

HIERARCHICAL BAYESIAN MODELING: EFFICIENT INFERENCE AND APPLICATIONS

CHONG WANG

A DISSERTATION
PRESENTED TO THE FACULTY
OF PRINCETON UNIVERSITY
IN CANDIDACY FOR THE DEGREE
OF DOCTOR OF PHILOSOPHY

RECOMMENDED FOR ACCEPTANCE
BY THE DEPARTMENT OF
COMPUTER SCIENCE
ADVISER: DAVID M. BLEI

SEPTEMBER 2012

© Copyright by Chong Wang, 2012.

All Rights Reserved

Abstract

Appropriate tools for managing large-scale data, like online texts, images and user profiles, are becoming increasingly important. Hierarchical Bayesian models provide a natural framework for building these tools due to their flexibility in modeling real-world data. In this thesis, we describe a suite of efficient inference algorithms and novel models under the hierarchical Bayesian modeling framework.

We first present a novel online inference algorithm for the hierarchical Dirichlet process. The hierarchical Dirichlet process (HDP) is a Bayesian nonparametric model that can be used to model mixed-membership data with a potentially infinite number of components. Our online variational inference algorithm is easily applicable to massive and streaming data and significantly faster than traditional inference algorithms.

Second, we present a generic approximation framework for variational inference in a large family of nonconjugate models. For example, this includes multi-level logistic regression/generalized linear models and correlated topic models. With this, developing variational inference algorithm for many nonconjugate models is much easier.

Finally, we describe two novel models for real-world applications. This first application is about simultaneous image classification and annotation. We show that image classification and annotation can be integrated together using the same underlying probabilistic model. The second application is to better disseminate scientific information using recommendations. Compared with traditional recommendation algorithms, our algorithm not only improves the recommendation accuracy, but also provides interpretable structure of users and scientific articles. This interpretability provides lots of potential for designing better recommender systems.

Acknowledgements

The life of as a PhD at Princeton has been the best part of my life and it is sad to say goodbye. The resulting dissertation is impossible without the help and support from many great colleagues and friends.

Foremost, I thank my advisor David Blei. Dave has been a superb and supportive advisor, a wonderful friend and a role model. I also thank Fei-Fei Li, who was in Princeton and took me into the world of computer vision. I also thank my dissertation committee, Christiane Fellbaum, Andrea LaPaugh, Robert Schapire, Olga Troyanskaya and Eric Xing, for their commentes and suggesstions that improved this dissertation. I would like to thank Eric for the help in the job search process.

I am also very fortunate to interact with many past and current great members in sl@p, the statistical learning group at Princeton. Thanks to to Jordan Boyd-Graber, Jonathan Chang, Sean Gerrish, Sam Gershman, Lauren Hannah, Matt Hoffman, David Mimno, John Paisley, Gungor Polatkan and many other members in the Computer Science department who helped me in various ways during past years.

I want to thank the generous supports from King Peh Kwoh fellowship, Google PhD fellowship and Siebel Scholar fellowship for my graduate study. These made my life at Princeton a joyful experience.

Finally, I thank all my family members for being with me all time. My parents and sister, although far way, have given me lifetime love and support. And my wife Bo and my daughter Amy have become the new treasure of my life.

Dedicated to Amy Guan Wang.

Contents

Abstract	iii
Acknowledgements	iv
1 Introduction	1
2 Hierarchical Bayesian Modeling	3
2.1 Hierarchical Bayesian Modeling	3
2.2 Approximate Inference	6
2.2.1 Variational inference	6
2.2.2 Markov Chain Monte Carlo	8
2.3 Bayesian Nonparametric modeling	10
2.3.1 Dirichlet Process	10
2.3.2 Hierarchical Dirichlet Process	11
2.4 Topic Modeling	12
3 Online Variational Inference for Hierarchical Dirichlet Process	14
3.1 A Stick Breaking Construction of the HDP	16
3.2 Online Variational Inference for the HDP	20
3.2.1 A New Coordinate-ascent Variational Inference	21
3.2.2 Online Variational Inference	24
3.3 Experimental results	26
3.3.1 Data and Metric	26
3.3.2 Results	28

3.4	Discussion	32
4	Variational Inference in Nonconjugate Models	33
4.1	Variational Inference in Nonconjugate models	34
4.1.1	Inference in Nonconjugate Models	36
4.1.2	Laplace Variational Inference	37
4.1.3	Delta Method Variational Inference	39
4.1.4	Updating $q(z)$	41
4.1.5	The Algorithm	43
4.2	Example Models	43
4.3	Empirical Study	46
4.4	Conclusions	49
5	Simultaneous Image Classification and Annotation	50
5.1	Models and Algorithms	51
5.1.1	Modeling images, labels and annotations	52
5.1.2	Approximate inference	54
5.1.3	Parameter estimation	57
5.1.4	Classification and annotation	58
5.2	Related work	60
5.3	Empirical results	60
5.4	Discussion	64
6	Collaborative Topic Modeling for Recommendations	66
6.1	Background	69
6.1.1	Recommendation tasks	69
6.1.2	Recommendation by matrix factorization	70
6.1.3	Probabilistic topic models	72
6.2	Collaborative topic regression	73

6.3	Empirical study	78
6.4	Conclusions and future work	87
7	Conclusions	89

Chapter 1

Introduction

The last two decades have witnessed the explosion of the data that human beings can not process without the help of appropriate software. For example, iTunes for music or online web search. These (semi-)automatic tools for organizing and discovering information from the data are becoming increasingly important. For example, automatic document categorization and image annotation can be used for enhancing the data quality to aid browsing. Large-scale hidden structure discovery of documents can be used to improve the results of information retrieval and recommendations. Reasonable and accurate results of clustering can be useful for efficient data indexing.

To achieve these goals, researchers build models that are able to capture the desired properties from the data. My thesis focuses on hierarchical Bayesian models. Hierarchical Bayesian models naturally take advantage of statistical information shared among the data. This coincides with the property of real-world applications. For example, in document modeling, different documents can share similar topics; in image modeling, different images can share similar objects and annotations; in user modeling, different users can share similar interests. This is essentially the reason we can generalize our model on unseen data.

Second, the volume of real-world data, such as online texts, images and users, is usually very large. Traditional algorithms that work well on small-scale data might not be scalable to the much larger datasets. Thus, we need efficient algorithms that can tackle the large-scale

computational problems arising in modern applications of hierarchical Bayesian modeling.

This thesis aims to cover the two aspects of hierarchical Bayesian modeling discussed above. I will present a suite of efficient inference algorithms and several novel models under the hierarchical Bayesian modeling framework.¹ The chapters are organized as follows,

- In Chapter 2, we first review the concept of hierarchical Bayesian modeling and approximate inference algorithms. Next, we introduce the Bayesian nonparametric framework, with Dirichlet process (DP) and hierarchical Dirichlet process (HDP) as examples. Finally, we describe the topic modeling as an example of hierarchical Bayesian models.
- In Chapter 3, we present a novel online inference algorithm for HDP. The HDP, a Bayesian nonparametric model, can be used for mixed-membership models with an unbounded number of components. Our online variational inference algorithm that is easily applicable to massive and streaming data. This is based on our work [109].
- In Chapter 4, we present a novel framework for variational inference in nonconjugate models. We describe a generic approximation framework for variational inference with non-conjugate priors for a large family of models. With this, developing variational inference algorithm for non-conjugate models has a general guidance and is much easier. This is based on our work [108].
- In Chapter 5, we describe a novel model for simultaneous image classification and annotation. We show that image classification and annotation can be integrated together using the same underlying probabilistic model. This is based on our work [106].
- In Chapter 6, we present a novel model to better disseminate scientific information using recommendation techniques. Our model not only improves the recommendation accuracy, but also provides interpretable latent structure of users and scientific articles. This is based on our work [107].
- In Chapter 7, we summarize the thesis and discuss some future directions.

¹All my code about the content in this thesis can be found at <http://www.cs.princeton.edu/~chongw>.

Chapter 2

Hierarchical Bayesian Modeling

In this chapter, we discuss the background of hierarchical Bayesian modeling. This includes what kind of models we focus in this thesis, approximate inference for these models and Bayesian nonparametric extensions. Finally, we describe topic models as an example of hierarchical Bayesian models.

2.1 Hierarchical Bayesian Modeling

Hierarchical Bayesian modeling (HBM) can be considered as a guideline of building statistical models [47]. In these models, the key assumption is that inference about one unobserved quantity (for example, a document, an image or a user) affects inference about another unobserved quantity. For example, suppose we want to build a model of user interests in a recommendation engine. When we have new users, it is reasonable to assume that the information we have gathered from the existing users is helpful to model the new users and vice versa. A good model should be able to leverage different observations and improve overall predications.

In this thesis, we are mostly considering one type of hierarchical Bayesian models that have a specific structure depicted in Figure 2.1 using graphical models. This can also be described using the following generative process,

1. Draw corpus-level (global) hidden variable $\beta \sim p(\beta | \alpha)$, where α indicates the fixed

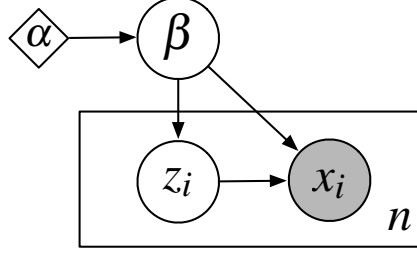


Figure 2.1: A general graphical model using to illustrate hierarchical Bayesian modeling. Here, parameter α indicates the fixed hyper-parameters; unshaded nodes β and z_i are hidden variables, shaded nodes x_i are the observed data and n is the number of data points in the collection. In general, we want to infer the posterior of hidden variables β and z_i , so that we use them to explore the collection or make prediction on future data.

hyper-parameters.

2. For each observed data point $x_i, i = 1, \dots, n$,
 - (a) Draw data-specific (local) hidden variable $z_i \sim p(z_i | \beta)$.
 - (b) Draw observed data point $x_i \sim p(x_i | z_i, \beta)$.

The joint probability of all the hidden and observed variables is,

$$p(x_{1:n}, z_{1:n}, \beta | \alpha) = p(\beta | \alpha) \prod_{i=1}^n p(z_i | \beta) p(x_i | \beta, z_i). \quad (2.1)$$

Given the observed data $x_{1:n}$ and the fixed hyper-parameter α , we would like to infer the posterior distributions over the hidden variables β and $z_{1:n}$, termed as,

$$p(z_{1:n}, \beta | x_{1:n}, \alpha) = \frac{p(x_{1:n}, z_{1:n}, \beta | \alpha)}{p(x_{1:n} | \alpha)}. \quad (2.2)$$

The marginal probability of the observed data, $p(x_{1:n} | \alpha)$, is also called the “evidence” of the data. After we obtain the posterior distribution $p(z_{1:n}, \beta | x_{1:n}, \alpha)$, we can use it to understand the data collection or make predictions on future data. For example, if we have a

new data point x_{n+1} , the predictive distribution given previous observed data $x_{1:n}$ is

$$p(x_{n+1} | x_{1:n}, \alpha) = \int p(x_{n+1} | z_{n+1}, \beta) p(z_{n+1} | \beta) p(\beta | x_{1:n}, \alpha) dz_{n+1} d\beta,$$

where we note that $p(\beta | x_{1:n}, \alpha)$ is the marginalized posterior distribution by integrating out $z_{1:n}$ from $p(z_{1:n}, \beta | x_{1:n}, \alpha)$.

This general model described using the generative process above covers a wide range of popular models, such as mixture models [75], mixed-membership models [26] and latent factor models [50]. Let us take an example of Bayesian Gaussian mixture models (GMM) with K mixture components. For simplicity, we assume the variance of each Gaussian mixture component is fixed as 1. Given this model, the global hidden variable $\beta = \{\mu_1, \mu_2, \dots, \mu_K, \pi\}$, where $\mu_k, k = 1, 2, \dots, K$ indicates the mixture mean locations and π is the mixture proportions and the fixed hyper-parameter $\alpha = \{\mu_0, \eta\}$, where μ_0 is the Gaussian prior mean parameter for $\mu_k, k = 1, 2, \dots, K$, and η is the Dirichlet parameter for mixture proportions π . The generative process for this Bayesian GMM is as follows,

1. Draw mixture proportions $\pi | \eta \sim \text{Dirichlet}(\eta)$.
2. For $k = 1, 2, \dots, K$, draw mixture mean location $\mu_k | \mu_0 \sim \mathcal{N}(\mu_0, 1)$.
3. For each observed data point $x_i, i = 1, \dots, n$,
 - (a) Draw mixture index $z_i | \pi \sim \text{Multinomial}(\pi)$.
 - (b) Draw observed data point $x_i | \mu_{z_i} \sim \mathcal{N}(\mu_{z_i}, 1)$.

The joint probability of the hidden and observed variables is,

$$p(x_{1:n}, z_{1:n}, \mu_{1:K}, \pi | \mu_0, \eta) = p(\pi | \eta) \prod_{k=1}^K p(\mu_k | \mu_0) \prod_{i=1}^n p(z_i | \pi) p(x_i | \mu_{z_i}).$$

And this completes the modeling process of a simple Bayesian GMM model.

2.2 Approximate Inference

In many hierarchical Bayesian models, even simply like Bayesian mixture models, the posterior distribution described in Eq. 2.2 is intractable. For convenience, we replicate Eq. 2.2 again as follows,

$$p(z_{1:n}, \beta \mid x_{1:n}, \alpha) = \frac{p(x_{1:n}, z_{1:n}, \beta \mid \alpha)}{p(x_{1:n} \mid \alpha)}.$$

The main reason is that the evidence term $p(x_{1:n} \mid \alpha)$ is intractable. That is

$$p(x_{1:n} \mid \alpha) = \int p(\beta \mid \alpha) \prod_{i=1}^n p(z_i \mid \beta) p(x_i \mid \beta, z_i) dz_{1:n} d\beta. \quad (2.3)$$

Again suppose we have a Bayesian mixture model with K components, where $z_{1:n}$ indicates the mixture indexes for data points $x_{1:n}$; the brute-force computation of $p(x_{1:n} \mid \alpha)$ has a complexity of $O(K^n)$, since the integration over β couples mixture index $z_{1:n}$ all together. (An alternative is usually impossible as well—the summing over $z_{1:n}$ is possible given β ; however this usually leads the integration over β impossible.)

In next two sections, we will describe two popular approximate inference algorithms for posterior computation, variational inference and Markov Chain Monte Carlo. The idea behind both of these methods is to form an approximate posterior distribution over the latent variables that is used as a proxy for the true posterior.

2.2.1 Variational inference

Variational inference [61] turns the posterior inference problem into optimization. We posit a family of distributions (usually simpler than the true posterior) over the hidden variables that is indexed by free parameters. We optimize those free parameters to find the member of the family that is the closest to the posterior of interest. Closeness is measured with Kullback-Leibler (KL) divergence. All the content in this thesis relies the variational-inference type of algorithms.

Variational inference has some nice properties. It is a deterministic algorithm and easy to assess the convergence. The downside of variational inference is also obvious—since it uses a simpler distribution to approximate the true posterior, it is difficult to know how biased the variational distribution is.

We will briefly describe simplest variational inference algorithm, mean-field variational inference. Variational inference does not optimize the KL divergence directly because, like the posterior, it is intractable to compute. Rather, the algorithm optimizes a lower bound (the evidence lower bound, ELBO) on the log probability of the observations $\log p(x_{1:n} | \alpha)$. According to the Jensen's inequality, we lower bound the $\log p(x_{1:n} | \alpha)$ as follows,

$$\begin{aligned}
\log p(x_{1:n} | \alpha) &= \log \int p(\beta | \alpha) \prod_{i=1}^n p(z_i | \beta) p(x_i | \beta, z_i) dz_{1:n} d\beta \\
&= \log \int \frac{p(\beta | \alpha) \prod_{i=1}^n p(z_i | \beta) p(x_i | \beta, z_i)}{q(\beta, z_{1:n})} q(\beta, z_{1:n}) dz_{1:n} d\beta \\
&= \log \mathbb{E}_q \left[\frac{p(\beta | \alpha) \prod_{i=1}^n p(z_i | \beta) p(x_i | \beta, z_i)}{q(\beta, z_{1:n})} \right] \\
&\geq \mathbb{E}_q [\log p(\beta | \alpha) \prod_{i=1}^n p(z_i | \beta) p(x_i | \beta, z_i)] - \mathbb{E}_q [\log q(\beta, z_{1:n})] \\
&= \mathcal{L}(q) \\
&= \log p(x_{1:n} | \alpha) - \text{KL}(q(\beta, z_{1:n}) || p(\beta, z_{1:n} | x_{1:n}, \alpha)),
\end{aligned}$$

where the term $\mathcal{L}(q)$ is denoted as ELBO and $q(\beta, z_{1:n})$ is the variational distribution. Maximizing the ELBO is equivalent to minimizing the KL-divergence between the variational distribution and the true posterior. If no restriction is assumed on the form of variational distribution $q(\beta, z_{1:n})$, the optimal $q(\beta, z_{1:n})$ is just the true posterior. However this does not solve our problem. The idea of variational inference restricts the form of variational distribution $q(\beta, z_{1:n})$ to be the family so that the ELBO $\mathcal{L}(q)$ is tractable.

In mean-field variational inference, we have the simplest form of the variational distribu-

tion that is fully factorized,

$$q(\beta, z_{1:n}) = q(\beta) \prod_{i=1}^n q(z_i).$$

Setting the partial derivatives with respect to $q(\beta)$ and $q(z_i)$ to zero, we obtain the optimal mean-field variational distribution must satisfy [16],

$$\begin{aligned} q(\beta) &\propto \exp \left\{ \mathbb{E}_{q(z_{1:n})} [\log p(\beta | \alpha) \prod_{i=1}^n p(z_i | \beta) p(x_i | \beta, z_i)] \right\}, \\ q(z_i) &\propto \exp \left\{ \mathbb{E}_{q(\beta)} [\log p(\beta | \alpha) p(z_i | \beta) p(x_i | \beta, z_i)] \right\} \end{aligned}$$

This actually defines iterative algorithm by alternately optimizing $q(\beta)$ and $q(z_i)$. The exact forms of $q(\beta)$ and $q(z_i)$ depend on the model specifications. When a model is conditionally conjugate— $p(\beta | \alpha)$ is the conjugate prior to $p(z_i | \beta)$ and $p(z_i | \beta)$ the conjugate prior to $p(x_i | z_i, \beta)$ —both $q(\beta)$ and $q(z_i)$ will be in closed-form, and they are in the same family as their node distributions $p(\beta)$ and $p(z_i | \beta)$ [12]. This leads to the traditional coordinate ascent algorithm, where we alternate between optimizing $q(\beta)$ and $q(z_i)$. In Chapter 4, we will describe approach that relaxes the conditional conjugacy assumption, allowing variational inference to be applied more widely.

2.2.2 Markov Chain Monte Carlo

Unlike variational inference, Markov Chain Monte Carlo (MCMC) is a sampling-based method [6]. In MCMC, the approximate posterior is formed as an empirical distribution of samples from a Markov chain whose stationary distribution is the posterior of interest. After the burn-in phase, we can collect samples and use these samples as the empirical distribution of the posterior. Theoretically, the estimation of the posterior distribution using the samples would be unbiased. However, due to the sampling nature, MCMC could be slow and it is difficult to assess its converge in practice [80, 88]. Nevertheless, MCMC is still a very powerful tool for Bayesian computation.

We will briefly describe the simplest MCMC algorithm, Gibbs sampling [48], for hierarchical Bayesian modeling. In general MCMC algorithms, designing an efficient proposal distribution is important, otherwise it will lead to high rejection rate. Unlike general MCMC algorithms, Gibbs sampling always accepts the proposed move. In Gibbs sampling, we sample each hidden variable using the conditional distribution given all the other hidden variables and the observations. For the model we consider in Figure 2.1, to sample the global hidden variable β , we have its conditional distribution,

$$p(\beta|z_{1:n}, x_{1:n}, \alpha) \propto p(\beta | \alpha) \prod_{i=1}^n p(z_i | \beta) p(x_i | \beta, z_i).$$

To sample each local hidden variable z_i , we have its conditional distribution,

$$p(z_i|\beta, z_{-i}, x_{1:n}, \alpha) \propto p(z_i | \beta) p(x_i | \beta, z_i).$$

Note sampling z_i does not depend on other local hidden variables z_j , $j \neq i$ due to the conditional independence given the global hidden variable β .

An improved version of Gibbs sampling can be achieved by integrating out certain hidden variables, leading to faster mixing [71]. For example, in many practical models, such as mixture models and mixed-membership models, it is possible to integrate out the global hidden variable β , leading to a better Gibbs sampling algorithm, usually called collapsed Gibbs sampler. In this case, we will need the conditional distribution,

$$p(z_i|z_{-i}, x_{1:n}, \alpha) \propto \int p(z_i | \beta) p(x_i | \beta, z_i) p(\beta | z_{-i}, x_{-i}, \alpha) d\beta.$$

This collapsed sampler removes the step of sampling β . However, since all the local hidden variables z_i , $i = 1, 2, \dots, n$ are tightly coupled, making it difficult for parallelization for large scale datasets.

2.3 Bayesian Nonparametric modeling

How do we choose the number of clusters/topics in a document collection? One common way is to use cross validation. Recently, Bayesian nonparametric models have received much attention as an appealing alternative to this kind of problems. These models encode a huge (usually unbounded) family of models to mitigate underfitting, and treat model selection as posterior inference to avoid overfitting. Next, we will describe two popular Bayesian nonparametric priors, Dirichlet process and hierarchical Dirichlet process as the priors for infinite mixture modeling and mixed-membership modeling.

2.3.1 Dirichlet Process

Dirichlet process (DP) [45] is a stochastic process whose sample draws are probabilistic measures. Let (E, \mathcal{E}) be a measurable space. Suppose that G_0 a probability measure on the space and α is a positive scalar. A random probability measure G on the space (E, \mathcal{E}) is said to be a draw from a DP with base measure αG_0 , if for any finite measurable partition (A_1, A_2, \dots, A_m) of E ,

$$(G(A_1), \dots, G(A_m)) \sim \text{Dirichlet}(\alpha G_0(A_1), \dots, \alpha G_0(A_m)), \quad (2.4)$$

where $\text{Dirichlet}(\cdot)$ is the Dirichlet distribution. For simplicity, we write $G \sim \text{DP}(\alpha G_0)$.

Two related perspectives of DP are the stick-breaking construction [94] and the Chinese restaurant process through the *Pólya urn scheme* [18]. We briefly introduce the stick-breaking construction.

Stick-breaking construction. As shown in [94], a draw G from $\text{DP}(\alpha G_0)$ can be described as

$$v_i \sim \text{Beta}(1, \alpha), \quad \pi_i = v_i \prod_{j=1}^{i-1} (1 - v_j), \quad y_i \sim G_0, \quad \text{for } i = 1, 2, \dots$$

$$G = \sum_{i=1}^{\infty} \pi_i \delta_{y_i},$$

where π are the stick lengths, and $\sum_{i=1}^{\infty} \pi_i = 1$ almost surely. This representation also illuminates the discreteness of a distribution drawn from a DP.

Dirichlet process mixtures. A popular application of DP is the infinite mixture models for density estimation [43]. Given $G \sim \text{DP}(\alpha G_0)$, we assume the observed data $\mathcal{D} = \{x_1, \dots, x_n\}$ is generated as follow, for $j = 1, \dots, n$,

$$\begin{aligned}\theta_j | G &\sim G, \\ x_j | \theta_j &\sim H(\theta_j),\end{aligned}\tag{2.5}$$

where $H(\theta)$ is the data generation distribution given the parameter θ . (For example, a Gaussian distribution given mean and variance.) Since G is a discrete measure, different data points have the chance to share the same θ , leading to the effect of mixture models. And G has infinite many atoms, allowing the number of mixtures unbounded. For approximate posterior inference for DP mixture models, Gibbs sampling [81] and variational inference [21] are two popular approaches.

2.3.2 Hierarchical Dirichlet Process

The hierarchical Dirichlet process [99] is a hierarchical generalization of the DP on random distributions [45]. We will focus on a two-level HDP, which can be used in an infinite capacity mixed-membership model. In a mixed-membership model, data are groups of observations, and each exhibits a shared set of mixture components with different proportion. For example, if we focus on text-based topic modeling, the data are observed words from a vocabulary grouped into documents; the mixture components are distributions over terms called “topics.”

In an HDP model mixed-membership model, each group is associated with a draw from

a shared DP whose base distribution is also a draw from a DP,

$$G_0 \sim \text{DP}(\gamma H)$$

$$G_j | G_0 \sim \text{DP}(\alpha_0 G_0), \text{ for each } j,$$

where j is a group index. At the top level, the distribution G_0 is a draw from a DP with concentration parameter γ and base distribution H . It is almost surely discrete, placing its mass on atoms drawn independently from H [45]. At the bottom level, this discrete distribution is used as the base distribution for each per-group distribution G_j . Though they may be defined on a continuous space (e.g., the simplex), this ensures that the per-group distributions G_j share the same atoms as G_0 .

As for the stick-breaking construction for the HDP and its application to topic modeling, we will cover them in more details when we present the novel online variational inference algorithm for large scale data in Chapter 3.

2.4 Topic Modeling

Now we describe the topic modeling as an real example of hierarchical Bayesian models. The idea of topic modeling has been used extensively in this thesis. Topic modeling algorithms are used to discover a set of “topics” from a large collection of documents, where a topic is a distribution over terms that is biased around those associated under a single theme [54, 32, 24]. Topic models provide an interpretable low-dimensional representation of the documents [35]. They have been used for tasks like corpus exploration, document classification, and information retrieval.

We use latent Dirichlet allocation (LDA) [26] as the representative of topic models. Suppose there are K topics, each of which is a distribution over a fixed vocabulary. The generative process of LDA is as follows.

1. Draw topic distribution $\beta_k \sim \text{Dirichlet}(\eta)$, for $k = 1, 2, \dots, K$.

2. For each article w_j in the corpus,
 - (a) Draw topic proportions $\theta_j \sim \text{Dirichlet}(\alpha)$.
 - (b) For each word n ,
 - i. Draw topic assignment $z_{jn} \sim \text{Mult}(\theta_j)$.
 - ii. Draw word $w_{jn} \sim \text{Mult}(\beta_{z_{jn}})$.

This process reveals how the words of each document are assumed to come from a mixture of topics (mixed-membership): the topic proportions are document-specific, but the set of topics is shared by the corpus.

Given a collection, the posterior distribution of the topics reveals the K topics that likely generated its documents. Unlike a clustering model, where each document is assigned to one cluster, LDA allows documents to exhibit multiple topics. For example, LDA can capture that one article might be about biology and statistics, while another might be about biology and physics. Since LDA is unsupervised, the themes of “physics” “biology” and “statistics” can be discovered from the corpus; the mixed-membership assumptions lead to sharper estimates of word co-occurrence patterns.

Chapter 3

Online Variational Inference for Hierarchical Dirichlet Process

As we briefly discussed in Chapter 2, the hierarchical Dirichlet process (HDP) [99] is a powerful mixed-membership model for the unsupervised analysis of grouped data. Applied to document collections, the HDP provides a nonparametric topic model where documents are viewed as groups of observed words, mixture components (called topics) are distributions over terms, and each document exhibits the topics with different proportions. Given a collection of documents, the HDP topic model finds a low-dimensional latent structure that can be used for tasks like classification, exploration, and summarization. Unlike its finite counterpart, latent Dirichlet allocation [26], the HDP topic model infers the number of topics from the data.

Posterior inference for the HDP is intractable, and much research is dedicated to developing approximate inference algorithms [99, 100, 70]. These methods are limited for massive scale applications, however, because they require multiple passes through the data and are not easily applicable to streaming data.¹ In this chapter, we develop a new approximate inference algorithm for the HDP. Our algorithm is designed to analyze much larger data sets than the existing state-of-the-art allows and, further, can be used to analyze *streams* of data.

¹One exception that may come to mind is the particle filter [33, 89]. However, this algorithm still requires periodically resampling a variable for every data point. Data cannot be thrown away as in a true streaming algorithm.

This is particularly apt to the HDP topic model. Topic models promise to help summarize and organize large archives of texts that cannot be easily analyzed by hand and, further, could be better exploited if available on streams of texts such as web APIs or news feeds.

Our method—online variational Bayes for the HDP— was inspired by the recent online variational Bayes algorithm for LDA [53]. Online LDA allows LDA models to be fit to massive and streaming data, and enjoys significant improvements in computation time without sacrificing model quality. Our motivation for extending this algorithm to the HDP is that LDA requires choosing the number of topics in advance. In a traditional setting, where fitting multiple models might be viable, the number of topics can be determined with cross validation or held-out likelihood. However, these techniques become impractical when the data set size is large, and they become impossible when the data are streaming. Online HDP provides the speed of online variational Bayes with the modeling flexibility of the HDP.

The idea behind online variational Bayes in general is to optimize the variational objective function of [61] with stochastic optimization [87]. Optimization proceeds by iteratively taking a random subset of the data, and updating the variational parameters with respect to the subset. Online variational Bayes is particularly efficient when using the natural gradient [5] on models in which traditional variational Bayes can be performed by simple coordinate ascent [93]. (This is the property that allowed [53] to derive an efficient online variational Bayes algorithm for LDA.) In this setting, online variational Bayes is significantly faster than traditional variational Bayes [10], which must make multiple passes through the data.

The challenge we face is that the existing coordinate ascent variational Bayes algorithms for the HDP require complicated approximation methods or numerical optimization [100, 70, 29]. We will begin by reviewing Sethuraman’s stick-breaking construction of the HDP [46]. We show that this construction allows for coordinate-ascent variational Bayes without numerical approximation, which is a new and simpler variational inference algorithm for the HDP. We will then use this approach in an online variational Bayes algorithm, allowing the

HDP to be applied to massive and streaming data. Finally, on two large archives of scientific articles, we will show that the online HDP topic model provides a significantly better fit than online LDA. Online variational Bayes lets us apply Bayesian nonparametric models at much larger scales.

3.1 A Stick Breaking Construction of the HDP

We describe the stick-breaking construction of the HDP [46] using the Sethuraman’s construction for the DP [94]. This is amenable to simple coordinate-ascent variational inference, and we will use it to develop online variational inference for the HDP.

A two-level hierarchical Dirichlet process (HDP) [99] (the focus of this chapter) is a collection of Dirichlet processes (DP) [45] that share a base distribution G_0 , which is also drawn from a DP. Mathematically,

$$G_0 \sim \text{DP}(\gamma H) \tag{3.1}$$

$$G_j \sim \text{DP}(\alpha_0 G_0), \text{ for each } j, \tag{3.2}$$

where j is an index for each group of data. A notable feature of the HDP is that all DPs G_j share the same set of atoms and only the atom weights differ. This is a result of the almost sure discreteness of the top-level DP.

In the HDP topic model—which is the focus of this chapter—we model groups of words organized into documents. The variable w_{jn} is the n th word in the j th document; the base distribution H is a symmetric Dirichlet over the vocabulary simplex; and the atoms of G_0 , which are independent draws from H , are called topics.

The HDP topic model contains two additional steps to generate the data. First we generate the topic associated with the n th word in the j th document; then we generate the word from that topic,

$$\theta_{jn} \sim G_j, \quad w_{jn} \sim \text{Mult}(\theta_{jn}). \tag{3.3}$$

The discreteness of the corpus-level draw G_0 ensures that all documents share the same set of topics. The document-level draw G_j inherits the topics from G_0 , but weights them according to document-specific topic proportions.

Teh’s Stick-breaking Construction. The definition of the HDP in Eq. 3.1 is implicit. [99] propose a constructive representation of the HDP using two stick-breaking representations of a Dirichlet distribution [94]. For the corpus-level DP draw, this representation is

$$\begin{aligned}\beta'_k &\sim \text{Beta}(1, \gamma), \\ \beta_k &= \beta'_k \prod_{l=1}^{k-1} (1 - \beta'_l), \\ \phi_k &\sim H, \\ G_0 &= \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k}.\end{aligned}\tag{3.4}$$

Thus, G_0 is discrete and has support at the atoms $\phi = (\phi_k)_{k=1}^{\infty}$ with weights $\beta = (\beta_k)_{k=1}^{\infty}$. The distribution for β is also written as $\beta \sim \text{GEM}(\gamma)$ [85].

The construction of each document-level G_j is

$$\begin{aligned}\pi'_{jk} &\sim \text{Beta}\left(\alpha_0 \beta_k, \alpha_0 \left(1 - \sum_{l=1}^k \beta_l\right)\right), \\ \pi_{jk} &= \pi'_{jk} \prod_{l=1}^{k-1} (1 - \pi'_{jl}), \\ G_j &= \sum_{k=1}^{\infty} \pi_{jk} \delta_{\phi_k},\end{aligned}\tag{3.5}$$

where $\phi = (\phi_k)_{k=1}^{\infty}$ are the same atoms as G_0 in Eq. 3.4.

This construction is difficult to use in an online variational inference algorithm. Online variational inference is particularly efficient when the model is also amenable to coordinate ascent variational inference, and where each update is available in closed form. In the construction above, the stick-breaking weights are tightly coupled between the bottom and top-level DPs. As a consequence, it is not amenable to closed form variational updates [100, 70].

Sethuraman’s Stick-breaking Construction. To address this issue, we describe an alternative stick-breaking construction for the HDP that allows for closed-form coordinate-ascent variational inference due to its full conjugacy. (This construction was also briefly described in [46].)

The construction is formed by twice applying Sethuraman’s stick-breaking construction of the DP. We again construct the corpus-level base distribution G_0 as in Eq. 3.4. The difference is in the document-level draws. We use Sethuraman’s construction for each G_j ,

$$\begin{aligned}\psi_{jt} &\sim G_0, \\ \pi'_{jt} &\sim \text{Beta}(1, \alpha_0), \\ \pi_{jt} &= \pi'_{jt} \prod_{l=1}^{t-1} (1 - \pi'_{jl}), \\ G_j &= \sum_{t=1}^{\infty} \pi_{jt} \delta_{\psi_{jt}},\end{aligned}\tag{3.6}$$

Notice that each document-level atom (i.e., topic) ψ_{jt} maps to a corpus-level atom ϕ_k in G_0 according to the distribution defined by G_0 . Further note there will be multiple document-level atoms ψ_{jt} which map to the same corpus-level atom ϕ_k , but we can verify that G_j contains all of the atoms in G_0 almost surely.

A second way to represent the document-level atoms $\psi_j = (\psi_{jt})_{t=1}^{\infty}$ is to introduce a series of indicator variables, $c_j = (c_{jt})_{t=1}^{\infty}$, which are drawn i.i.d.,

$$c_{jt} \sim \text{Mult}(\beta),\tag{3.7}$$

where $\beta \sim \text{GEM}(\gamma)$ (as mentioned above). Then let

$$\psi_{jt} = \phi_{c_{jt}},\tag{3.8}$$

Thus, we do not need to explicitly represent the document atoms ψ_j . This further simplifies online inference.

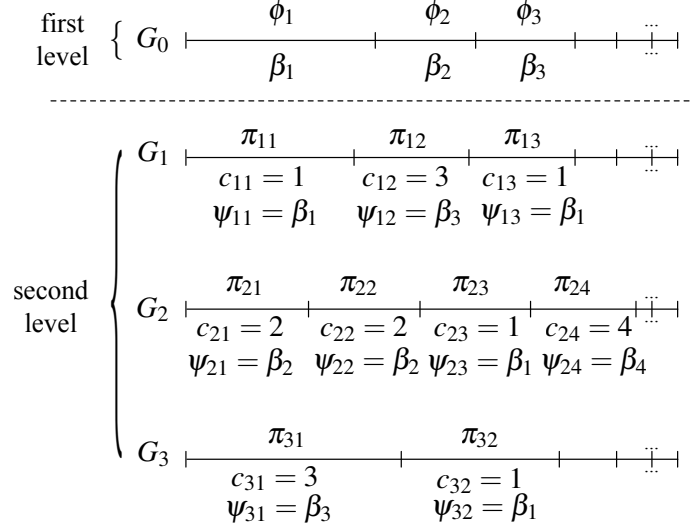


Figure 3.1: Illustration of the Sethuraman's stick-breaking construction of the two-level HDP. In the first level, $\phi_k \sim H$ and $\beta \sim \text{GEM}(\gamma)$; in the second level, $\pi_j \sim \text{GEM}(\alpha_0)$, $c_{jt} \sim \text{Mult}(\beta)$ and $\psi_{jt} = \phi_{c_{jt}}$.

The property that multiple document-level atoms ψ_{jt} can map to the same corpus-level atom ϕ_k in this representation is similar in spirit to the Chinese restaurant franchise (CRF) [99], where each restaurant can have multiple tables serving the *same* dish ϕ_k . In the CRF representation, a hierarchical Chinese restaurant process allocates dishes to tables. Here, we use a series of random indicator variables c_j to represent this structure. Figure 3.1 illustrates the concept.

Given the representation in Eq. 3.6, the generative process for the observed words in j th document, w_{jn} , is as follows,

$$z_{jn} \sim \text{Mult}(\pi_j), \quad (3.9)$$

$$\theta_{jn} = \psi_{jz_{jn}} = \phi_{c_{jz_{jn}}}, \quad (3.10)$$

$$w_{jn} \sim \text{Mult}(\theta_{jn}). \quad (3.11)$$

The indicator z_{jn} selects topic parameter ψ_{jt} , which maps to one topic ϕ_k through the indicators c_j . This also provides the mapping from topic θ_{jn} to ϕ_k , which we need in Eq. 3.3.

3.2 Online Variational Inference for the HDP

With Sethuraman’s construction of the HDP in hand, we now turn to our original aim—approximate posterior inference in the HDP for massive and streaming data. Given a large collection of documents, our goal is to approximate the posterior distribution of its latent topic structure.

We will use online variational inference [93]. Traditional variational inference approximates the posterior over the hidden variables by positing a simpler distribution which is optimized to be close in Kullback-Leibler (KL) divergence to the true posterior [61]. This problem is (approximately) solved by optimizing a function equal up to a constant to the KL of interest. In online variational inference, we optimize that function with stochastic approximation.

Online variational inference enjoys a close relationship with *coordinate-ascent variational inference*. Consider a model with latent variables and observations for which the posterior is intractable to compute. One strategy for variational inference is the mean-field approach: posit a distribution where each latent variable is independent and governed by its own parameter, and optimize the variational parameters with coordinate ascent.

Now, suppose that those coordinate ascent updates are available in closed form and consider updating them in parallel. (Note this is no longer coordinate ascent.) It turns out that the vector of parallel coordinate updates is exactly the natural gradient of the variational objective function under conjugate priors [93]. This insight makes stochastic optimization of the variational objective, based on a subset of the data under analysis, a simple and efficient alternative to traditional coordinate-ascent.

Let us now return to the HDP topic model. We will first show that Sethuraman’s representation of the HDP above allows for closed-form coordinate-ascent updates for variational inference. Then, we will derive the corresponding online algorithm, which provides a scalable method for HDP posterior inference.

3.2.1 A New Coordinate-ascent Variational Inference

When applied to Bayesian nonparametric models, variational methods are usually based on stick-breaking representations—these representations provide a concrete set of hidden variables on which to place an approximate posterior [21, 66, 100]. Furthermore, the approximate posterior is usually truncated. The user first sets a truncation on the number of topics to allow, and then relies on variational inference to infer a smaller number that are used in the data. (Two exceptions are found in [67, 104], who developed methods that allow the truncation to grow.) Note that setting a truncation level is different from asserting a number of components in a model. When set large, the HDP assumptions encourage the approximate posterior to use fewer components.

We use a fully factorized variational distribution and perform mean-field variational inference. The hidden variables that we are interested in are the top-level stick proportions $\beta' = (\beta'_k)_{k=1}^\infty$, bottom-level stick proportions $\pi'_j = (\pi'_{jt})_{t=1}^\infty$ and the vector of indicators $\mathbf{c}_j = (c_{jt})_{t=1}^\infty$ for each G_j . We also infer atom/topic distributions $\phi = (\phi_k)_{k=1}^\infty$, topic index z_{jn} for each word w_{jn} . Thus our variational distribution has the following form,

$$q(\beta', \pi', \mathbf{c}, \mathbf{z}, \phi) = q(\beta')q(\pi')q(\mathbf{c})q(\mathbf{z})q(\phi). \quad (3.12)$$

This further factorizes into

$$\begin{aligned} q(\mathbf{c}) &= \prod_j \prod_t q(c_{jt} | \varphi_{jt}), \\ q(\mathbf{z}) &= \prod_j \prod_n q(z_{jn} | \zeta_{jn}), \\ q(\phi) &= \prod_k q(\phi_k | \lambda_k), \end{aligned}$$

where the variational parameters are φ_{jt} (multinomial), ζ_{jn} (multinomial) and λ_k (Dirichlet).

The factorized forms of $q(\beta')$ and $q(\pi')$ are

$$\begin{aligned} q(\beta') &= \prod_{k=1}^{K-1} q(\beta'_k | u_k, v_k), \\ q(\pi') &= \prod_j \prod_{t=1}^{T-1} q(\pi'_{jt} | a_{jt}, b_{jt}), \end{aligned} \quad (3.13)$$

where (u_k, v_k) and (a_{jt}, b_{jt}) are parameters of beta distributions. We set the truncations for the corpus and document levels to K and T . Here, T can be set much smaller than K , because in practice each document G_j requires far fewer topics than those needed for the entire corpus (i.e., the atoms of G_0). With this truncation, our variational distribution has $q(\beta'_K = 1) = 1$ and $q(\pi'_{jT} = 1) = 1$, for all j .

Using standard variational theory [61], we lower bound the marginal log likelihood of the observed data $\mathcal{D} = (\mathbf{w}_j)_{j=1}^D$ using Jensen's inequality,

$$\begin{aligned} \log p(\mathcal{D} | \gamma, \alpha_0, \eta) &\geq \mathbb{E}_q [\log p(\mathcal{D}, \beta', \pi', \mathbf{c}, \mathbf{z}, \phi)] + H(q) \\ &= \sum_j \left\{ \mathbb{E}_q [\log (p(\mathbf{w}_j | \mathbf{c}_j, \mathbf{z}_j, \phi) p(\mathbf{c}_j | \beta') p(\mathbf{z}_j | \pi'_j) p(\pi'_j | \alpha_0))] \right. \\ &\quad \left. + H(q(\mathbf{c}_j)) + H(q(\mathbf{z}_j)) + H(q(\pi'_j)) \right\} \\ &\quad + \mathbb{E}_q [\log p(\beta') p(\phi)] + H(q(\beta')) + H(q(\phi)) \\ &= \mathcal{L}(q), \end{aligned} \quad (3.14)$$

where $H(\cdot)$ is the entropy term for the variational distribution. This is the variational objective function, which up to a constant is equivalent to the KL to the true posterior. Taking derivatives of this lower bound with respect to each variational parameter, we can derive the following coordinate ascent updates.

Document-level Updates: At the document level we update the parameters to the per-document stick, the parameters to the per word topic indicators, and the parameters to the

per document topic indices,

$$a_{jt} = 1 + \sum_n \zeta_{jnt}, \quad (3.15)$$

$$b_{jt} = \alpha_0 + \sum_n \sum_{s=t+1}^T \zeta_{jns}, \quad (3.16)$$

$$\varphi_{jtk} \propto \exp \left(\sum_n \zeta_{jnt} \mathbb{E}_q [\log p(w_{jn} | \phi_k)] + \mathbb{E}_q [\log \beta_k] \right), \quad (3.17)$$

$$\zeta_{jnt} \propto \exp \left(\sum_{k=1}^K \varphi_{jtk} \mathbb{E}_q [\log p(w_{jn} | \phi_k)] + \mathbb{E}_q [\log \pi_{jt}] \right). \quad (3.18)$$

Corpus-level Updates: At the corpus level, we update the parameters to top-level sticks and the topics,

$$u_k = 1 + \sum_j \sum_{t=1}^T \varphi_{jtk}, \quad (3.19)$$

$$v_k = \gamma + \sum_j \sum_{t=1}^T \sum_{l=k+1}^K \varphi_{jtl}, \quad (3.20)$$

$$\lambda_{kw} = \eta + \sum_j \sum_{t=1}^T \varphi_{jtk} (\sum_n \zeta_{jnt} I[w_{jn} = w]), \quad (3.21)$$

The expectations involved above are taken under the variational distribution q , and are

$$\mathbb{E}_q [\log \beta_k] = \mathbb{E}_q [\log \beta'_k] + \sum_{l=1}^{k-1} \mathbb{E}_q [\log(1 - \beta'_l)],$$

$$\mathbb{E}_q [\log \beta'_k] = \Psi(u_k) - \Psi(u_k + v_k),$$

$$\mathbb{E}_q [\log(1 - \beta'_k)] = \Psi(v_k) - \Psi(u_k + v_k),$$

$$\mathbb{E}_q [\log \pi_{jt}] = \mathbb{E}_q [\log \pi'_{jt}] + \sum_{s=1}^{t-1} \mathbb{E}_q [\log(1 - \pi'_{js})],$$

$$\mathbb{E}_q [\log \pi'_{jt}] = \Psi(a_{jt}) - \Psi(a_{jt} + b_{jt}),$$

$$\mathbb{E}_q [\log(1 - \pi'_{jt})] = \Psi(b_{jt}) - \Psi(a_{jt} + b_{jt}),$$

$$\mathbb{E}_q [\log p(w_{jn} = w | \phi_k)] = \Psi(\lambda_{kw}) - \Psi(\sum_w \lambda_{kw}),$$

where $\Psi(\cdot)$ is the digamma function.

Unlike previous variational inference methods for the HDP [100, 70], this method only contains simple closed-form updates due to the full conjugacy of the stick-breaking

construction. (We note that, even in the batch setting, this is a new posterior inference algorithm for the HDP.)

3.2.2 Online Variational Inference

We now develop online variational inference for an HDP topic model. In online variational inference, we apply stochastic optimization to the variational objective. We subsample the data (in this case, documents), compute an approximation of the gradient based on the subsample, and follow that gradient with a decreasing step-size. The key insight behind efficient online variational inference is that coordinate ascent updates applied in parallel precisely form the *natural gradient* of the variational objective function [93, 53].

Our approach is similar to that described in [53]. Let D be the total number of documents in the corpus, and define the variational lower bound for document j as

$$\begin{aligned}\mathcal{L}_j = & \mathbb{E}_q [\log (p(\mathbf{w}_j|\mathbf{c}_j, \mathbf{z}_j, \phi)p(\mathbf{c}_j|\beta')p(\mathbf{z}_j|\pi'_j)p(\pi'_j|\alpha_0))] + H(q(\mathbf{c}_j)) + H(q(\mathbf{z}_j)) + H(q(\pi'_j)) \\ & + \frac{1}{D} [\mathbb{E}_q [\log p(\beta')p(\phi)] + H(q(\beta')) + H(q(\phi))].\end{aligned}$$

We have taken the corpus-wide terms and multiplied them by $1/D$. With this expression, we can see that the lower bound \mathcal{L} in Eq. 3.14 can be written as

$$\mathcal{L} = \sum_j \mathcal{L}_j = \mathbb{E}_j[D\mathcal{L}_j],$$

where the expectation is taken over the empirical distribution of the data set. The expression $D\mathcal{L}_j$ is the variational lower bound evaluated with D duplicate copies of document j .

With the objective construed as an expectation over our data, online HDP proceeds as follows. Given the existing corpus-level parameters, we first sample a document j and compute its optimal document-level variational parameters $(\mathbf{a}_j, \mathbf{b}_j, \psi_j, \zeta_j)$ by coordinate ascent (see Eq. 3.15 to 3.18.). Then, take the gradient of the corpus-level parameters $(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{v})$ of $D\mathcal{L}_j$, which is a noisy estimate of the gradient of the expectation above. We follow that

gradient according to a decreasing learning rate, and repeat.

Natural Gradients. The gradient of the variational objective contains, as a component, the covariance matrix of the variational distribution. This is a computational problem in topic modeling because each set of topic parameters involves a $V \times V$ covariance matrix, where V is the size of the vocabulary (e.g., 5,000). The *natural gradient* [5]—which is the inverse of the Riemannian metric [5] multiplied by the gradient—has a simple form in the variational setting [93] that allows for fast online inference.

Multiplying the gradient by the inverse of Riemannian metric cancels the covariance matrix of the variational distribution, leaving a natural gradient which is much easier to work with. Specifically, the natural gradient is structurally equivalent to the coordinate updates of Eq 3.19 to 3.21 taken in parallel. (And, in stochastic optimization, we treat the sampled document j as though it is the whole corpus.) Let $\partial\boldsymbol{\lambda}(j)$, $\partial\boldsymbol{u}(j)$ and $\partial\boldsymbol{v}(j)$ be the natural gradients for $D\mathcal{L}_j$. Using the analysis in [93, 53], the components of the natural gradients are

$$\partial\lambda_{kw}(j) = -\lambda_{kw} + \eta + D \sum_{t=1}^T \varphi_{jtk} (\sum_n \zeta_{jnt} I[w_{jn} = w]), \quad (3.22)$$

$$\partial u_k(j) = -u_k + 1 + D \sum_{t=1}^T \varphi_{jtk}, \quad (3.23)$$

$$\partial v_k(j) = -v_k + \gamma + D \sum_{t=1}^T \sum_{l=k+1}^K \varphi_{jtl}. \quad (3.24)$$

In online inference, an appropriate learning rate ρ_{t_o} is needed to ensure the parameters to converge to a stationary point [93, 53]. Then the updates of $\boldsymbol{\lambda}$, \boldsymbol{u} and \boldsymbol{v} become

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \rho_{t_o} \partial\boldsymbol{\lambda}(j), \quad (3.25)$$

$$\boldsymbol{u} \leftarrow \boldsymbol{u} + \rho_{t_o} \partial\boldsymbol{u}(j) \quad (3.26)$$

$$\boldsymbol{v} \leftarrow \boldsymbol{v} + \rho_{t_o} \partial\boldsymbol{v}(j), \quad (3.27)$$

-
- 1: Initialize $\boldsymbol{\lambda} = (\lambda_k)_{k=1}^K$, $\boldsymbol{u} = (u_k)_{k=1}^{K-1}$ and $\boldsymbol{v} = (v_k)_{k=1}^{K-1}$ randomly. Set $t_o = 1$.
 - 2: **while** Stopping criterion is not met **do**
 - 3: Fetch a random document j from the corpus.
 - 4: Compute \boldsymbol{a}_j , \boldsymbol{b}_j , $\boldsymbol{\varphi}_j$ and $\boldsymbol{\zeta}_j$ using variational inference using document-level updates, Eq. 3.15 to 3.18.
 - 5: Compute the natural gradients, $\partial\boldsymbol{\lambda}(j)$, $\partial\boldsymbol{u}(j)$ and $\partial\boldsymbol{v}(j)$ using Eq. 3.22 to 3.24.
 - 6: Set $\rho_{t_o} = (\tau_0 + t_o)^{-\kappa}$, $t_o \leftarrow t_o + 1$.
 - 7: Update $\boldsymbol{\lambda}$, \boldsymbol{u} and \boldsymbol{v} using Eq. 3.25 to 3.27.
 - 8: **end while**
-

Figure 3.2: Online variational inference for the HDP

where the learning rate ρ_{t_o} should satisfy

$$\sum_{t_o=1}^{\infty} \rho_{t_o} = \infty, \quad \sum_{t_o=1}^{\infty} \rho_{t_o}^2 < \infty, \quad (3.28)$$

which ensures convergence [87]. In our experiments, we use $\rho_{t_o} = (\tau_0 + t_o)^{-\kappa}$, where $\kappa \in (0.5, 1]$ and $\tau_0 > 0$. Note that the natural gradient is essential to the efficiency of the algorithm. The online variational inference algorithm for the HDP topic model is illustrated in Figure 3.2.

Mini-batches. To improve stability of the online learning algorithm, practitioners typically use multiple samples to compute gradients at a time—a small set of documents in our case. Let \mathcal{S} be a small set of documents and $S = |\mathcal{S}|$ be its size. In this case, rather than computing the natural gradients using $D\mathcal{L}_j$, we use $(D/S) \sum_{j \in \mathcal{S}} \mathcal{L}_j$. The update equations can then be similarly derived.

3.3 Experimental results

In this section, we evaluate the performance of online variational HDP compared with batch variational HDP and online variational LDA.²

3.3.1 Data and Metric

Data Sets. Our experiments are based on two datasets:

²<http://www.cs.princeton.edu/~blei/downloads/onlineLDAVB.tar>

- *Nature*: This dataset contains 352,549 documents, with about 58 million tokens and a vocabulary size of 4,253. These articles are from the years 1869 to 2008.
- *PNAS*: The *Proceedings of the National Academy of Sciences* (PNAS) dataset contains 82,519 documents, with about 46 million tokens and a vocabulary size of 6,500. These articles are from the years 1914 to 2004.

Standard stop words and those words that appear too frequently or too rarely are removed.

Evaluation Metric. We use the following evaluation metric to compare performance. For each dataset, we held out 2000 documents as a test set $\mathcal{D}_{\text{test}}$, with the remainder as training data $\mathcal{D}_{\text{train}}$. For testing, we split document \mathbf{w}_j in $\mathcal{D}_{\text{test}}$ into two parts, $\mathbf{w}_j = (\mathbf{w}_{j1}, \mathbf{w}_{j2})$, and compute the predictive likelihood of the second part \mathbf{w}_{j2} (10% of the words) conditioned on the first part \mathbf{w}_{j1} (90% of the words) and on the training data. This is similar to the metrics used in [100, 9], which tries to avoid comparing different hyperparameters. The metric is

$$\text{likelihood}_{\text{pw}} = \frac{\sum_{\mathbf{j} \in \mathcal{D}_{\text{test}}} \log p(\mathbf{w}_{j2} | \mathbf{w}_{j1}, \mathcal{D}_{\text{train}})}{\sum_{\mathbf{j} \in \mathcal{D}_{\text{test}}} |\mathbf{w}_{j2}|},$$

where $|\mathbf{w}_{j2}|$ is the number of tokens in \mathbf{w}_{j2} and “pw” means “per-word.” Exact computation is intractable, and so we use the following approximation. For all algorithms, let $\bar{\phi}$ be the variational expectation of ϕ given $\mathcal{D}_{\text{train}}$. For LDA, let $\bar{\pi}_j$ be the variational expectation given \mathbf{w}_{j2} and α be its Dirichlet hyperparameter for topic proportions. The predictive marginal probability of \mathbf{w}_{j1} is approximated by

$$p(\mathbf{w}_{j2} | \mathbf{w}_{j1}, \mathcal{D}_{\text{train}}) \approx \prod_{w \in \mathbf{w}_{j2}} \sum_k \bar{\pi}_{jk} \bar{\phi}_{kw}.$$

To use this approximation for the HDP, we set the Dirichlet hyperparameter to $\bar{\alpha} = \alpha_0 \bar{\beta}$, where $\bar{\beta}$ is the variational expectation of β , obtained from the variational expectation of β' .

3.3.2 Results

Experimental Settings. For the HDP, we set $\gamma = \alpha_0 = 1$, although using priors is also an option. We set the top-level truncation $K = 150$ and the second level truncation $T = 15$. Here $T \ll K$, since documents usually don’t have many topics. For online variational LDA, we set its Dirichlet hyperparameter $\alpha = (1/K, \dots, 1/K)$, where K is the number of topics; we set $K = \{20, 40, 60, 80, 100, 150\}$.³ We set $\tau_0 = 64$ based on the suggestions in [53], and vary $\kappa = \{0.6, 0.8, 1.0\}$ and the batch size $S = \{16, 64, 256, 1024, 2048\}$. We collected experimental results during runs of 6 hours each.⁴

Nature Corpus. In Figure 3.3, we plot the per-word log likelihood as a function of computation time for online HDP, online LDA, and batch HDP. (For the online algorithms, we set $\kappa = 0.6$ and the batch size was $S = 256$.) This figure shows that online HDP performs better than online LDA. The HDP uses about 110 topics out of its potential 150. In contrast, online LDA uses almost all the topics and exhibits overfitting at 150 topics. Note that batch HDP is only trained on a subset of 20,000 documents—otherwise it is too slow—and its performance suffers.

In Figure 3.5, we plot the per-word likelihood after 6 hours of computation, exploring the effect of batch size and values of κ . We see that, overall, online HDP performs better than online LDA. (This matches the reported results in [100], which compares batch variational inference for the HDP and LDA.) Further, we found that small κ favors larger batch sizes. (This matches the results seen for online LDA in [53].)

We also ran online HDP on the full *Nature* dataset using only one pass (with $\kappa = 0.6$ and a batch size $S = 1024$) by sequentially processing the articles from the year 1869 to 2008. Table 3.4 tracks the most probable ten words from two topics as we encounter more articles in the collection. Note that the HDP here is not a dynamic topic model [22, 105]; we show these results to demonstrate the online inference process.

³This is different from the top level truncation K in the HDP.

⁴The python package will be available at first author’s homepage.

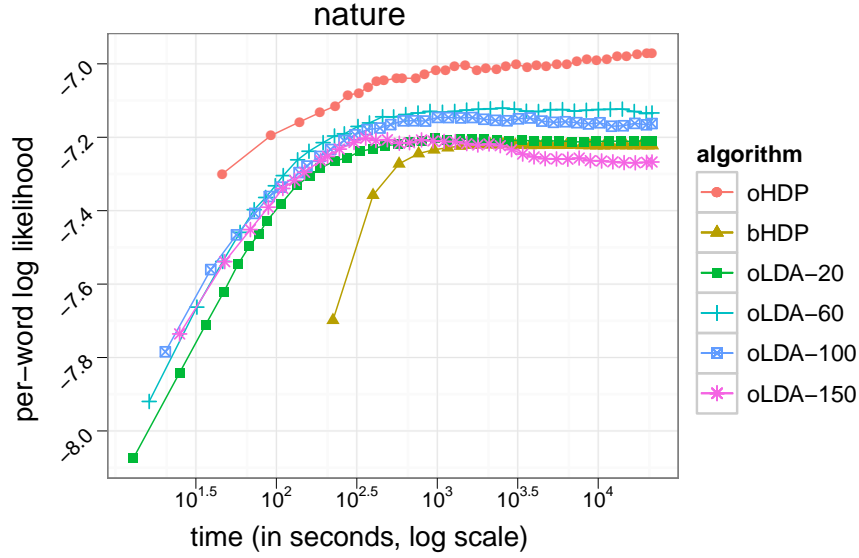


Figure 3.3: Experimental results on *Nature* with $\kappa = 0.6$ and batch size $S = 256$ (for the online algorithms). Points are sub-sampled for better view. The label “oLDA-20” indicates online LDA with 20 topics. (Not all numbers of topics are shown; see Figure 3.5 for more details.) Online HDP performs better than online LDA and batch HDP.

These results show that online inference for streaming data finds different topics at different speeds, since the relevant information for each topic does not come at the same time. In this sequential setting, some topics are rarely used until there are documents that can provide enough information to update them (see the top topic in Figure 3.4). Other topics are updated throughout the stream because relevant documents occur throughout the whole collection (see the bottom topic in Figure 3.4).

PNAS Corpus We ran the same experiments on the *PNAS* corpus. Since *PNAS* is smaller than *Nature*, we were able to run batch HDP on the whole data set. Figure 3.6 shows the result with $\kappa = 0.6$ and batch size $S = 2048$. Online HDP performs better than online LDA. Here batch HDP performs a little better than online HDP, but online HDP is much faster. Figure 3.7 plots the comparison between online HDP and online LDA across different batch sizes and values of κ .

40,960	81,920	122,880	163,840	204,800	245,760	286,720	327,680	352,549
author	author	due	weight	rats	rats	rats	rats	neurons
series	series	series	due	response	mice	response	brain	rats
vol	due	distribution	birds	blood	response	dose	memory	memory
due	sea	author	response	sec	dose	saline	dopamine	brain
your	vol	sea	series	weight	drug	injection	mice	dopamine
latter	latter	carried	average	dose	brain	brain	subjects	response
think	hand	statistical	sample	mice	injection	females	neurons	mice
sun	carried	sample	soil	average	food	treated	drug	behavioural
sea	fact	average	population	food	saline	food	induced	training
feet	appear	soil	frequency	controls	females	rat	response	responses
stars	stars	stars	stars	stars	stars	star	stars	galaxies
star	observatory	observatory	observatory	observatory	observatory	arc	galaxy	stars
observatory	star	sun	solar	solar	radio	emission	star	galaxy
sun	sun	star	sun	sun	star	stars	emission	star
magnitude	magnitude	solar	astronomical	astronomical	optical	optical	galaxies	emission
solar	solar	astronomical	star	star	objects	spectrum	optical	optical
comet	motion	greenwich	greenwich	earth	magnitude	image	redshift	redshift
spectrum	comet	earth	eclipse	radio	solar	images	images	spectrum
motion	eclipse	eclipse	instrument	greenwich	positions	ray	image	images
photographs	spectrum	magnitude	royal	motion	plates	magnitude	objects	objects

Figure 3.4: The top ten words from two topics, displayed after different numbers of documents have been processed for inference. The two topics are separated by the dashed line. The first line of the table indicates the number of articles seen so far (beginning from the year 1869). The topic on the top (which could be labeled “neuroscience research on rats”) does not have a clear meaning until we have analyzed 204,800 documents. This topic is rarely used in the earlier part of the corpus and few documents provide useful information about it. In contrast, the topic on the bottom (which could be labeled “astronomy research”) has a clearer meaning from the beginning. This subject is discussed earlier in *Nature* history.

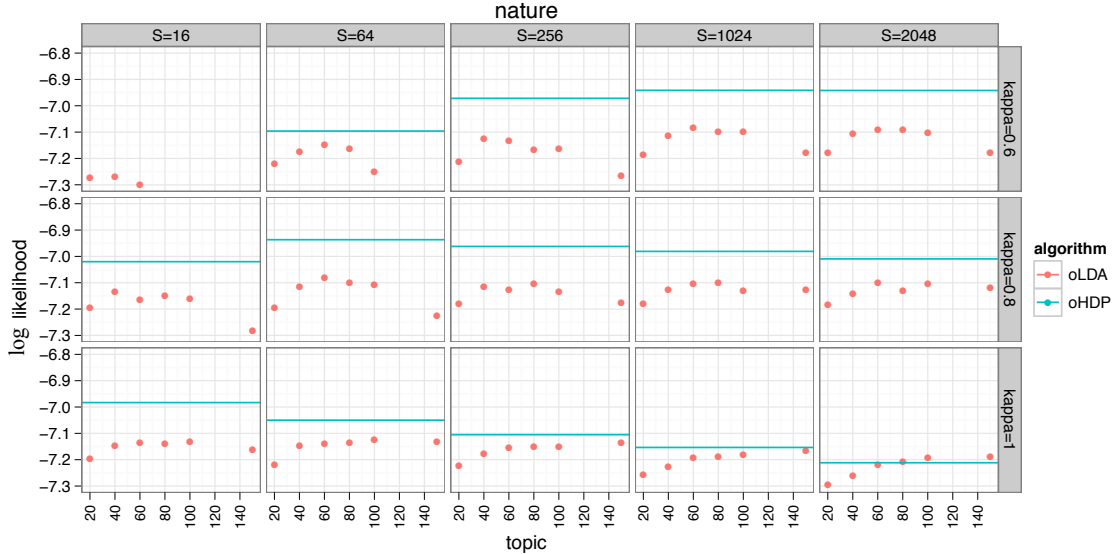


Figure 3.5: Comparisons of online LDA and online HDP on the *Nature* corpus under various settings of batch size S and parameter κ (kappa), run for 6 hours each. (Some lines for online HDP and points for online LDA do not appear due to figure limits.) The best result among all is achieved by online HDP.

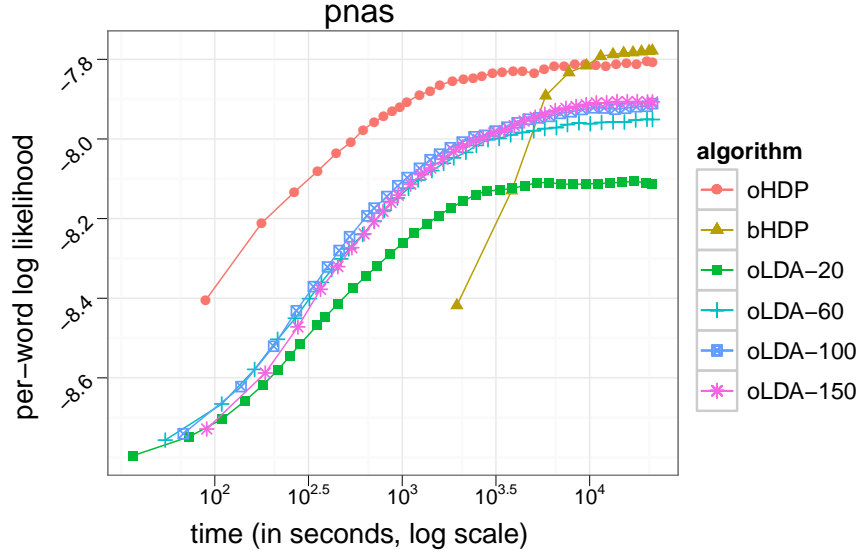


Figure 3.6: Experimental results on *PNAS* $\kappa = 0.6$ and batch size $S = 2048$ (for the online algorithms). Points are sub-sampled for better view. (Not all numbers of topics are shown, please see Figure 3.7 for more details.) Online HDP performs better than online LDA, and slightly worse than batch HDP. Unlike in the *Nature* experiment, batch HDP is trained on the whole training set.

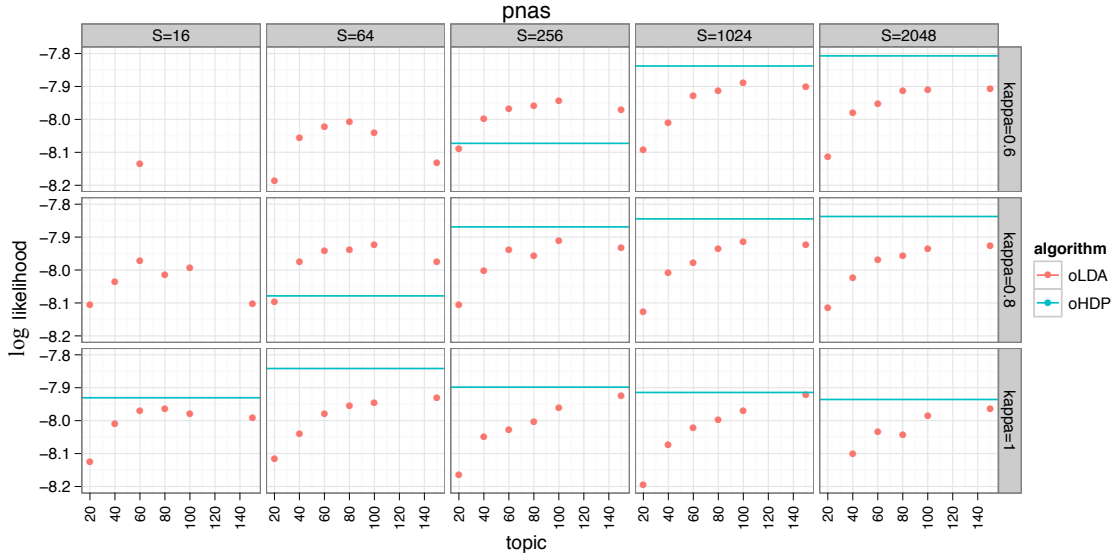


Figure 3.7: Comparisons of online LDA and online oHDP on the *PNAS* corpus under various settings of batch size S and parameter κ (kappa), run for 6 hours each. (Some lines for online HDP and points for online LDA do not appear due to figure limits.) The best result among all is achieved by online HDP.

3.4 Discussion

We developed an online variational inference algorithm for the hierarchical Dirichlet process topic model. Our algorithm is based on a stick-breaking construction of the HDP that allows for closed-form coordinate ascent variational inference, which is a key factor in developing the online algorithm. Our experimental results show that for large-scale applications, the online variational inference for the HDP can address the model selection problem for LDA and avoid overfitting.

The application of natural gradient learning to online variational inference may be generalized to other Bayesian nonparametric models, as long as we can construct variational inference algorithms with closed form updates under conjugacy. For example, the Indian Buffet process (IBP) [51, 98, 39] might be another model that can use an efficient online variational inference algorithm for large and streaming data sets.

Chapter 4

Variational Inference in Nonconjugate Models

Variational inference, especially the mean-field variational inference [61] has been used extensively in my research. Mean-field variational inference lets us efficiently estimate posterior distributions in complex probabilistic models [61]. The idea is to optimize the parameters of a factorized distribution so that it is close to the true posterior. When each variable of the model is part of a conjugate pair, we can easily derive a coordinate ascent algorithm to perform this optimization. This is the principle behind code bases like VIBES [17], which let us quickly define models of our data and run variational inference.

Many models of interest, however, do not enjoy the conjugacy properties required to take advantage of this easily derived algorithm. Such models include Bayesian logistic regression [59] (or more broadly, Bayesian generalized linear models [112]), discrete choice models [30], item response models [37], and some probabilistic topic models [23]. Analyzing data in these settings with variational inference requires algorithms tailored to the specific model at hand. Researchers have developed a variety of ways of handling nonconjugate priors in specific settings. These include approximations [30, 3], alternative bounds [23, 63], and quadrature [57]. The recent work of [64] presents a message passing algorithm for nonconjugate models, which has been implemented in Infer.NET [78].¹

In this chapter, we aim to provide easily derived algorithms for mean-field variational

¹The technique of [64] applies to a subset of models described in this chapter. It may be generalizable to the full set, however, how to find the expectations required by [64] would still have to be determined.

inference in a large class of nonconjugate models. We develop two strategies. We first develop *Laplace variational inference*. It embeds Laplace approximations [73]—a technique for univariate distributions from Bayesian statistics—within a variational optimization algorithm. We then develop *delta method variational inference*. This approach optimizes a Taylor approximation of the variational objective. The details of the algorithm depend on how the approximation is formed. Formed one way, it recovers Laplace variational inference. Formed another way, it is equivalent to using a multivariate delta approximation [15] of the variational objective.²

We studied our algorithms with two nonconjugate models: Bayesian logistic regression [59] and correlated topic models [23]. We found that our methods usually give better results than those obtained through special purpose techniques. Further, we show that Laplace variational inference usually outperforms the delta method approach. These methods significantly expand the class of models for which mean-field variational inference can be easily applied.

4.1 Variational Inference in Nonconjugate models

Consider a generic model with the following joint distribution,

$$p(x, z, \theta) = p(x|z)p(z|\theta)p(\theta). \quad (4.1)$$

We assume this model has *hidden* variables θ and z , and *observed* variable x . The inference problem is to compute the posterior of θ and z , $p(z, \theta|x)$. This is intractable for many models and we must resort to approximations of the posterior.

Variational inference approximates the posterior by minimizing the Kullback-Leibler (KL) divergence between a simpler distribution and the true posterior $p(\theta, z|x)$ [61]. Mean-field variational inference is simplest and most widely used. It uses a fully factorized

²The delta method was first used in variational inference by [30] in the context of the discrete choice model. Our method generalizes their approach.

variational distribution,

$$q(z, \theta) = q(z)q(\theta).$$

Under the standard variational theory, minimizing the KL-divergence between $q(z, \theta)$ and $p(\theta, z|x)$ is equivalent to maximizing a lower bound of the log marginal likelihood of the observed data x . We obtain this bound with Jensen's inequality,

$$\log p(x) = \log \int p(x, z, \theta) dz d\theta \geq \mathbb{E}_q [q] \log p(x, Z, \theta) + H[q] = \mathcal{L}(q), \quad (4.2)$$

where $\mathbb{E}_q [q] \cdot$ is the expectation taken with respect to q and $H[q]$ the entropy. Setting $\partial \mathcal{L}(q) / \partial q = 0$ shows that the optimal solution satisfies the following [16],

$$q^*(\theta) \propto \exp \{ \mathbb{E}_q [q(z)] \log p(Z|\theta) p(\theta) \}, \quad (4.3)$$

$$q^*(z) \propto \exp \{ \mathbb{E}_q [q(\theta)] \log p(x|z) p(z|\theta) \}. \quad (4.4)$$

A model is *conditionally conjugate*—when $p(\theta)$ is the conjugate prior to $p(z|\theta)$ and $p(z|\theta)$ the conjugate prior to $p(x|z)$ —in this setting, both $q^*(\theta)$ and $q^*(z)$ will be in closed-form, and they are in the same family as their prior distributions $p(\theta)$ and $p(z|\theta)$. This leads to the traditional coordinate ascent algorithm, where we alternate between optimizing $q(\theta)$ and $q(z)$ [16].

However, when the variable θ is not part of a conjugate pair, we cannot find closed-form updates for either distribution. In the next section, we define a wider class of models that includes Bayesian logistic regression, correlated topic models [23] and discrete choice models [30] as special cases. We develop variational inference for this more general class of models.

4.1.1 Inference in Nonconjugate Models

Before describing our algorithms, we present the modeling assumptions with reference to Eq. 4.1.

1. Recall that θ is real-valued. The distribution $p(\theta)$ is twice differentiable with respect to θ . If we require $\theta > \theta_0$ (θ_0 is a constant), we may define a distribution over $\log(\theta - \theta_0)$.
2. The distribution $p(z|\theta)$ is in the exponential family [31],

$$p(z|\theta) = h(z) \exp \left\{ \eta(\theta)^\top t(z) - a(\eta(\theta)) \right\}, \quad (4.5)$$

where $h(z)$ is a function of z ; $t(z)$ is the sufficient statistic; $\eta(\theta)$ is the natural parameter; and $a(\eta(\theta))$ is log partition function. We emphasize that $p(\theta)$ is not necessarily the conjugate prior.

3. The distribution $p(x|z)$ is in the exponential family with z as the natural parameter,

$$p(x|z) = h(x) \exp \left\{ z^\top t(x) - a(z) \right\}, \quad (4.6)$$

and we require the distribution $p(z|\theta)$ is conjugate [13]. Consequently, the term $t(z)$ in Eq. 4.5 is

$$t(z) = [z, -a(z)]. \quad (4.7)$$

Note that $t(x)$ and $a(z)$ are overloaded. They are different from $t(z)$ and $a(\eta(\theta))$.

These assumptions are weaker than those of conditional conjugacy. This family includes nonconjugate models like the correlated topic model (CTM) [23], Bayesian logistic regression [59] and discrete choice models [30]. For these models, we can no longer implement the closed form updates for the variational distributions in Eq. 4.3 and Eq. 4.4. Because $p(\theta)$ is not assumed conjugate to $p(z|\theta)$, the update in Eq. 4.3 does not necessarily have the form of an exponential family we can work with. As a further consequence, it is difficult to

use $\mathbb{E}_q[q(\theta)]p(z|\theta)$ in Eq. 4.4.

We will develop two variational inference algorithms for this class of models. They both use coordinate ascent to optimize the variational parameters, iterating between updating $q(\theta)$ and $q(z)$. First in *Laplace variational inference*, we use Laplace approximations [73] within the coordinate ascent updates of Eq. 4.3 and Eq. 4.4—we approximate the update for $q(\theta)$ in Eq. 4.3. Second, in *delta method variational inference*, we apply Taylor approximations to approximate the variational objective in Eq. 4.2, then derive the corresponding updates. Different ways of taking the Taylor approximation lead to different algorithms. Formed one way, it also recovers the Laplace algorithm. Formed another way, it is equivalent to using a multivariate delta approximation [15] of the variational objective function—we name this way *delta method variational inference*.

In both algorithms, the variational distribution $q(\theta)$ is a Gaussian; the distribution $q(z)$ is in the same family as $p(z|\theta)$ in Eq. 4.5. These forms emerge from the derivation of Laplace variational inference; they are assumed in the derivation of delta method variational inference. We will first derive the algorithms for updating $q(\theta)$. We then show how to update $q(z)$.

4.1.2 Laplace Variational Inference

We first review the Laplace approximation [73]. Then we show how to use it in variational inference.

The Laplace Approximation. Laplace approximations use a Gaussian to approximate an intractable density [73]. Consider approximating an intractable posterior $p(\theta|x)$. (There is no hidden variable z in this set up.) Assume the joint distribution $p(x, \theta) = p(x|\theta)p(\theta)$ is tractable. Laplace approximations use a Taylor approximation around the maximum a posteriori (MAP) to construct a Gaussian proxy for the posterior. First, notice the posterior is proportional to the exponentiated log joint

$$p(\theta|x) = \exp\{\log p(\theta|x)\} \propto \exp\{\log p(\theta, x)\}.$$

Let $\hat{\theta}$ be the MAP of $p(\theta|x)$, found by maximizing $\log p(\theta, x)$. A Taylor expansion around $\hat{\theta}$ gives

$$\log p(\theta|x) \approx \log p(\hat{\theta}|x) + \frac{1}{2}(\theta - \hat{\theta})^\top H(\hat{\theta})(\theta - \hat{\theta}). \quad (4.8)$$

The term $H(\hat{\theta})$ is the Hessian of $\log p(\theta|x)$ evaluated at $\hat{\theta}$, $H(\hat{\theta}) \triangleq \nabla^2 \log p(\theta|x)|_{\theta=\hat{\theta}}$.

In the Taylor expansion of Eq. 4.8, the first-order term $(\theta - \hat{\theta})^\top \nabla \log p(\theta|x)|_{\theta=\hat{\theta}}$ equals zero. The reason is that $\hat{\theta}$ is the maximum of $\log p(\theta|x)$ and so its gradient $\nabla \log p(\theta|x)|_{\theta=\hat{\theta}}$ is zero. Exponentiating Eq. 4.8 gives the approximate Gaussian posterior

$$p(\theta|x) \approx \frac{1}{C} \exp \left\{ -\frac{1}{2}(\theta - \hat{\theta})^\top \left(-H(\hat{\theta}) \right) (\theta - \hat{\theta}) \right\},$$

where C is a normalizing constant. In other words, $p(\theta|x)$ can be approximated by

$$p(\theta|x) \approx \mathcal{N}(\hat{\theta}, -H(\hat{\theta})^{-1}). \quad (4.9)$$

This is the Laplace approximation. It is difficult to use in multivariate settings, for example, when there are discrete hidden variables. Now we describe how we use Laplace approximations as part of variational inference for a complex model.

Laplace Updates. We use Laplace approximations to update the variational distribution $q(\theta)$. First, we combine the coordinate update in Eq. 4.3 with the exponential family assumption in Eq. 4.5,

$$q(\theta) \propto \exp \left\{ \eta(\theta)^\top \mathbb{E}_q[q(z)] t(z) - a(\eta(\theta)) + \log p(\theta) \right\}. \quad (4.10)$$

Now define

$$\bar{t}_z \triangleq \mathbb{E}_q [q(z)] t(Z), \quad (4.11)$$

$$f(\theta) \triangleq \eta(\theta)^\top \bar{t}_z - a(\eta(\theta)) + \log p(\theta). \quad (4.12)$$

We approximate $q(\theta)$ by taking a second-order Taylor approximation of $f(\theta)$ around its maximum, following the same logic as from Eq. 4.8. Let $\hat{\theta}$ be the value that maximizes $f(\theta)$ and $\nabla^2 f(\hat{\theta})$ be the Hessian matrix evaluated at $\hat{\theta}$. Adapting Eq. 4.9 and Eq. 4.10 to this setting gives

$$q(\theta) \approx \mathcal{N}(\hat{\theta}, -\nabla^2 f(\hat{\theta})^{-1}). \quad (4.13)$$

The Gaussian form of $q(\theta)$ stems from using the Taylor approximation. This is an approximate update for $q(\theta)$ and can be embedded in a coordinate ascent algorithm for a nonconjugate model. Notice we need to use numerical optimization to obtain $\hat{\theta}$.

4.1.3 Delta Method Variational Inference

In Laplace variational inference, the variational distribution $q(\theta)$ Eq. 4.13 is solely a function of $\hat{\theta}$, the maximum of $f(\theta)$ in Eq. 4.12. A natural question is, would other values of θ be suitable as well?

To consider such alternatives, we describe a different way of performing variational inference. We approximate the variational objective \mathcal{L} in Eq. 4.2 and then optimize that approximation.

Again we focus on $q(\theta)$ and postpone the discussion of $q(z)$. The variational distribution $q(\theta)$ is a Gaussian $q(\theta) = \mathcal{N}(\mu, \Sigma)$. We isolate the terms related to $q(\theta)$ in the objective, then substitute the exponential family assumption about $p(z|\theta)$ in Eq. 4.5 into $\mathcal{L}(q)$ in

Eq. 4.2,

$$\mathcal{L}(q(\theta)) = \mathbb{E}_q [q(\theta)] \eta(\theta)^\top \mathbb{E}_q [q(z)] t(z) - a(\eta(\theta)) + \log p(\theta) + H[q(\theta)].$$

The entropy of the Gaussian is $H[q(\theta)] = \frac{1}{2} \log |\Sigma| + C$, where C is a constant. Notice the first three terms are the same function $f(\theta)$ defined in Eq. 4.12. We simplify the lower bound $\mathcal{L}(q(\theta))$,

$$\mathcal{L}(q(\theta)) = \mathbb{E}_q [q(\theta)] f(\theta) + \frac{1}{2} \log |\Sigma|.$$

We cannot easily compute the expectation in the first term. We use a Taylor expansion of $f(\theta)$ around a chosen value $\hat{\theta}$,

$$f(\theta) \approx f(\hat{\theta}) + \nabla f(\hat{\theta})(\theta - \hat{\theta}) + \frac{1}{2}(\theta - \hat{\theta})^\top \nabla^2 f(\hat{\theta})(\theta - \hat{\theta}).$$

With this Taylor approximation, $\mathcal{L}(q(\theta))$ can be approximated with

$$\begin{aligned} \mathcal{L}(q(\theta)) &\approx f(\hat{\theta}) + \nabla f(\hat{\theta})^\top (\mu - \hat{\theta}) + \frac{1}{2}(\mu - \hat{\theta})^\top \nabla^2 f(\hat{\theta})(\mu - \hat{\theta}) \\ &\quad + \frac{1}{2}(\text{Tr} \left\{ \nabla^2 f(\hat{\theta}) \Sigma \right\} + \log |\Sigma|), \end{aligned} \tag{4.14}$$

where $\text{Tr}(\cdot)$ is the Trace operator. This is the function we optimize w.r.t. the variational parameters of $q(\theta)$, $\{\mu, \Sigma\}$. The form of this optimization, however, depends on how we choose $\hat{\theta}$, the point around which to approximate $f(\theta)$. Unlike in Laplace variational inference, $\hat{\theta}$ can be any value.

We will describe three options. First, we can choose $\hat{\theta}$ to be the maximum of $f(\theta)$. Then maximizing the approximation in Eq. 4.14 gives $\mu = \hat{\theta}$ and $\Sigma = -\nabla^2 f(\hat{\theta})^{-1}$. This is the update derived in Section 4.1.2. It gives a different derivation of Laplace inference.

A second choice is $\hat{\theta} = \mu$, i.e., the mean of the variational distribution $q(\theta)$. With this

choice, the variable around which we center the Taylor approximation becomes part of the optimization problem. The objective in Eq. 4.14 is

$$\mathcal{L}(q(\theta)) \approx f(\mu) + \frac{1}{2} \text{Tr} \{ \nabla^2 f(\mu) \Sigma \} + \frac{1}{2} \log |\Sigma|. \quad (4.15)$$

This is the multivariate delta method for evaluating $\mathbb{E}_q [q(\theta)] f(\theta)$ [15]. We call this choice *delta method variational inference*. To optimize $\mathcal{L}(q(\theta))$, we use coordinate ascent on μ and Σ . We use gradient methods to find μ . Note this is more expensive than Laplace inference because it requires the third derivative $\nabla^3 f(\theta)$. Given a value of μ , we optimize Σ in closed form, $\Sigma = -\nabla^2 f(\mu)^{-1}$. [30] is the first to use the delta method in a variational inference algorithm, developing this technique for the discrete choice model. If we assume $p(\theta)$ is Gaussian then we recover their algorithm. With the ideas presented here, we can now use this strategy in many models.

A final choice is to guess $\hat{\theta}$, for example, as the mean of the variational distribution from the previous iteration of coordinate ascent. If $p(\theta)$ is a Gaussian distribution, this recovers the updates derived in [3] for the correlated topic model.³ We found this simple guess did not work well on our data. (It did not always converge.) We thus focus on Laplace and delta method variational inference.

4.1.4 Updating $q(z)$

We have derived variational updates for $q(\theta)$ using two methods. We now turn to the update for $q(z)$. We will show that Laplace (Section 4.1.2) and delta method variational inference (Section 4.1.3) lead to the same update form for $q(z)$. Further, we have implicitly assumed that $\mathbb{E}_q [q(z)] t(z)$ in Eq. 4.11 and Eq. 4.12 is easy to compute. We will confirm this as well.

³In their paper, these updates were derived from the perspective of generalized mean-field theory [113].

Laplace variational inference. First, we apply the exponential family form in Eq. 4.5 to the exact update of Eq. 4.4,

$$\log q(z) = C + \log p(x|z) + \log h(z) + \mathbb{E}_q [q(\theta)] \eta(\theta)^\top t(z),$$

where C is a constant not depending on z . Now we use $p(x|z)$ from Eq. 4.6 and $t(z)$ from Eq. 4.7 to obtain

$$q(z) \propto h(z) \exp \left\{ (\mathbb{E}_q [q(\theta)] \eta(\theta) + [t(x), 1])^\top t(z) \right\}, \quad (4.16)$$

which is in the same family as $p(z|\theta)$ in Eq. 4.5. This is the update for $q(z)$. Further, because it is in a simple exponential family form, we can compute $\mathbb{E}_q [q(z)] t(z)$ from Eq. 4.11. Note we have additionally assumed the tractability of $\mathbb{E}_q [q(\theta)] \eta(\theta)$. To approximate this, we take a Taylor approximation of $\eta(\theta)$ around the variational parameter μ ,

$$\eta(\theta) \approx \eta(\mu) + \nabla \eta(\mu)^\top (\theta - \mu) + \frac{1}{2} (\theta - \mu)^\top \nabla^2 \eta(\mu) (\theta - \mu).$$

Since $q(\theta) \approx \mathcal{N}(\mu, \Sigma)$, this means that

$$\mathbb{E}_q [q(\theta)] \eta(\theta) \approx \eta(\mu) + \frac{1}{2} \text{Tr} \{ \nabla^2 \eta(\mu) \Sigma \}. \quad (4.17)$$

(Note that the linear term $\mathbb{E}_q [q(\theta)] \nabla \eta(\mu)^\top (\theta - \mu) = 0$.)

Delta method variational inference. Using the delta method to update $q(\theta)$, the update for $q(z)$ is identical to that in Laplace variational inference. We isolate the relevant terms in Eq. 4.2,

$$\mathcal{L}(q(z)) = \mathbb{E}_q [q(z)] \log p(x|z) + \log h(z) + \mathbb{E}_q [q(\theta)] \eta(\theta)^\top t(z) + H[q(z)].$$

-
- 1: Initialize variational distribution $q(\theta)$ and $q(z)$.
 - 2: **repeat**
 - 3: Compute the statistics $\mathbb{E}_q [q(z)] t(z)$ in Eq. 4.11.
 - 4: To obtain $q(\theta)$,
 - Option a): in Laplace variational inference, compute Eq. 4.13.
 - Option b): in delta method variational inference, optimize Eq. 4.15.
 - 5: Approximate the statistics $\mathbb{E}_q [q(\theta)] \eta(\theta)$ in Eq. 4.17.
 - 6: Update $q(z)$ in Eq. 4.16.
 - 7: **until** Mean of $|\text{change in } \mathbb{E}_q [q] \theta| < 0.0001$.
 - 8: **return** $q(\theta)$ and $q(z)$.
-

Figure 4.1: The approximation framework in variational inference.

Setting the partial gradient $\partial \mathcal{L}(q(z))/\partial q(z) = 0$ gives the same optimal $q(z)$ of Eq. 4.4.

Computing this update reduces to the approach for Laplace variational inference.

4.1.5 The Algorithm

We have described the updates for $q(\theta)$ and $q(z)$. Alternately between these updates defines an iterative algorithm. Our algorithms no longer guarantee a strict lower bound of the true variational objective. However, as also observed in [30], we found that the algorithms converge in practice.

See Figure 4.1 for an outline of this algorithm. We have reduced deriving variational updates to somewhat mechanical work—calculating derivatives and calling a numerical optimization library.

Regarding the variants of our algorithm, Laplace variational inference is simpler to derive because it only requires first derivatives of the function in Eq. 4.12, while delta variational inference requires third derivatives. We empirically study the differences between these methods in Section 4.3.

4.2 Example Models

We now discuss how several nonconjugate models from the research literature on which we can use the generic algorithm described above.

Correlated topic models. The correlated topic model (CTM) [23] is an extension of the latent Dirichlet allocation (LDA) [26] topic model. Topic models are hierarchical mixed membership models of documents. Each document is treated as a collection of observed words that are drawn from a mixture model. The mixture components, called “topics,” are distributions over terms. Each document exhibits its own topic proportions, but the corpus shares the same set of topics.

The CTM extends LDA by replacing the Dirichlet prior on the topic proportions with a logistic normal prior [4]. This captures correlations between the components. While the CTM is more expressive, it is no longer conditionally conjugate. Suppose there are K topic parameters $\beta_{1:K}$ (fixed for now), each of which is a distribution over V terms. Let π be the topic proportions for a document and n be the index of an observed word x_n . The CTM assumes the following generative process of a document,

$$\begin{aligned}\theta &\sim \mathcal{N}(\mu_0, \Sigma_0), \pi \propto \exp(\theta) \\ z_n \mid \pi &\sim \text{Mult}(\pi), x_n \mid z_n, \beta \sim \text{Mult}(\beta_{z_n}), \forall n.\end{aligned}$$

In this model, the topic proportions π are drawn from a logistic normal distribution. Their correlation structure is captured in Σ_0 . The variable z_n indicates which topic the n th word is drawn from.

Our variational distribution for parameter θ is $q(\theta) = \mathcal{N}(\mu, \Sigma)$. In delta method inference, as in [30], we assume the covariance matrix Σ is diagonal. Laplace inference does not require this. The detailed derivations are in Appendix A.

Note that this algorithm is only for inference at the document level. As in [23], we estimate the corpus-level topic parameters $\beta_{1:K}$ and logistic normal parameters (μ_0, Σ_0) with variational EM.

Bayesian Logistic regression. Standard *Bayesian logistic regression* is a well-studied model for binary classification [59].⁴ Let t_n is be a p -dimensional observed feature vector for the n th sample and z_n be its class (an indicator vector of length two). Let θ be the real-valued parameter vector in \mathbb{R}^p ; there is a component for each feature. The usual Bayesian logistic regression is the following:

$$\begin{aligned} p(\theta) &= \mathcal{N}(\mu_0, \Sigma_0), \\ p(z_n | \theta, t_n) &= \sigma(\theta^\top t_n)^{z_{n,1}} \sigma(-\theta^\top t_n)^{z_{n,2}}, \quad \forall n \end{aligned} \tag{4.18}$$

where $\sigma(y) \triangleq 1 / (1 + \exp(-y))$ is the logistic function. The variable z is observed and there is no additional variable x downstream as assumed in Section 4.1.1. Our variational distribution for parameter θ is $q(\theta) = \mathcal{N}(\mu, \Sigma)$. If we use Laplace variational inference, this becomes the standard Laplace approximation for Bayesian logistic regression [16]. Delta method inference provides an alternative. The detailed derivations are in Appendix B.

Hierarchical Bayesian logistic regression considers multiple related logistic regression problems and solves them together [47]. Taking an empirical Bayes approach, we treat the parameters for each problem from a shared prior and then estimate the hyperparameters of that prior with maximum likelihood or MAP. If we have M related problems, we construct the following hierarchical model,

$$\begin{aligned} \mu_0 &\sim \mathcal{N}(0, \hat{\Sigma}_0), \quad \Sigma_0^{-1} \sim \text{Wishart}(\hat{\nu}, \hat{\Phi}_0), \\ p(\theta_m) &= \mathcal{N}(\mu_0, \Sigma_0), \quad \forall m, \quad p(z_{mn} | \theta_m, t_{mn}) = \sigma(\theta_m^\top t_{mn})^{z_{mn,1}} \sigma(-\theta_m^\top t_{mn})^{z_{mn,2}}, \quad \forall m, n. \end{aligned} \tag{4.19}$$

We construct $f(\theta_m)$ in Eq. 4.12 separately for each m , and fit the hyperparameters μ_0 and Σ_0 using reguarlized variational EM [16]. This amounts to MAP estimation, with priors as specified above.

⁴Logistic regression is a generalized linear model with a binary response and canonical link function [74]. It is straightforward to use our algorithm with other Bayesian generalized linear models.

4.3 Empirical Study

We studied the Laplace and delta method variational inference for the CTM and Bayesian logistic regression model on several real-world datasets.

Correlated topic models (CTM). For the CTM, we analyzed two document collections. The *Associated Press* data contains 2,246 documents from the *Associated Press*, with 436K observed words and a vocabulary size of 10,473 terms. The *New York Times* data contains 9,238 documents from the *New York Times*, with 2.3 million observed words and a vocabulary size of 10,760 terms.

We measured per-word held-out log likelihood to evaluate the models' fit to the data. For each collection, we held out 20% of the documents as a test set. We split each document \mathbf{w}_d in $\mathcal{D}_{\text{test}}$ into halves, $\mathbf{w}_i = (\mathbf{w}_{d1}, \mathbf{w}_{d2})$, and computed the log likelihood of \mathbf{w}_{d2} conditioned on \mathbf{w}_{d1} and $\mathcal{D}_{\text{train}}$, similar to [9]. A better predictive distribution gives higher likelihood to the second half.

Let $\bar{\pi}_d$ be the variational expectation of the topic proportions given \mathbf{w}_{d1} . The predictive probability of \mathbf{w}_{d2} given \mathbf{w}_{d1} is approximated by $p(\mathbf{w}_{d2}|\mathbf{w}_{d1}, \mathcal{D}_{\text{train}}) \approx \prod_{w \in \mathbf{w}_{d2}} \sum_k \bar{\pi}_{dk} \beta_{kw}$. Then the per-word held-out log likelihood is

$$\text{likelihood}_{\text{pw}} \triangleq \sum_{d \in \mathcal{D}_{\text{test}}} \log p(\mathbf{w}_{d2}|\mathbf{w}_{d1}, \mathcal{D}_{\text{train}}) / \sum_{d \in \mathcal{D}_{\text{test}}} |\mathbf{w}_{d2}|.$$

This evaluation lets us compare methods regardless of whether they provide a bound. It is a measure of the quality of the estimated predictive distribution. We compared the original method of [23] to Laplace and delta method variational inference. We varied the number of topics from 20 to 80. We stopped fitting when the relative change of the (approximated) lower bound in the corpus-level EM algorithm was smaller than $10\text{e-}5$.

Figure 4.2(a) shows per-word held-out log likelihood as a function of the number of topics. On both data sets, the original algorithm of [23], tailored for this model, usually performed worse than both generic algorithms. Our conjecture is that [23] gives a strict

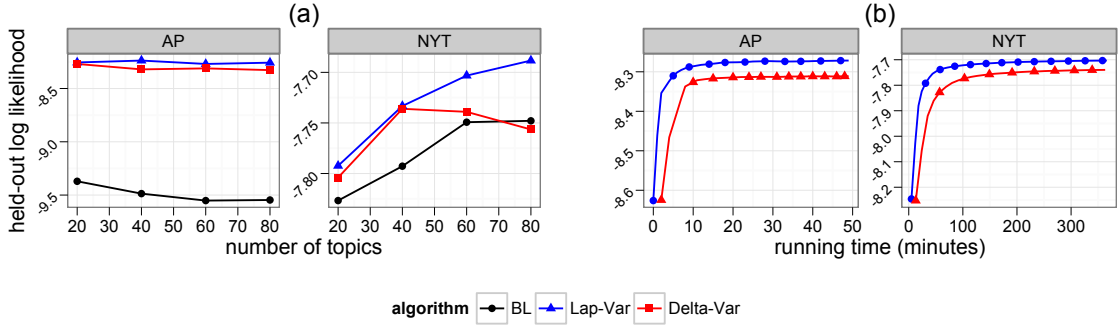


Figure 4.2: Comparison on per-word held-out log likelihood. Higher numbers are better. Laplace inference is “Lap-Var”; delta method inference is “Delta-Var”; the method by Blei and Lafferty in [23] is BL. (a) Held-out likelihood against number of topics. Delta-Var performs better BL. Lap-Var performs best. (b) Held-out likelihood against running time when $K = 60$. Lap-Var is faster and gives performs better than Delta-Var.

lower bound, which might be loose; our method uses an approximation which, while not a bound, might be closer to the objective and give better predictive distributions. Laplace inference was always better than both other algorithms. This might be because the delta method uses a diagonal covariance matrix of the variational distribution and Laplace method can use a full matrix. (This might also explain when $K = 80$ for NYT, the delta method performs slightly worse than [23].)

We also compared the running times of Laplace inference and delta method inference. For a model with 60 topics, Figure 4.2(b) shows the per-word held-out log likelihood as a function of time.⁵ (Other topic settings looked similar.) In addition to finding a better fit to the data, Laplace inference is faster.

Bayesian logistic regression. We studied our algorithms on Bayesian logistic regression in both standard and hierarchical settings. In the standard setting, we analyzed two datasets. The *Yeast* data [42] is formed by micro-array expression data and phylogenetic profiles. There are 1500 genes in the training set and 917 genes in the test set. The input dimension is 103. One gene is associated with up to 14 different edges (14 labels), corresponding to 14 independent binary classification problems. The *Scene* data [28] contains 1,211 training and 1,196 test images, with 294 images features and up to 6 scene labels per image,

⁵We did not formally compare [23]’s method on running time because we used the authors’ C implementation while ours in Python. We observed, however, that their method took about 5 times longer.

corresponding to 6 independent binary classification problems.

In the hierarchical setting, we analyzed the *School* data. This data was from the Inner London Education Authority and consists of examination records from 139 secondary schools in 1985, 1986 and 1987. It is a random 50% sample with 15,362 students. The students’ features contain four student-dependent features (year of the exam, gender, VR band and ethnic group) and four school-dependent features (percentage of students eligible for free school meals, percentage of students in VR band 1, school gender and school denomination). We coded the binary indicator of whether each was below the median (“bad”) or above (“good”). We use the same 10 random splits of the data as [8]. In this data, we can either treat each school as a separate classification problem, consider all the schools together as a single big school, or analyze them with hierarchical logistic regression.

We used two metrics: 1) accuracy, i.e., the proportion of examples correctly labeled, 2) averaged log predictive likelihood. Given test input t with label z , the predictive likelihood is computed as

$$\log p(z | \mu, t) = z_1 \log \sigma(\mu^\top t) + z_2 \log \sigma(-\mu^\top t),$$

where μ is the mean of variational distribution $q(\theta) = \mathcal{N}(\mu, \Sigma)$. Higher likelihood is better.

We compared the following methods on (hierarchical) Bayesian logistic regression. (1) *Jaakkola*: the variational approach in [59]. (2) *Lap-Var*: Laplace variational inference in Section 4.1.2. (3) *Delta-Var*: delta method variational inference in Section 4.1.3. (4) *Hier-Lap-Var*: the hierarchical version of Lap-Var. This is only performed on *School*. (5) *Hier-Delta-Var*: the hierarchical version of Delta-Var. This is only performed on *School*.

In Jaakkola, Lap-Var and Delta-Var, we set $\mu_0 = 0$ and $\Sigma_0 = I$ in Eq. 4.18 to favor sparsity. In Hier-Lap-Var and Hier-Delta-Var, we set $\hat{\Sigma}_0 = 0.01I$, $\hat{\nu} = p + 100$, $\hat{\Phi}_0 = 0.01I$ in Eq. 4.19 also to favor sparsity. Here p is the dimensionality of the input feature.

Table 4.1 shows the results on standard Bayesian logistic regression on Yeast and Scene data. Lap-Var and Delta-Var gave slightly better accuracy but much better log predictive

Table 4.1: Comparison of different methods on standard Bayesian logistic regression using accuracy (Acc.) and averaged log predictive likelihood (Lik.)—the higher, the better. Results are averaged from five random starts. Variance is too small to show. The best results are highlighted. Lap-Var and Delta-Var gives slightly better accuracy but much better predictive log likelihood.

Yeast		
Algorithm	Acc.	Lik.
Jaakkola	79.7%	-0.678
Lap-Var	80.1%	-0.449
Delta-Var	80.2%	-0.450
Scene		
Algorithm	Acc.	Lik.
Jaakkola	87.4%	-0.670
Lap-Var	89.4%	-0.259
Delta-Var	89.5%	-0.265

Table 4.2: Comparison of different methods on School data using accuracy (Acc.) and averaged log predictive likelihood (Lik.). Results are averaged from 10 random splits. Variance is too small to show. The best results are highlighted. Lap-Var-all indicates the case where we treats all schools together. The hierarchical approach performs the best.

School		
Algorithm	Acc.	Lik.
Jaakkola	70.5%	-0.684
Lap-Var	70.8%	-0.569
Delta-Var	70.8%	-0.571
Jaakkola-all	71.2%	-0.685
Lap-Var-all	71.3%	-0.557
Delta-Var-all	71.3%	-0.557
Hier-Lap-Var	71.9%	-0.549
Hier-Delta-Var	71.9%	-0.559

likelihood than the Jaakkola method.⁶

Table 4.2 shows the results on School data. The hierarchical Bayesian logistic regression performed the best on both accuracy and predictive log likelihood.

4.4 Conclusions

We have developed two strategies for variational inference in a large class of nonconjugate models, Laplace and delta method variational inference. Although our methods no longer guarantee a strict lower bound, they work well in practice, forming approximate posteriors that lead to good predictions. In the examples we analyzed, these methods work better than methods tailored for specific models. Further, we showed that Laplace inference is better and faster than delta method inference. One area of future work is that in more complicated models, we may not want to assume full conditional conjugacy of z . In those settings, we may be able to use moment matching [101] to develop efficient variational algorithms.

⁶Previous literature, e.g., [114, 7] also treat *Yeast* and *Scene* as multi-task problems. This is slightly non-standard, since each sub-problem does share the same input features. We found the standard Bayesian logistic regression performed competitively with the multi-task algorithms reported in [7] in terms of accuracy. We also found the hierarchical version did worse in this setting.

Chapter 5

Simultaneous Image Classification and Annotation

Starting from this chapter, we focus on developing novel models for real-world applications. In this chapter, we consider the problem of modeling image data that are both labeled with a category and annotated with free text. In such data, the class label tends to globally describe each image, while the annotation terms tend to describe its individual components. For example, an image in the *outdoor* category might be annotated with “tree,” “flower,” and “sky.”

Image classification and image annotation are typically treated as two independent problems. Our motivating intuition, however, is that these two tasks should be connected. An image annotated with “car” and “pedestrian” is unlikely to be labeled as a *living room* scene. An image labeled as an *office* scene is unlikely to be annotated with “swimming pool” or “sunbather.” In this chapter, we develop a probabilistic model that simultaneously learns the salient patterns among images that are predictive of their class labels and annotation terms. For new unknown images, our model provides predictive distributions of both class and annotation.

We build on recent machine learning and computer vision research in probabilistic topic models, such as latent Dirichlet allocation (LDA) [26] and probabilistic latent semantic indexing [55] (pLSI). Probabilistic topic models find a low dimensional representation of data under the assumption that each data point can exhibit multiple components or “topics.”

While topic models were originally developed for text, they have been successfully adapted and extended to many computer vision problems [20, 11, 40, 44, 27].

Our model finds a set of image topics that are predictive of both class label and annotations. The two main contributions of this work are:

1. We extended supervised topic modeling [25] (sLDA) to classification problems. SLDA was originally developed for predicting continuous response values, via a linear regression. We note that the multi-class extension presented here is not simply a “plug-and-play” extension of [25]. As we show in Section 5.1.2, it requires substantial development of the underlying inference and estimation algorithms.
2. We embed a probabilistic model of image annotation into the resulting supervised topic model. This yields a *single coherent* model of images, class labels and annotation terms, allowing classification and annotation to be performed using the *same* latent topic space.

We find that a single model, fit to images with class labels and annotation terms, provides state-of-the-art annotation performance and exceeds the state-of-the-art in classification performance. This shows that image classification and annotation can be performed simultaneously.

This chapter is organized as follows. In Section 2, we describe our model and derive variational algorithms for inference, estimation, and prediction. In Section 3, we describe related work. In Section 4, we study the performance of our models on classification and annotation for two real-world image datasets. We summarize our findings in Section 5.

5.1 Models and Algorithms

In this section, we develop two models: *multi-class sLDA* and *multi-class sLDA with annotations*. We derive a variational inference algorithm for approximating the posterior distribution, and an approximate parameter estimation algorithm for finding maximum

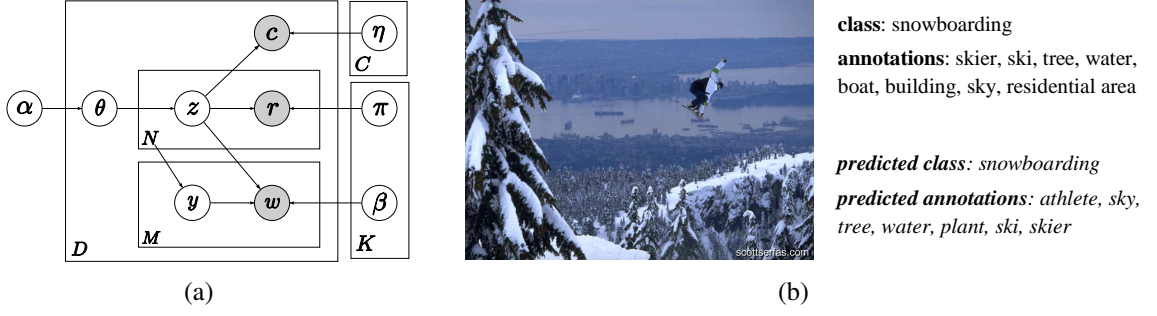


Figure 5.1: (a). A graphical model representation of our model. Nodes represent random variables; edges denote possible dependence between random variables; plates denote replicated structure. Note that in this model, the image class c and image annotation w_m are dependent on the topics that generated the image codewords r_n . (b). An example image with the class label and annotations from the UIUC-Sport dataset [69]. The italic words are the predicted class label and annotations, using our model.

likelihood estimates of the model parameters. Finally, we derive prediction algorithms for using these models to label and annotate new images.

5.1.1 Modeling images, labels and annotations

The idea behind our model is that class and annotation are related, and we can leverage that relationship by finding a latent space predictive of both. Our training data are images that are categorized and annotated. In testing, our goal is to predict the category and annotations of a new image.

Each image is represented as a bag of “codewords” $r_{1:N}$, which are obtained by running the k-means algorithm on patches of the images [62, 72]. (See Section 5.3 for more details about our image features.) The category c is a discrete class label. The annotation $w_{1:M}$ is a collection of words from a fixed vocabulary.

We fix the number of topics K and let C denote the number of class labels. The parameters of our model are a set of K image topics $\pi_{1:K}$, a set of K annotation topics $\beta_{1:K}$, and a set of C class coefficients $\eta_{1:C}$. Each coefficient η_c is a K -vector of real values. Each “topic” is a distribution over a vocabulary, either image codewords or annotation terms. Our model assumes the following generative process of an image, its class label, and its annotation.

1. Draw topic proportions $\theta \sim \text{Dir}(\alpha)$.

2. For each image region r_n , $n \in \{1, 2, \dots, N\}$:
 - (a) Draw topic assignment $z_n \mid \theta \sim \text{Mult}(\theta)$.
 - (b) Draw region codeword $r_n \mid z_n \sim \text{Mult}(\pi_{z_n})$.
3. Draw class label $c \mid z_{1:N} \sim \text{softmax}(\bar{z}, \eta)$, where $\bar{z} = \frac{1}{N} \sum_{n=1}^N z_n$ is the empirical topic frequencies and the softmax function provides the following distribution,

$$p(c \mid \bar{z}, \eta) = \exp(\eta_c^T \bar{z}) / \sum_{l=1}^C \exp(\eta_l^T \bar{z}).$$

4. For each annotation term w_m , $m \in \{1, 2, \dots, M\}$:
 - (a) Draw region identifier $y_m \sim \text{Unif}\{1, 2, \dots, N\}$
 - (b) Draw annotation term $w_m \sim \text{Mult}(\beta_{z_{y_m}})$.

Figure 5.1(a) illustrates our model as a graphical model.

We refer to this model as *multi-class sLDA with annotations*. It models both the image class and image annotation with the same latent space.

Consider step 3 of the generative process. In modeling the class label, we use a similar set-up as supervised LDA (sLDA) [25]. In sLDA, a response variable for each “document” (here, an image) is assumed drawn from a generalized linear model with input given by the empirical distribution of topics that generated the image patches. In [25], that response variable is real valued and drawn from a linear regression, which simplified inference and estimation.

However, a continuous response is not appropriate for our goal of building a classifier. Rather, we consider a class label response variable, drawn from a softmax regression for classification. This complicates the approximate inference and parameter estimation algorithms (see Section 5.1.2 and 5.1.3), but provides an important extension to the sLDA framework. We refer to this multi-class extension of sLDA (without the annotation portion) as *multi-class sLDA*. We note that multi-class sLDA can be used in classification problems

outside of computer vision.

We now turn to step 4 of the generative process. To model annotations, we use the same generative process as correspondence LDA (corr-LDA) [20], where each annotation word is assumed to be drawn from one of the topics that is associated with an image patch. For example, this will encourage words like “blue” and “white” to be associated with the image topics that describe patches of sky.

We emphasize that Corr-LDA and sLDA were developed for different purposes. Corr-LDA finds topics predictive of annotation words; sLDA finds topics predictive of a global response variable. However, both approaches employ similar statistical assumptions. First, generate the image from a topic model. Then, generate its annotation or class label from a model conditioned on the topics which generated the image. Our model uses the *same* latent topic space to generate both the annotation and class label.

5.1.2 Approximate inference

In posterior inference, we compute the conditional distribution of the latent structure given a model and a labeled annotated image. As for LDA, computing this posterior exactly is not possible [26]. We employ mean-field variational methods for a scalable approximation algorithm.

Variational methods consider a simple family of distributions over the latent variables, indexed by free variational parameters, and try to find the setting of those parameters that minimizes the Kullback-Leibler (KL) divergence to the true posterior [61]. In our model, the latent variables are the per-image topic proportions θ , the per-codeword topic assignment z_n , and the per-annotation word region identifier y_m . Note that there are no latent variables explicitly associated with the class; its distribution is wholly governed by the per-codeword topic assignments.

The mean-field variational distribution is,

$$q(\theta, \mathbf{z}, \mathbf{y}) = q(\theta|\gamma) \prod_{n=1}^N q(z_n|\phi_n) \prod_{m=1}^M q(y_m|\lambda_m), \quad (5.1)$$

where ϕ_n is a variational multinomial over the K topics, γ is a variational Dirichlet, and λ_m is a variational multinomial over the image regions. We fit these parameters with coordinate ascent to minimize the KL divergence between q and the true posterior. (This will find a local minimum.)

Let $\Theta = \{\alpha, \beta_{1:K}, \eta_{1:C}, \pi_{1:K}\}$. Following Jordan et al. [61], we bound the log-likelihood of a image-class-annotation triple, $(\mathbf{r}, c, \mathbf{w})$. We have:

$$\begin{aligned} \log p(\mathbf{r}, c, \mathbf{w}|\Theta) &= \log \int \frac{p(\theta, \mathbf{z}, \mathbf{y}, \mathbf{r}, c, \mathbf{w}|\Theta)q(\theta, \mathbf{z}, \mathbf{y})}{q(\theta, \mathbf{z}, \mathbf{y})} d\theta d\mathbf{z} d\mathbf{y} \\ &\geq \mathbb{E}_q [\log p(\theta, \mathbf{z}, \mathbf{y}, \mathbf{r}, c, \mathbf{w}|\Theta)] - \mathbb{E}_q [q(\theta, \mathbf{z}, \mathbf{y})] \\ &= \mathcal{L}(\gamma, \phi, \lambda; \Theta). \end{aligned} \quad (5.2)$$

The coordinate ascent updates for γ and λ are the same as those in [26], which uses the same notation:

$$\gamma = \alpha + \sum_{n=1}^N \phi_n \quad (5.3)$$

$$\lambda_{mn} \propto \exp \left(\sum_{i=1}^K \phi_{ni} \log \beta_{i,w_m} \right). \quad (5.4)$$

We next turn to the update for the variational multinomial ϕ . Here, the variational method derived in [25] cannot be used because the expectation of the log partition function for softmax regression (i.e., multi-class classification) cannot be exactly computed. The terms in \mathcal{L} containing ϕ_n are:

$$\begin{aligned} \mathcal{L}_{[\phi_n]} &= \sum_{i=1}^K \phi_{ni} \left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^K \gamma_j\right) + \log \pi_{i,r_n} + \sum_{m=1}^M \lambda_{mn} \log \beta_{i,w_m} \right) + \frac{1}{N} \eta_c^T \phi_n \\ &\quad - \mathbb{E}_q \left[\log \left(\sum_{l=1}^C \exp(\eta_l^T \bar{\mathbf{z}}) \right) \right] - \sum_{i=1}^K \phi_{ni} \log \phi_{ni}. \end{aligned} \quad (5.5)$$

The central issue here is that exactly computing $-\mathbb{E}_q \left[\log \left(\sum_{l=1}^C \exp(\eta_l^T \bar{\mathbf{z}}) \right) \right]$ takes $O(K^N)$

time. To address this, we lower bound this term with Jensen's inequality. This gives:

$$\begin{aligned}\mathbb{E}_q \left[\log \left(\sum_{l=1}^C \exp(\eta_l^T \bar{z}) \right) \right] &\geq -\log \left(\sum_{l=1}^C \mathbb{E}_q [\exp(\eta_l^T \bar{z})] \right) \\ &= -\log \left(\sum_{l=1}^C \prod_{n=1}^N \left(\sum_{j=1}^K \phi_{nj} \exp \left(\frac{1}{N} \eta_{lj} \right) \right) \right). \quad (5.6)\end{aligned}$$

Plugging Equation 5.6 into Equation 5.5, we obtain a lower bound of $\mathcal{L}_{[\phi_n]}$, which we will denote $\mathcal{L}'_{[\phi_n]}$.

We present a fixed-point iteration for maximizing this proxy. The idea is that given an old estimation of ϕ_n^{old} , a lower bound of $\mathcal{L}'_{[\phi_n]}$ is constructed so that this lower bound is tight on ϕ_n^{old} [77]. Then maximizing this lower bound of $\mathcal{L}'_{[\phi_n]}$ is solved in closed-form and ϕ_n^{old} is updated correspondingly. We note that $\sum_{l=1}^C \prod_{n=1}^N \left(\sum_{j=1}^K \phi_{nj} \exp \left(\frac{1}{N} \eta_{lj} \right) \right)$ is only a linear function of ϕ_n , thus can be written as $h^T \phi_n$, where $h = [h_1, \dots, h_i, \dots, h_K]^T$ and does not contain ϕ_n . For convenience, define b_i as follows,

$$b_i = \Psi(\gamma_i) - \Psi\left(\sum_{j=1}^K \gamma_j\right) + \log \pi_{i,r_n} + \sum_{m=1}^M \lambda_{mn} \log \beta_{i,w_m}.$$

Now, the lower bound $\mathcal{L}'_{[\phi_n]}$ can be written as

$$\mathcal{L}'_{[\phi_n]} = \sum_{i=1}^K \phi_{ni} b_i + \frac{1}{N} \eta_c^T \phi_n - \log(h^T \phi_n) - \sum_{i=1}^K \phi_{ni} \log \phi_{ni}.$$

Finally, suppose we have a previous value ϕ_n^{old} . For $\log(x)$, we know $\log(x) \leq \zeta^{-1}x + \log(\zeta) - 1, \forall x > 0, \zeta > 0$, where the equality holds if and only if $x = \zeta$. Set $x = h^T \phi_n$ and $\zeta = h^T \phi_n^{\text{old}}$. Immediately, we have:

$$\mathcal{L}'_{[\phi_n]} \geq \sum_{i=1}^K \phi_{ni} b_i + \frac{1}{N} \eta_c^T \phi_n - (h^T \phi_n^{\text{old}})^{-1} h^T \phi_n - \log(h^T \phi_n^{\text{old}}) + 1 - \sum_{i=1}^K \phi_{ni} \log \phi_{ni}. \quad (5.7)$$

This lower bound of $\mathcal{L}'_{[\phi_n]}$ is tight when $\phi_n = \phi_n^{\text{old}}$. Maximizing Equation 5.7 under the constraint $\sum_{i=1}^K \phi_{ni} = 1$ leads to the fixed point update,

$$\phi_{ni} \propto \pi_{i,r_n} \exp \left(\Psi(\gamma_i) + \sum_{m=1}^M \lambda_{mn} \log \beta_{i,w_m} + \frac{1}{N} \eta_{ci} - (h^T \phi_n^{\text{old}})^{-1} h_i \right). \quad (5.8)$$

Observe how the per-feature variational distribution over topics ϕ depends on both class label c and annotation information w_m . The combination of these two sources of data has naturally led to an inference algorithm that uses both. The full variational inference procedure repeats the updates of Equations 5.3, 5.4 and 5.8 until Equation 5.2, the lower bound on the log marginal probability $\log p(\mathbf{r}, c, \mathbf{w} | \Theta)$, converges.

5.1.3 Parameter estimation

Given a corpus of image data with class labels and annotations, $\mathcal{D} = \{(\mathbf{r}_d, \mathbf{w}_d, c_d)\}_{d=1}^D$, we find the maximum likelihood estimation for image topics $\pi_{1:K}$, text topics $\beta_{1:K}$ and class coefficients $\eta_{1:C}$. We use variational EM, which replaces the E-step of expectation-maximization with variational inference to find an approximate posterior for each data point. In the M-step, as in exact EM, we find approximate maximum likelihood estimates of the parameters using expected sufficient statistics computed from the E-step.

Recall $\Theta = \{\alpha, \beta_{1:K}, \eta_{1:C}, \pi_{1:K}\}$. The corpus log-likelihood is,

$$\mathcal{L}(\mathcal{D}) = \sum_{d=1}^D \log p(\mathbf{r}_d, c_d, \mathbf{w}_d | \Theta). \quad (5.9)$$

(We do not optimize α .) Again, we maximize the lower bound of $\mathcal{L}(\mathcal{D})$ by plugging Equations 5.2 and 5.6 into Equation 5.9.

Let V_r denote the number of codewords, the terms containing $\pi_{1:K}$ (with Lagrangian multipliers) are:

$$\mathcal{L}_{[\pi_{1:K}]}(\mathcal{D}) = \sum_{d=1}^D \sum_{n=1}^{N_d} \sum_{i=1}^K \phi_{dni} \log \pi_{i,r_n} + \sum_{i=1}^K \mu_i \left(\sum_{f=1}^{V_r} \pi_{if} - 1 \right).$$

Setting $\partial \mathcal{L}_{[\pi_{1:K}]}(\mathcal{D}) / \partial \pi_{if} = 0$ leads to

$$\pi_{if} \propto \sum_{d=1}^D \sum_{n=1}^{N_d} 1[r_n = f] \phi_{dni}. \quad (5.10)$$

Next, let V_w denote the number of total annotations, and the terms containing $\beta_{1:K}$ (with Lagrangian multipliers) are:

$$\mathcal{L}_{[\beta_{1:K}]}(\mathcal{D}) = \sum_{m=1}^M \sum_{n=1}^N \sum_{i=1}^K \lambda_{mn} \phi_{ni} \log \beta_{i,w_m} + \sum_{i=1}^K \nu_i \left(\sum_{w=1}^{V_w} \beta_{iw} - 1 \right).$$

Setting $\partial \mathcal{L}_{[\beta_{1:K}]}(\mathcal{D}) / \partial \beta_{iw} = 0$ leads to

$$\beta_{iw} \propto \sum_{d=1}^D \sum_{m=1}^M 1[w_m = w] \sum_n \phi_{dni} \lambda_{dmn}. \quad (5.11)$$

Finally, terms containing $\eta_{1:C}$ are:

$$\mathcal{L}_{[\eta_{1:C}]}(\mathcal{D}) = \sum_{d=1}^D \left(\eta_{c_d}^T \bar{\phi}_d - \log \left(\sum_{c=1}^C \prod_{n=1}^{N_d} \left(\sum_{i=1}^K \phi_{dni} \exp \left(\frac{1}{N_d} \eta_{ci} \right) \right) \right) \right).$$

Setting $\partial \mathcal{L}_{[\eta_{1:C}]}(\mathcal{D}) / \partial \eta_{ci} = 0$ does not lead to a closed-form solution. We optimize with conjugate gradient [82]. Let $\kappa_d = \sum_{c=1}^C \prod_{n=1}^{N_d} \left(\sum_{i=1}^K \phi_{dni} \exp \left(\frac{1}{N_d} \eta_{ci} \right) \right)$. Conjugate gradient only requires the derivatives:

$$\begin{aligned} \frac{\partial \mathcal{L}_{[\eta_{1:C}]}(\mathcal{D})}{\partial \eta_{ci}} &= \sum_{d=1}^D (1[c_d = c] \bar{\phi}_{di}) - \\ &\sum_{d=1}^D \left(\kappa_d^{-1} \prod_{n=1}^{N_d} \left(\sum_{j=1}^K \phi_{dnj} \exp \left(\frac{1}{N_d} \eta_{cj} \right) \right) \sum_{n=1}^{N_d} \left(\frac{\frac{1}{N_d} \phi_{dni} \exp \left(\frac{1}{N_d} \eta_{ci} \right)}{\sum_{j=1}^K \phi_{dnj} \exp \left(\frac{1}{N_d} \eta_{cj} \right)} \right) \right). \end{aligned} \quad (5.12)$$

5.1.4 Classification and annotation

With inference and parameter estimation algorithms in place, it remains to describe how to perform prediction, i.e. predicting both a class label and annotations from an unknown

image. The first step is to perform variational inference given the unknown image. We can use a variant of the algorithm in Section 5.1.2 to determine $q(\theta, \mathbf{z})$. Since the class label and annotations are not observed, we remove the λ_{mn} terms from the variational distribution (Equation 5.1) and the terms involving η_c from the updates on the topic multinomials (Equation 5.8).

In classification, we estimate the probability of the label c by replacing the true posterior $p(\mathbf{z}|\mathbf{w}, \mathbf{r})$ with the variational approximation

$$\begin{aligned} p(c|\mathbf{r}, \mathbf{w}) &\approx \int \exp \left(\eta_c^T \bar{\mathbf{z}} - \log \left(\sum_{l=1}^C \exp(\eta_l^T \bar{\mathbf{z}}) \right) \right) q(\mathbf{z}) d\mathbf{z} \\ &\geq \exp \left(\mathbb{E}_q [\eta_c^T \bar{\mathbf{z}}] - \mathbb{E}_q \left[\log \left(\sum_{l=1}^L \exp(\eta_l^T \bar{\mathbf{z}}) \right) \right] \right), \end{aligned}$$

where the last equation comes from Jensen's inequality, and q is the variational posterior computed in the first step. The second term in the exponent is constant with respect to class label. Thus, the prediction rule is

$$c^* = \arg \max_{c \in \{1, \dots, C\}} \mathbb{E}_q [\eta_c^T \bar{\mathbf{z}}] = \arg \max_{c \in \{1, \dots, C\}} \eta_c^T \bar{\boldsymbol{\phi}}. \quad (5.13)$$

There are two approximations at play. First, we approximate the posterior with q . Second, we approximate the expectation of an exponential using Jensen's inequality. While there are no theoretical guarantees here, we evaluate this classification procedure empirically in Section 5.3.

The procedure for predicting annotations is the same as in [20]. To obtain a distribution over annotation terms, we average the contributions from each region,

$$p(w|\mathbf{r}, c) \approx \sum_{n=1}^N \sum_{z_n} p(w|z_n, \beta) q(z_n). \quad (5.14)$$

5.2 Related work

Image classification and annotation are both important problems in computer vision and machine learning. Much previous work has explored the use of global image features for scene (or event) classification [83, 102, 97, 103, 69], and both discriminative and generative techniques have been applied to this problem. Discriminative methods include the work in [36, 116, 111, 68]. Generative methods include the work in [44, 34, 86, 69]. In the work of [27], the authors combine generative models for latent topic discovery [56] and discriminative methods for classification (k-nearest neighbors). LDA-based image classification was introduced in [44], where each category is identified with its own Dirichlet prior, and that prior is optimized to distinguish between them. The *multi-class sLDA* model combines the generative and discriminative approaches, which may be better for modeling categorized images (see Section 5.3).

For image annotation, several studies have explored the use of probabilistic models to learn the relationships between images and annotation terms [11, 40, 60]. Our model is most related to the family of models based on LDA, which were introduced to image annotation in [40]. But the idea that image annotation and classification might share the same latent space has not been studied. We will compare the performance of our model to corr-LDA [20]. (Corr-LDA was shown to provide better performance than the previous LDA-based annotation models in [11] and [40].)

5.3 Empirical results

We test our models with two real-world data sets that contain class labels and annotations: a subset from LabelMe [90] and the UIUC-Sport data from [69]. In the LabelMe data, we used the on-line tool to obtain images from the following 8 classes: “highway,” “inside city,” “tall building,” “street,” “forest,” “coast,” “mountain,” and “open country.” We first only kept the images that were 256×256 pixels, and then randomly selected 200 images for each class. (In doing this, we attempted to obtain the same image data as described

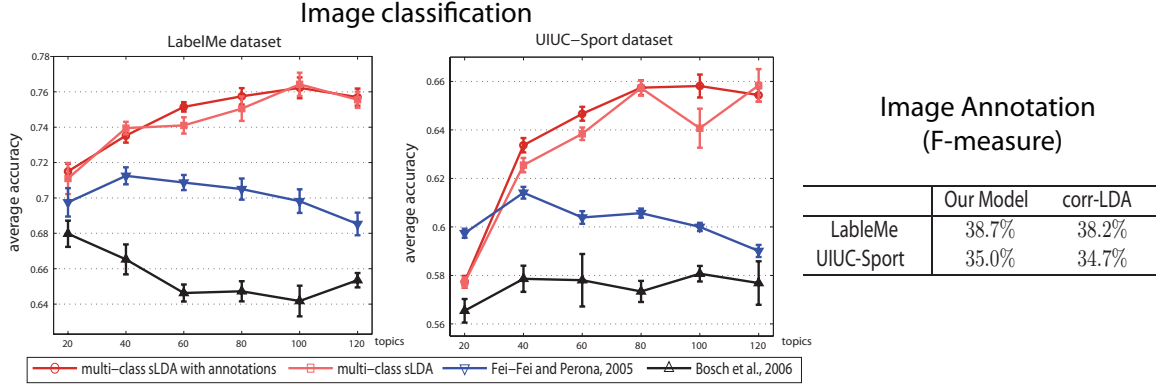


Figure 5.2: Comparisons of average accuracy over all classes based on 5 random train/test subsets. *multi-class sLDA with annotations* and *multi-class sLDA* (red curves in color) are both our models. **left.** Accuracy as a function of the number of topics on the LabelMe dataset. **right.** Accuracy as a function of the number of topics on the UIUC-Sport dataset. in [44].) The total number of images is 1600. The UIUC-Sport dataset [69] contains 8 types of sports: “badminton,” “bocce,” “croquet,” “polo,” “rockclimbing,” “rowing,” “sailing” and “snowboarding.” The number of images in each class varies from 137 (bocce) to 250 (rowing). The total number of images is 1792.

Following the setting in [44], we use the 128-dimensional SIFT [72] region descriptors selected by a sliding grid (5×5). We ran the k-means algorithm [62] to obtain the codewords and codebook. We report on a codebook of 240 codewords. (Other codebook sizes gave similar performance.) In both data sets, we removed annotation terms that occurred less than 3 times. On average, there are 6 terms per annotation in the LabelMe data, and 8 terms per annotation in the UIUC-Sport data. Finally, We evenly split each class to create the training and testing sets.

Our procedure is to train the *multi-class sLDA with annotations* on labeled and annotated images, and train the *multi-class sLDA* model on labeled images. All testing is on unlabeled and unannotated images. See Figure 5.4 for example annotations and classifications from the *multi-class sLDA with annotations*.

Image Classification. To assess our models on image classification, we compared the following methods,

1. *Fei-Fei and Perona, 2005*: This is the model from [44]. It is trained on labeled images without annotation.
2. *Bosch et al., 2006*: This is the model described in [27]. It first employs pLSA [56] to learn latent topics, and then uses the k-nearest neighbor (KNN) classifier for classification. We use unsupervised LDA¹ to learn the latent topics and, following [27], set the number of neighbors to be 10. As for the other models considered here, we use SIFT features. We note that [27] use other types of features as well.
3. *multi-class sLDA*: This is the multi-class sLDA model, described in this chapter.
4. *multi-class sLDA with annotations*: This is multi-class sLDA with annotations, described in this chapter.

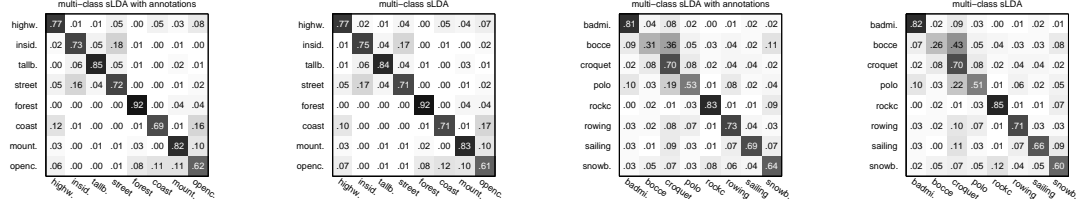
Note all testing is performed on unlabeled and unannotated images.

The results are illustrated in the graphs of Figure 5.2 and in the confusion matrices of Figure 5.3.² Our models—multi-class sLDA and multi-class sLDA with annotations—perform better than the other approaches. They reduce the error of *Fei-Fei and Perona, 2005* by at least 10% on both data sets, and even more for *Bosch et al., 2006*. This demonstrates that multi-class sLDA is a better classifier, and that joint modeling does not negatively affect classification accuracy when annotation information is available. In fact, it usually increases the accuracy.

Observe that the model of [27], unsupervised LDA combined with KNN, gives the worst performance of these methods. This highlights the difference between finding topics that are predictive, as our models do, and finding topics in an unsupervised way. The accuracy of unsupervised LDA might be increased by using some of the other visual features suggested by [27]. Here, we restrict ourselves to SIFT features in order to compare models, rather than feature sets.

¹According to [96], pLSA performs similarly to unsupervised LDA in practice.

²Other than the topic models listed, we also tested an SVM-based approach using SIFT image features. The SVM yielded much worse performance than the topic models (47% for the LabelMe data, and 20% for the UIUC-Sport data). These are not marked on the plots.



(a) LabelMe: avg. accu- (b) LabelMe: avg. accu- (c) UIUC-Sport: avg. ac- (d) UIUC-Sport: avg. ac-
racy: 76% racy: 76% curacy: 66% curacy: 65%

Figure 5.3: Comparisons using confusion matrices, all from the 100-topic models using *multi-class sLDA with annotations* and *multi-class sLDA*. (a) *multi-class sLDA with annotations* on the LabelMe dataset. (b) *multi-class LDA* on the LabelMe dataset. (c) *multi-class sLDA with annotations* on the UIUC-Sport dataset. (d) *multi-class sLDA* model on the UIUC-Sport dataset.

As the number of topics increases, the multi-class sLDA models (with and without annotation) do not overfit until around 100 topics, while *Fei-Fei and Perona, 2005* begins to overfit at 40 topics. This suggests that multi-class sLDA, which combines aspects of both generative and discriminative classification, can handle more latent features than a purely generative approach. On one hand, a large number of topics increases the possibility of overfitting; on the other hand, it provides more latent features for building the classifier.

Image Annotation. In the case of multi-class sLDA with annotations, we can use the same trained model for image annotation. We emphasize that our models are designed for simultaneous classification and annotation. For image annotation, we compare following two methods,

1. *Blei and Jordan, 2003*: This is the corr-LDA model from [19], trained on annotated images.
2. *multi-class sLDA with annotations*: This is exactly the *same* model trained for image classification in the previous section. In testing annotation, we observe only images.

To measure image annotation performance, we use an evaluation measure from information retrieval. Specifically, we examine the top- N F-measure³, denoted as F-measure@ N , where we set $N = 5$. We find that *multi-class sLDA with annotations* performs slightly

³F-measure is defined as $2 * \text{precision} * \text{recall} / (\text{precision} + \text{recall})$.

better than corr-LDA over *all* the numbers of topics tested (about 1% relative improvement). For example, considering models with 100 topics, the LabelMe F-measures are 38.2% (corr-LDA) and 38.7% (multi-class sLDA with annotations); on UIUC-Sport, they are 34.7% (corr-LDA) and 35.0% (multi-class sLDA with annotations).

These results demonstrate that our models can perform classification and annotation with the same latent space. With a single trained model, we find the annotation performance that is competitive with the state-of-the-art, and classification performance that is superior.

5.4 Discussion

We have developed a new graphical model for learning the salient patterns in images that are simultaneously predictive of class and annotations. In the process, we have derived the multi-class setting of supervised topic models and studied its performance for computer vision problems. On real-world image data, we have demonstrated that the proposed model is on par with state-of-the-art image annotation methods and outperforms current state-of-the-art image classification methods. Guided by the intuition that classification and annotation are related, we have illustrated that the same latent space can be used to predict both.





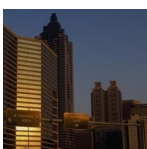
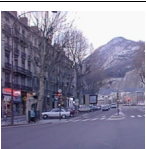
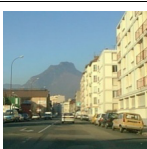
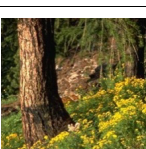

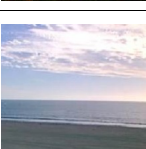
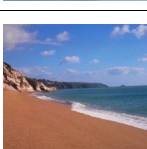
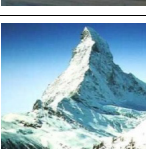
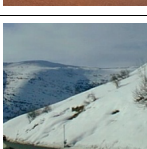

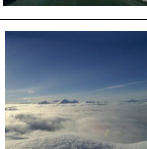
Correct classification with predicted annotations		Incorrect classification (correct class) with predicted annotations
<i>highway</i> car, sign, road		 <i>coast (highway)</i> car, sand beach, tree
<i>inside city</i> buildings, car, sidewalk		 <i>street (inside city)</i> window, tree, building occluded
<i>tall building</i> trees, buildings occluded, window		 <i>inside city (tall building)</i> tree, car, sidewalk
<i>street</i> tree, car, sidewalk		 <i>highway (street)</i> car, window, tree
<i>forest</i> tree trunk, trees, ground grass		 <i>mountain (forest)</i> snowy mountain, tree trunk
<i>coast</i> sand beach, cloud		 <i>open country (coast)</i> sea water, buildings
<i>mountain</i> snowy mountain, sea water, field		 <i>highway (mountain)</i> tree, snowy mountain
<i>open country</i> cars, field, sand beach		 <i>coast (open country)</i> tree, field, sea water

Figure 5.4: Example results from the LabelMe dataset. For each class, left side contains examples with correct classification and predicted annotations, while right side contains wrong ones (the class label in the bracket is the right one) with the predicted annotations. The italic words indicate the class label, while the normal words are associated predicted annotations.

Chapter 6

Collaborative Topic Modeling for Recommendations

In this chapter, we consider building a model for recommending scientific articles for the researchers. In this internet era, modern researchers have easy access to large archives of scientific articles. These archives are growing as new articles are placed online and old articles are scanned and indexed. While this growth has allowed researchers to quickly access more scientific information, it has also made it more difficult for them to find articles relevant to their interests. Modern researchers need new tools for managing what is available to them.

Historically, one way that researchers find articles is by following citations in other articles that they are interested in. This is an effective practice—and one that we should continue—but it limits researchers to specific citation communities, and it is biased towards heavily cited papers. A statistician may miss a relevant paper in economics or biology because the two literatures rarely cite each other; and she may miss a relevant paper in statistics because it was also missed by the authors of the papers that she has read. One of the opportunities of online archives is to inform researchers about literature that they might not be aware of.

A complementary method of finding articles is keyword search. This is a powerful approach, but it is also limited. Forming queries for finding new scientific articles can be

difficult as a researcher may not know what to look for; search is mainly based on content, while good articles are also those that many others found valuable; and search is only good for directed exploration, while many researchers would also like a “feed” of new and interesting articles.

Recently, websites like CiteULike¹ and Mendeley² allow researchers to create their own reference libraries for the articles they are interested in and share them with other researchers. This has opened the door to using recommendation methods [65] as a third way to help researchers find interesting articles. In this chapter, we develop an algorithm for recommending scientific articles to users of online archives. Each user has a library of articles that he or she is interested in, and our goal is to match each user to articles of interest that are not in his or her library.

We have several criteria for an algorithm to recommend scientific articles. First, recommending older articles is important. Users of scientific archives are interested in older articles for learning about new fields and understanding the foundations of their fields. When recommending old articles, the opinions of other users plays a role. A foundational article will be in many users’ libraries; a less important article will be in few.

Second, recommending new articles is also important. For example, when a conference publishes its proceedings, users would like see the recommendations from these new articles to keep up with the state-of-the-art in their discipline. Since the articles are new, there is little information about which or how many other users placed the articles in their libraries, and thus traditional collaborative filtering methods has difficulties making recommendations. With new articles, a recommendation system must use their content.

Finally, exploratory variables can be valuable in online scientific archives and communities. For example, we can summarize and describe each user’s preference profile based on the content of the articles that he or she likes. This lets us connect similar users to enhance the community, and indicate why we are connecting them. Further, we can describe articles

¹<http://www.citeulike.org>

²<http://www.mendeley.com>

in terms of what kinds of users like them. For example, we might detect that a machine learning article is of strong interest to computer vision researchers. If enough researchers use such services, these variables might also give an alternative measure of the impact of an article within a field.

With these criteria in mind, in this chapter, we develop a machine learning algorithm for recommending scientific articles to users in an online scientific community. Our algorithm uses two types of data—the other users’ libraries and the content of the articles—to form its recommendations. For each user, our algorithm can find both older papers that are important to other similar users and newly written papers whose content reflects the user’s specific interests. Finally, our algorithm gives interpretable representations of users and articles.

Our approach combines ideas from collaborative filtering based on latent factor models [91, 92, 65, 1, 115] and content analysis based on probabilistic topic modeling [26, 35, 99, 2]. Like latent factor models, our algorithm uses information from other users’ libraries. For a particular user, it can recommend articles from other users who liked similar articles. Latent factor models work well for recommending known articles, but cannot generalize to previously unseen articles.

To generalize to unseen articles, our algorithm uses topic modeling. Topic modeling provides a representation of the articles in terms of latent themes discovered from the collection. When used in our recommender system, this component can recommend articles that have similar content to other articles that a user likes. The topic representation of articles allows the algorithm to make meaningful recommendations about articles before anyone has rated them.

We combine these approaches in a probabilistic model, where making a recommendation for a particular user is akin to computing a conditional expectation of hidden variables. We will show how the algorithm for computing these expectations naturally balances the influence of the content of the articles and the libraries of the other users. An article that has

not been seen by many will be recommended based more on its content; an article that has been widely seen will be recommended based more on the other users.

We studied our algorithm with data from CiteULike: 5,551 users, 16,980 articles, and 204,986 bibliography entries. We will demonstrate that combining content-based and collaborative-based methods works well for recommending scientific articles. Our method provides better performance than matrix factorization methods alone, indicating that content can improve recommendation systems. Further, while traditional collaborative filtering cannot suggest articles before anyone has rated them, our method can use the content of new articles to make predictions about who will like them.

6.1 Background

We first give some background. We describe two types of recommendation problems we address; we describe the classical matrix factorization solution to recommendation; and we review latent Dirichlet allocation (LDA) for topic modeling of text corpora.

6.1.1 Recommendation tasks

The two elements in a recommender system are users and items. In our problem, items are scientific articles and users are researchers. We will assume I users and J items. The rating variable $r_{ij} \in \{0, 1\}$ denotes whether user i includes article j in her library [58]. If it is in the library, this means that user i is interested in article j . (This differs from some other systems where users explicitly rate items on a scale.) Note that $r_{ij} = 0$ can be interpreted into two ways. One way is that user i is not interested in article j ; the other is that user i does not know about article j .

For each user, our task is to recommend articles that are not in her library but are potentially interesting. There are two types of recommendation: *in-matrix prediction* and *out-of-matrix prediction*. Figure 6.1 illustrates the idea.

In-matrix prediction. Figure 6.1 (a) illustrates in-matrix prediction. This refers to the problem of making recommendations about those articles that have been rated by at least

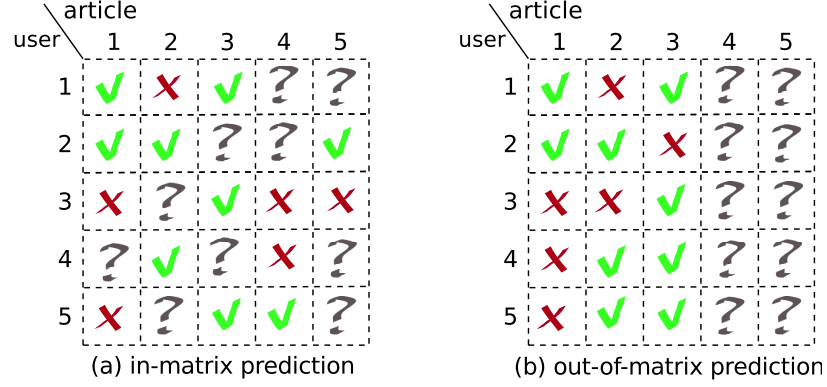


Figure 6.1: Illustration of the two tasks for scientific article recommendation systems, where \checkmark indicates “like”, \times “dislike” and ? “unknown”.

one user in the system. This is the task that traditional collaborative filtering can address.

Out-of-matrix prediction. Figure 6.1 (b) illustrates out-of-matrix prediction, where articles 4 and 5 have never been rated. (This is sometimes called “cold start recommendation.”) Traditional collaborative filtering algorithms cannot make predictions about these articles because those algorithms only use information about other users’ ratings. This task is important for online scientific archives, however, because users want to see new articles in their fields. A recommender system that cannot handle out-of-matrix prediction cannot recommend newly published papers to its users.

6.1.2 Recommendation by matrix factorization

The traditional approach to recommendation is collaborative filtering (CF), where items are recommended to a user based on other users with similar patterns of selected items. (Note that collaborative filtering does not use the content of the items.) Most successful recommendation methods are *latent factor models* [91, 92, 65, 1, 115], which provide better recommendation results than the *neighborhood methods* [52, 65]. In this chapter, we focus on latent factor models.

Among latent factor methods, matrix factorization performs well [65]. In matrix factorization, we represent users and items in a shared latent low-dimensional space of dimension K —user i is represented by a latent vector $u_i \in \mathbb{R}^K$ and item j by a latent vector $v_j \in \mathbb{R}^K$.

We form the prediction of whether user i will like item j with the inner product between their latent representations,

$$\hat{r}_{ij} = u_i^T v_j. \quad (6.1)$$

Biases for different users and items can also be incorporated [65].

To use matrix factorization, we must compute the latent representations of the users and items given an observed matrix of ratings. The common approach is to minimize the regularized squared error loss with respect to $U = (u_i)_{i=1}^I$ and $V = (v_j)_{j=1}^J$,

$$\min_{U,V} \sum_{i,j} (r_{ij} - u_i^T v_j)^2 + \lambda_u \|u_i\|^2 + \lambda_v \|v_j\|^2, \quad (6.2)$$

where λ_u and λ_v are regularization parameters.

This matrix factorization for collaborative filtering can be generalized as a probabilistic model [92]. In probabilistic matrix factorization (PMF), we assume the following generative process,

1. For each user i , draw user latent vector $u_i \sim \mathcal{N}(0, \lambda_u^{-1} I_K)$.
2. For each item j , draw item latent vector $v_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K)$.
3. For each user-item pair (i, j) , draw the response

$$r_{ij} \sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1}), \quad (6.3)$$

where c_{ij} is the precision parameter for r_{ij} .

(Note that I_K is a K -dimensional identity matrix.) This is the interpretation of matrix factorization that we will build on.

When $c_{ij} = 1$, for $\forall i, j$, the maximum a posteriori estimation (MAP) of PMF corresponds to the solution in Eq. 6.2. Here, the precision parameter c_{ij} serves as a confidence parameter for rating r_{ij} . If c_{ij} is large, we trust r_{ij} more. As we mentioned above, $r_{ij} = 0$ can be

interpreted into two ways—the user i is either not interested in item j or is unaware of it. This is thus a “one-class collaborative filtering problem,” similar to the TV program and news article recommendation problems studied in [58] and [84]. In that work, the authors introduce different confidence parameters c_{ij} for different ratings r_{ij} . We will use the same strategy to set c_{ij} a higher value when $r_{ij} = 1$ than when $r_{ij} = 0$,

$$c_{ij} = \begin{cases} a, & \text{if } r_{ij} = 1, \\ b, & \text{if } r_{ij} = 0, \end{cases} \quad (6.4)$$

where a and b are tuning parameters satisfying $a > b > 0$.

We fit a CF model by finding a locally optimal solution of the user variables U and item variables V , usually with an iterative algorithm [58]. We then use Eq. 6.1 to predict the ratings of the articles outside of each user’s library.

There are two main disadvantages to matrix factorization for recommendation. First, the learnt latent space is not easy to interpret; second, as mentioned, matrix factorization only uses information from other users—it cannot generalize to completely unrated items.

6.1.3 Probabilistic topic models

Topic modeling algorithms [24] are used to discover a set of “topics” from a large collection of documents, where a topic is a distribution over terms that is biased around those associated under a single theme. Topic models provide an interpretable low-dimensional representation of the documents [35]. They have been used for tasks like corpus exploration, document classification, and information retrieval. Here we will exploit the discovered topic structure for recommendation.

The simplest topic model is latent Dirichlet allocation (LDA) [26]. Assume there are K topics $\beta = \beta_{1:K}$, each of which is a distribution over a fixed vocabulary. The generative process of LDA is as follows. For each article w_j in the corpus,

1. Draw topic proportions $\theta_j \sim \text{Dirichlet}(\alpha)$.

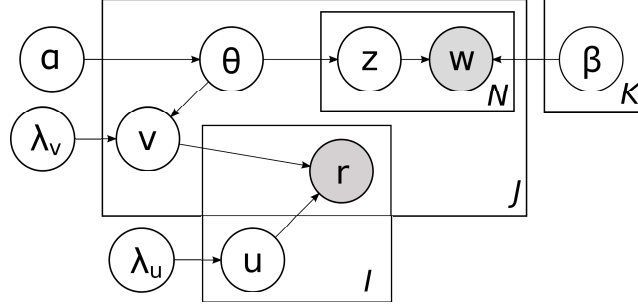


Figure 6.2: The graphical model for the CTR model.

2. For each word n ,

- (a) Draw topic assignment $z_{jn} \sim \text{Mult}(\theta_j)$.
- (b) Draw word $w_{jn} \sim \text{Mult}(\beta_{z_{jn}})$.

This process reveals how the words of each document are assumed to come from a mixture of topics: the topic proportions are document-specific, but the set of topics is shared by the corpus.

Given a corpus of documents, we can use variational EM to learn the topics and decompose the documents according to them [26]. Further, given a new document, we can use variational inference to situate its content in terms of the topics. Our goal is to use topic modeling to give a content-based representation of items in a recommender system.

6.2 Collaborative topic regression

In this section, we describe the collaborative topic regression (CTR) model. CTR combines traditional collaborative filtering with topic modeling.

A first approach to combining collaborative filtering and topic modeling is to fit a model that uses the latent topic space to explain both the observed ratings and the observed words. For example, we can use the topic proportion θ_j in place of the latent item latent vector v_j in Eq. 6.3,

$$r_{ij} \sim \mathcal{N}(u_i^T \theta_j, c_{ij}^{-1}). \quad (6.5)$$

(We note that [95] proposed a similar approach to Eq. 6.5, but based on correlated topic models [23]. It showed modest improvement over matrix factorization on several movie recommendation datasets.)

This model suffers from the limitation that it cannot distinguish topics for explaining recommendations from topics important for explaining content. Consider two articles A and B that are both about machine learning applied to social networks. They are similar and, therefore, have similar topic proportions θ_A and θ_B . Now further suppose that these articles are interesting to different kinds of users: Article A might give an interesting machine learning algorithm that is applied to social network applications; article B uses standard machine learning techniques, but gives an important piece of data analysis on social network data.

Users that work in machine learning will prefer article A and rarely consider article B; users that work in social networks will prefer the opposite. However, using the topic proportions as in Eq. 6.5 will be likely to make similar recommendations for both articles to both types of users. Collaborative topic regression can detect this difference—that one type of user likes the first article and another type likes the second.

As above, collaborative topic regression (CTR) represents users with topic interests and assumes that documents are generated by a topic model. CTR additionally includes a latent variable ϵ_j that offsets the topic proportions θ_j when modeling the user ratings. As more users rate articles, we have a better idea of what this offset is. This offset variable can explain, for example, that article A is more interesting to machine learning researchers than it is to social network analysis researchers. How much of the prediction relies on content and how much it relies on other users depends on how many users have rated the article.

Figure 6.2 shows the graphical model. Again, assume there are K topics $\beta = \beta_{1:K}$. The generative process of CTR is as follows,

1. For each user i , draw user latent vector $u_i \sim \mathcal{N}(0, \lambda_u^{-1} I_K)$.
2. For each item j ,

- (a) Draw topic proportions $\theta_j \sim \text{Dirichlet}(\alpha)$.
 - (b) Draw item latent offset $\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K)$ and set the item latent vector as $v_j = \epsilon_j + \theta_j$.
 - (c) For each word w_{jn} ,
 - i. Draw topic assignment $z_{jn} \sim \text{Mult}(\theta)$.
 - ii. Draw word $w_{jn} \sim \text{Mult}(\beta_{z_{jn}})$.
3. For each user-item pair (i, j) , draw the rating

$$r_{ij} \sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1}). \quad (6.6)$$

The key property in CTR lies in how the item latent vector v_j is generated. Note that $v_j = \epsilon_j + \theta_j$, where $\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K)$, is equivalent to $v_j \sim \mathcal{N}(\theta_j, \lambda_v^{-1} I_K)$, where we assume the item latent vector v_j is close to topic proportions θ_j , but could diverge from it if it has to. Note that the expectation of r_{ij} is a linear function of θ_j ,

$$\mathbb{E}[r_{ij} | u_i, \theta_j, \epsilon_j] = u_i^T (\theta_j + \epsilon_j).$$

This is why we call the model collaborative topic regression.

Learning the parameters. Given topic parameter β , computing the full posterior of u_i , v_j and θ_j is intractable. We develop an EM-style algorithm to learn the maximum a posteriori (MAP) estimates.

Maximization of the posterior is equivalent to maximizing the complete log likelihood of U , V , $\theta_{1:J}$, and R given λ_u , λ_v and β ,

$$\begin{aligned} \mathcal{L} = & -\frac{\lambda_u}{2} \sum_i u_i^T u_i - \frac{\lambda_v}{2} \sum_j (v_j - \theta_j)^T (v_j - \theta_j) \\ & + \sum_j \sum_n \log \left(\sum_k \theta_{jk} \beta_{k, w_{jn}} \right) - \sum_{i,j} \frac{c_{ij}}{2} (r_{ij} - u_i^T v_j)^2. \end{aligned} \quad (6.7)$$

We have omitted a constant and set $\alpha = 1$. We optimize this function by coordinate ascent,

iteratively optimizing the collaborative filtering variables $\{u_i, v_j\}$ and topic proportions θ_j .

For u_i and v_j , maximization follows in a similar fashion as for basic matrix factorization [58]. Given the current estimate of θ_j , taking the gradient of \mathcal{L} with respect to u_i and v_j and setting it to zero leads to (recall the matrix definition $U = (u_i)_{i=1}^I$ and $V = (v_j)_{j=1}^J$)

$$u_i \leftarrow (VC_iV^T + \lambda_u I_K)^{-1}VC_iR_i \quad (6.8)$$

$$v_j \leftarrow (UC_jU^T + \lambda_v I_K)^{-1}(UC_jR_j + \lambda_v \theta_j). \quad (6.9)$$

where C_i is a diagonal matrix with $c_{ij}, j = 1 \dots, J$ as its diagonal elements and $R_i = (r_{ij})_{j=1}^J$ for user i . For item j , C_j and R_j are similarly defined. Eq. 6.9 shows how topic proportions θ_j affects item latent vector v_j , where λ_v balances this effect. Finally, we note that the complexity is linear in the number of articles in the users' libraries. This follows from the special structure of c_{ij} defined in Eq. 6.4. (See [58] for details.)

Given U and V , we now describe how to learn the topic proportions θ_j .³ We first define $q(z_{jn} = k) = \phi_{jnk}$. Then we separate the items that contain θ_j and apply Jensen's inequality,

$$\begin{aligned} \mathcal{L}(\theta_j) &\geq -\frac{\lambda_v}{2}(v_j - \theta_j)^T(v_j - \theta_j) + \sum_n \sum_k \phi_{jnk} (\log \theta_{jk} \beta_{k,w_{jn}} - \log \phi_{jnk}) \\ &= \mathcal{L}(\theta_j, \phi_j). \end{aligned} \quad (6.10)$$

Let $\phi_j = (\phi_{jnk})_{n=1, k=1}^{N \times K}$. The optimal ϕ_{jnk} satisfies

$$\phi_{jnk} \propto \theta_{jk} \beta_{k,w_{jn}}. \quad (6.11)$$

The $\mathcal{L}(\theta_j, \phi_j)$ gives the *tight* lower bound of $\mathcal{L}(\theta_j)$. We cannot optimize θ_j analytically, so we use projection gradient [14]. We use coordinate ascent to optimize the remaining parameters, U , V , $\theta_{1:J}$ and $\phi_{1:J}$.

³On our data, we found that simply fixing θ_j as the estimate from vanilla LDA gives comparable performance and saves computation.

After we estimate U , V and ϕ , we can optimize β ,

$$\beta_{kw} \propto \sum_j \sum_n \phi_{jnk} 1[w_{jn} = w]. \quad (6.12)$$

Note this is the same M-step update for topics as in LDA [26].

Prediction. After all the (locally) optimal parameters U^* , V^* , $\theta_{1:J}^*$ and β^* are learned, the CTR model can be used for both in-matrix and out-of-matrix prediction. Let D be the observed data, in general each prediction is estimated as

$$\mathbb{E}[r_{ij}|D] \approx \mathbb{E}[u_i | D]^T (\mathbb{E}[\theta_j | D] + \mathbb{E}[\epsilon_j | D]). \quad (6.13)$$

For in-matrix prediction, we use the point estimate of u_i , θ_j and ϵ_j to approximate their expectations,

$$r_{ij}^* \approx (u_i^*)^T (\theta_j^* + \epsilon_j^*) = (u_i^*)^T v_j^*, \quad (6.14)$$

where recall that $v_j = \theta_j + \epsilon_j$.

In out-of-matrix prediction the article is new, and no other ratings are available. Thus, $\mathbb{E}[\epsilon_j] = 0$ and we predict with

$$r_{ij}^* \approx (u_i^*)^T \theta_j^*. \quad (6.15)$$

To obtain the topic proportions θ_j^* for a new article, we optimize Eq. 6.10. The first term is dropped because $v_j = \theta_j$.

Related work. Several other work uses content for recommendation [79, 76, 1, 2]. Among these, the closest work to ours is fLDA by [2]. FLDA generalizes the supervised topic model (sLDA) [25], using the empirical topic proportions $\bar{z}_j = (1/N) \sum_{n=1}^N z_{jn}$ as well as several other covariates to form predictions. In our settings, where we do not have additional

covariates, their approach is roughly akin to setting $v_j = \theta_j$. We show in Section 6.3 that a similar setting does not perform as well as the CTR model because it largely ignores the other users ratings.

Other recent work considers the related problem of using topic modeling to predict legislative votes [110, 49]. Neither of these methods introduces offset terms to account for votes (i.e., ratings). Legislative votes might be an interesting application for the CTR model.

Most recently, [41] presented an influence-based approach to find relevant articles given a set of query articles. This is similar to the recommendation scenario we consider in this chapter, although no specific users were used in [41].

6.3 Empirical study

We demonstrate our model by analyzing a real-world community of researchers and their citation files.⁴

Dataset. Our data are users and their libraries of articles obtained from CiteULike.⁵ At CiteULike, registered users create personal reference libraries; each article usually has a title and abstract. (The other information about the articles, such as the authors, publications and keywords, is not used in this chapter.)

We merged duplicated articles, removed empty articles, and removed users with fewer than 10 articles to obtain a data set of 5,551 users and 16,980 articles with 204,986 observed user-item pairs. (This matrix has a sparsity of 99.8%; it is highly sparse.) On average, each user has 37 articles in the library, ranging from 10 to 403. 93% of the users have fewer than 100 articles.

For each article, we concatenate its title and abstract. We remove stop words and use tf-idf to choose the top 8,000 distinct words as the vocabulary [24]. This yielded a corpus of 1.6M words. These articles were added to CiteULike between 2004 and 2010. On average, each article appears in 12 users' libraries, ranging from 1 to 321. 97% of the articles appear

⁴A demo of the results can be found at <http://www.cs.princeton.edu/~chongw/citeulike/>

⁵<http://www.citeulike.org/faq/data.adp>

in fewer than 40 libraries.

Evaluation. In our experiments, we will analyze a set of articles and user libraries. We will evaluate recommendation algorithms on sets of held-out articles and ratings. We will (hypothetically) present each user with M articles sorted by their predicted rating and evaluate based on which of these articles were actually in each user’s library.

Two possible metrics are precision and recall. However, as we discussed earlier, zero ratings are uncertain. They may indicate that a user does not like an article or does not know about it. This makes it difficult to accurately compute precision. Rather, since ratings of $r_{ij} = 1$ are known to be true positives, we focus on recall. Recall only considers the positively rated articles within the top M —a high recall with lower M will be a better system. For each user, the definition of $recall@M$ is

$$recall@M = \frac{\text{number of articles the user likes in top } M}{\text{total number of article the user likes}}.$$

The recall for the entire system can be summarized using the average recall from all users.

The recall above we defined is user-oriented. We also consider article-oriented recall for testing the system’s predictive performance on a particular article. For article j , we consider the population of users that like the article and the proportion of those for whom that article appears in their top M recommended articles. This evaluates the predictive power of the system on a chosen set of articles.

As we discussed in section 6.1, we consider two recommendation tasks users, *in-matrix prediction* and *out-of-matrix prediction*.

In-matrix prediction. In-matrix prediction considers the case where each user has a set of articles that she has not seen, but that at least one other user has seen. We ask the question, how good is each system at rating that set of articles for each user?

As discussed in section 6.1.1, this task is similar to traditional collaborative filtering. We split the data into a training set and test set, ensuring that all articles in the

test set have appeared in the training set. Content information is not required to perform recommendations—though we will see that it helps—and thus matrix factorization can be used.

We use 5-fold cross-validation. For every article that appears at least 5 times in the users’ libraries, we evenly split their user-item pairs (both 1’s and 0’s) into 5 folds. We iteratively consider each fold to be a test set and the others to be the training set. For those articles that appear fewer than 5 times, we always put them into the training set. This guarantees that all articles in the test set must appear in the training set. (9% of the articles are always in the training set, since they appear fewer than 5 times.)

For each fold, we fit a model to the training set and test on the within-fold articles for each user. (Note: each user has a different set of within-fold articles.) We form predictive ratings for the test set, and generate a list of the top M recommended articles.

Out-of-matrix prediction. Out-of-matrix prediction considers the case where a new set of articles is published and no one has seen them. Again we ask, how good is each system at rating that set of articles for each user?

We again use 5-fold cross validation. First, we evenly group all articles into 5 folds. For each fold, we fit the model to the submatrix formed by the out-of-fold articles and then test the recommendations for each user on the within-fold articles. Note that in this case, each user has the same set of within-fold articles and we are guaranteed that none of these articles is in the training set for any user. Again, we form predictive ratings for the test set, and generate a list of the top M recommended articles.

These two experimental set-ups—in-matrix and out-of-matrix predictions—are designed to be comparable—the top M articles are computed from the same size of candidate populations.

Experimental settings. For matrix factorization for collaborative filtering (CF), we used grid search to find that $K = 200$, $\lambda_u = \lambda_v = 0.01$, $a = 1$, $b = 0.01$ gives good performance on held out recommendations. We use CF to denote this method.

For collaborative topic regression (CTR), we set the parameters similarly as for CF, $K = 200$, $\lambda_u = 0.01$, $a = 1$ and $b = 0.01$. In addition, the precision parameter λ_v balances how the article’s latent vector v_j diverges from the topic proportions θ_j . We vary $\lambda_v \in \{10, 100, 1000, 10000\}$, where a larger λ_v increases the penalty of v_j diverging from θ_j .

We also compare to the model that only uses LDA-like features, as we discussed in the beginning of section 6.2. This is equivalent to fixing the per-item latent vector $v_j = \theta_j$ in the CTR model. This is a nearly content-only model—while the per-user vectors are fit to the ratings data, the document vectors θ_j are only based on the words of the document.⁶ We use LDA to denote this method. (Note that we use the resulting topics and proportions of LDA to initialize the CTR model.)

The baseline is the random model, where a user see M random recommended articles. We note that the expected recall for the random method from a pool of M_{tot} articles is irrelevant to library size. It is always M/M_{tot} .

Comparisons. Figure 6.3 shows the overall performance for in-matrix and out-of-matrix prediction, when we vary the number of returned articles $M = 20, 40, \dots, 200$. For CTR, we pick $\lambda_v = 100$; Figure 6.4 shows the performs when we change λ_v for the CTR model compared with CF and LDA when we fix $M = 100$.

Figure 6.3 and 6.4 shows that matrix factorization works well for in-matrix prediction, but adding content with CTR improves performance. The improvement is greater when the number of returned documents M is larger. The reason is as follows. Popular articles are more likely to be recommended by both methods. However, when M becomes large, few user ratings are available to ensure that CF gives good recommendations; the contribution of the content becomes more important.

Compared to both CF and CTR, LDA suffers for in-matrix prediction. It does not

⁶We also implemented a pure text-based baseline method. We represent a user’s interest using all the documents in the library using the tf-idf representation of the articles. For recommendations, we find the most similar documents to those in users’ library. We found this baseline performed slightly better for out-of-matrix prediction but still much worse for in-matrix prediction.

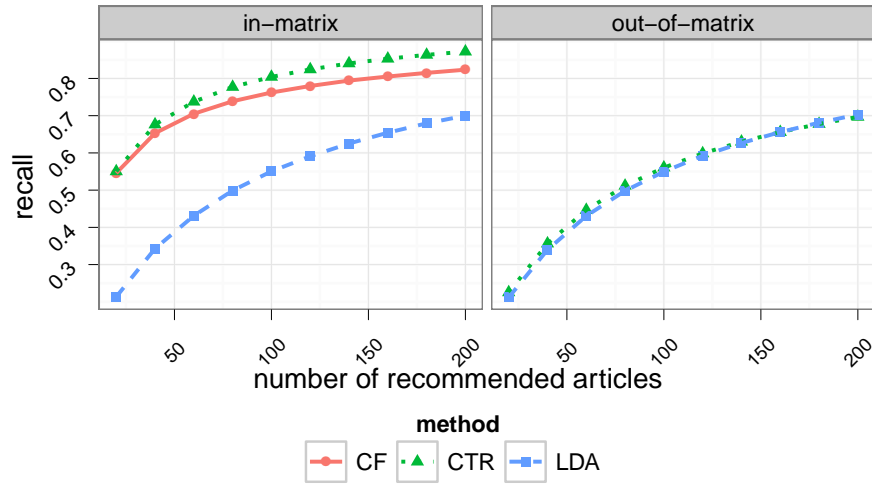


Figure 6.3: Recall comparison on in-matrix and out-of-matrix prediction tasks by varying the number of recommended articles. For CTR, we set $\lambda_v = 100$. Error bars are too small to show. The maximum expected recall for random recommendation is about 6%. CF can not do out-of-matrix prediction. CTR performs best.

account enough for the users' information in forming its predicted ratings. The gap between CF and LDA is interesting—other users provide a better assessment of preferences than content alone.

Out-of-matrix prediction is a harder problem, as shown by the relatively lower recall. In this task, CTR performs slightly better than LDA. Matrix factorization cannot perform out-of-matrix prediction. (Note also that LDA performs almost the same on both in-matrix and out-of-matrix predictions. This is expected because, in both settings, it makes its recommendations almost entirely based on content.) Overall, CTR is the best model.

In Figure 6.4 we study the effect of the precision parameter λ_v . When λ_v is small in CTR, the per-item latent vector v_j can diverge significantly from the topic proportions θ_j . Here, CTR behaves more like matrix factorization where no content is considered. When λ_v increases, CTR is penalized for v_j diverging from the topic proportions; this brings the content into the recommendations. When λ_v is too large, v_j is nearly the same as θ_j and, consequently, CTR behaves more like LDA.

We next study the relationship, across models, between recommendation performance

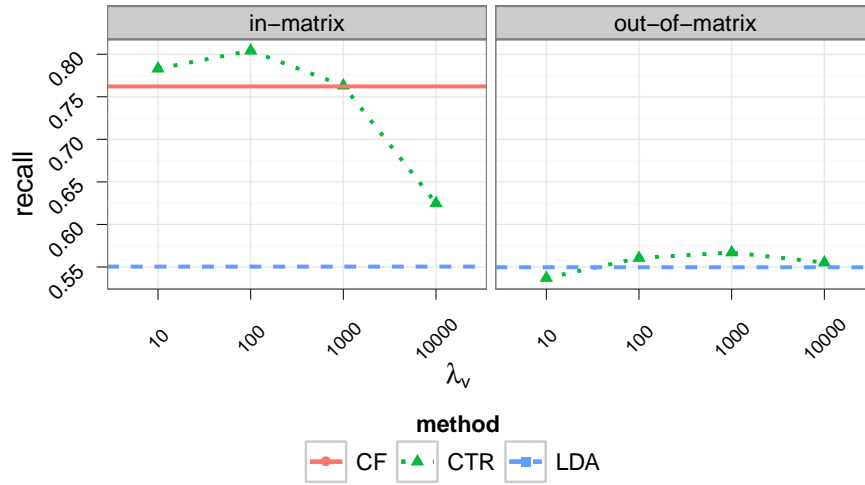


Figure 6.4: Recall comparison on in-matrix and out-of-matrix prediction tasks by fixing the number of recommended articles at $M = 100$. Error bars are too small to show. This shows how the precision parameter λ_v affects the performance of CTR. The expected recall of random recommendation is about 3%. CF can not do out-of-matrix prediction.

and properties of the users and articles. For this study we set the number of recommended articles $M = 100$ and the precision $\lambda_v = 100$. Figure 6.5 shows how the performance varies as a function of the number of articles in a user's library; Figure 6.6 shows how the performance varies as a function of the number of users that like an article.

As we see from Figure 6.5, for both in-matrix and out-of-matrix prediction, users with more articles tend to have less variance in their predictions. Users with few articles tend to have a diversity in the predictions, whose recall values vary around the extreme values of 0 and 1. In addition, we see that recall for users with more articles have a decreasing trend. This is reasonable because when a user has more articles then there will be more infrequent ones. As we see next, these articles are harder to predict.

From Figure 6.6, on in-matrix prediction for CF, CTR and LDA articles with high frequencies tend to have high recalls for in-matrix prediction and their predictions have less variance. This is because these articles have more collaborative information than infrequent ones, and, furthermore, CF and CTR make use of this information. For LDA, this trend is much smaller. In out-of-matrix predictions, since predictions are made on new articles,

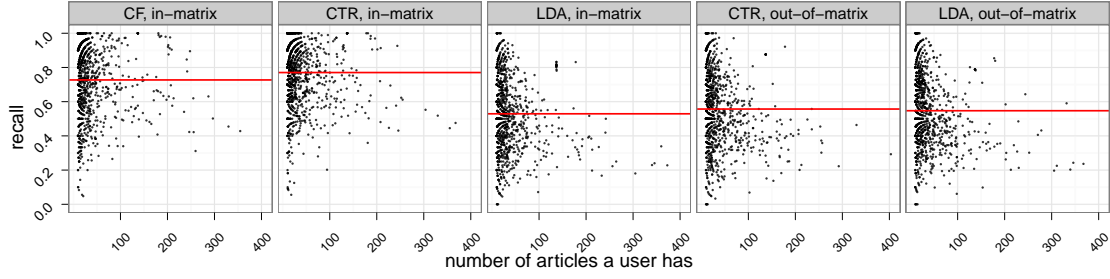


Figure 6.5: These scatter plots show how the number of articles a user has affects his or her recall. Red lines indicate the average. In these plots, the number of recommended articles is 100. CF can not do out-of-matrix prediction. This shows that CTR performs the best over user-oriented recall.

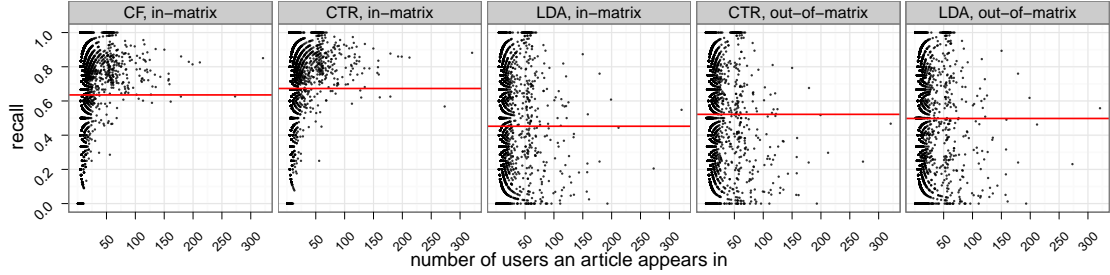


Figure 6.6: These scatter plots show how the number of users that like an article affects its recall. Red lines indicate the average. In these plots, the number of recommended articles is 100. CF can not do out-of-matrix prediction. This shows that CTR performs best over article-oriented recall as well.

these frequencies do not have an effect on training the model.

We now turn to an exploratory analysis of our results on the CTR model. (In the following, the precision $\lambda_v = 100$.)

Examining User Profiles. One advantage of the CTR model is that it can explain the user latent space using the topics learned from the data. For one user, we can find the top matched topics by ranking the entries of her latent vector u_i . Table 6.1 shows two example users and their top 3 matched topics along with their top 10 preferred articles as predicted by the CTR model.

The learned topics serve as a summary of what users might be interested in. For user I, we see that he or she might be a researcher working on machine learning and its applications

	user I	in user's lib?
top 3 topics	1. image, measure, measures, images, motion, matching, transformation 2. learning, machine, training, vector, learn, machines, kernel, learned 3. sets, objects, defined, categories, representations, universal, category	
top 10 articles	1. Information theory inference learning algorithms 2. Machine learning in automated text categorization 3. Artificial intelligence a modern approach 4. Data mining: practical machine learning tools and techniques 5. Statistical learning theory 6. Modern information retrieval 7. Pattern recognition and machine learning, information science and statistics 8. Recognition by components: a theory of human image understanding 9. Data clustering a review 10. Indexing by latent semantic analysis	✓ ✓ × × × ✓ ✓ × ✓ ✓
	user II	in user's lib?
top 3 topics	1. users, user, interface, interfaces, needs, explicit, implicit, usability 2. based, world, real, characteristics, actual, exploring, exploration, quite 3. evaluation, collaborative, products, filtering, product, reviews, items	
top 10 articles	1. Combining collaborative filtering with personal agents for better ... 2. An adaptive system for the personalized access to news 3. Implicit interest indicators 4. Footprints history-rich tools for information foraging 5. Using social tagging to improve social navigation 6. User models for adaptive hypermedia and adaptive educational systems 7. Collaborative filtering recommender systems 8. Knowledge tree: a distributed architecture for adaptive e-learning 9. Evaluating collaborative filtering recommender systems 10. Personalizing search via automated analysis of interests and activities	× ✓ × ✓ ✓ ✓ ✓ ✓ ✓ ✓

Table 6.1: Two example users. We show their position in latent space via their highest weighted topics. We list the top 10 preferred articles as predicted by CTR. The last column shows whether each article is in the user's library.

to texts and images. Although the predicted top 10 articles don't contain a vision article, we see such articles when more articles are retrieved. For user II, he or she might be a researcher who is interested in user interfaces and collaborative filtering.

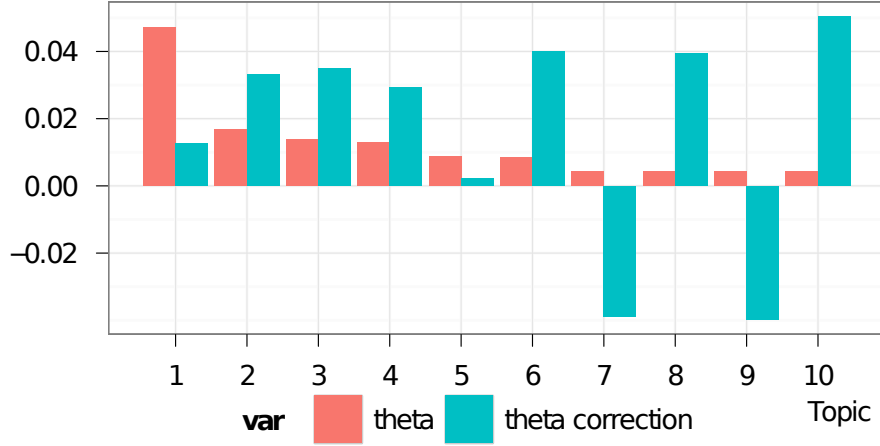
Examining the Latent Space of Articles. We can also examine the latent space of articles beyond their topic proportions. Here we inspect the articles with the largest overall offsets ϵ_j . Table 6.2 shows the top 10 articles with the largest offsets measured by the distance between v_j and θ_j , $\epsilon_j^T \epsilon_j = (v_j - \theta_j)^T (v_j - \theta_j)$. The last two columns show the average of predicted ratings over those users who actually have that article (avg-like) and those users who do not have that article (avg-dislike).

title	# dataset	# Google	avg-like	avg-dis.
1. The structure and function of complex networks	212	5,192	0.909	0.052
2. Emergence of scaling in random networks	193	8,521	0.899	0.058
3. R: a language and environment for statistical computing	113	837	0.827	0.047
4. A mathematical theory of communication	129	39,401	0.817	0.062
5. Maximum likelihood from incomplete data via the EM algorithm	157	22,874	0.864	0.055
6. A tutorial on hidden Markov models and selected applications ...	135	11,929	0.822	0.048
7. The structure of collaborative tagging systems	321	648	0.903	0.055
8. Why most published research findings are false	161	713	0.846	0.049
9. Phase-of-firing coding of natural visual stimuli in ...	8	64	1.057	-0.004
10. Defrosting the digital library bibliographic tools ...	179	37	0.840	0.042

Table 6.2: The top 10 articles with the largest discrepancy between their item latent vector v_j and topic proportions θ_j , measured by $(v_j - \theta_j)^T(v_j - \theta_j)$. Column 2 and 3 show how often they appear in the data, as well as the number of citations (retrieved from Google Scholar on Feb 17, 2011). Most of these articles are popular. The last two columns give the average values of “predicted” ratings over those users who have the article (avg-like) in the library and those who do not (avg-dis.).

These articles are popular in this data. Among the top 50 articles by this measure, 94% of them have at least 50 appearances. Articles with large offsets enjoy readership from different areas, and their item latent vectors have to diverge from the topic proportions to account for this. For example, Figure 6.7 illustrates the article that is the main citation for the expectation-maximization algorithm, “*Maximum likelihood from incomplete data via the EM algorithm*” [38]. Its top topic (found by $k = \arg \max_k \theta_{jk}$), is shown as topic 1. It is about “parameter estimation,” which is the main focus of this article. We can also examine the topics that are offset the most, $k = \arg \max_k |\epsilon_{jk}| = \arg \max_k |v_{jk} - \theta_{jk}|$. The maximum offset is for topic 10, a topic about “Bayesian statistics.” Topic 10 has a low value in θ_j —the EM paper is not a Bayesian paper—but readers of Bayesian statistics typically have this paper in their library.

Examining the offset can yield the opposite kind of article. For example, consider the article “*Phase-of-firing coding of natural visual stimuli in primary visual cortex*” in Figure 6.8. Its most probable topic is topic 1 (about “Computational Neuroscience”). Taking into account the offset, the most probable topic does not change and nor are new topics brought in. This indicates that the offset ϵ_j only adjusts v_j so that the objective function is



topic 1: estimate, estimates, likelihood, maximum, estimated, missing, distances
topic 10: parameters, Bayesian, inference, optimal, procedure, prior, assumptions

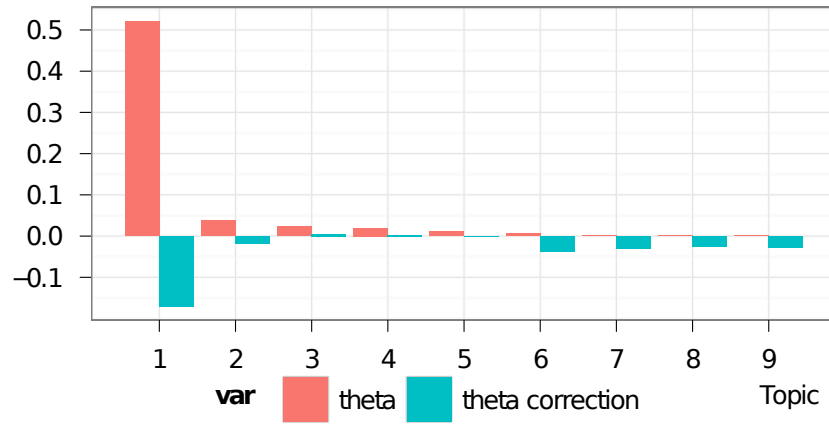
Figure 6.7: *Maximum likelihood from incomplete data via the EM algorithm.* Here, “theta” denotes θ_j and “theta correction” denotes the offset ϵ_j . The 10 topics are obtained by joining the top 5 topics ranked by θ_{jk} and another top 5 topics ranked by $|\epsilon_{jk}|$, $k = 1, \dots, K$. Under CTR, an article of wide interest is likely to exhibit more topics than its text exhibits. For example, this article brings in several other topics, including one on “Bayesian statistics” (topic 10). Note that the EM article is mainly about parameter estimation (topic 1), though is frequently referenced by Bayesian statisticians (and scholars in other fields as well).

well minimized. This article is not as interesting to users outside of Neuroscience.

6.4 Conclusions and future work

We proposed an algorithm for recommending scientific articles to users based on both content and other users’ ratings. Our study showed that this approach works well relative to traditional matrix factorization methods and makes good predictions on completely unrated articles.

Further, our algorithm provides interpretable user profiles. Such profiles could be useful in real-world recommender systems. For example, if a particular user recognizes her profile as representing different topics, she can choose to “hide” some topics when seeking recommendations about a subject.



topic 1: neurons, responses, neuronal, spike, cortical, stimuli, stimulus

Figure 6.8: *Phase-of-firing coding of natural visual stimuli in primary visual cortex*. This figure was created in the same way as Figure 6.7. It shows that a less popular article might also have a high offset value ϵ_j . In this case, it changes the actual magnitudes in θ_j , but does not bring in other topics.

Chapter 7

Conclusions

In this thesis, we have developed a suite of efficient inference algorithms and novel models under the hierarchical Bayesian modeling framework. These algorithms and models were applied to real-world data, including texts, images and user profiles.

The first algorithm we considered was an online variational inference algorithm for one type of Bayesian nonparametric models, hierarchical Dirichlet process. We showed our algorithm can handle millions of documents efficiently.

Second, we developed a novel algorithm to allow easier doing variational inference for nonconjugate models. This proposed algorithm would likely be integrated to a general toolbox for variational inference.

Finally, we described two novel models for real-world applications. The first was an approach for joint modeling image classification and annotation using topic modeling. The resulting model and software have become a standard baseline for many new research papers. The second application we considered was scientific article recommendation. We have developed a model that was able to present interpretable user and article profiles.

For future work, I would like to work on the following directions:

- *Online inference for changing data distributions.* We have considered the online inference problem by assuming data comes from a stationary distribution. However, many real-world data, such as online news articles and blog posts, is consistently

changing. New events are happening all the time. Extending the current online inference for Bayesian nonparametric models for data with changing distributions is an important next step for real-world applications. For example, we can design some hybrid algorithm of Gibbs sampling and variational inference; we can use Gibbs sampling to better explore the uncertain of the model space to detect new events and use the fastness of variational inference to refine the old events.

- *Novel ways of fitting hierarchical models.* Our motivation for this direction is as follows. In Bayesian inference, the usual claim is the uncertain around the posterior distribution is helpful for predictions. This is effective for not-so-large scale data. However when we apply Bayesian inference to very large-scale data, the posterior distribution obtained is usually very peaky, which effectively reduces Bayesian inference to point estimation. There is no problem if the model we build is the true generative model of the data, which unfortunately is usually not true. Our proposal is to a per-data predictive distribution as the criterion, which allows the posterior-like distribution not collapsed into a point estimation. Our preliminary results have shown this method is very competitive compared with traditional Bayesian inference.
- *Recommendation and exploration.* In Chapter 6, we have shown it would be very helpful to help users to engage into the exploration of results if the recommendation algorithm can provide better representations of the recommended results. By continuing this line of research, I would like to explore new recommendation algorithms that can lead to better representations. Specifically I want to focus on the following two directions, 1) a hierarchical representation of recommended items. I will work on an algorithm that can provide a personalized coarse-to-fine hierarchy. This will greatly help the users to navigate the recommendation results. 2) a sequential representation of recommended items. I will work on an algorithm that is able to predict the best sequence of recommendations. For example, what would be a good sequence of articles to read to understand one subject? In addition, these approaches also aim

to change the current evaluation of recommender systems mostly using quantitative metrics like RMSE by introducing some novel qualitative metrics, which could be equally important in designing recommender systems.

Appendix: Derivations for Variational Inference for Nonconjugate Models

A. Derivations for correlated topic models

Suppose there are K topic parameters $\beta_{1:K}$ (fixed for now), each of which is a distribution over V terms. Let π be the topic proportions for a document and n be the index of an observed word x_n . The CTM assumes the following generative process of a document,

$$\begin{aligned}\theta &\sim \mathcal{N}(\mu_0, \Sigma_0), \pi \propto \exp(\theta) \\ z_n | \pi &\sim \text{Mult}(\pi), x_n | z_n, \beta \sim \text{Mult}(\beta_{z_n}).\end{aligned}$$

In this model, the topic proportions π are drawn from a logistic normal distribution. Their correlation structure is captured in Σ_0 . The variable z_n indicates which topic the n th word is drawn from. We can now identify the quantities from Eq. 4.5 to Eq. 4.7 that we need to compute $f(\theta)$ in Eq. 4.12,

$$h(z) = 1, \quad a(z) = 0, \quad t(z) = \sum_n z_n, \quad \eta(\theta) = \theta - \log \{\sum_k \exp\{\theta_k\}\}, \quad a(\eta(\theta)) = 0.$$

With this notation, function $f(\theta)$ defined in Eq. 4.12 is,

$$f(\theta) = \eta(\theta)^\top \bar{t}_z - \frac{1}{2}(\theta - \mu_0)^\top \Sigma_0^{-1}(\theta - \mu_0),$$

where \bar{t}_z is the expected word counts of each topic under the variational distribution $q(z)$.

Using $\partial\pi_i/\partial\theta_j = \pi_i(1_{[i=j]} - \pi_j)$, we obtain the gradient and Hessian for function $f(\theta)$ in CTM,

$$\begin{aligned}\nabla f(\theta) &= \bar{t}_z - \pi \sum_{k=1}^K [\bar{t}_z]_k - \Sigma_0^{-1}(\theta - \mu_0), \\ \nabla^2 f(\theta)_{ij} &= (-\pi_i 1_{[i=j]} + \pi_i \pi_j) \sum_{k=1}^K [\bar{t}_z]_k - (\Sigma_0^{-1})_{ij}.\end{aligned}$$

where $1_{[i=j]} = 1$ if $i = j$ and 0 otherwise. Note the first $\nabla f(\theta)$ is enough for Laplace variational inference.

In Delta method variational inference, we also need to compute the derivative of

$$\text{Trace} \{ \nabla^2 f(\theta) \Sigma \} = \left(- \sum_{k=1}^K \pi_k \Sigma_{kk} + \pi^T \Sigma \pi \right) \sum_{k=1}^K [\bar{t}_z]_k - \text{Trace}(\Sigma_0^{-1} \Sigma).$$

We also assume Σ is diagonal for the delta method for simplicity [30]. (Note for Laplace method, we don't need this assumption.) This gives us

$$\frac{\partial \text{Trace} \{ \nabla^2 f(\theta) \Sigma \}}{\partial \theta_i} = \pi_i (1 - 2\pi_i) (\sum_k \pi_k \Sigma_{kk} - 1)$$

The algorithm in Figure 4.1 for CTM only applies to a single document.

B. Derivations for Bayesian logistic regression

In Bayesian logistic regression, let t_n is be a p -dimensional observed feature vector for the n th sample and z_n be its class (represented as an indicator vector of length two). Let θ be the real-valued parameter vector in \mathbb{R}^p ; there is a component for each feature. The usual Bayesian logistic regression model is the following:

$$\begin{aligned}p(\theta) &= \mathcal{N}(\mu_0, \Sigma_0), \\ p(z_n | \theta, t_n) &= \sigma(\theta^\top t_n)^{z_{n,1}} \sigma(-\theta^\top t_n)^{z_{n,2}}, \quad \forall n\end{aligned}$$

where $\sigma(y) \triangleq 1 / (1 + \exp(-y))$ is the logistic function.

In Bayesian logistic regression, we can fit the distribution of the observations $z_{1:N}$ into the exponential family with

$$h(z) = 1, \quad a(z) = 0, \quad \bar{t}_z = t(z) = [z_1, \dots, z_N],$$

$$\eta(\theta)^\top = [\log \sigma(\theta^\top t_n), \log \sigma(-\theta^\top t_n)]_{n=1}^N, \quad a(\eta(\theta)) = 0,$$

In this set up, \bar{t}_z represents the whole set of labels. With this notation, the $f(\theta)$ defined in Eq. 4.12

$$f(\theta) = \eta(\theta)^\top \bar{t}_z - \frac{1}{2}(\theta - \mu_0)^\top \Sigma_0^{-1}(\theta - \mu_0).$$

The gradient and Hessian for function $f(\theta)$ in Bayesian logistic regression are

$$\nabla f(\theta) = \sum_{n=1}^N t_n (z_{n,1} - \sigma(\theta^\top t_n)) - \Sigma_0^{-1}(\theta - \mu_0),$$

$$\nabla^2 f(\theta) = - \sum_{n=1}^N \sigma(\theta^\top t_n) \sigma(-\theta^\top t_n) t_n t_n^\top - \Sigma_0^{-1}.$$

Note the first $\nabla f(\theta)$ is enough for Laplace variational inference.

For delta variational inference, we also need the gradient for $\text{Trace} \{ \nabla^2 f(\theta) \Sigma \}$ is

$$\frac{\partial \text{Trace} \{ \nabla^2 f(\theta) \Sigma \}}{\partial \theta_i} = - \sum_{n=1}^N \sigma(\theta^\top t_n) \sigma(-\theta^\top t_n) (1 - 2\sigma(\theta^\top t_n)) t_n t_n^\top \Sigma t_n.$$

Note that the “diagonal” assumption for Σ is not needed.

Bibliography

- [1] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 19–28, New York, NY, USA, 2009. ACM.
- [2] D. Agarwal and B.-C. Chen. flda: matrix factorization through latent Dirichlet allocation. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 91–100, New York, NY, USA, 2010. ACM.
- [3] A. Ahmed and E. Xing. On tight approximate inference of the logistic normal topic admixture model. In *AISTATS*, 2007.
- [4] J. Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society, Series B*, 44(2):139–177, 1982.
- [5] S. Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.
- [6] C. Andrieu, N. de Freitas, A. Doucet, and M. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003.
- [7] C. Archambeau, S. Guo, and O. Zoeter. Sparse Bayesian multi-task learning. In *NIPS*, 2011.
- [8] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Mach. Learn.*, 73:243–272, December 2008.

- [9] A. Asuncion, M. Welling, P. Smyth, and Y. Teh. On smoothing and inference for topic models. In *UAI*, 2009.
- [10] H. Attias. A variational Bayesian framework for graphical models. In *Advances in Neural Information Processing Systems 12*, 2000.
- [11] K. Barnard, P. Duygulu, N. de Freitas, D. Forsyth, D. Blei, and M. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.
- [12] M. Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.
- [13] J. Bernardo and A. Smith. *Bayesian theory*. John Wiley & Sons Ltd., Chichester, 1994.
- [14] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.
- [15] P. Bickel and K. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics*, volume 1. Pearson Prentice Hall, Upper Saddle River, NJ, 2nd edition, 2007.
- [16] C. Bishop. *Pattern Recognition and Machine Learning*. Springer New York., 2006.
- [17] C. Bishop, D. Spiegelhalter, and J. Winn. VIBES: A variational inference engine for Bayesian networks. In *NIPS*, 2003.
- [18] D. Blackwell and J. MacQueen. Ferguson distributions via Pólya urn schemes. *The Annals of Statistics*, 1(2):353–355, 1973.
- [19] D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In *NIPS*, 2003.
- [20] D. Blei and M. Jordan. Modeling annotated data. In *Proceedings of the 26th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 127–134. ACM Press, 2003.

- [21] D. Blei and M. Jordan. Variational methods for the Dirichlet process. In *21st International Conference on Machine Learning*, 2004.
- [22] D. Blei and J. Lafferty. Dynamic topic models. In *ICML*, 2006.
- [23] D. Blei and J. Lafferty. A correlated topic model of Science. *Annals of Applied Statistics*, 1(1):17–35, 2007.
- [24] D. Blei and J. Lafferty. Topic models. In A. Srivastava and M. Sahami, editors, *Text Mining: Theory and Applications*. Taylor and Francis, 2009.
- [25] D. Blei and J. McAuliffe. Supervised topic models. In *Neural Information Processing Systems*, 2007.
- [26] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *JMLR*, 3:993–1022, January 2003.
- [27] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via pLSA. In *ECCV*, 2006.
- [28] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757 – 1771, 2004.
- [29] J. Boyd-Graber and D. Blei. Syntactic topic models. In *Neural Information Processing Systems*, 2009.
- [30] M. Braun and J. McAuliffe. Variational inference for large-scale models of discrete choice. *JASA*, 105(489), 2007.
- [31] L. Brown. *Fundamentals of Statistical Exponential Families*. Institute of Mathematical Statistics, Hayward, CA, 1986.

- [32] W. Buntine and A. Jakulin. Applying discrete PCA in data analysis. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 59–66. AUAI Press, 2004.
- [33] K. Canini, L. Shi, and T. Griffiths. Online inference of topics with latent Dirichlet allocation. In *Artificial Intelligence and Statistics*, 2009.
- [34] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. In *CVPR*, 2007.
- [35] J. Chang, J. Boyd-Graber, S. Gerrish, C. Wang, and D. Blei. Reading tea leaves: How humans interpret topic models. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *NIPS*, pages 288–296, 2009.
- [36] Y. Chen and J. Z. Wang. Image categorization by learning and reasoning with regions. *JMLR*, 5:913–939, 2004.
- [37] J. Clinton, S. Jackman, and D. Rivers. The statistical analysis of roll call data. *American Political Science Review*, 98(2):355–370, 2004.
- [38] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.
- [39] F. Doshi-Velez, K. Miller, J. Van Gael, and Y. Teh. Variational inference for the Indian buffet process. In *Proceedings of the Intl. Conf. on Artificial Intelligence and Statistics*, pages 137–144, 2009.
- [40] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*, 2002.
- [41] K. El-Arini and C. Guestrin. Beyond keyword search: discovering relevant scientific literature. In *Proceedings of the 17th ACM SIGKDD international conference on*

- Knowledge discovery and data mining*, KDD '11, pages 439–447, New York, NY, USA, 2011. ACM.
- [42] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *NIPS*, 2001.
 - [43] M. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.
 - [44] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. *IEEE Computer Vision and Pattern Recognition*, pages 524–531, 2005.
 - [45] T. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1:209–230, 1973.
 - [46] E. Fox, E. Sudderth, M. Jordan, and A. Willsky. An HDP-HMM for systems with state persistence. In *International Conference on Machine Learning*, 2008.
 - [47] A. Gelman and J. Hill. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge Univ. Press, 2007.
 - [48] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.
 - [49] S. M. Gerrish and D. M. Blei. Predicting legislative roll calls from text. In *Proceedings of the 28th Annual International Conference on Machine Learning*, ICML '11, 2011.
 - [50] Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixtures of factor analysers. In *NIPS*, 2000.
 - [51] T. Griffiths and Z. Ghahramani. Infinite latent feature models and the Indian buffet process. In *NIPS*, 2006.

- [52] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 230–237, New York, NY, USA, 1999. ACM.
- [53] M. Hoffman, D. Blei, and F. Bach. Online inference for latent Dirichlet allocation. In *NIPS*, 2010.
- [54] T. Hofmann. Probabilistic latent semantic indexing. *Research and Development in Information Retrieval*, pages 50–57, 1999.
- [55] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, 1999.
- [56] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Mach. Learn.*, 42(1-2):177–196, 2001.
- [57] A. Honkela and H. Valpola. Unsupervised variational bayesian learning of nonlinear models. In *NIPS*, 2004.
- [58] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.
- [59] T. Jaakkola and M. Jordan. A variational approach to Bayesian logistic regression models and their extensions. In *AISTATS*, 1996.
- [60] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance models. In *SIGIR*, 2003.
- [61] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.
- [62] T. Kadir and M. Brady. Saliency, scale and image description. *IJCV*, 45(2):83–105, 2001.

- [63] M. E. Khan, B. Marlin, G. Bouchard, and K. Murphy. Variational bounds for mixed-data factor analysis. In *NIPS*, 2010.
- [64] D. A. Knowles and T. Minka. Non-conjugate variational message passing for multinomial and binary regression. In *NIPS*, 2011.
- [65] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.
- [66] K. Kurihara, M. Welling, and Y. Teh. Collapsed variational Dirichlet process mixture models. In *IJCAI*, 2007.
- [67] K. Kurihara, M. Welling, and N. Vlassis. Accelerated variational Dirichlet process mixtures. In *NIPS*, 2007.
- [68] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.
- [69] L.-J. Li and L. Fei-Fei. What, where and who? Classifying event by scene and object recognition. In *ICCV*, 2007.
- [70] P. Liang, S. Petrov, D. Klein, and M. Jordan. The infinite PCFG using hierarchical Dirichlet processes. In *Empirical Methods in Natural Language Processing*, 2007.
- [71] J. Liu. The collapsed Gibbs sampler in Bayesian computations with application to a gene regulation problem. *Journal of the American Statistical Association*, 89(958–966), 1994.
- [72] D. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.
- [73] D. J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Comput.*, 4:448–472, May 1992.

- [74] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. London: Chapman and Hall, 1989.
- [75] G. McLachlan and D. Peel. *Finite mixture models*. Wiley-Interscience, 2000.
- [76] P. Melville, M. R., and R. Nagaraja. Content-boosted collaborative filtering for improved recommendations. In *American Association for Artificial Intelligence*, pages 187–192, 2002.
- [77] T. Minka. Estimating a Dirichlet distribution. Technical report, M.I.T., 2000.
- [78] T. Minka, J. Winn, J. Guiver, and D. Knowles. Infer.NET 2.4, 2010. Microsoft Research Cambridge. <http://research.microsoft.com/infernet>.
- [79] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204, New York, NY, USA, 2000. ACM.
- [80] R. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, 1993.
- [81] R. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.
- [82] J. Nocedal and S. Wright. *Numerical Optimization, Second Edition*. Springer, 2006.
- [83] A. Oliva and A. B. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.
- [84] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 502–511, Washington, DC, USA, 2008. IEEE Computer Society.

- [85] J. Pitman. Poisson-Dirichlet and GEM invariant distributions for split-and-merge transformations of an interval partition. *Combinatorics, Probability, and Computing*, 11:501–514, 2002.
- [86] P. Quelhas, F. Monay, J. Odobez, D. Gatica-Perez, T. Tuyelaars, and L. Van Gool. Modeling scenes with local descriptors and latent aspects. In *ICCV*, 2005.
- [87] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):pp. 400–407, 1951.
- [88] C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer-Verlag, New York, NY, 2004.
- [89] A. Rodriguez. On-line learning for the infinite hidden Markov model. Technical Report UCSC-SOE-10-27, Department of Applied Mathematics and Statistics, University of California at Santa Cruz, 2010.
- [90] B. C. Russell, A. B. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: A database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008.
- [91] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th International Conference on Machine learning*, pages 880–887. ACM, 2008.
- [92] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. *Advances in Neural Information Processing Systems*, 20:1257–1264, 2008.
- [93] M. Sato. Online model selection based on the variational Bayes. *Neural Computation*, 13(7):1649–1681, 2001.
- [94] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

- [95] H. Shan and A. Banerjee. Generalized probabilistic matrix factorizations for collaborative filtering. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, pages 1025–1030, Washington, DC, USA, 2010. IEEE Computer Society.
- [96] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. Technical report, CSAIL, Massachusetts Institute of Technology, 2005.
- [97] M. Szummer and R. W. Picard. Indoor-outdoor image classification. In *IEEE International Workshop on Content-based Access of Image and Video Databases*, 1998.
- [98] Y. Teh, D. Gorur, and Z. Ghahramani. Stick-breaking construction for the Indian buffet process. In *AISTATS*, 2007.
- [99] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2007.
- [100] Y. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for HDP. In *NIPS*, 2007.
- [101] L. Tierney, R. E. Kass, and J. B. Kadane. Fully exponential laplace approximations to expectations and variances of nonpositive functions. *JASA*, 84(407), 1989.
- [102] A. Vailaya, M. A. T. Figueiredo, A. K. Jain, and H. Zhang. Image classification for content-based indexing. *IEEE Trans. on Image Processing*, 10(1):117–130, 2001.
- [103] J. Vogel and B. Schiele. A semantic typicality measure for natural scene categorization. In *DAGM-Symposium*, 2004.
- [104] C. Wang and D. Blei. Variational inference for the nested Chinese restaurant process. In *NIPS*, 2009.

- [105] C. Wang, D. Blei, and D. Heckerman. Continuous time dynamic topic models. In *Uncertainty in Artificial Intelligence (UAI)*, 2008.
- [106] C. Wang, D. Blei, and F. Li. Simultaneous image classification and annotation. In *Computer Vision and Pattern Recognition*, 2009.
- [107] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '11*, 2011.
- [108] C. Wang and D. M. Blei. Variational inference for nonconjugate models. In *NIPS*, *submitted*, 2012.
- [109] C. Wang, J. Paisley, and D. M. Blei. Online variational inference for the hierarchical Dirichlet process. In *AISTATS*, 2011.
- [110] E. Wang, D. Liu, J. Silva, D. Dunson, and L. Carin. Joint analysis of time-evolving binary matrices and associated documents. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2370–2378. 2010.
- [111] Y. Wang and S. Gong. Conditional random field for natural scene categorization. In *BMVC*, 2007.
- [112] M. T. Wells. Generalized linear models: A Bayesian perspective. *JASA*, 96(453):339–355, 2001.
- [113] E. Xing, M. Jordan, and S. Russell. A generalized mean field algorithm for variational inference in exponential families. In *UAI*, 2003.
- [114] Y. Xue, D. Dunson, and L. Carin. The matrix stick-breaking process for flexible multi-task learning. In *ICML*, 2007.

- [115] K. Yu, J. Lafferty, S. Zhu, and Y. Gong. Large-scale collaborative prediction using a nonparametric random effects model. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1185–1192, New York, NY, USA, 2009. ACM.
- [116] Z.-H. Zhou and M.-L. Zhang. Multi-instance multi-label learning with application to scene classification. In *NIPS*, 2006.