, probably because it has has trained on the oldest documents for many iterations.

In general, incremental training is fast for two reasons. First, extending the model itself is fast: Doing so only requires training a separate model on the new data, which is typically a fraction of the existing dataset. Second, the extended model only needs to be trained on the full dataset for a small number of stochastic variational inference iterations to allow extra components to be pared back.

7.5 Active Learning

The ability to train models incrementally is a vital tool for tasks where new data periodically become available for training. However, in many real-world situations, the volume of new data may be far larger than that on which the model can feasibly be trained, even with scalable methods like stochastic variational inference. Instead of using all new data, it will generally be necessary to sample new examples from the data stream. In this case, it is desirable to pick examples that are most useful for training, rather than ones already well understood by the existing model. Furthermore, if the topic model is being used to support a supervised learning task, the cost of labeling the new documents must be considered. Rather than labeling all new examples, it desirable to sample a subset of examples whose labels would be useful for classification.

To address such challenges, an extension of incremental variational inference based on *active learning* was developed in this dissertation. Active learning is typically performed with a classifier alone, where the goal is to maximize performance by acquiring labels for as few examples as possible. In the type of active learning considered in this dissertation, the system is actually a combination of a topic model and a classifier. In addition to minimizing the number of training examples that must be labeled, a goal is minimize the training time (or computational resources) required to train the topic model. Documents are actively selected both for labeling and inclusion in the topic model's training set. Such a method can be thought of as a special case of incremental variational inference where a classifier selects a subset of new documents for inclusion in the training set. The key idea is that, by choosing only the most informative examples (according to some criterion), the model-plus-classifier pair can achieve high performance with relatively small amounts of training data.

The proposed method is based on *pool-based active learning* [47]. In this sce-

Algorithm 2 Pool-based active learning with uncertainty sampling

```
Select a small initial training set $\mathcal{D}_{\text{train}}$.
Train the model $\mathcal{M}$ on $\mathcal{D}_{\text{train}}$.
while more labeled documents are needed do
    Run inference on unlabeled documents $\mathcal{D}_{\text{pool}}$ to generate features
    Set $\mathcal{D}'_{train}$ to top documents ranked by $S(\cdot)$.
    Set $\mathcal{D}_{\text{train}} \leftarrow \mathcal{D}_{\text{train}} \cup \mathcal{D}'_{train}$ and $\mathcal{D}_{\text{pool}} \leftarrow \mathcal{D}_{\text{pool}} \backslash \mathcal{D}'_{train}$
    Train the model on the new $\mathcal{D}_{\text{train}}$ for $n_{\text{train}}$ more iterations.
end while
```

nario, the system is given access to a pool of *unlabeled* documents and can iteratively select new examples to be labeled and included in the training set. Because the goal is to train models with high classification accuracy, it should be most beneficial to label documents that the classifier is most uncertain about labeling. This particular form of active learning is called *uncertainty sampling* [47]. Intuitively, uncertainty sampling should select documents that are different as possible from those already in the training set, at the exclusion of easy examples that would provide little new discriminative information to the model.

Algorithm 2 describes the pool-based uncertainty sampling algorithm for IWTM. The model is initially trained on a small amount of labeled data. Then, at each iteration of active learning, the model and an external classifier work in concert to select new examples to add to the model's training set. To select new examples, the model is first used to compute features for both the labeled training examples and the unlabeled pool. A classifier is trained on the labeled examples and used to score the examples in the unlabeled pool. (All examples are represented using the features generated from the model). The top-scoring example is then labeled and added to the training set. The classifier can then be re-trained and used to select another example, as necessary.

In this topic model active learning method, new training examples are selected using the *margin sampling* criterion, defined as

$$S(\mathbf{x}) = p(y = \hat{y}_2 \,|\, x) - p(y = \hat{y}_1 \,|\, x)$$

where $\hat{y}_1$ and $\hat{y}_2$ are the most and second-most likely class labels according to the classifier. By construction, it selects examples that lie near a decision boundary in the classifier, i.e. ones that the classifier is uncertain about how to label.

The active learning algorithm described above is fairly general. The procedure can be used with any topic model that can be trained incrementally and any classifier from which class label probabilities can be derived. Thus, both IWTM and LDA may benefit from active learning. However, one appealing property of IWTM is that the model can adapt its vocabulary and vocabulary size as more documents are added to the training set; LDA, on the other hand must use a fixed vocabulary defined offline by K-means. Thus, IWTM is expected to achieve higher gains in accuracy as a result of actively selecting the dataset.

7.6 Active Learning Experiments

Active learning for IWTM is a special case of incremental variational inference where new training documents are selected by a classifier. It is therefore evaluated on the 13-scene dataset using an experimental setup similar to that of section 7.4.

7.6.1 Method

As in the incremental training experiments, IWTM is first trained on a seed set of 65 documents for 100 iterations. Subsequently, 65 new training documents are periodically selected from an unlabeled pool, labeled, and added to IWTM's training set. In one set of models (IWTM active), the new documents are selected by the classifier's margin criterion using the method described in the previous section. As

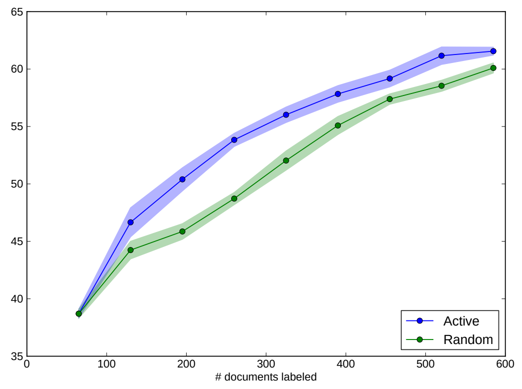


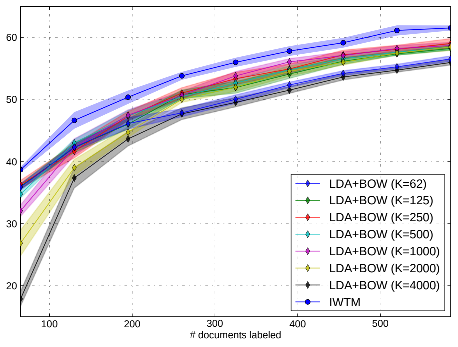
Figure 7.4: Active learning for IWTM versus random selection. Actively selecting new labeled training documents results in higher classification accuracy with smaller amounts of data than random selection, validating the topic model active learning method.

a baseline, a second set of models (IWTM random) receive a random sample of new documents from the unlabeled pool. For both selection strategies, IWTM is extended using incremental variational inference each time the training set grows. As in the previous experiments, the new data model is trained for 50 iterations of stochastic variational inference and, after merging, the extended model is trained for 20 additional iterations.

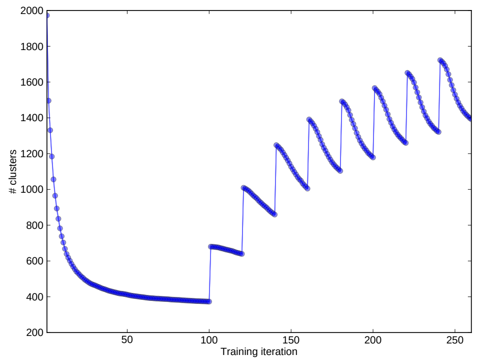
As a second baseline, active learning for IWTM is compared to a similar procedure for LDA. The vocabulary size is varied $K \in \{62, 125, 250, 500, 1000, 2000, 4000\}$. As in previous chapters, a one-versus-all SVM classifier is trained on topic weights. The SVM soft-margin parameter C for each classifier is selected using cross-validation. New documents are selected using the margin criterion. Both models have $T = 25$ topics. The unlabeled pool contains 1300 unlabeled documents.

7.6.2 Results

Figure 7.4 shows the results of active and random selection in IWTM. Models whose training sets were selected using active learning achieved significantly higher accu-



(a) IWTM active vs. LDA active



(b) Number of clusters in IWTM

Figure 7.5: (left) Test accuracies of IWTM and LDA during active learning. (right) Number of clusters used by IWTM as training progresses. Each LDA model uses a static visual vocabulary that is fixed the start of training. The relative performance of different vocabulary sizes varies as the training set grows; some that are initially near-optimal (e.g., $K = 62$) perform poorly later. In contrast, IWTM adapts its clustering as training progresses. As a result, it consistently outperforms all LDA models for the entirety of the experiment.

racy across the entire run of the experiment. The performance difference between active and random selection starts small and gradually increases as more examples are added to the training set. At 260 labeled documents, active selection has its greatest gains over the baseline, achieving an 5.1% higher absolute accuracy. The margin between the two methods begins to close towards the end of the experiment. This is for two reasons. First, the accuracy of any classifier, as a function of training set size, naturally asymptotes. As the training set grows, additional examples provide smaller marginal gains. Second, active learning is a greedy algorithm and selects the most discriminative training examples in early iterations. Doing so leaves a smaller pool of less informative examples for later iterations, making active selection progressively less effective. This degradation can be mitigated by providing the learner with a larger unlabeled pool.

Figure 7.5(a) compares the performance of active learning in IWTM and LDA. LDA's accuracy varies to some degree with a combination of the vocabulary and training set sizes. One important observation is that the relative performance of a particular vocabulary size sometimes changes as the dataset grows. For example, on the initial training set, LDA does well with a very small vocabulary ($K = 62$) and worse with larger vocabularies such as $K = 500$ or $K = 1000$, which overfit. However, as more documents are added to the training set, the small vocabulary models fall to among the worst (because they now underfit the data), while the larger vocabulary models become the best. This phenomenon complicates the selection of an optimum vocabulary size for active learning in LDA. Indeed, since different vocabulary sizes "peak" with different amounts of data, there generally will not be a vocabulary size that is optimal during the entire course of active learning.

In contrast, IWTM has the ability both to pick a vocabulary size that fits the initial training set and, thanks to IVI, change it dynamically as the dataset grows. Figure 7.5(b) shows how the number of clusters changes over time during the course of active learning. IWTM uses approximately 400 clusters to fit the initial 65 documents and about 1400 for the final dataset. As a result, IWTM maintains about 2-4% better absolute accuracy than LDA – with *any* vocabulary size – across the entirety of the experiment. Another interpretation of the same results is that IWTM can achieve the same level of performance as LDA using a smaller number of documents. For most iterations, IWTM matches the accuracy of the best LDA model with about 65 more documents. Towards the end of the experiment, IWTM matches the best LDA model overall using about 130 fewer training documents.

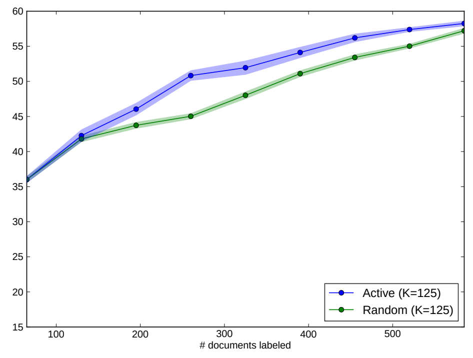
Although LDA's active learning performance is outmatched by IWTM's, active selection in LDA is indeed effective. Figure 7.6 compares the performance of LDA under active learning and random selection, breaking down the results by different vocabulary sizes. In a few cases ($K = 125, 250, 2000$), active learning is no more effective than random selection until the training corpus has grown to 200 documents. These cases occur when the training set is small and the vocabulary is very large or very small. Such results suggests that over- or under-fitting the visual vocabulary leads to poor quality topic representations, which makes it difficult for the classifier to identify informative unlabeled examples.

Nevertheless, actively selecting the training set nearly always yields higher test accuracy than choosing new examples at random, regardless of the vocabulary size used. Thus, this style of active learning – with a combination of an unsupervised model and external classifier – is not specific to IWTM and may be useful for topic modeling in general. In particular, active learning with LDA might be useful for subsampling large text corpora, where the vocabulary size is static.

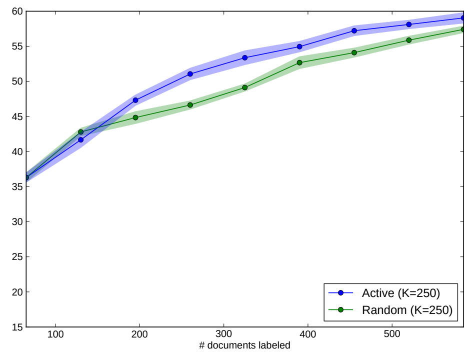
7.7 Conclusions

This chapter presented two contributions of this dissertation that help accommodate the challenge of learning on growing datasets. Incremental variational inference allows Bayesian non-parametric models to be trained efficiently on growing datasets while supporting the property of them that makes them so powerful – namely, that the model complexity is a random quantity that grows dynamically with the dataset size. IVI is a general method that can be applied to a variety of BNP models based on stick-breaking constructions. Furthermore, it is simpler to implement than existing methods, needing only an implementation of (stochastic) variational inference for the model of interest. Building on variational inference in such a design makes IVI fast and allows document-level inference to be parallelized over multiple cores or machines. Active learning is one application of IVI in supervised learning settings. Rather than using adding all available documents, it intelligently selects new ones to add to the topic model and classifier so that the system performs better. Doing so allows the system to perform better while using a smaller training set.

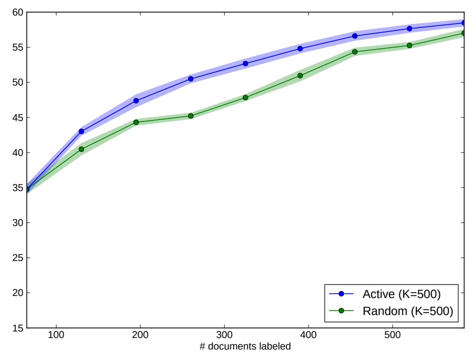
Overall, the results in this chapter and chapter 6 establish that powerful models like IWTM, despite their additional complexity, can be trained efficiently



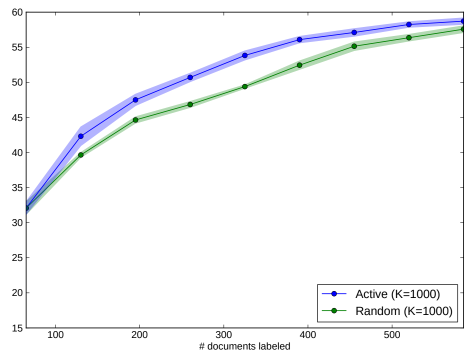
(a) K=125



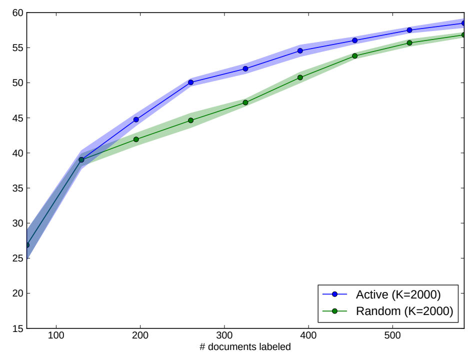
(b) K=250



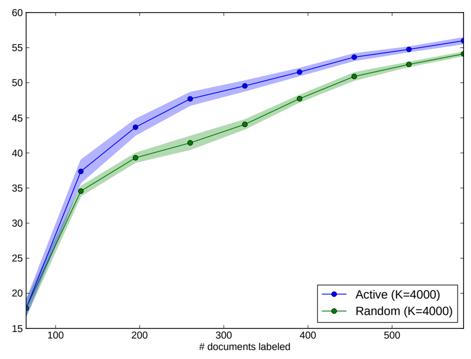
(c) K=500



(d) K=1000



(e) K=2000



(f) K=4000

Figure 7.6: LDA active learning vs. random selection for different vocabulary sizes. When the vocabulary size is too small or large relative to the initial training set ($K = 125, 250, 2000$), active learning must go through multiple iterations before it is more effective than random selection. These exceptions notwithstanding, active learning almost always outperforms random selection, regardless of the vocabulary size used. Such results demonstrate that active learning with topic models is effective beyond IWTM, and suggests that it may be applied to topic models in general.

and at scale. The next chapter shifts to reviewing the lessons learned during the development of this research and discussing avenues for future work.

Chapter 8

Discussion and Future Work

Previous chapters described the Infinite-Word Topic Model, its inference procedures, and tools to handle challenges of real-world applications, including partially labeled and growing datasets. These contributions constitute progress towards the larger goal of automated, large-scale analysis of digital media, but there is still more work to do. This chapter discusses some directions for future work, including model extensions, improvements to the inference procedure, and further applications of the methods developed in this dissertation.

8.1 Infinite Word Infinite Topic Models

A key feature of IWTM is its ability to infer the visual vocabulary size to use based on the data. One issue not addressed in this dissertation is how to select the number of *topics* in the model. Experimentally, its effect seems to be less important than vocabulary size; nonetheless, the number of topics must be chosen somehow, and ideally it should be done in a principled way. Can IWTM be extended to infer the number of topics as well as the vocabulary size?

Such a model, dubbed the Infinite Word Infinite Topic Model (IWITM), can be developed, although it is more complex than IWTM. It consists of an infinite set of topics over an infinite-sized vocabulary of clusters, where the same clusters are shared among all topics, and the same topics are shared among all documents. Generating such structure requires a two-level hierarchy of Dirichlet processes. It is written as

$$\begin{aligned} G_0 &\sim \mathrm{DP}(\gamma H) \\ G_1 &\sim \mathrm{DP}(\alpha \mathrm{DP}(\gamma G_0)). \end{aligned} \tag{8.1}$$

Here, as in IWTM, H is a prior over the cluster parameters. Random variable G_0 is an infinite mixture over cluster parameters drawn from H. Random variable G_1 is also an infinite mixture; each of its atoms is a draw from $\mathrm{DP}(\gamma G_0)$ and is thus a reweighted copy of G_0. Thus, G_1 is an infinite mixture over reweighted copies of G_0. In the language of IWTM, it is an infinite collection of *topics* over an infinitely-sized vocabulary. The construction in equation (8.1) is a type of Nested Dirichlet Process [44]. Specifically, it is a Nested Hierarchical Dirichlet Process, a construction recently proposed to build topic models with tree-structured topics [41].

The remainder of IWITM generates a mixture of topics for each document, a topic for each feature, a cluster from the feature's selected topic, and, finally, the datum itself:

$$\begin{array}{llll} G_d \mid \rho, G_1 & \sim \mathrm{DP}(\rho G_1), & d \in 1 \ldots N_{\mathrm{doc}}, & \text{(per-document topic mixture)} \\ \boldsymbol{\phi}_{d,i} \mid G_d & \sim G_d, & i \in 1 \ldots N_d, & \text{(topic)} \\ \Omega_{d,i} \mid \boldsymbol{\phi}_{d,i} & \sim \boldsymbol{\phi}_{d,i}, & i \in 1 \ldots N_d, & \text{(cluster parameters)} \\ \mathbf{x}_{d,i} \mid \Omega_{d,i} & \sim F(\Omega_{d,i}), & i \in 1 \ldots N_d. & \text{(data)} \end{array}$$

Unlike IWTM, IWITM does not model the per-document topic weights explicitly; rather, they are implicitly the weights of G_d, which is a document-specific mixture over topics. All documents share the same topic definitions because, by construction, the atoms of G_d are inherited from G_1, the global topic mixture.

Like IWTM, IWITM can be expressed using only conjugate exponential family distributions by applying the stick-breaking construction to each DP. An efficient stochastic variational inference procedure can therefore be derived for it. Due to the need to maintain a large number of topics and clusters simultaneously, stochastic variational inference for D-IWTM would benefit greatly from the simulated reordering technique that was developed for IWTM. In short, IWITM is a natural extension of several parts of this dissertation, making it possible to infer both the vocabulary and topics nonparametrically. This extension will be explored in future work.

8.2 Flexible Training

In general, topic modeling of digitial media – whether with IWTM or with LDA and K-means – is more computationally demanding than topic modeling of text corpora. The reason is that the documents' local features must be clustered before a topic structure on them can be learned. That is, clustering is computationally expensive, but the topic modeling itself is much less so. The difference between the two steps is seen most clearly in the combination of LDA and K-means. For example, running K-means on the descriptors of 1300 images from 13-scene typically takes seven to ten hours, depending on the value of K, while LDA itself takes around ten minutes to train on the resulting bags of words. The division of labor is less clear in IWTM, which performs both clustering and topic modeling in a single model. However, profiling the inference procedure shows that clustering-related operations consume most of the CPU time. In particular, the most computationally demanding operation is computing the expected log likelihood of each datum with respect to each cluster.

When using LDA and K-means, there is a straightforward way to speed up the modeling process: train K-means for fewer iterations. (Of course, doing so may hinder the performance of the topic model, but it will still be faster.) Is there a similar way to speed up IWTM? That is, can the clustering portion of the model be turned off after some number of iterations of inference?

The answer is yes, and the solution to doing so involves an optimization already present in the variational inference implementation. "Turning off" the clustering layer of IWTM amounts to fixing the estimates of the cluster parameters ($\tilde{\mathbf{m}}, \tilde{\nu}, \tilde{\chi}_1$, and $\tilde{\chi}_2$) to their current values. By fixing the clusters, the expected data log likelihoods $\mathbb{E}[\log p(\mathbf{x}_{di} \mid \boldsymbol{\mu}, \boldsymbol{\lambda})]$ become fixed as well. That is, there is no point in recomputing them in each iteration of inference – they can be cached. Caching *all* of the expected log likelihoods, of course, would be very expensive: There are $N_d \times C$ of them per document, and storing them for thousands of documents would require many gigabytes of storage. Fortunately, there is no need to do so because, as discussed in section XXX, the inference procedure only uses the $C_{\text{top}} \ll C$ most likely clusters per datum. The top log likelihoods are stored in an efficient sparse matrix format using $\mathcal{O}(N_d)$ memory per document, where entries not explicitly represented as treated as $-\infty$. When the clustering portion of IWTM is turned off, these matrices effectively become the new representation of the documents. They only need to be computed once, and then can be cached in memory or piped to and from secondary storage. Such a representation can be thought of as a probabilistic generalization of bag of words. Indeed, when $C_{\text{top}} = 1$, the sparse matrix representation becomes equivalent to a bag of words, and the inference procedure assigns each datum entirely to its most likely cluster.

Optimizations like the one just described require some technical skill to implement, but are not theoretically challenging to derive. In general, the point is that using more complex probabilistic models does not have to translate to an intolerably slow inference procedure as long as inference is implemented in a smart manner.

8.3 Applications

This section describes future applications that are made possible by various parts of this dissertation but that have not yet been explored.

8.3.1 Incremental Variational Inference

This dissertation developed incremental variational inference (IVI), a general method for training Bayesian non-parametric models incrementally with variational inference. IVI is a powerful method: It allows BNP models to be trained efficiently on large, growing datasets while preserving their ability to grow the model complexity with the dataset size.

IVI has two main advantages over truncation-free variational inference (TFVI) [58], the only existing work that addresses this problem for general BNP models. First, IVI is simpler to implement. TVFI requires embedding a Gibbs sampler to estimate document-level latent variables inside of variational inference, while IVI only requires a variational inference procedure for the model of interest. Second, IVI is faster and more scalable than TFVI. The latter's Gibbs sampling steps are slow for complex models like IWTM and must be performed serially. In contrast, IVI is built on variational inference, which is fast and allows document-level inference to be distributed among multiple cores or machines.

Although IVI so far has been applied only to the Infinite-Word Topic Model, similar procedures can be built for a wide class of Bayesian non-parametric models; section 7.3 provided a general recipe for doing using variational inference (or stochastic variational inference) as a subroutine. IVI for Gaussian Dirichlet Process mixtures was already developed in the course of this research, although it was not evaluated in this dissertation. Another candidate application for IVI is HDP-LDA [53]. IVI would allow HDP-LDA to be trained incrementally while using more *topics* as its corpus grows. The resulting system could, for example, receive batches of new news articles on an hourly basis and automatically add new topics for emerging issues that it discovers.

8.3.2 Active Learning

In section 7.5, an active learning method was developed. The purpose was to do better on a supervised learning task such as natural scene classification. Active learning demonstrates that models do not always have to be trained on *bigger* datasets to perform better; instead, they can be trained on a relatively small number of informative examples.

Results of this sort are curious in an age of big data. As datasets continue to grow to unprecedented sizes, it sometimes seems necessary to develop models and inference methods that scale along with them. Active learning shows that such scaling is not always necessary, at least when the goal is a supervised task: By quantifying how much each example will improve the task performance, the useful information in a large dataset can be distilled into small set of its most informative examples. This observation begs the question – what if there is no supervised task? The goal of modeling might be, for example, exploratory data analysis, clustering, or dimensionality reduction. In such cases, is it possible to distill large datasets into small samples without resorting to random sampling?

Although intelligent subsampling is a difficult (and possibly ill-posed) problem in general, the methods developed in this dissertation allow doing it with topic models. In the future, this approach can be developed into a more general method of *unsupervised active learning.* It would behave in the same way as the method in section 7.5, except it would select documents according to an unsupervised metric, i.e. one that does not require training documents to be labeled. In particular, it could use metrics that are functions of documents' topic weights, which are mid-level representations of their content. For example, new training documents could be selected on the basis of how much their topic weights differ from documents already in the training corpus. Such a criterion could conceivably construct small datasets with high coverage, i.e. ones with documents from many semantically distinct categories. It would also be an intelligent way to generate subsets without duplicates or near-duplicates. If a natural scene database contains several similar pictures of the same river, for example, there is probably little point in training the model on all of them.

In addition to developing unsupervised active learning, in the future the existing, *supervised* active learning method could be applied to datasets where it will be more effective. While results on 13-scene are promising, active learning would have a more dramatic effect on the SUN task (section 6.5), where the amount of data available for training is extremely large.

8.3.3 Topic-Based Segmentation

IWTM's ability to group together semantically-related image patches (section 6.5) suggests that its features may be useful for other image analysis tasks, in particular image segmentation. Such an idea is supported by existing work where topic models have been applied successfully to segmentation. Two examples are the spatially-coherent latent topic model (Spatial-LTM) [17] and the topic random field (TRF) model [63]. In both models, the most likely topic assignment of each image region is used as its segment label. Because these topic models were designed with segmentation in mind, they have additional layers in their graphical models that enforce spatial coherence in the topic labeling; that is, adjacent image regions are *a priori* more likely to be assigned to the same topic. In future work, spatial coherence could also be added to IWTM to make it more suitable for segmentation. The resulting model would inherit the strengths of IWTM, including its ability to infer the feature vocabulary non-parametrically. Furthermore, a stochastic variational inference procedure could be developed for it by modifying IWTM's, allowing it to be trained efficiently and at scale.

8.4 Conclusions

Although the contributions of this dissertation constitute progress towards automated, large-scale analysis of digital media, there are many avenues for future work. Some, but not all, have been enumerated in this chapter. In particular, applications in non-text domains besides images have not been considered, although the widespread attention topic modeling has attracted suggests there would be many. Additionally, the Infinite-Word Topic Model itself can be extended in an open-ended manner: It can be embedded inside of other models that have not yet been envisioned to discover more complex types of latent structure in the data.

The next and final chapter concludes. It reviews the contributions made by this dissertation and puts them in perspective of the larger goal of automated content analysis.

Chapter 9

Conclusions

This final chapter summarizes the contributions of this dissertation: the Infinite-Word Topic Model, the inference procedures used to train it, and the tools that allow it to accommodate the challenges of real-word settings. It then broadly relates this dissertation to the field's goal of automated content analysis.

9.1 Summary of Contributions

Infinite-Word Topic Model The Infinite-Word Topic Model itself is the primary contribution of this dissertation. IWTM is motivated by the shortfalls of existing methods for topic modeling in non-text domains, typified by Latent Dirichlet Allocation and K-means. IWTM has a number of advantages over such an approach. Rather than performing clustering as a separate preprocessing step, it clusters data probabilistically and integrates clustering into the topic model itself. By making use of the hierarchical Dirichlet process, IWTM infers an appropriate number of clusters to use, based on the size and complexity of the dataset, and allows sharing between topics. Altogether, IWTM is a simpler and more powerful method for topic modeling in non-text domains than existing methods. Experiments show that it achieves higher performance than LDA and K-means while streamlining the modeling process and obviating the need to tune the vocabulary size. While this dissertation focuses on the application of IWTM to images, the model is sure to have use in other digital domains such as speech, audio, and video analysis.

Stochastic Variational Inference for IWTM This dissertation developed a stochastic variational inference procedure for IWTM that is fast, easily parallelizable, and scalable to large datasets. (The general theory of stochastic variational inference is not novel to this dissertation, but rather originates from recent work by Hoffman et al. [26].) As demonstrated experimentally, SVI makes IWTM approximately as efficient to train as LDA and K-means where multiple values of K are evaluated. Efficiency does not deteriorate the quality of the models produced by SVI: IWTM still outperforms LDA, regardless of the vocabulary size used in the latter. Perhaps most importantly, SVI allows IWTM to be trained on large datasets like SUN, with thousands of images and millions of patch-level features.

Building efficient variational-inference procedures for large-scale Bayesian non-parametric models has certain challenges. In particular, models that contain dependent Dirichlet processes with many thousands of atoms each (like IWTM) require an efficient method for storing and accessing the mappings between DPs. This dissertation developed a new technique for doing so, wherein mapping parameters are shared between DPs and their sticks undergo simulated reordering. In IWTM, this technique makes inference faster and reduces its memory overhead by a factor of T (i.e. the number of topics; $T = 100$ in the SUN experiments). It is vital to scaling IWTM to large datasets. In the future, it could also be used to improve inference procedures for other large HDP-based models, such as HDP-LDA with many topics.

SVI is also an improvement over coordinate-ascent variational inference (CAVI) for IWTM. Not only is SVI faster than coordinate ascent, it actually produces models that fit the data better. Each iteration of SVI examines a fraction of the documents in the corpus and is thus faster and more memory-efficient than CAVI. Unlike CAVI, SVI has the ability to escape (some) local optima, thanks to noise in its natural gradient estimates. As a result, SVI produces smaller models that achieve a significantly better fit of the data.

Incremental variational inference This dissertation developed incremental variational inference, a general method for training Bayesian non-parametric models *incrementally*. In general, mean-field variational inference is unsuitable for incremental training because the model cmplexity decreases monotonically over time. BNP priors assume that the model complexity should grow as a function of dataset size, however, variational inference always shrinks it over time, and will not grow it even if new data are added to the training set. IVI overcomes this limitation by forcing the model complexity to grow when the training set grows. Experiments show that incremental training with IVI takes a fraction of the time that training a new model from scratch does, and it produces fits of the data that are just as good. IVI enables BNP models to be trained incrementally within the fast, scalable framework of variational inference. Thus, it helps clear a roadblock to the large-scale application of BNP models.

Active learning with topic models Building upon incremental variational inference, this dissertation developed a method for active learning in topic models. Rather than training on all new data, an external classifier is used to pick out the most informative examples to train on. Experiments demonstrated that active learning in both IWTM and LDA outperformed random selection. While LDA's active learning performance depends on the size of the training set and visual vocabulary, IWTM achieved higher accuracy than all LDA models. This result is due in part to IWTM's ability to grow its complexity as new training documents are added. The fact that active learning was successful in both IWTM and LDA suggests that it can be applied to a broad class of topic models.

9.2 Closing Thoughts

Topic models have become a powerful, general purpose tool for learning latent structure in document collections. This dissertation overcomes several hurdles to their large-scale deployment in non-text domains. IWTM itself helps extend topic modeling to such domains by removing assumptions that do not make sense for them – primarily, that documents are collections of discrete, mutually exclusive word indicators from a fixed vocabulary. Doing so allows IWTM to match or exceed the performance of existing methods while simultaneously simplifying the modeling process. Stochastic variational inference makes IWTM approximately as fast to train as LDA with bags of words, and allows the model to be applied to large datasets. Finally, incremental variational inference and active learning address another challenge of real-world applications: training models efficiently on growing datasets.

In conclusion, IWTM uses statistical structure within a corpus to infer semantic representations of documents. It is flexible and can be used to analyze collections of video, audio, and images. Although the model is more complex than existing ones like LDA, it can still be trained efficiently and at scale. Therefore, although machines are still far from the ultimate goal of automated content analysis, this dissertation takes a step in that direction.

Bibliography

[1] Flickr blog: 6,000,000. `http://blog.flickr.net/en/2011/08/04/6000000000/`.

[2] YouTube press statistics. `http://www.youtube.com/t/press_statistics`.

[3] D. Aloise, A. Deshpande, P. Hansen, and P. Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75(2):245–248, 2009.

[4] Shun-ichi Amari. Differential geometry of curved exponential families-curvatures and information loss. *Annals of Statistics*, pages 357–385, 1982.

[5] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[6] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43, 2003.

[7] C.E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Satistics*, pages 1152–1174, 1974.

[8] Annalisa Barla, Francesca Odone, and Alessandro Verri. Histogram intersection kernel for image classification. In *Image Processing, 2003. ICIP 2003.*

Proceedings. 2003 International Conference on, volume 3, pages III–513. IEEE, 2003.

[9] Dimitri P Bertsekas. *Nonlinear programming.* Athena Scientific, 1999.

[10] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer, 1st ed. 2006. corr. 2nd printing edition, October 2007.

[11] David Blackwell and James B MacQueen. Ferguson distributions via pólya urn schemes. *The annals of statistics*, pages 353–355, 1973.

[12] D Blei and J McAuliffe. Supervised topic models. *Advances in Neural Information Processing*, 2008.

[13] D.M. Blei and M.I. Jordan. Modeling annotated data. In *ACM SIGIR*, 2003.

[14] DM Blei, AY Ng, and MI Jordan. Latent Dirichlet Allocation. *JMLR*, 2003.

[15] Léon Bottou. Online learning and stochastic approximations. *On-line learning in neural networks*, 17:9, 1998.

[16] W Buntine and M Hutter. A Bayesian review of the Poisson-Dirichlet process. *Arxiv preprint arXiv:1007.0296*, 2010.

[17] Liangliang Cao and Li Fei-Fei. Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes. In *International Conference on Computer Vision (ICCV) 2007.* IEEE, 2007.

[18] George Casella and Roger L Berger. *Statistical inference*, volume 70. Duxbury Press Belmont, CA, 1990.

[19] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[20] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition*, 2005.

[21] Jack Edmonds and Richard M Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

[22] L Fei-Fei and P Perona. A Bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition*, 2005.

[23] D. Grangier, F. Monay, and S. Bengio. A discriminative approach for the retrieval of images from text queries. In *European Conference on Machine Learning*, 2006.

[24] Thomas L Griffiths and Zoubin Ghahramani. The Indian buffet process: An introduction and review. *Journal of Machine Learning Research*, 12:1185–1224, 2011.

[25] Greg Hamerly. Making k-means even faster. In *SDM*, pages 130–140, 2010.

[26] Matt Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *arXiv preprint arXiv:1206.7051*, 2012.

[27] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning*, volume 99, pages 200–209, 1999.

[28] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python. `http://www.scipy.org/`.

[29] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

[30] K Kurihara, M Welling, and YW Teh. Collapsed variational Dirichlet process mixture models. In *International Joint Conference on Artificial Intelligence*, volume 20, page 19, 2007.

[31] Kenichi Kurihara, Max Welling, and Yee Whye Teh. Collapsed variational dirichlet process mixture models. In *IJCAI*, volume 7, pages 2796–2801, 2007.

[32] Kenichi Kurihara, Max Welling, and Nikos A Vlassis. Accelerated variational dirichlet process mixtures. In *Advances in Neural Information Processing Systems*, pages 761–768, 2006.

[33] Simon Lacoste-Julien, Fei Sha, and Michael I Jordan. Disclda: Discriminative learning for dimensionality reduction and classification. In *Advances in Neural Information Processing Systems*, pages 897–904, 2008.

[34] D. Larlus and F. Jurie. Latent mixture vocabularies for object categorization and segmentation. *Image and Vision Computing*, 27(5):523–534, 2009.

[35] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178. IEEE, 2006.

[36] L. Li, M. Zhou, G. Sapiro, and L. Carin. On the Integration of Topic Modeling and Dictionary Learning. In *International Conference on Machine Learning*, 2011.

[37] J.S. Liu. The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association*, pages 958–966, 1994.

[38] Stuart Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.

[39] D.G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.

[40] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[41] John Paisley, Chong Wang, David M Blei, and Michael I Jordan. Nested hierarchical dirichlet processes. *arXiv preprint arXiv:1210.6738*, 2012.

[42] Ian Porteous, Alexander T Ihler, Padhraic Smyth, and Max Welling. Gibbs sampling for (coupled) infinite mixture models in the stick breaking representation. In *Uncertainty in Artificial Intelligence*, 2006.

[43] Joseph Reisinger, Austin Waters, Bryan Silverthorn, and Raymond Mooney. Spherical topic models. In *International Conference on Machine Learning*, 2010.

[44] Abel Rodriguez, David B Dunson, and Alan E Gelfand. The nested dirichlet process. *Journal of the American Statistical Association*, 103(483), 2008.

[45] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[46] J. Sethuraman. A constructive definition of dirichlet priors. *Statistica Sinica*, 4:639650, 1994.

[47] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[48] Erick Shonfeld. Who has the most photos of them all? hint: It is not facebook. *Techcrunch*, 2009.

[49] B. Sigurbjörnsson and R. Van Zwol. Flickr tag recommendation based on collective knowledge. In *International Conference on World Wide Web*, pages 327–336, 2008.

[50] J Sivic. Discovering Objects and their Location in Images. pages 1–8, October 2005.

[51] E Sudderth, A Torralba, W Freeman, and A Willsky. Describing visual scenes using transformed Dirichlet processes. In *Advances in Neural Information Processing Systems*, 2006.

[52] YW Teh and MI Jordan. Hierarchical Bayesian nonparametric models with applications. *Bayesian Nonparametrics: Principles and Practice. Cambridge University Press, Cambridge*, 2009.

[53] YW Teh, MI Jordan, MJ Beal, and DM Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[54] YW Teh, K Kurihara, and M Welling. Collapsed variational inference for HDP. volume 20, pages 1481–1488, 2008.

[55] Romain Thibaux and Michael I Jordan. Hierarchical beta processes and the indian buffet process. In *International conference on artificial intelligence and statistics*, pages 564–571, 2007.

[56] C Wang and D Blei. Decoupling sparsity and smoothness in the discrete hierarchical Dirichlet process. In *Advances in Neural Information Processing Systems*, 2009.

[57] C Wang, D Blei, and F.F. Li. Simultaneous image classification and annotation. In *Computer Vision and Pattern Recognition*, 2009.

[58] Chong Wang and David Blei. Truncation-free online variational inference for bayesian nonparametric models. In *Advances in Neural Information Processing Systems*, pages 422–430, 2012.

[59] Chong Wang, John Paisley, and David M Blei. Online variational inference for the hierarchical dirichlet process. In *AISTATS*, 2011.

[60] S. Williamson, C Wang, K. Heller, and D Blei. The IBP compound Dirichlet process and its application to focused topic modeling. In *International Conference on Machine Learning*, 2010.

[61] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer Vision and Pattern Recognition*, pages 3485–3492. IEEE, 2010.

[62] Eric P Xing, Kyung-ah Sohn, Michael I Jordan, and Yee W Teh. Bayesian multi-population haplotype inference via a hierarchical dirichlet process mixture. In *International Conference on Machine Learning*, pages 1049–1056, 2006.

[63] Bin Zhao, Li Fei-Fei, and Eric P Xing. Image segmentation with topic random field. In *European Conference on Computer Vision*, 2010.

[64] Xiaojin Zhu. Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison*, 2:3, 2006.

[65] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.