

UNIVERSITY OF CALIFORNIA,
IRVINE

Networks of Mixture Blocks for Non Parametric Bayesian Models
with Applications

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY
in Information and Computer Science

by

Ian Porteous

Dissertation Committee:
Professor Max Welling, Chair
Professor Alex Ihler
Professor Hal Stern

2010

DEDICATION

To my wife Grete, whose love and support made this possible.

TABLE OF CONTENTS

	Page
LIST OF FIGURES	v
LIST OF TABLES	ix
ACKNOWLEDGMENTS	x
CURRICULUM VITAE	xi
ABSTRACT OF THE DISSERTATION	xii
1 Introduction	1
2 Overview	3
3 Related Work	17
4 Networks of Mixture Blocks	19
4.1 Bayesian Mixture of Gaussians	19
4.2 Mixture Blocks	21
4.3 Network of Mixture Blocks	25
5 Inference for Mixture Block Models	28
5.1 Gibbs Sampling	28
5.2 Network of Mixture Block Sampling	30
5.3 Encapsulation of Inference in Mixture Blocks	32
5.3.1 Joint Sampling	34
6 Collaborative Filtering through Matrix Factorization	35
6.1 Multi-LDA , Multi-HDP	36
6.1.1 Bi-LDA	38
6.1.2 Bi-LDA Gibbs Sampling	41
6.1.3 Description of Inference	41
6.1.4 Inference in Terms of Mixture Block Objects	43
6.1.5 Prediction	43
6.1.6 Multi-LDA.	45
6.1.7 Multi-LDA with Mixture Blocks	47

6.1.8	Bi-LDA Experiments	48
6.1.9	Bi-HDP	51
6.1.10	Bi-HDP Experiments	57
6.1.11	Multi-LDA/Multi-HDP Discussion	61
6.1.12	Sampling Augmented Variables	61
6.1.13	Sampling Hyper-parameters	62
6.2	Bayesian Matrix Factorization with Side Information and Dirichlet Process Mixtures	63
6.2.1	Related work	65
6.2.2	Bayesian probabilistic matrix factorization	66
6.2.3	Bayesian matrix factorization with side information	68
6.2.4	Inference	70
6.2.5	BMFSI with Dirichlet process mixture prior	71
6.2.6	Experiments	73
6.2.7	BMFSI Discussion	77
6.2.8	BMFSI Gibbs sampling algorithm	78
6.2.9	Gibbs sampling BMFSI with Dirichlet process mixture sampling algorithm	80
7	Flexible Nonparametric Priors	86
7.1	FLEXIBLE PRIORS OVER PARTITIONS	90
7.2	Learning Flexible Priors	91
7.3	ILLUSTRATIVE EXAMPLE	94
7.4	HIERARCHICAL FLEXIBLE PRIORS	96
7.5	AN EXPERIMENT WITH TEXT DATA	100
7.6	Flexible Priors Discussion	103
8	Conclusions	105
	Bibliography	107

LIST OF FIGURES

	Page
2.1 Graphical model for Dirichlet compound multinomial.	4
2.2 Dirichlet compound multinomial mixture component. The set of words $W = \{w_1, \dots, w_N\}$ is partitioned into K groups based on the document $d_i \in D$ that word $w_i \in W$ is from. Given the partitioning of W into K groups, the probability of all the words W is the product of the probability of each subset of words that belong to a specific document.	5
2.3 Graphical model for Latent Dirichlet Allocation (LDA).	7
2.4 Latent Dirichlet Allocation Model drawn as network of mixture blocks. $D = d_1, \dots, d_n$ is the document id for each word. k is the id of the mixture group in each DCM mixture block	7
2.5 Graphical model for pachinko allocation model (PAM).	8
2.6 Pachinko Allocation Model drawn as NMB. $D = d_1, \dots, d_n$ is the document id for each word. K is the set of possible topics in each component	9
2.7 Chinese restaurant representation of the Dirichlet process. Customers, z_i , sit at tables in a Chinese restaurant with an infinite number of tables. Each customer sits at a table proportional to the number of customers already sitting at the table.	10
2.8 Two graphical models for the Dirichlet process. Left: The stick breaking construction for the Dirichlet process mixture model, $\theta \sim GEM(\alpha)$ [34]. Right: Dirichlet process as a distribution over partitions. The variables Z are drawn from $CRP(\alpha)$, and define a partition of N items, S^*	12
2.9 Two Dirichlet process mixture models. Left: A DP mixture model. Right: N_D independent DP mixture models, one for each value of D . Since the value z_i only represents the equivalence class with respect to the d_i CRP, the mixture group for x is a function of both d_i, z_i	13

2.10	Chinese restaurant franchise representation of an HDP. Each box represents a restaurant (CRP) with an infinite number of tables. Customers come in and sit down at tables according to the CRP distribution. The vertically oriented restaurants each represent a bottom level partition, e.g. a topic distribution for a document where the tables represent the document-topic. The words are assigned to document topics via z variables. Each z variable is assigned to a table according to a CRP. The horizontal restaurant represents the franchise, and each table represents a topic. The variables for each table in each restaurant, k , are assigned a topic from the franchise according to a CRP distribution.	14
2.11	Two HDP mixture models. On the left there are two DP mixture blocks which are used to define the assignment of X variables to mixture components. On the right each set of variables Z_d represents a different partition, but since they are from an HDP they refer to a set of equivalence classes shared by all documents.	15
2.12	Block diagram of the functional components that go into one of the mixture blocks in a network of mixture blocks	15
4.1	Bayesian model Mixture of Gaussians. Square nodes represent parameters of the model. Red outline designates the sub-graphs that correspond to the mixture blocks for Z and X	20
4.2	MoG. \mathcal{MD} stands for a multinomial distribution whose parameters have a Dirichlet prior. $\mathcal{NN}T^{-1}$ indicates a Normal distribution with a Normal-scaled inverse gamma prior over its parameters.	22
6.1	Bayesian graphical model for Bi-LDA.	39
6.2	NMB for Bi-LDA. U is the number of users. M is the number of movies, N_Z is the number of movie groups times the number of user groups	40
6.3	RMSE of Netflix prediction versus cumulative RMSE of prediction with calculated variance less than X	49
6.4	Top: The RMSE of predictions binned by their predicted variance. The first bar is predictions with variance less than .25, the second bar is the RMSE of predictions with variance between .25 and .5, and so on by increments of .25. Bottom: The percentage of predictions that fall within each bin. . .	49
6.5	Bi-LDA model learned on the Netflix data. Each box corresponds to a movie group. Each box shows the titles of the five movies responsible for the most data points in each group and the percentage of data points in that group associated with the movie.	50
6.6	Prototype digit '5' as a combination of strokes (best viewed in color). The Bi-LDA image prototype (image group) on the left. Strokes (pixel groups) that were active for this prototype on the right.	52

6.7	a) Original training data b) Bi-LDA strokes c) Bi-LDA image-groups d) Bi-LDA reconstruction e) URP-LDA image-groups f) URP-LDA reconstruction. The white frame in image b) is a pixel group for the black pixels which surround most digits	53
6.8	NMB for Multi-HDP using direct assignment HDP blocks	55
6.9	NMB for Multi-HDP using Hierarchy of CRP.	55
6.10	Graphical model for Bi-HDP	57
6.11	Year groups on NIPS data. Two groups are shown. Top: years that contributed the most data points to the group shown. For each year the percentage of points accounted for in the year group is shown. Below, the three most popular topics for that year group are shown. For each topic the 10 most popular words are shown within a box.	59
6.12	Author groups on NIPS data. Four groups are shown. Top: authors that contributed the most data points to the group shown. Below, the four most popular topics for that author group are shown. For each topic, the 10 most popular words are shown within a box.	81
6.13	Item groups learned by the Bi-HDP model compared to ground truth item groups. Each table corresponds to a single learned item group. The 10 most popular item groups learned by Bi-HDP are shown. In each table, the top row shows the most popular items for that item group, in order of decreasing popularity. The bottom row shows the ground truth item group corresponding to the learned item group. The items in a ground truth group have no associated weight; therefore, they were ordered to facilitate comparison with the top row. Non-matching items are shown in boldface. As can be seen, in most cases a learned item group corresponds accurately to a true item group.	82
6.14	Performance of Bi-HDP and NMF on the market basket data. <i>Y</i> axis: RMSE (smaller values are better). <i>X</i> axis: the number of factors for NMF. Dashed curve: NMF. Solid curve: Bi-HDP. Since Bi-HDP determines the number of factors automatically, a horizontal curve is shown.	83
6.15	LEFT: Graphical model for Bayesian probabilistic matrix factorization (BPMF). RIGHT: Graphical model for Bayesian matrix factorization with side information (BMFSI).	83
6.16	BMFSI with Dirichlet process mixture	84
6.17	BMFSI with DP as NMB. The <i>U</i> and <i>V</i> mixture blocks are hybrid mixture objects because their child <i>R</i> is not a mixture block.	84
6.18	(a) Left: Probe RMSEs during burn-in, for various <i>D</i> . (b) Right: Probe RMSE after burn-in, with online averaging of samples.	85
6.19	Coefficients learned on 45D run.	85

7.1	(a) Prior distribution over \mathbf{m} for DP prior (solid) and learned prior (dashed). (b) Posterior distribution over \mathbf{m} for both models. Note the peak at very small clusters for the DP. (c) Parameter values for λ for both models. (d) Error in association between datapoints as a function of α . Note that even for optimal α performance is worse than for the learned prior because small clusters have been suppressed.	93
7.2	We replace the CRP mixture block in the NMB for the Dirichlet process mixture model with a FP mixture block.	94
7.3	Two FP mixture blocks are used to define the assignment of X variables to super-topics in HFP.	97
7.4	Graphical representation of HFP model. Each box represents a restaurant (FP). Customers jointly sit down at tables according to the FP distribution. The vertically oriented restaurants each represent a document with N_d customers and N_d tables. Each table represents a document-topic. The horizontal restaurant represents the franchise restaurant and each table represents a super-topic. There are $N = \sum_d N_d$ table representatives in the franchise restaurant, one for each possible table. A unique super-topic is assigned to each z_{id} by asking the representative of their table what super-topic ($t_{r_{z_{id}d}}$) is assigned to their table ($r_{z_{id}d}$) in the franchise restaurant. The blue customers in the horizontal franchise restaurant are representatives for tables in the vertical restaurants that have no customers (not all representatives are shown). If there are L current occupied tables, then there are $N - L$ blue representatives.	99
7.5	Resulting λ from HFP learning a, b, c in equation 7.9, and HDP learning α	101
7.6	Comparison of the test perplexity of HDP vs HFP for the KOS text corpus. The x and y coordinates of each point are the perplexity of HDP (\mathbf{x}) and HFP (\mathbf{y}) on one subset of the KOS text corpus. Points below the diagonal are tests where HFP had lower perplexity than HDP	104

LIST OF TABLES

	Page
6.1 The rating for user i of movie j , r_{ij} , comes from the product of the augmented feature vectors U_{ij}^p, V_{ji}^p . In this table the feature vectors are arranged so that the sections of vectors that are multiplied together are adjacent to each other.	70
6.2 RMSEs for the model with/without side information (SI) and the Dirichlet process (DP).	75
6.3 Largest movies by number of ratings for four different movie clusters.	75
6.4 Comparison between our model and other factorization techniques . .	76

ACKNOWLEDGMENTS

I am very grateful to my adviser, Max Welling, for his support, guidance, collaboration and enthusiasm throughout my years at UCI. Max Welling has been a great mentor and teacher. I appreciated that he found the time for friendly chats about the future of artificial intelligence as well as helping me understand the details of Hamiltonian MCMC methods. I would also like to acknowledge the contributions of professors Alex Ihler and Padhraic Smyth who's guidance and collaboration contributed greatly to my time at UCI and to this dissertation. I would also like to thank my committee members professor Hal Stern, Alex Ihler and Max Welling for their for their insightful feedback.

I would also like to recognize the contributions of several co-authors that I have had the good fortune to collaborate with. I would to thank David Newman and Arthur Asuncion for their research contributions and many fruitful discussions. I would also like to thank Pietro Perona and Evgeniy Bart for introducing me to computer vision research.

Finally, I would like to thank my parents for a lifetime of love and support.

This work is supported in part by NSF grants IIS-0447903 and IIS-0914783 as well as ONR/MURI grant 00014-06-1-073.

CURRICULUM VITAE

Ian Porteous

EDUCATION

Doctor of Philosophy in Information and Computer Science	2010
University of California, Irvine	<i>Irvine, California</i>
Master of Science in Information and Computer Science	2007
University of California, Irvine	<i>Irvine, California</i>
Bachelor of Arts in Computer Science	1994
University of California, Berkeley	<i>Berkeley, California</i>

REFEREED CONFERENCE PUBLICATIONS

Bayesian Matrix Factorization with Side Information and Dirichlet Process Mixtures	2010
AAAI	
Fast Collapsed Gibbs Sampling For Latent Dirichlet Allo- cation	2008
KDD	
Multi-LDA/HDP A Non Parametric Bayesian Model for Tensor Factorization	2008
AAAI	
Unsupervised learning of visual taxonomies	2008
CVPR	
Infinite State Bayesian Networks for Structured Domains	2007
NIPS	
Gibbs Sampling for (Coupled) Infinite Mixture Models in the Stick Breaking Representation	2006
UAI	

ABSTRACT OF THE DISSERTATION

Networks of Mixture Blocks for Non Parametric Bayesian Models
with Applications

By

Ian Porteous

Doctor of Philosophy in Information and Computer Science

University of California, Irvine, 2010

Professor Max Welling, Chair

This study brings together Bayesian networks, topic models, hierarchical Bayes modeling and nonparametric Bayesian methods to build a framework for efficiently designing and implementing a family of (non)parametric Bayesian mixture models. Bayesian mixture models, including Bayesian topic models, have shown themselves to be a useful tool for modeling and discovering latent structure in a number of domains. We introduce a modeling framework, networks of mixture blocks, that brings together these developments in a way that facilitates the definition and implementation of complex (non)parametric Bayesian networks for data with partitioned structure. Networks of mixture blocks can be viewed as Bayesian networks that have been factored into a network of sub-models, mixture blocks, which are conditionally independent of each other given the introduction of auxiliary partition variables. We use this framework to develop several novel nonparametric Bayesian models for collaborative filtering and text modeling.

Chapter 1

Introduction

Bayesian networks are a cornerstone of modern Machine Learning and AI. They have been applied across a wide range of problem and application domains. One focus of recent work in Bayesian networks has been topic-models for collections of text, images and other structured corpora. Topic models can be viewed as hierarchical structured mixture models where the latent mixtures components are treated as semantic topics. Research in the area of topic models has inspired the creation of ever more complex Bayesian networks with increasing layers of hierarchy.

Bayesian networks for topic-models are often quite large but lend themselves to short descriptions and efficient inference algorithms. Topic models often have a network size that is a multiple of the number of words in the corpus. However, these large networks are also usually made up of simple networks repeated many times. This observation hints at the fact that instead of specifying each new model in terms of its Bayesian network and deriving a new inference procedure, we can take advantage of the repeated structure. Specifically, we observe that for many models the repeated structure is a mixture model, and many different models use the same sub-models,

each sub-model only differing in the way its variables are partitioned into mixture components.

Our work creates a framework that simplifies the construction of these hierarchical structured mixture models by factoring the full models into sub-models we call mixture blocks. Each mixture block is itself a hierarchical Bayesian mixture model defined in terms of how its variables are partitioned and the family of distributions for the mixture components. Furthermore, we note that conditioned on the partitioning of its variables, each mixture block is independent of other mixture blocks in the network. We take advantage of the conditional independence to specify inference in terms of the mixture blocks.

Our framework can be used to describe and build a variety of previously proposed topic models and define their inference algorithms, but it is not restricted to these tasks. In our work we have used the framework to build models for collaborative filtering, visual classification, shopping cart identification, as well as text.

The resulting framework encapsulates many of the complications of building hierarchical structured mixture models within reusable mixture block objects. This encapsulation makes it easier and more intuitive to both describe and implement a broad family of hierarchical Bayesian mixture models, encouraging the invention of new models and variants of existing models.

Chapter 2

Overview

Before moving on to the details of our framework we will walk through the description of a few topic models to illustrate how they can be factored into simpler mixture models. We will see that the interface between these mixture blocks is based on how a child mixture block partitions its variables as function of the variables of a parent mixture block. We will also see how one model leads to the next by replacement or modification of the mixture blocks. This tour through example models will lead us to the basic premise of our framework: defining and implementing complex hierarchical Bayesian (non)parametric mixture models out of mixture blocks which vary by the family of distribution (Gaussian, multinomial, nonparametric, etc) and how variables are partitioned into mixture groups. In chapter 4 and 5 we will give a formal definition of Network of Mixture Blocks (NMB) framework we propose.

We start with the Dirichlet compound multinomial model (DCM), a bag of words model for documents that can be used for document classification [27]. The DCM model assumes that each document has a separate distribution over words in the vocabulary, from which the words in the document are drawn *iid*. DCM has the

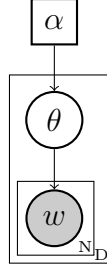


Figure 2.1: Graphical model for Dirichlet compound multinomial.

following generative description:

1. For every document d draw a distribution over words $\theta_d \sim \text{Dirichlet}(\alpha)$
2. Draw words in the document *iid* given θ_d , $w_i \sim \text{Multinomial}(\theta_d)$

where θ_d is a vector of probabilities for each word in the vocabulary. The graphical model for the DCM is in figure 2.1 ¹.

Another view of the Dirichlet compound multinomial is as a hierarchical Bayesian mixture model that we will be able to reuse as a component when building more complex topic models. In this view the DCM is a mixture model where the words are divided into sets based on which document they belong to and the words in each set are drawn *iid* from a multinomial distribution with parameters θ_d . Each θ_d in turn is drawn from a Dirichlet distribution with parameters α . Let us abbreviate this description of the model with the node in figure 2.2. The top half of the node in figure 2.2, W/ \sim_D , indicates that the variables, W , assigned to this node are partitioned by the equivalence relation \sim_D and the equivalence relation depends on variables in the set D . The equivalence relation \sim_D says that if $w_i \sim_D w_j$ then

¹The graphical model in figure 2.1 and subsequent graphical models use plate notation [8] to represent variables that repeat. In plate notation a rectangle is drawn around a subgraph of the model that repeats. A number is drawn in the rectangle to indicate the number of repetitions of that subgraph.

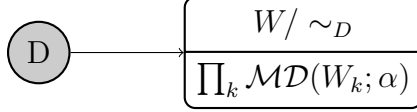


Figure 2.2: Dirichlet compound multinomial mixture component. The set of words $W = \{w_1, \dots, w_N\}$ is partitioned into K groups based on the document $d_i \in D$ that word $w_i \in W$ is from. Given the partitioning of W into K groups, the probability of all the words W is the product of the probability of each subset of words that belong to a specific document.

they belong to the same subset and are drawn *iid* from the same distribution. The bottom half of the node, $\prod_k \mathcal{MD}(W; \alpha)$, indicates that given the partition of W into sets, the joint distribution of the variables in the set W is a mixture of DCM models. \mathcal{MD} is abbreviation for a multinomial (\mathcal{M}) model with a Dirichlet (\mathcal{D}) prior over the parameters of the multinomial. In more detail, $W = \{W_1, \dots, W_K\}$, where W_1, \dots, W_K are the subsets of W which form a partition given by W / \sim_D . The joint distribution of W is the following:

$$P(W|\alpha, W/\sim_D) = \prod_{k=1}^K \int \prod_{w_i \in W_k} \mathcal{M}(w_i|\theta_k) \mathcal{D}(\theta_k|\alpha) d\theta_k \quad (2.1)$$

$$:= \prod_k \mathcal{MD}(W_k; \alpha) \quad (2.2)$$

Using the DCM component we will now build a more complicated topic model, latent Dirichlet allocation (LDA). LDA is a generative model for topic discovery [6] in which documents can be viewed as distributions over topics. Each topic is in turn a distribution over words shared across all documents. LDA has the following generative description:

1. For all T topics sample a distribution over words $\phi_t \sim \text{Dirichlet}(\beta)$
2. For all D documents sample a distribution over topics $\theta_d \sim \text{Dirichlet}(\alpha)$
3. For every word $w_i, i \in 1, \dots, N$
 - (a) sample a topic $z_i \sim \text{Multinomial}(\theta_{d_i})$
 - (b) sample a word $w_i \sim \text{Multinomial}(\phi_{z_i})$

The graphical model for LDA is in figure 2.3. LDA can be built using two DCM models. The first DCM model for the Z variables is a distribution over topic assignments. The second DCM model for the W variables is a distribution over words given the the partition of W into W_1, \dots, W_K induced by Z . Building the LDA model using two DCM components is shown in figure 2.4. The difference between the two DCM mixture blocks is that in the DCM model for topic assignment variables Z , the partitioning of variables is a function of the document id of a variable and in the second DCM model the partitioning of word tokens W is based on the value of the corresponding variable in Z . That is, the equivalence relation Z / \sim_D assigns variables z_i and z_j to the same group if they are in the same document and W / \sim_Z assigns w_i and w_j to the same group if z_i and z_j have the same value. Given the partitioning of W and Z their distribution is again a mixture of DCM models, equation 2.1.

One of the goals of this tour through models is to show how seemingly complicated models can be described in terms of simpler sub-models using the NMB framework. Therefore, we will now briefly describe an even more complicated topic model, the Pachinko allocation model (PAM). PAM [24] is another model for topic discovery which captures correlations between topics as well as words. For each word a chain or path of topics is sampled, where the probability of sampling a topic at each step

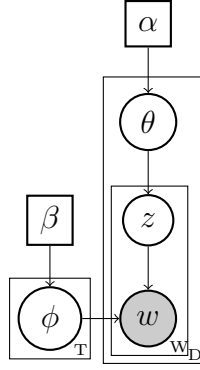


Figure 2.3: Graphical model for Latent Dirichlet Allocation (LDA).

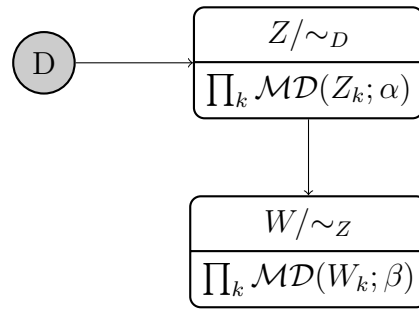


Figure 2.4: Latent Dirichlet Allocation Model drawn as network of mixture blocks. $D = d_1, \dots, d_n$ is the document id for each word. k is the id of the mixture group in each DCM mixture block

in the chain depends on the topic chosen at the previous step. That is, for every document there is a distribution over topics, super-topics, super-super-topics, and so on. For every word in every document a chain of topics is sampled and then given the last topic in the chain, a word is sampled. PAM has the following generative description:

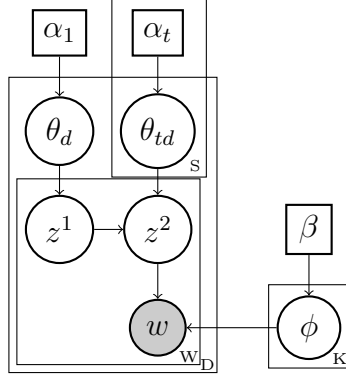


Figure 2.5: Graphical model for pachinko allocation model (PAM).

1. For all T topics sample a distribution over words $\phi_t \sim \text{Dirichlet}(\beta)$
2. For all D documents sample a distribution over root topics $\theta_d \sim \text{Dirichlet}(\alpha_1)$
3. For all D documents and L levels > 1 sample $\theta_{dl} \sim \text{Dirichlet}(\alpha_l)$
4. For every word $w_i, i \in 1, \dots, N$
 - (a) For all L levels sample
 - i. sample a topic $z_i^l \sim \text{Multinomial}(\theta_{z_i^{l-1}})$
 - (b) sample a word $w_i \sim \text{Multinomial}(\phi_{z_i^L})$

The graphical model for a “4-level” PAM is shown in figure 2.5. From the graphical model one can see that it consists of three DCM models, one for each of the variables Z^1, Z^2 and W . A PAM is described in terms of DCM mixture blocks in figure 2.6. Like in the LDA model, the difference between the three DCM models is only how the variables in each mixture block are partitioned. Unlike the LDA model where a variables assignment to a mixture group always depends on one other variable, in the middle DCM model for the variables Z^2 , the partitioning of variables now

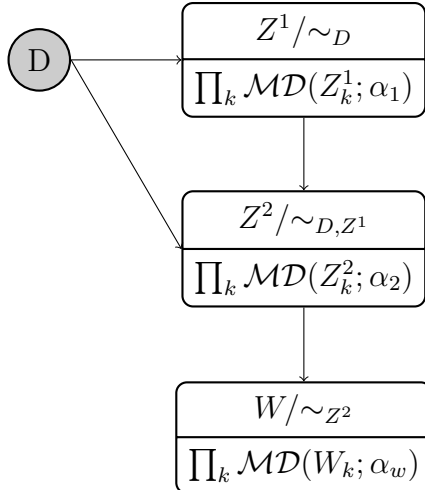
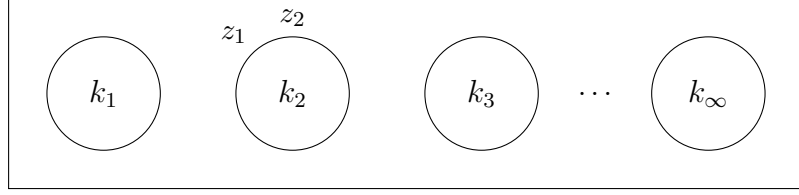


Figure 2.6: Pachinko Allocation Model drawn as NMB. $D = d_1, \dots, d_n$ is the document id for each word. K is the set of possible topics in each component

depends on two parent variables. Specifically, z_i^2 is assigned to a mixture component indexed by both d_i and z_i^1 , the document id and the corresponding z^1 variable. In the NMB framework the equivalence relation \sim_{D, Z^1} (figure 2.6) depends on two variables instead of one.

Although we have only discussed the parametric DCM mixture block so far, nonparametric Bayesian mixture models also fit in the network of mixture blocks (NMB) framework. Nonparametric mixture blocks will force us to reconsider how the variables for a child block are assigned to mixture groups. We will no longer be able to always treat a parent variable as simply an index for a mixture group.

In nonparametric Bayesian models a DCM distribution is often replaced by a Dirichlet process (DP). For example, the nonparametric extension of LDA, the Hierarchical Dirichlet Process (HDP), replaces the DCM distribution over topics for each document with a DP. However, simply replacing the DCM model for each document with a DP is not sufficient. Before explaining why replacing DCM model for each document with



$$P(z_3 = k) = \frac{N_k}{N+\alpha}$$

$$P(z_3 = k_{new}) = \frac{\alpha}{N+\alpha}$$

Figure 2.7: Chinese restaurant representation of the Dirichlet process. Customers, z_i , sit at tables in a Chinese restaurant with an infinite number of tables. Each customer sits at a table proportional to the number of customers already sitting at the table.

a DP is not sufficient we will quickly review the Chinese restaurant process ² (CRP) representation of a DP. In the CRP we imagine forming a partition of N customers by having N customers sit at tables in a Chinese restaurant with an infinite number of tables. We assign the customers (where the customers are represented by integers $1, \dots, N$) to the same set in the partition if they sit at the same table. In the CRP the first customer to arrive at the restaurant sits at one of the infinite number of tables with uniform probability. Subsequent customers sit at a table with probability proportional to the number of customers already sitting at a table. This process is depicted in figure 2, where the third customer, z_3 , is sitting at table k proportional to the number of customers, N_k , already sitting at the table. N is the total number of customers already in the restaurant. With some probability $\frac{\alpha}{N+\alpha}$, each new customer sits at an unoccupied table.

In the LDA model, if instead of sampling z_i from a multinomial for each document we sample it from a Chinese restaurant process (CRP) for each document, we run into a problem. Imagine the first customer sitting at a table in restaurant 1 and the first customer sitting at a table in restaurant 2. Since they both pick from an infinite

²The Chinese Restaurant Process (CRP) is one of several forms of the Dirichlet Process. It is often used because of its intuitive description.

number of tables with uniform probability, there is no chance they will sit at tables with the same index k . Consequently, if table indexes represent topics, there is no chance that different documents will share the same topic. In other words, if z_i is drawn from a CRP its value represents an equivalence class instead of a specific class, as is the case when z_i is drawn from a multinomial. In the case where z_i 's are sampled from a CRP if $z_i = z_j$, all we can say is which w_i and w_j should be assigned to the same mixture group but not which mixture group. This is not a problem until, as in the case of LDA, we want to relate mixture groups between different DPs. In LDA we say that if in document 1, word 1 is assigned to topic 1, and in document 2, word 1 is assigned to topic 1, then both words are assigned to the same topic. However, if topic assignments for each document are distributed according to separate CRPs, we can no longer say anything about how topic assignments in document 1 relate to topic assignments in document 2. The solution to this problem is the Hierarchical Dirichlet Process [43] which uses a hierarchical model that allows several DPs to share the same mixture components. In the case of the nonparametric version of LDA this means that mixture groups in several DPs can refer to the same topic.

When using nonparametric mixture blocks the essential issue is how to map variables drawn from a CRP in one block to a partition of variables in another block. The NMB framework remains agnostic to the mapping, but there are four basic patterns. The first, most basic structure is the Dirichlet process mixture model. To review, there are several graphical models for the Dirichlet process mixture model, one that corresponds with the stick breaking construction [39] is in figure 2.8 (left). However, it is easier to understand the role of a DP mixture block in the NMB framework as a prior distribution over all possible *partitions* S^* of N data-items [37]. A graphical model of this view is in figure 2.8 (right). In this view a partition, S^* , is drawn from a $CRP(\alpha)$. If z_i is the set that item i is assigned to in partition S^* , then the distribution for X is a mixture model where each $x_i \in X$ is assigned to a mixture

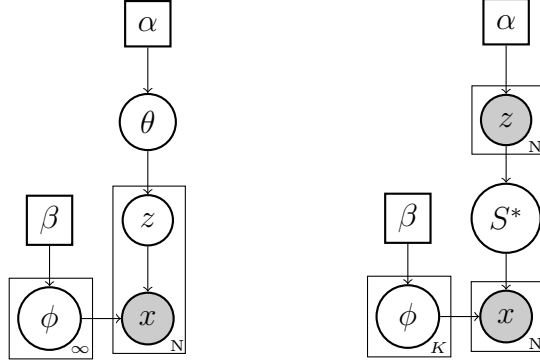


Figure 2.8: Two graphical models for the Dirichlet process. Left: The stick breaking construction for the Dirichlet process mixture model, $\theta \sim GEM(\alpha)$ [34]. Right: Dirichlet process as a distribution over partitions. The variables Z are drawn from $CRP(\alpha)$, and define a partition of N items, S^* .

group according to z_i . The NMB version of the model uses two mixture blocks, figure 2.9 (left). The first CRP mixture block, Z , is a distribution over partitions. The partition is represented by the variables Z . Using the partition drawn from the Z block, the X block is the same DCM block that we have already used.

The second structure we could build out of nonparametric mixture blocks is in figure 2.9 (right). Now instead of having one partition, the Z block has a different partition for every value of D . We get N_D independent partitions by first partitioning Z into N_D groups, $Z = \{Z_1, \dots, Z_d\}$, and then drawing each set of variables Z_d from a different CRP.

However, if we want to share mixture components between different DP mixture models then we need to be able to build HDP models using mixture blocks. The third and fourth patterns make up the two basic ways to build HDP models using mixture blocks. In the third representation we build an HDP using the Chinese restaurant franchise representation, see figure 2.10. The assignment of X variables to mixture components is then defined hierarchically as the following: x_{ji} is assigned to the mixture group $k_{jz_{ji}}$, where j is one of the vertical restaurants, and z_{ji} picks

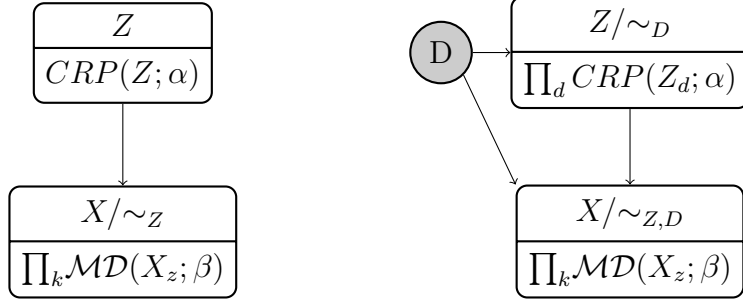


Figure 2.9: Two Dirichlet process mixture models. Left: A DP mixture model. Right: N_D independent DP mixture models, one for each value of D . Since the value z_i only represents the equivalence class with respect to the d_i CRP, the mixture group for x is a function of both d_i, z_i

the table for the i^{th} item in restaurant j , and $k_{jz_{ji}}$ picks the dish for the table. From the Chinese restaurant franchise representation in figure 2.10, where each restaurant is a CRP, one can see that HDP can be constructed using a number of CRPs. In the NMB representation we have one block for the Z variables, the vertical restaurants in figure 2.10, and one block for the K variables, the horizontal restaurants in figure 2.10. The NMB representation of this structure is in figure 2.11 (left).

In the fourth pattern, figure 2.11 (right), we introduce a direct assignment HDP (DAHDP) mixture block. This DAHDP block encapsulates the structure of the Z block and the K block in the Chinese restaurant franchise representation in figure 2.11 (left) into one block. However, the DAHDP block uses a different representation of the HDP model (see HDP inference using direct assignment in [43]). As a consequence of encapsulating the hierarchical HDP structure within a single block, the Z variables in the DAHDP block can be treated as assignments to a common set of equivalence classes across all documents instead of a separate set for each document.

In general, when building networks of mixture models, it might be difficult to anticipate all the variants of how a sub model partitions its variables. Consequently, in our framework instead of choosing a one-to-one mapping between parent variables

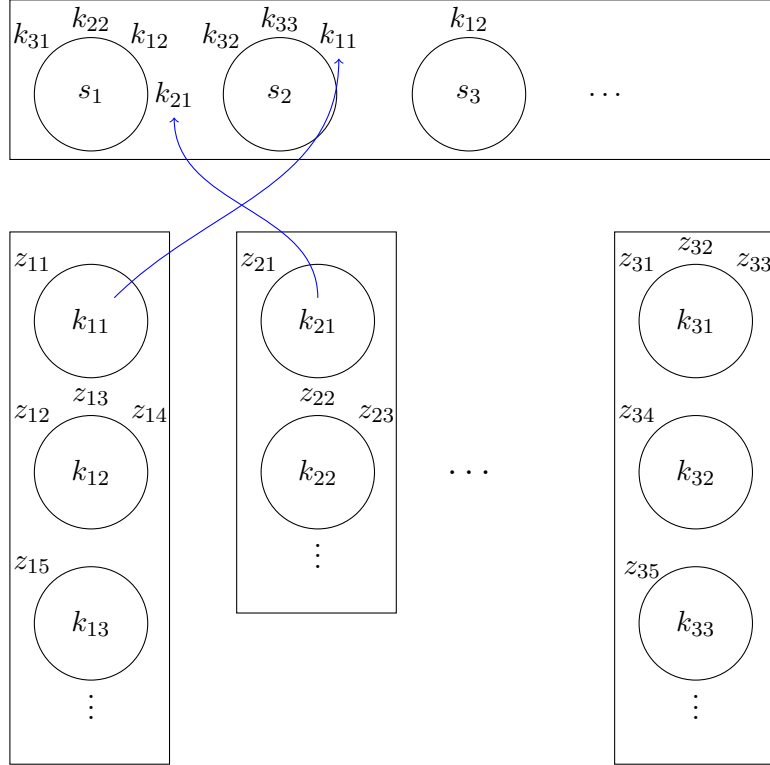


Figure 2.10: Chinese restaurant franchise representation of an HDP. Each box represents a restaurant (CRP) with an infinite number of tables. Customers come in and sit down at tables according to the CRP distribution. The vertically oriented restaurants each represent a bottom level partition, e.g. a topic distribution for a document where the tables represent the document-topic. The words are assigned to document topics via z variables. Each z variable is assigned to a table according to a CRP. The horizontal restaurant represents the franchise, and each table represents a topic. The variables for each table in each restaurant, k , are assigned a topic from the franchise according to a CRP distribution.

and mixture assignments, we will allow an equivalence relation to define the mapping between parent variables and the mixture group assignments.

These examples lead us to the basic form of a mixture blocks in our framework, illustrated in figure 2.12. First, each mixture block forms a partition for the variables assigned to it based on the value of other variables in the network. Given that partition, the mixture block defines a Bayesian mixture distribution for the variables. Finally, the variables of a mixture block may be used by other mixture blocks. We can

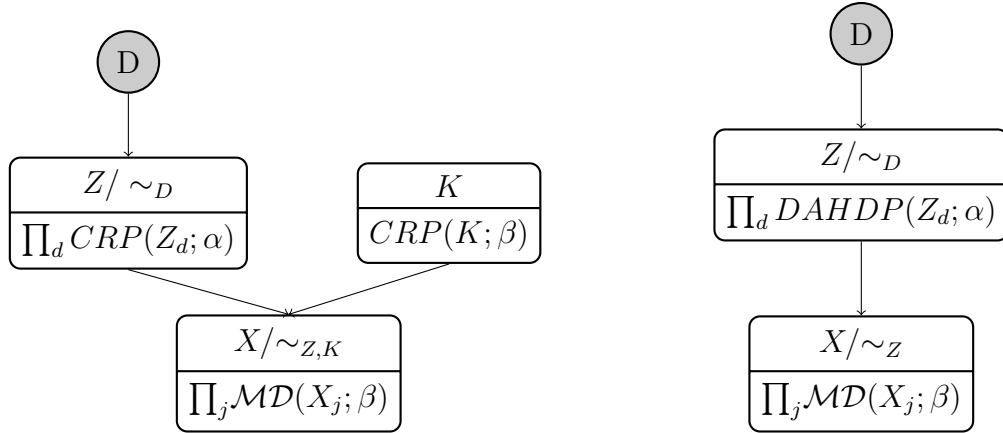


Figure 2.11: Two HDP mixture models. On the left there are two DP mixture blocks which are used to define the assignment of X variables to mixture components. On the right each set of variables Z_d represents a different partition, but since they are from an HDP they refer to a set of equivalence classes shared by all documents.

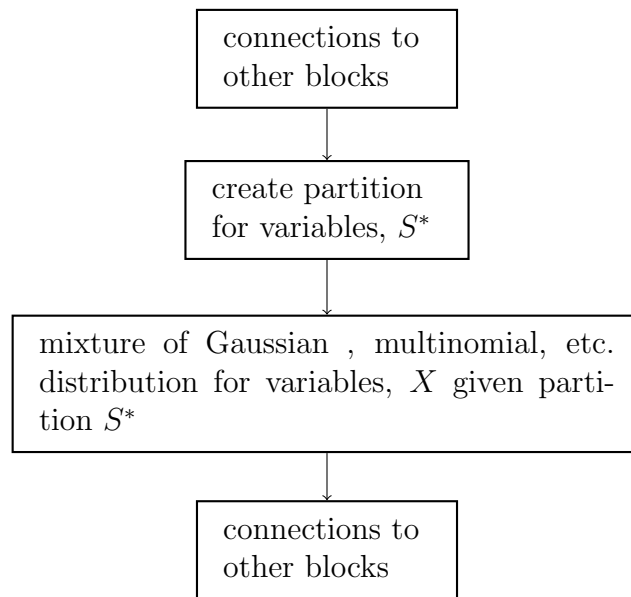


Figure 2.12: Block diagram of the functional components that go into one of the mixture blocks in a network of mixture blocks

form new networks of these mixture blocks by changing how variables are partitioned once we have a block for each family of mixture distributions. Later in section 5 we will explain how inference for these NMB models can also be done in terms of mixture blocks.

The rest of the thesis is organized as follows: We will start by describing the network of mixture blocks (NMB) framework in detail. First we will describe individual mixture blocks and then the joint model defined by a network of connected mixture blocks. Second, we specify approximate inference for a NMB using MCMC sampling and how that inference can be written in terms of mixture block objects. Third, we will describe several novel hierarchical Bayesian mixture models using the flexibility of the framework. Our first new model is a hierarchical Bayesian mixture model for collaborative filtering, Multi-LDA. The second new model is the nonparametric version of Multi-LDA, Multi-HDP, which we create by replacing the parametric mixture blocks with nonparametric mixture blocks. Next, we will explore another model for collaborative filtering, Bayesian probabilistic matrix factorization with side information that incorporates regression with latent factors in one Bayesian model. Fourth, we will introduce a new type of topic model by replacing the nonparametric mixture blocks in a hierarchical Dirichlet process topic-model with a new mixture block based on a new type of Flexible nonparametric prior.

Chapter 3

Related Work

The NMB framework is built upon a broad range of work in Bayesian networks, nonparametric Bayesian models, topic-models, product partition models, and MCMC sampling methods.

The way mixture blocks are defined in our framework is partially inspired by the definition of product partition models, [5, 37]. Product partition models (PPMs) take the form $p(Y|S^*)p(S^*)$, where Y is factored into K sets by the partition S^* and $p(Y|S^*) = \prod_k f(Y_k)$. Additionally, $p(S^*)$ can be factored as the product of cohesion's of the subsets $p(S^*) = c_0 \prod_{k=1}^K c(S_k)$. Given the partition S^* , an individual mixture blocks is the same as a PPM. However, in a NMB the partitions are formed hierarchically, the partition for a mixture blocks is a function of the variables in other mixture blocks.

Previous work on networks of infinite state Bayesian networks for structured domains (ISBN) [46], is also related in that it describes how to build networks and perform inference on Bayesian networks of nonparametric mixture models. However, it does not generalize the interface between sub-models in terms of a partition. Instead

partitions are defined hierarchically in terms of the value of parent variables with a matching index. Mapping from the variable value in one sub-model to the index in another sub-model is more limiting. For example since the value in a CRP sub-model represents only an equivalence class, it can't be blindly used as an index in another sub-model. Also ISBN does not emphasize factoring networks into reusable components.

In the sense the NMB tries to automate or at least simplify inference for a family of probabilistic models, there are a number alternative frameworks. One example is the Bayesian Logic (BLOG) inference engine [32]. Another example is the Hierarchical Bayes Compiler (HBC) [18]. Compared to these projects, NMB is less ambitious and more practical. Instead of trying to provide a language in which models can be specified and inference performed automatically, NMB simply describes how hierarchical Bayesian structured mixture models can be factored into reusable components. This approach is more practical in the sense that for a particular implementation of a NMB, one may still need to make model specific assumptions that make inference faster for the specific model. For example one may make assumptions that certain operations can be done in parallel in order to implement a model for a large data set as we did in section 6.2.6.

Chapter 4

Networks of Mixture Blocks

In this chapter we describe the network of mixture blocks framework. First we will briefly review the hierarchical Bayesian mixture of Gaussians model, which we will use as a concrete example of a NMB. Next we look at the definition of individual mixture blocks. Finally, the joint model defined by a network of the individual mixture blocks is described.

It is worth noting at this point that, although we will use a parametric example to describe mixture blocks and inference procedures, the framework does not distinguish between parametric and nonparametric mixture blocks. Essentially mixture blocks view each other through partitions and it does not matter to a mixture block if the blocks on the other side of the partition are nonparametric blocks.

4.1 Bayesian Mixture of Gaussians

To illustrate the description of our framework, we will use the familiar example of a hierarchical Bayes mixture of Gaussians model (MoG). The MoG model has the

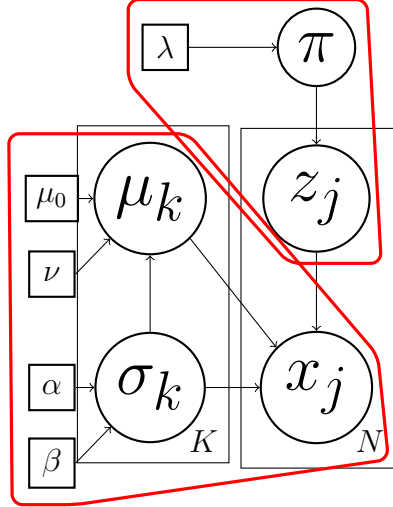


Figure 4.1: Bayesian model Mixture of Gaussians. Square nodes represent parameters of the model. Red outline designates the sub-graphs that correspond to the mixture blocks for Z and X

following generative description for a set of observed data $X = \{x_1, \dots, x_N\}$.

For each of the K topics

$$\text{Draw } \mu_k, \sigma_k \sim \mathcal{N}(\mu_k; \mu_0, \sigma_k/\nu) \Gamma^{-1}(\sigma_k; \alpha, \beta)$$

For each observation $x_j \in X$

$$\text{Draw a topic: } z_j \in \{1, \dots, K\} \sim \mathcal{M}(\pi)$$

$$\text{Draw an observation from the topic: } x_j \sim \mathcal{N}(\mu_{z_j}, \sigma_{z_j})$$

where π are the parameters of a multinomial distribution drawn from a Dirichlet prior with parameters $\lambda_1, \dots, \lambda_k = \lambda/K$. The Bayesian network using plate notation is shown in figure 4.1.

The joint density for the MoG model, $P(X, Z, \pi, \lambda, \boldsymbol{\mu}, \boldsymbol{\sigma}, \alpha, \beta, \mu_0)$, is the following:

$$p(X, Z, \pi, \lambda, \boldsymbol{\mu}, \boldsymbol{\sigma}, \alpha, \beta, \mu_0, \nu) = \left[\prod_{j=1}^N p(x_j | z_j, \mu_{z_j}, \sigma_{z_j}) p(z_j | \pi) \right] \times \left[\prod_{k=1}^K p(\mu_k | \mu_0, \sigma_k / \nu) p(\sigma_k | \alpha, \beta) \right] \times p(\pi | \lambda)$$

where $\boldsymbol{\sigma} = \{\sigma_1, \dots, \sigma_K\}$, $\boldsymbol{\mu} = \{\mu_1, \dots, \mu_K\}$ and $Z = \{z_1, \dots, z_N\}$.

4.2 Mixture Blocks

The basic units of our modeling framework are compound distributions organized into entities we call mixture blocks. Mixture blocks allow us decompose a family of complex Bayesian networks into conditionally independent mixture blocks, each of which is itself a hierarchical Bayesian network. These mixture models, which we call mixture blocks, help us encapsulate the complexity and implementation of hierarchical Bayesian mixture models.

Each mixture block defines a distribution over variables associated with it. The blocks are composed of two parts, a partition part and a distribution part. The partition part specifies how the set of random variables assigned to a block will be partitioned into conditionally independent sets. The distribution part of the block specifies the probability density function for the set of variables associated with a block. For example, the MoG model described in terms of mixture blocks is composed of two blocks. The first block is responsible for the cluster assignments, i.e. the hierarchical Bayes model for $Z = \{z_1, \dots, z_N\}$. The second block is responsible for the hierarchical Bayes model for the data X , conditioned on the partition induced by the cluster assignments Z . The mixture block diagram for the MoG model is shown in figure

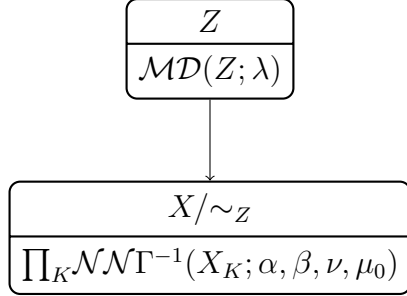


Figure 4.2: MoG. \mathcal{MD} stands for a multinomial distribution whose parameters have a Dirichlet prior. $\mathcal{NN}\Gamma^{-1}$ indicates a Normal distribution with a Normal-scaled inverse gamma prior over its parameters.

4.2.

The partition part of a mixture block defines a partitioning of the variables in the mixture block into conditionally independent sets of exchangeable variables. For example, the partition part of the X mixture block in figure 4.2 partitions the set of variables X into groups, each of which is associated with a different distribution. The partitions are a key concept in the overall framework because they allow the mixture blocks to be conditionally independent no matter how complicated the mapping between parent variables and mixture groups assignments becomes. For this reason it is easier to define the framework if we introduce an auxiliary variable S^* whose value is a partition of the variables in a mixture block. Introducing the partition variable S^* allows us to easily describe the joint model of a network of mixture blocks in terms of separate mixture blocks. Using partitions also unifies parametric and nonparametric models in one framework.

Describing the joint model in terms of mixture blocks will lead us to factoring inference for the full model in terms of mixture block operations. Although the S^* variables are useful in specifying the framework, in section 5 we will eventually write the inference procedure without the S^* variables.

Using the equivalence kernel of a function ρ we will define a mapping from the domain of the parent variables to the domain of partitions S^* . The equivalence kernel of the function ρ has the following form:

$$x_a \sim x_b \iff \rho(\pi(x_a)) = \rho(\pi(x_b)).$$

where ρ is a function whose arguments, $\pi(x)$, are the parent variables which will determine the subset a variable belongs to in the partition S^* . The set $\pi(x)$ is also the set of parents of a variable in the corresponding Bayesian network. In the block diagram this is abbreviated X/\sim_Z . X/\sim_Z is used to denote that the set X is partitioned by an equivalence relation \sim_Z . The Z subscript denotes that the equivalence relation depends on the parents of X , which are in the set Z . That is $x_a \sim_Z x_b \iff \rho(\pi(x_a)) = \rho(\pi(x_b))$ and $\forall_c \pi(x_c) \in Z$. An arrow is also used from the parent mixture block to the child mixture block to indicate the parent child relationship in the corresponding Bayesian network. In the MoG model the variables in X are assigned to partitions depending on the value of their parent assignment variable, $\pi(x_a) = z_a$ and $\rho(\pi(x_a)) = z_a$. That is, if the parent value $z_a = k$ then $P(x_a) = \mathcal{N}(\mu_k, \sigma_k)$. Therefore, two variables x_a, x_b are in the same subset if $z_a = z_b$. Formally, a mixture block partitions the data and defines a distribution factored by the partition in the following way:

$$S^* = \{S_1, \dots, S_K\} \text{ is a partition of } \{1, \dots, N\}$$

$$S_k = \{i : \rho(\pi(x_i)) = k\} \text{ for MoG } S_k = \{i : z_i = k\}$$

$$X_k = \{x_i : i \in S_k\}$$

$$X^* = \{X_1, \dots, X_K\}$$

$$X = \{x_1, \dots, x_N\}$$

$$P(X|S^*) = \prod_{k=1}^K P(X_k)$$

The second part of a mixture block is the distribution part. It defines a distribution over the set variables assigned to that mixture block as the product of distributions for each subset X_k . Conditioned on the the partition S^* induced by the equivalence kernel of the function ρ , the distribution part is equivalent to the definition of a product partition model conditioned on S^* [37, 5]. That is the data in each subset is assumed to be drawn *iid* from some parametric family f with parameters θ_k ,

$$P(X|S^*) = \prod_{k=1}^K \int \prod_{i \in S_k} f(x_i|\theta_k) dG_0(\theta_k) \quad (4.1)$$

where $f(x_i|\theta_k)$ is the likelihood and $(\theta_1, \dots, \theta_k)$ is drawn *iid* from the prior G_0 .

To specify the full mixture block form of the MoG model, we will use the convention that a superscript indexes the mixture block version of a variable. For example, there are two mixture blocks X and Z for the MoG model, so S^{Z^*} is the partition for the Z mixture block and S^{X^*} is the partition for the X mixture block. The following is a full specification of the conditional mixture block distributions for MoG ¹:

$$\begin{aligned} S^{Z^*} &= \{1, \dots, N\} \\ S_1^Z &= S^{Z^*} \text{ (note: there is only one set)} \\ Z_1 &= \{z_i : i \in S_1^Z\} \\ Z &= \{Z_1\} \\ P(Z|S^{Z^*}, \lambda) &= \int \prod_{i \in S^{Z^*}} \mathcal{M}(z_i; \theta^Z) \mathcal{D}(\theta^Z; \lambda) d\theta^Z \end{aligned}$$

¹For the Z mixture block, the Z 's are not being partitioned, but to stay consistent with the general specification of a mixture block we will describe it in terms of a partition with only one subset.

$S^{X^*} = \{S_1^X, S_2^X, \dots, S_K^X\}$ is a partition of $\{1, \dots, N\}$

$S_k^X = \{i : \rho^X(\pi^X(x_i)) = k\}$ for MoG $S_k^X = \{i : z_i = k\}$

$X_k = \{x_i : i \in S_k^X\}$

$X = \{X_1, \dots, X_K\}$

$\theta_k^X = \{\mu_k, \sigma_k\}$

$$P(X|\mu_0, \sigma_0, \nu, S^{X^*}) = \prod_{k=1}^K \int \prod_{i \in S_k^X} \mathcal{N}(x_i; \mu_k, \sigma_k) \mathcal{N}(\mu_k | \mu_0, \sigma_k / \nu) \Gamma^{-1}(\sigma_k | \alpha, \beta) d\theta_k^X$$

4.3 Network of Mixture Blocks

We will now describe the complete model created by connecting several mixture blocks. Given the partitions S^{X^*}, S^{Z^*} the mixture blocks are independent. Thus, conditioned on the partitions the joint probability of X, Z is simply the product of their probabilities separately. If a indexes the mixture block and γ^a is the set of hyperparameters for G_0^a , then conditioned on the partitions, $S^{a^*}, a = \{1, \dots, M\}$, the joint probability of the random variables for a network of M mixture block models is the following:

$$P(X^1, \dots, X^M | S^{1^*}, \dots, S^{M^*}, \gamma^1, \dots, \gamma^M) = \prod_{a=1}^M P(X^a | S^{a^*}, \gamma^a)$$

Sometimes the partition variables S^{a^*} are observed, as in the case where a text corpus is partitioned into documents and we know which words belong to which documents. However, we are often interested in the case where the partition is unknown. Furthermore, we may be interested in learning the partition given the observed data. A common approach to this problem is to introduce hidden or latent variables which

represent the partition, such as the component indicator variables Z in the MoG model.

Let us consider the distribution over partitions S^* for a single mixture block associated with the data X . Let Σ^* be a random variable which maps the sample space made up of the parent values of X to partitions of $\{1, \dots, N\}$. Also, let the probability of a particular partition S^* be the normalized probability of the parent variables taking on a value such that ρ maps the parent values to S^* . In more detail, Let \mathbf{z} be a random variable made of the values taken by all the parent variables of the set X . In the MoG example $\mathbf{z} = \{z_1, \dots, z_N\}$ for the X mixture block. A is the set of possible parent values that are mapped by ρ to the specific partition $S^* = \{S_1, \dots, S_K\}$, $A = \{\mathbf{z} : \forall_{k=1}^K \forall_{i \in S_k} \pi(x_i) \in \mathbf{z} \wedge \rho(\pi(\mathbf{x}_i)) = \mathbf{k}\}$. Then the probability of a partition S^* is the sum over the probability of all possible parent values that lead to that partition, $P_{\Sigma^*}(S^*) = D \sum_{\mathbf{z} \in \mathbf{Z}} \mathbf{1}_A(\mathbf{z})f(\mathbf{z})$, where $f(\mathbf{z})$ is the probability mass function of \mathbf{z} , and D is the normalizing constant. Here we are assuming the usual case where \mathbf{z} is discrete, but it can be extended to the case where \mathbf{z} is continuous.

The joint probability of the partitions, S^{1*}, \dots, S^{M*} , and the variables, X^1, \dots, X^M has the following form:

$$P(X^1, \dots, X^M, S^{1*}, \dots, S^{M*} | \gamma^1, \dots, \gamma^M) = \prod_{a=1}^M P(X^a | S^{a*}, \gamma^a) P(S^{a*} | X^{\pi(a)}) \quad (4.2)$$

where $X^{\pi(a)}$ is the set of variables that are parents in the corresponding Bayesian network of the variables in the set X^a . $P(S^{a*} | X^{\pi(a)}) = \mathbf{1}_A(X^{\pi(a)})$, the indicator function that takes value 1 if the function ρ^a maps $X^{\pi(a)}$ to the partition S^{a*} . That is, the partition S^{a*} is deterministic given the values of the parents $X^{\pi(a)}$ ². The joint probability now looks very similar to the definition of the joint probability for a

²It can be the case, as in a Hidden Markov Model, that X^a and $X^{\pi(a)}$ overlap.

Bayesian network. In fact, we could have written the same distribution by applying ρ to the possible values of $X^{\pi(a)}$ and explicitly writing down the conditional probability table of $P(X^a|X^{\pi(a)}, \gamma^a)$. Note that it is still the case that the dependencies between variables X^1, \dots, X^M and partition variables S^{*1}, \dots, S^{*M} form a directed acyclic graph. If a variable x^a has a role in how x^b is grouped in a mixture model, where there would have been a directed edge between two variables $x^a \rightarrow x^b$ in the Bayesian network, there is now an intermediate partition variable $x^a \rightarrow S^* \rightarrow x^b$.

If we can write a mixture block model down in terms of a Bayesian network, the natural question to ask is why should we use the NMB framework in the first place. The answer is that the NMB framework allows us to encapsulate some of the complexity of the full model within the mixture blocks. This will make it easier to design and implement complex Bayesian networks, including non-parametric Bayesian networks. Additionally, the NMB can be mapped to an efficient MCMC inference scheme made up of modules that implement individual mixture blocks. Another way to look at it is that NMB allows us to partition a complex Bayesian network into sub graphs, where the distribution over variables in each sub graph only depends on variables outside the sub graph (mixture block) through the equivalence kernel of ρ .

The NMB decomposition will make the problem of designing and implementing complex models easier both from the top down and from the bottom up. From the bottom up, in designing and implementing these mixture blocks we will only need to consider the conditional distribution of variables owned by a block. From the top down, we will be able to build complex new models by picking from a set of available mixture blocks, and defining the structure of the generative process with the ρ functions.

Chapter 5

Inference for Mixture Block Models

In this section we will examine how, given a network specified in terms of mixture blocks, we can construct a Markov Chain Monte Carlo (MCMC) procedure for inference that takes advantage of the mixture block structure. For non-trivial mixture block models inference is intractable so an approximate method must be used. We choose to use MCMC methods because it allows us to decompose the sampling procedure into algorithms which are local to each mixture component. That is, we can build an inference algorithm for the joint model using local inference algorithms defined only on the individual parts.

5.1 Gibbs Sampling

We start by noting that Gibbs sampling can be performed for Bayesian networks by sampling a new value for each variable conditioned on its Markov blanket (its children,

its parents, and the other parents of its children). To perform inference with Gibbs sampling, we sequentially sample each variable conditioned on all the other variables. We repeat this process until we have gathered sufficient samples or we decide to stop due to a time limit. The sequence of samples is called a chain. The distribution of the samples produced by this procedure is guaranteed to converge to the distribution of the variables in the limit. For Bayesian networks, where variables are conditionally independent of all the rest of the network given their Markov blanket, the Gibbs sampling algorithm is given by

For epoch $e = 1 \dots E$

For all variables $i = 1 \dots N$

$$\text{Sample a new value for } Y_i^{t+1} \sim P(Y_i^{t+1} | Y_{-i}^t) \propto P(y_i^{t+1} | \pi(y_i)) \prod_{\{y_j: y_i \in \pi(y_j)\}} P(y_j^t | \pi(y_j)) \quad t \leftarrow t+1 \quad (5.1)$$

where $Y = \{y_1, \dots, y_N\}$ is now the set of all random variables in the model, Y_{-i} is the set Y not including the variable y_i , $\pi(y_i)$ is the set of variables that are the parents of y_i , and the superscript t indexes the value of the variables at step t of the algorithm. Each epoch e is a pass through the entire set of variable. Parents, $\pi(y_i)$, of y_i are those variables in the Bayesian network for which there is an edge from the parent variable to y_i .

We can then estimate the probability of any subset of variables $Y_k \subseteq Y$ from the chain of samples. For the discrete case:

$$\frac{1}{T} \sum_{t=1}^T \mathbb{I}[Y_k^t = y] \rightarrow P(Y_k = y) \text{ as } T \rightarrow \infty$$

5.2 Network of Mixture Block Sampling

We will now write an MCMC inference procedure for NMB similar to algorithm 5.1. It is worth noting that even though we are working with models that can be expressed as a Bayesian network, we are often sampling a set of variables that no longer have conditional independencies. The directed acyclic graph of conditional independencies is lost when we analytically marginalize over (collapse) the parameters of the likelihood functions. That is, we solve the integral in equation 5.2 analytically.

We need to rewrite $P(Y_i^{t+1}|Y^t)$ in equation 5.1 in terms of the variables that make up the NMB, $\Omega = \{X^a, S^{a*}, \theta^a, \lambda^a\}$; $a = 1, \dots, M$ where a indexes the mixture block and there are M mixture blocks. Now we will write the joint distribution in terms of the individual variables. In equation 4.2 the probability of X^a was conditioned on the partitions S^{*a} , but for an individual variable x_i , conditioning on $\rho(\pi(x_i))$ is equivalent to conditioning on the partition S^* , i.e. $P(x_i|S^*, \gamma) = P(x_i|\rho(\pi(x_i)), \gamma)$. Thus the full joint distribution of an NMB network written in terms of the conditional probability of individual variables is the following:

$$P(X^1, \dots, X^M, S^{1*}, \dots, S^{M*} | \gamma^1, \dots, \gamma^M) = \prod_{a=1}^M \left[\prod_{i=1}^{N^a} 1_B(\pi(x_i^a)) \right] \left[\prod_{k=1}^{K^a} \int \prod_{i \in S_k^a} f^a(x_i^a; \theta_k^a) G^a(\theta_k^a) d\theta_k^a \right] \quad (5.2)$$

$1_B(\pi(x_i^a))$ is the indicator function for the set of partitions S^{*a} which are consistent with the parent values, $X^{\pi(a)}$, or equivalently if $S^{*a} = \{S_1^a, \dots, S_K^a\}$ then $(B = \{\mathbf{z} : \forall_{k=1}^K \forall_{i \in S_k} \pi(x_i) = z \wedge \rho(\pi(x_i)) = k\})$. That is, $1_B(\pi(x_i^a))$ assigns 0 probability to all partitions where element i is in a different group than $k = \rho^a(\pi(x_i))$. For Gibbs sampling we need the distribution $P(x_i^a | \Omega_{-x_i^a})$. However, equation 5.2 instead gives us the joint probability of \mathbf{X} and \mathbf{S}^* , where $\mathbf{S}^* = \{S^{1*}, \dots, S^{M*}\}$, $\mathbf{X} = \{X^1, \dots, X^M\}$.

Fortunately, because of the term $1_B(\pi(x_i^a))$, we can trivially marginalize over the partitions, S^{*1}, \dots, S^{*M} . Specifically, the density in equation 5.2 is non zero only when $1_B(\pi(x_i^a))$ is 1 for all i and a , so after marginalizing over \mathbf{S}^* we only have to consider the case where the partitions are consistent with the functions ρ^1, \dots, ρ^M applied to the variables $x \in \{X^1, \dots, X^M\}$. Let $\hat{\mathbf{S}}^*$ indicate the value of \mathbf{S}^* for which $P(\mathbf{X}, \mathbf{S}^*) > 0$. First, if we consider the case where we do not marginalize over the likelihood parameters then the conditional distribution we want for Gibbs sampling is $P(x_i^a | \mathbf{X}_{-x_i^a}, \Theta, \Gamma) = P(x_i^a, \hat{\mathbf{S}}^* | \mathbf{X}_{-x_i^a}, \Theta, \Gamma)$, where $\Theta = \{\Theta^1, \dots, \Theta^M\}$, $\Theta^a = \{\theta_1^a, \dots, \theta_{K^a}^a\}$, $\Gamma = \{\gamma^1, \dots, \gamma^M\}$. The density for the conditional distribution of x_i^a is the following:

$$\begin{aligned} P(x_i^a, \mathbf{X}_{-x_i^a} | \Theta, \Gamma) &= \sum_{\mathbf{S}^* \in \Sigma^*} P(x_i^a, \mathbf{X}_{-x_i^a}, \mathbf{S}^* | \Theta, \Gamma) \\ &= P(x_i^a, \mathbf{X}_{-x_i^a}, \hat{\mathbf{S}}^* | \Theta, \Gamma) \end{aligned} \quad (5.3)$$

$$\begin{aligned} P(x_i^b | \mathbf{X}_{-x_i^b}, \Theta, \Gamma) &= \frac{P(x_i^b, \mathbf{X}_{-x_i^b}, \hat{\mathbf{S}}^* | \Theta, \Gamma)}{\sum_v P(x_i^b = v, \hat{\mathbf{X}}_{-x_i^b}, \hat{\mathbf{S}}^* | \Theta, \Gamma)} \\ &= \frac{\prod_{a=1}^M \prod_{k=1}^{K^a} G^a(\theta_k^a) \prod_{j \in \hat{S}_k^a} f^a(x_j^a; \theta_k^a)}{\sum_{x_i^b \in V} f^b(x_i^b; \theta^b) \prod_{a=1}^M \prod_{k=1}^{K^a} G^a(\theta_k^a) \prod_{j \in \hat{S}_k^a} f^a(x_j^a; \theta_k^a)^{(1-[a=b \wedge i=j])}} \end{aligned} \quad (5.4)$$

Here, V is the domain of x_i^b . All of the terms in equation 5.4 that do not change when x_i^b changes can be canceled. After canceling terms, in the numerator we are left with only the likelihood, $f(x_i^b; \theta_{\rho^b(\pi(x_i^b))}^b)$, and the terms for which x_i^b affects the class assignment. Specifically, x_i^b affects the assignment of x_j^a where $\{a, j\}$ are in the set of children $C = \{\{a, j\} : x_i^b \in \pi(x_j^a)\}$.

$$P(x_i^b | \mathbf{X}_{-x_i^b}, \Theta, \Gamma) \propto f(x_i^b; \theta_{\rho^b(\pi(x_i^b))}^b) \prod_{\{a, j\} \in C} f(x_j^a; \Theta_{\rho^a(\pi^a(x_i^b))}^a) \quad (5.5)$$

$P(x_i^a | \mathbf{X}_{x_i^a}, \gamma^1, \dots, \gamma^M)$, is now the product of the term for x_i^a given $\{\hat{S}^{*a}, \mathbf{X}_{-x_i^a}\}$, and the terms for the children whose group assignment is affected by x_i^a , $\{x_j : x_j^a \in \pi(x_i^a)\}$. This leads us to the NMB Markov blanket property. Analogous to the Markov blanket property for Bayesian networks, the NMB blanket property is the following: x_i^a is conditionally independent of variables outside of a mixture block given the variables which affect its class assignment in partition S^{a*} (x_i^a 's NMB parents) and the variables for which x_i^a affects their class assignment (x_i^a 's NMB children), and the parents of x_i^a 's NMB children.

The difference between the Markov blanket property and the NMB blanket property is that the latter is in terms of the partition variables S^* and by extension the equivalence functions ρ .

5.3 Encapsulation of Inference in Mixture Blocks

One of the main benefits of using the NMB is that inference can be abstracted in terms of mixture block objects. A mixture block object, MBO, encapsulates the details of the different mixture distributions and the variables associated with the block behind a common interface. For example a mixture block object for a mixture of Gaussians or one for a mixture of multinomials will have the same interface. Gibbs sampling of full NMB can then be done in terms of the common interface provided by each MBO. The minimum interface for a mixture block object provides for updating the value of a variable, and provides the terms of equation 5.1. Specifically a mixture block (minimally) provides an interface with the following functions.

- Probability of a variable x_i given S^{m*}

$$p(x_i^m | S^{m*}, X_{-x_i^m}^m)$$

- Children of a variable i

$$C(i) = \{ \{a, j\} : x_i^m \in \pi^a(x_j^a) \}$$

- Parents of a variable i

$$\pi^m(x_i^m)$$

- Sample a new value for variable i according to equation 5.5

$$x_i^{m,t+1} \sim p(x_i^m | S^{m*}, X_{-x_i^m}^m) \prod_{\{a,j\} \in C(i)} p(x_i^a | S^{a*}, X_{-x_i^a}^a) \quad (5.6)$$

where t is the step in the MCMC chain.

- Sample a new value for Θ^m, Γ^m

(In the case where the parameters for the mixture distributions are not collapsed, they must be sampled.)

- Set the value of variable i to v

$$x_i = v$$

With this interface, inference is factored so that in order for a mixture block to implement its interface, it only depends on other mixture blocks through their interface. Thus the details of the inference are encapsulated within the mixture block objects. NMB models can then be built out of mixture block objects with this interface.

For example the MoG example uses two mixture block objects. The first is the Z MBO for a multinomial mixture distribution $\mathcal{MD}(Z; \lambda)$, where the multinomial parameters have a Dirichlet prior. The second is the X MBO for a Normal distribution with a Normal-scaled inverse gamma prior over its parameters, $\mathcal{NN}\Gamma^{-1}(X_z; \alpha, \beta, \nu, \mu_0)$. For the first MBO, $Z, \rho^z() = 1$, i.e. all of the variables are in the same in one set. For

MBO X , $\pi^X(x_i) = z_i$, $\rho^X(\pi^X(x_i)) = z_i$. Now Gibbs sampling for the NMB model for MOG built using mixture block objects can be implemented by repeatedly invoking the sample variable function, 5.6, of the Z mixture block object.

Once we have a library of MBOs, it becomes easy to implement interesting new NMB models by customizing the ρ and π functions. Approximate inference using Gibbs sampling can be performed by invoking the sample variable function for unobserved variables.

5.3.1 Joint Sampling

It is sometimes the case that we want to sample a number of variables jointly in order to improve the mixing rate. For example in Pachinko Allocation, instead of sampling the value of one step on the topic path, we want to sample a new path jointly. This can be done by taking the variables we want to sample jointly, e.g. z_{i1}, z_{i2} for PAM, and creating a new graph with only those variables. where the new graph is built using the conditional distribution of the variables in the NMB, equation 5.4. Using the new graph a new joint sample for the variables can be drawn using an approach such as the junction tree algorithm [21].

Chapter 6

Collaborative Filtering through Matrix Factorization

One of the main benefits of the NMB framework is that it allows one to rapidly develop new and interesting topic and mixture models that are efficient enough to be applied to large datasets while also providing an interface for rapid model development. Using the mixture block framework we will explore several novel models for matrix factorization applied to collaborative filtering.

Matrix factorization algorithms are frequently used in the machine learning community to find low dimensional representations of data. We introduce several generative Bayesian probabilistic models for unsupervised matrix and tensor factorization. The first set of models, Multi-LDA, Multi-HDP [36], will explore generative Bayesian non-negative matrix factorization. These models consists of several interacting LDA/HDP models, one for each modality. The second set of models introduces Bayesian matrix factorization with side information and a Dirichlet Process MoG prior for the latent factors, [35].

6.1 Multi-LDA , Multi-HDP

Matrix factorization refers to the problem of representing a given input matrix as the product of two lower rank matrices. In this section we focus on matrix factorization for the purpose of learning the latent or hidden structure of the data and prediction of missing elements of the input matrix. Collaborative filtering is an example of a problem where matrix factorization has been applied to achieve these goals. In collaborative filtering we assume a set of N users $1, \dots, N$, a set of M items $1, \dots, M$, and rating values which can be real or discrete. For example we might have N movie patrons (users) and M movies, for which we have sparse data on how some users rated some movies. We can represent this data as a matrix X with N rows and M columns most of which are unknown. Our goal may be to predict the missing values of X , (i.e. how much a user will enjoy a movie they have not seen), or we may want to find groups of users with similar tastes or movies with similar themes. If we factor X into the product of UWV^T we can generate values for the unknown elements of the input data.

Non probabilistic models for matrix factorization such as non-negative matrix factorization (NMF) and singular value decomposition (SVD) have been applied to collaborative filtering and work well for predicting missing values, or classification based on the low rank representation. However, generative probabilistic models have a number of advantages that these methods lack such as predictive *distributions* from which predictor intervals can be inferred, the possibility to include prior knowledge into the generative process and a principled framework to select model structure. Moreover, the inferred latent structure is directly tied to the generative process and therefore often easy to interpret.

Probabilistic models such as probabilistic latent semantic analysis (PLSA) and their

Bayesian extensions such as LDA [7] have been proposed as text models in the bag-of-words representation. These models have the benefit of learning a latent structure for the data and providing a probability distribution for the missing data predictions. LDA in particular has proved successful at revealing latent document topics [15]. Extensions of PLSA [17] and LDA [30] to the collaborative filtering case have been proposed. However, using movie ratings as an example, these models treat users and movies differently. In particular, they discover hidden structure for users but not for movies.

Our model fits into the class of models known as “blockmodels” [2]. The parametric, two modality version of our model, Bi-LDA, is similar to [2] with multinomial instead of Bernoulli mixture components. The extension to nonparametric methods has also been considered before with other blockmodels [19] and [28]. However these nonparametric block models have objects (users, movies) belonging to only a single group. In our model objects can belong to several groups (i.e. one for each rating).

[31] is another Bayesian matrix factorization method which does treat two modalities symmetrically but assumes a different structure for the data. They use a ‘factorial learning’ model that encourages a representation where an object is described by a diverse set of features. Our model on the other hand encourages a succinct representation where an object belongs to few groups akin to soft multi-modality clustering. Specifically, in our model we have that the more an object belongs to one group, the less it belongs to other groups. Another essential difference is that our model combines factors/topics in the probability domain, while the model of [31] combines factors in the log-probability domain.

The Multi-LDA model consists of two or more interacting Latent Dirichlet Allocation models, one for each modality, users and movies, and Multi-HDP consists of two or more interacting Hierarchical Dirichlet Process models. Using the mixture

block framework we describe an efficient collapsed Gibbs sampler composed of mixture blocks for inference. We also show how it is easy to convert from the parametric to the non parametric form of the model by replacing the Dirichlet multinomial mixture block for membership assignment variables with a hierarchical Dirichlet process mixture block for membership assignment variables. Experiments demonstrate that the model is useful for prediction of missing data with two or more modalities as well as learning the latent structure in the data.

6.1.1 Bi-LDA

It is easiest to describe the model by first describing the following generative process for ratings in the Bi-LDA model:

1. Choose $J \times K$ distributions over ratings $\phi_{jk} \sim \text{Dirichlet}(\beta)$
2. Choose a distribution over K user groups for each user $\pi_u^{user} \sim \text{Dirichlet}(\alpha^{user})$
3. Choose a distribution over J movie groups for each movie $\pi_m^{movie} \sim \text{Dirichlet}(\alpha^{movie})$
4. For each movie-user pair (mu)
 - (a) Choose a user group $z_{mu}^{user} \sim \text{Multinomial}(\pi_u^{user})$
 - (b) Choose a movie group $z_{mu}^{movie} \sim \text{Multinomial}(\pi_m^{movie})$
 - (c) Choose a rating $r_{mu} \sim p(r_{mu} | z_{mu}^{user}, z_{mu}^{movie}, \phi_{z_{mu}^{movie}, z_{mu}^{user}}^m)$

Where Φ_{jk} is the distribution over values $1..V$ for cluster j, k and has a Dirichlet prior with parameter β . π_m^{movies} and π_u^{users} are distributions over movie and user groups with Dirichlet priors using parameters $\alpha^{movies}, \alpha^{users}$ respectively. We also introduce

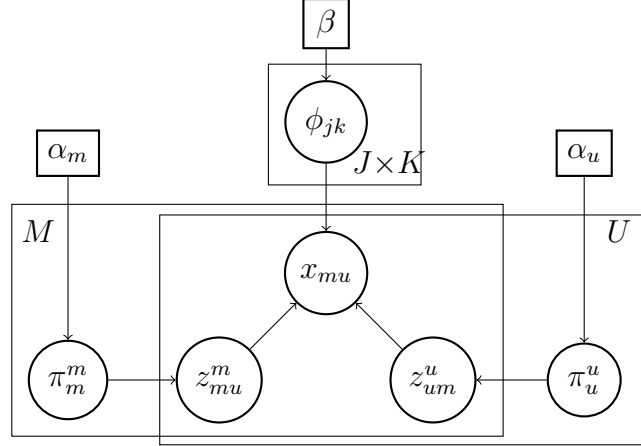


Figure 6.1: Bayesian graphical model for Bi-LDA.

indicator variables z_{mu}^{movies} and z_{mu}^{users} for each modality representing the group chosen for each movie-user pair. So z_{um}^{movie} represents the group that movie m picked for the rating given by user u . X_{mu} is the rating observed for user u and movie m . To reduce clutter m, u will be used instead of $movie, user$ in the superscript to indicate the modality ($\pi_u^u \equiv \pi_u^{user}$).

The graphical model for Bi-LDA is in Figure 6.1. To draw the Bayesian network for Bi-LDA in Figure 6.1. The corresponding figure using mixture model blocks is in Figure 6.2.

In the NMB framework we can describe the model as three interacting Dirichlet compound multinomial sub-models, figure 6.2. In Bi-LDA the Z^M variables are partitioned by the movie they are associated with, the Z^U variables are partitioned by the user they are associated with, and the ratings, X , are partitioned according by the movie and user group assigned to that rating by z_{um}^m and z_{mu}^u .

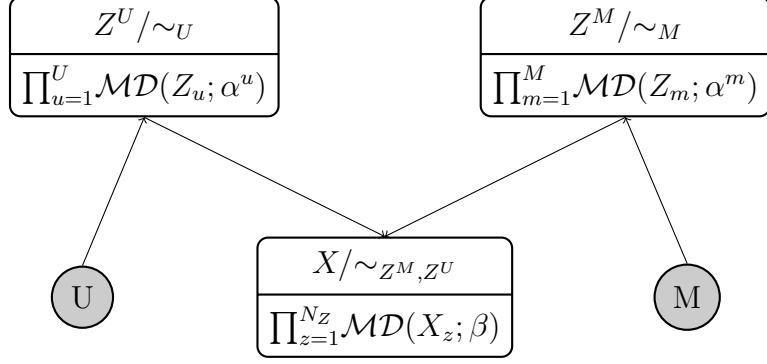


Figure 6.2: NMB for Bi-LDA. U is the number of users. M is the number of movies, N_Z is the number of movie groups times the number of user groups

The joint distribution described conventionally for the Bi-LDA model is:

$$\begin{aligned}
P(X, \mathbf{z}^m, \mathbf{z}^u, \Phi, \boldsymbol{\pi}^m, \boldsymbol{\pi}^u) &= \tag{6.1} \\
P(X|\mathbf{z}^m, \mathbf{z}^u, \Phi) P(\Phi|\boldsymbol{\beta}) P(\mathbf{z}^m|\boldsymbol{\pi}^m) \times \\
P(\mathbf{z}^u|\boldsymbol{\pi}^u) P(\boldsymbol{\pi}^m|\boldsymbol{\alpha}^m) P(\boldsymbol{\pi}^u|\boldsymbol{\alpha}^u) \\
&= \left[\prod_{m,u} \Phi_{z_{mu}^m, z_{mu}^u} [X_{mu}] \right] \times \left[\prod_{j,k} \text{Dirichlet}(\Phi_{jk}|\boldsymbol{\beta}) \right] \times \\
&\quad \left[\prod_{m,u} (\pi_m^m [z_{mu}^m] \pi_u^u [z_{mu}^u]) \right] \times \left[\prod_m \text{Dirichlet}(\boldsymbol{\pi}_m^m|\boldsymbol{\alpha}^m) \right] \times \\
&\quad \left[\prod_u \text{Dirichlet}(\boldsymbol{\pi}_u^u|\boldsymbol{\alpha}^u) \right]
\end{aligned}$$

Where bold variables represent the collection of individual variables, $\mathbf{z}^m \equiv \{z_{11}^m, z_{12}^m, \dots, z_{UM}^m\}$ (U is the number of users, and M is the number of movies). Now our goal is to perform inference on (6.1) to learn the posterior distribution of X and other variables of interest given the input data.

6.1.2 Bi-LDA Gibbs Sampling

Although exact inference is intractable in Bi-LDA as it is in LDA, we can derive an efficient collapsed Gibbs sampler in terms of mixture blocks. As in [14] in the mixture block objects we will analytically marginalize over the parameters of the multinomial distributions. For example, we will marginalize over the parameters π_u^u , for the user u , $P(z_u^u|\alpha_u^u) = \int d\pi_u^u P(z_u^u|\pi_u^u)P(\pi_u^u|\alpha_u^u)$. The basic idea is to analytically marginalize out all the conjugate distributions Φ, π^u, π^m in (6.1) and obtain an expression for the joint probability $P(X, \mathbf{z}^m, \mathbf{z}^u) = P(X|\mathbf{z}^m, \mathbf{z}^u)P(\mathbf{z}^m)P(\mathbf{z}^u)$, where each term on the RHS is maintained by a different mixture block.

6.1.3 Description of Inference

We will describe the inference procedure in two ways: first, in terms of Gibbs sampling using the conditional distributions of the variables and second, algorithmically in terms of operations on mixture block objects.

To begin with we will need the following counts to describe the Gibbs sampling process in detail:

$$N_{jkv} = \sum_{um} \mathbb{I}[X_{mu} = v] \mathbb{I}[z_{mu}^m = j] \mathbb{I}[z_{mu}^u = k] \quad (6.2)$$

$$N_{uk} = \sum_m \mathbb{I}[z_{um}^u = k] \quad (6.3)$$

$$N_{mj} = \sum_u \mathbb{I}[z_{um}^m = j] \quad (6.4)$$

In terms of the collaborative filtering model for movies and users, N_{jkv} represents the number of entries in the entire data-array that are assigned to user factor k and movie factor j , for which the rating has the value v . N_{uk} is the number of ratings for user

u assigned to factor k . N_{mj} is the number of ratings for movie m assigned to factor j . We will use a \cdot in place of a variable to indicate that we are taking the sum over the values of the variable (i.e. $N_{jk\cdot} = \sum_v N_{jkv}$). We will use the superscript $\neg(um)$ to denote that data-entry (um) is subtracted from the counts. In terms of these counts we find the following conditional distribution for movie indicator variables \mathbf{z}^m ,

$$\begin{aligned} P(z_{um}^m = j | \mathbf{z} \setminus z_{um}^m, X) &\propto P(z_{um}^m = j | \mathbf{z} \setminus z_{um}^m) P(x_{um} | X \setminus x_{um}, z_{um}^m = j, z_{um}^u = k) \\ &\propto \left(\frac{N_{jkv}^{\neg(um)} + \beta^v}{N_{jk}^{\neg(um)} + \beta} \right) \left(N_{mj}^{\neg(um)} + \frac{\alpha^m}{J} \right) \end{aligned} \quad (6.5)$$

Where J is the number of movie factors, $X_{um} = v$, $z_{um}^u = k$ and $\beta = \sum_v \beta^v$. The conditional distribution is the same for the user indicator variables with the role of user and movie reversed.

$$\begin{aligned} P(z_{um}^u = k | \mathbf{z} \setminus z_{um}^u, X) &\propto \\ &\left(\frac{N_{jk}^{v, \neg(um)} + \beta^v}{N_{jk}^{\neg(um)} + \beta} \right) \left(N_{uk}^{\neg(um)} + \frac{\alpha^u}{K} \right) \end{aligned} \quad (6.6)$$

Where K is the number of user factors, and $X_{um} = v$ and $z_{um}^m = j$. The Gibbs sampler thus cycles through the indicator variables $z_{um}^m, z_{um}^u \forall u, m$. Ratings are conditionally independent given Φ so we can marginalize out unobserved ratings from the model. The Gibbs sampler therefore only scans over the observed entries in the matrix X .

As an alternative, one may wish to sample the indicator variables for all modalities jointly. This improves mixing at the expense of increased computation. The equations are given by,

$$\begin{aligned} P(z_{um}^m = j, z_{um}^u = k | \mathbf{z}^{\neg(um)}, X) &\propto \\ &\left(\frac{N_{jk}^{v, \neg(um)} + \beta^v}{N_{jk}^{\neg(um)} + \beta} \right) \left(N_{mj}^{\neg(um)} + \frac{\alpha^m}{J} \right) \left(N_{uk}^{\neg(um)} + \frac{\alpha^u}{K} \right) \end{aligned} \quad (6.7)$$

6.1.4 Inference in Terms of Mixture Block Objects

In this section we assume that Bi-LDA is implemented in terms of a NMB as depicted in figure 6.2. There are three mixture blocks, one for each set of variables X, Z^M, Z^U . To specify the mixture block objects, we need the distribution family and the ρ and π functions which define the partition of the variables. All blocks use the same distribution family, Dirichlet compound multinomial. The functions ρ and π are defined as follows:

$$u_i = \text{user for rating } i$$

$$m_i = \text{movie for rating } i$$

$$\rho^{Z^U}(\pi(z_i^u)) = u_i$$

$$\rho^{Z^M}(\pi(z_i^m)) = m_i$$

$$\rho^X(\pi(x_i)) = \{z_i^u, z_i^m\}$$

Given this definition of the mixture block objects, collapsed Gibbs sampling for the model can be done by alternatively invoking function 5.6 for Z^U, Z^M . As a consequence of using the NMB framework, inference for the nonparametric version of the model (section 6.1.9) will be the same as for Bi-LDA except the distribution family for Z^U and Z^M will change.

6.1.5 Prediction

In a typical collaborative filtering scenario the product-consumer rating matrix is very sparse. In the movie user example, for each user the data will contain ratings for only a small fraction of the movies. One task is to estimate the ratings for movies the user has not seen. To make predictions, we will run the chain until it has converged and

then collect samples from the chain. Given a single sample from the chain for \mathbf{z} we start by calculating a mean estimate for $\Phi_{jk}, \pi_m^m, \pi_u^u$

$$\begin{aligned}\Phi_{jk}[v] &= \frac{N_{jk}^v + \beta^v}{N_{jk} + \beta} \\ \pi_m^m[j] &= \frac{N_{mj} + \alpha^m/J}{\sum_j N_{mj} + \alpha^m} \\ \pi_u^u[k] &= \frac{N_{uk} + \alpha^u/K}{\sum_k N_{uk} + \alpha^u}\end{aligned}\tag{6.8}$$

Then we calculate the expected value of X_{um}

$$E(X_{mu}) = \sum_{j,k} \left(\sum_v v \Phi_{jk}[v] \right) \pi_m^m[j] \pi_u^u[k].\tag{6.9}$$

To see the connection with matrix factorization define $\Phi_{jk} = \sum_v v \Phi_{jk}[v]$ as the core matrix if expected values for movie-user category pairs, $U = \boldsymbol{\pi}^u$ and $V = \boldsymbol{\pi}^m$. Then $E(X_{mu})$ is given by the following matrix multiplication

$$X \sim U \Phi V^t$$

It may also be useful to estimate how confident we should be in the predicted value X_{um} . The distribution of X_{um} is multinomial (6.10), so its variance can easily be calculated (formula not shown). Using the variance we could, for example, only make recommendation to a movie customer if the prediction distribution has a variance below a certain threshold.

$$P(X_{mu} = v | \Phi_{jk}, \pi_m^m, \pi_u^u) = \sum_{j,k} \Phi_{jk}[v] \pi_m^m[j] \pi_u^u[k]\tag{6.10}$$

In the previous calculations for the predicted value of X_{mu} , we used just a single sample. We would like to take many samples from the chain and use them to find estimates of $\Phi_{jk}, \pi_m^m, \pi_u^u$ marginalized over \mathbf{z} . However, there may be multiple equivalent modes of the distribution where the assignment variables \mathbf{z} have different values which would cause the average of equations (6.8) calculated over samples from different modes to be incorrect. In other words, there are multiple equivalent permutations for the group labels and the Gibbs sampler may mix between these artificial modes in the distribution. Instead we can marginalize over \mathbf{z} implicitly by averaging over predictions from many Gibbs iterations. Call Φ_s, π_s^m and π_s^u the mean estimates of Φ, π^m, π^u based on a single sample \mathbf{z}_s . We initialize $X = 0$. After scan number “s” through the data-matrix we update our average as follows,

$$\bar{X} \rightarrow \frac{s-1}{s}\bar{X} + \frac{1}{s}\Phi_s \prod_m \pi_s^m \tag{6.11}$$

where we suppressed summations to avoid clutter. We find this on-line averaging results in a significant improvement in prediction accuracy.

6.1.6 Multi-LDA.

In this section we extend the model to more than two modalities. Bi-LDA works well for the movie ratings example, but there are cases where more than two modalities make sense. For example, the release year of the movie may provide additional predictive information for ratings or we may be interested to learn if there was a shift in the movie groups users preferred over the years. Although it is not difficult to extend Bi-LDA to Multi-LDA, we introduced Bi-LDA first because it is the most common case and the description of Multi-LDA involves bookkeeping that clutters the description of the model.

In order to make the transition to Multi-LDA we need to replace the two modalities user-movie with a list of modalities 1..M. Thus instead of π^{user} and π^{movie} we have $\pi^m, m \in 1..M$. To index the observations associated with the combination of modalities we replace m, u with $i_1..i_M$. For example if i_1 is the user, i_2 is the movie, and i_3 is the rating date, then a rating is identified by $X_{i_1i_2i_3}$. In general the index for variables associated with a data entry of X becomes $i_1..i_M$, and a data entry is indexed by $X_{i_1..i_M}$. In Bi-LDA movie and user groups were indexed by j and k , in Multi-LDA j and k are replaced by $j_1..j_M$. For instance $z_{i_1..i_M}^1 = j_1$ tells us that for modality 1 and data item $i_1..i_M$ group j_1 is assigned. Also, $\Phi_{j_1..j_M}[v]$ is the probability of value v for the cluster identified by the factors $j_1..j_M$. Thus the equation for the Bi-LDA model 6.1 becomes the following for the Multi-LDA model

$$\begin{aligned}
P(X, \{\mathbf{z}^m\}, \Phi, \{\boldsymbol{\pi}^m\}) &= \tag{6.12} \\
P(X|\{\mathbf{z}^m\}, \Phi)P(\Phi|\boldsymbol{\beta}) \prod_m P(\mathbf{z}^m|\boldsymbol{\pi}^m) P(\boldsymbol{\pi}^m|\boldsymbol{\alpha}^m) \\
&= \left[\prod_{i_1..i_M} \Phi_{z_{i_1..i_M}^1..z_{i_1..i_M}^M} [X_{i_1..i_M}] \right] \times \\
&\quad \left[\prod_{j_1..j_M} \text{Dirichlet}(\Phi_{j_1..j_M}|\boldsymbol{\beta}) \right] \times \\
&\quad \left[\prod_m \prod_{i_m} \left(\prod_{i_1..i_M \setminus i_m} \pi_{i_m}[z_{i_1..i_M}^m] \right) \text{Dirichlet}(\boldsymbol{\pi}_{i_m}|\boldsymbol{\alpha}_m) \right]
\end{aligned}$$

Conversion from Bi-LDA to the Multi-LDA version of equations 6.2 6.3 6.4 6.13 6.6 6.7 6.11 6.9 is not difficult. First the conversion requires using the Multi-LDA indices $i_1..i_M$ and $j_1..j_M$ instead of u, m and j, k . Second the Multi-LDA version requires performing the multiplications and summations over all modalities $j_1..j_M$, instead of over just two modalities.

The conditional distribution for sampling \mathbf{z} becomes

$$P(z_{i_1..i_M}^m = j_m | \mathbf{z} \setminus z_{i_1..i_M}^m, X) \propto \left(\frac{N_{j_1..j_M}^{v, \neg(i_1..i_M)} + \beta^v}{N_{j_1..j_M}^{\neg(i_1..i_M)} + \beta} \right) \left(N_{j_1..j_M}^{\neg(i_1..i_M)} + \frac{\alpha^m}{J} \right) \quad (6.13)$$

where $X_{i_1..i_M} = v$ and $z_{i_1..i_M}^{m'} = j_{m'} \forall m' \setminus m$. The Bi-LDA equation for prediction 6.9 is replaced with:

$$E(X_{i_1..i_M}) = \sum_{j_1, \dots, j_M} \left[\left(\sum_v v P(x_{i_1..i_M} = v | \Phi_{j_1..j_M}[v]) \right) \prod_{m=1}^M \pi_{i_m}^m[j_m] \right] \quad (6.14)$$

Defining $\Phi = \sum_v v P(x = v | \Phi_{j_1..j_M})$ as the core, we see the relation to tensor factorization, just as Bi-LDA was related to matrix factorization.

6.1.7 Multi-LDA with Mixture Blocks

The extension from Bi-LDA to Multi-LDA using mixture block object is simple. For each modality there is a Dirichlet compound mixture block. The mixture blocks have the following ρ and π functions.

$$\begin{aligned} d_i^m &= \text{id in modality } m \text{ for rating } i \\ \rho^{Z^m}(\pi(z_i^m)) &= d_i^m \\ \rho^X(\pi(x_i)) &= \{z_i^1, \dots, z_i^M\} \end{aligned}$$

Collapsed Gibbs sampling for the model can be done by alternatively invoking function 5.6 for Z^1, \dots, Z^M

6.1.8 Bi-LDA Experiments

Netflix

In these experiments we evaluate Bi-LDA on the Netflix movie ratings dataset. The Netflix dataset consists of user-movie ratings provided by the Netflix corporation as part of a data-mining contest. The input data is movie-user rating pairs where the rating can take integer values 1 to 5. Consequently we have an array $X_{movie,user} \in 1, 2, 3, 4, 5$ for input into the model. We use the Netflix training dataset of approximately 100 million ratings for our training data. We used 10% of the probe dataset for our held out test data ¹. We use Root Mean Squared Error (RMSE) to measure the prediction performance on the test data. Using 50 user groups and 500 movie groups we ran the Gibbs sampler for 1600 epochs (an epoch is resampling all the variables once). The parameters $\alpha^m, \alpha^u, \beta$ were set to 1. The resulting RMSE on the test data set was .933. This compares to an RMSE of .947 from Netflix's own rating prediction system Cinematch.

One advantage of Bi-LDA over non probabilistic approaches to matrix factorization is that it provides a distribution for the predictions. In figure 6.3 we have a plot of the cumulative RMSE for all predictions less than the variance on the X axis. This plot indicates that the variance of the distribution for X_{mu} (6.10) is a good indicator of its accuracy. In fact if we look at figure 6.4 we can see that by ordering the predictions by variance, we can classify 90% of the data with an RMSE of .9 and 40% with an RMSE of .75. Ordering the prediction by variance would for example, enable a marketing department to effectively target the most accurately classified customers.

It is informative to look at the movie groups the Bi-LDA model learned for the Netflix

¹The probe dataset is provided with the Netflix dataset and specifies a subset of the data for testing. We removed the probe data from the original dataset so that it could not be used for training.

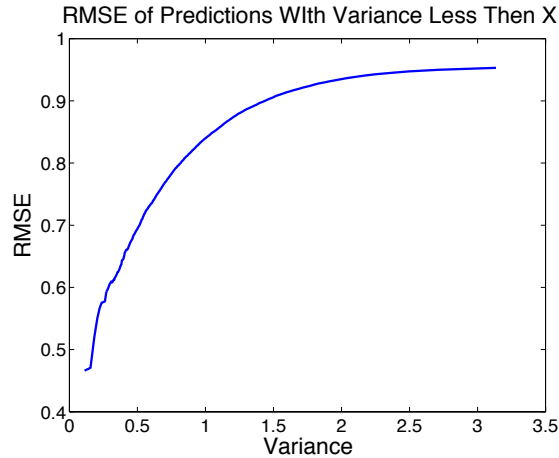


Figure 6.3: RMSE of Netflix prediction versus cumulative RMSE of prediction with calculated variance less than X.

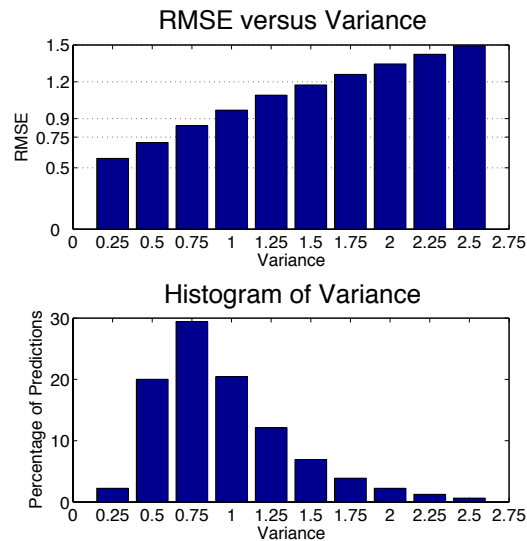


Figure 6.4: Top: The RMSE of predictions binned by their predicted variance. The first bar is predictions with variance less than .25, the second bar is the RMSE of predictions with variance between .25 and .5, and so on by increments of .25. Bottom: The percentage of predictions that fall within each bin.

dataset. Figure 6.5 illustrates the movie groups. For each group, we show the five movie titles that are responsible for the largest number of data points (with any rating) within that group. We chose four movie groups that were easy to interpret. Most other groups were also easily interpretable. We see that from only the ratings

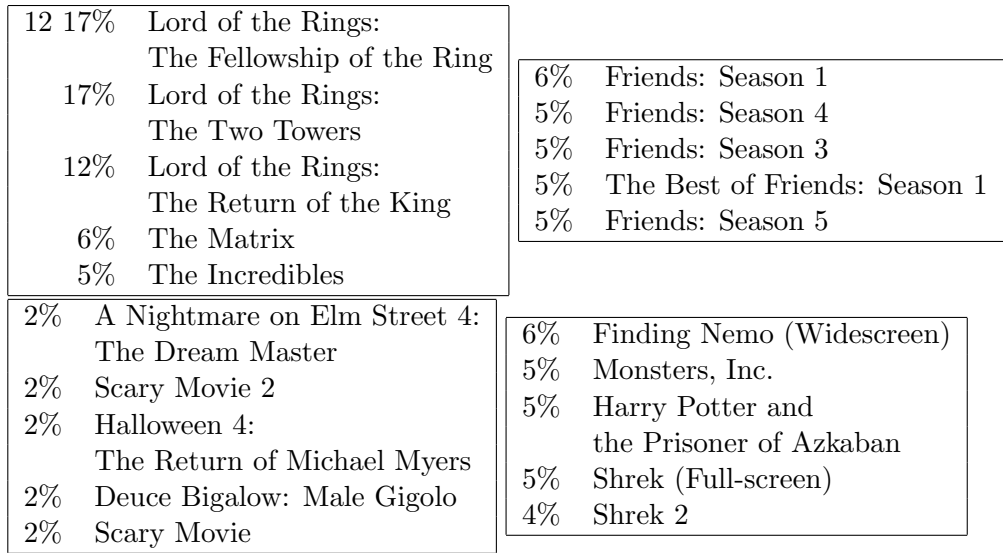


Figure 6.5: Bi-LDA model learned on the Netflix data. Each box corresponds to a movie group. Each box shows the titles of the five movies responsible for the most data points in each group and the percentage of data points in that group associated with the movie.

given by users Bi-LDA reveals the relationship between certain DVD titles. The fact that the various seasons from the “Friends” TV show are grouped together, and similarly for movies from the “Lord of The Rings” trilogy, validates that Bi-LDA infers a reasonable model for the data.

MNIST

In this section, we use the MNIST dataset [22] of handwritten digits to further investigate the structure revealed by Bi-LDA and compare it to the User Ratings Profile (URP) model [29]. The URP model can be shown to be a special case of the Bi-LDA model where we set the number of movie factors equal to the number of movies and fix the movie indicator variables such that each movie has its own factor, $z_m^m = m$.

MNIST consists of 28×28 grayscale bitmaps of handwritten digits (see figure 6.7) of which we used the first 4000 digits. The data was first converted to binary and then 50% of the pixels were removed at random to simulate missing data. Consequently

we have an array $X_{pixel,image} \in 0, 1$ as input to the model. We trained Bi-LDA with 25 pixel factors and 90 image factors and $\alpha^p = 1, \alpha^I = 1$. As previously noted URP is a special case of the Bi-LDA model, so to run Bi-LDA as URP we set the number of pixel factors to 784, the number of pixels. The URP model was trained using 25 image factors and $\alpha = 1$.

In figure 6.6 and 6.7 we show the reduced dimensionality representations learned by Bi-LDA and URP. For URP the image-groups, $\Phi_{pk}[1]$ are “holistic”, i.e. entire prototypes of digits (k is the image group). To reconstruct an image it will pick a single such prototype or combine very few of them. Hence, URP entertains a *sparse latent representation* on this dataset. Note also that URP did not use all the factors available to it. For Bi-LDA we observe that pixel-groups, $\pi_p^p[j]$ form strokes. Strokes are combined into digit-templates that represent the image groups, $\theta_{pk}^I = \sum_j \Phi_{jk}[1]\pi_p^p[j]$. θ_{pk}^I is the gray scale value of pixel p in image group k . If we plot the θ_{pk}^I for all the pixels in an image group, we get a graphic visualization of the image group, figure 6.6. Since image-groups are again entire digits the latent representation is sparse at the image-group level. However, since many strokes need to be combined into digits the latent representation is *distributed* at the pixel-group level (see figure 6.6). We conclude that Bi-LDA builds a hierarchical representation, from pixels to strokes, from strokes to digit prototypes and from prototypes to images.

6.1.9 Bi-HDP

One consideration when applying the Bi-LDA model to a new data set, is how to choose the number of groups for each modality. Not only does one need to consider the number of groups for each modality, one needs to consider how the number of groups in one modality effects another. For example, having many user groups and

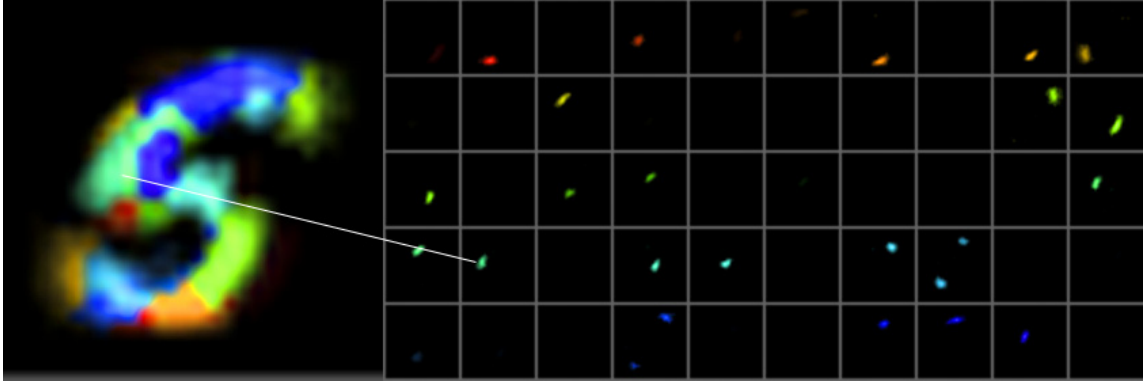


Figure 6.6: Prototype digit ‘5’ as a combination of strokes (best viewed in color). The Bi-LDA image prototype (image group) on the left. Strokes (pixel groups) that were active for this prototype on the right.

few movie groups will result in a different analysis of the data than if many movie groups and few user groups are used. As a consequence the space over which one wants to search the number of groups grows exponentially. Nonparametric Bayesian models offer an elegant solution, providing a prior over a countably infinite number of groups.

Using the movie ratings example, again we note that given z^{movie} , the user-branch of the Bi-LDA model is an LDA model. In other words, if we hold the assignment variables for the movie modality constant, inference in the user-branch is the same as inference in an LDA model. This observation suggests an easy procedure to take the infinite limit: replace each LDA branch with the nonparametric version of LDA, the Hierarchical Dirichlet Process (HDP) [43]. This change corresponds with replacing the Dirichlet compound multinomial mixture blocks with hierarchical Dirichlet process mixture blocks in the NMB representation.

In the finite model π^m is drawn from a Dirichlet distribution $\text{Dirichlet}(\alpha^m/J, \dots, \alpha^m/J)$. We might first try taking the limit as $J \rightarrow \infty$. However, if $z_{mu} = a$ for some movie

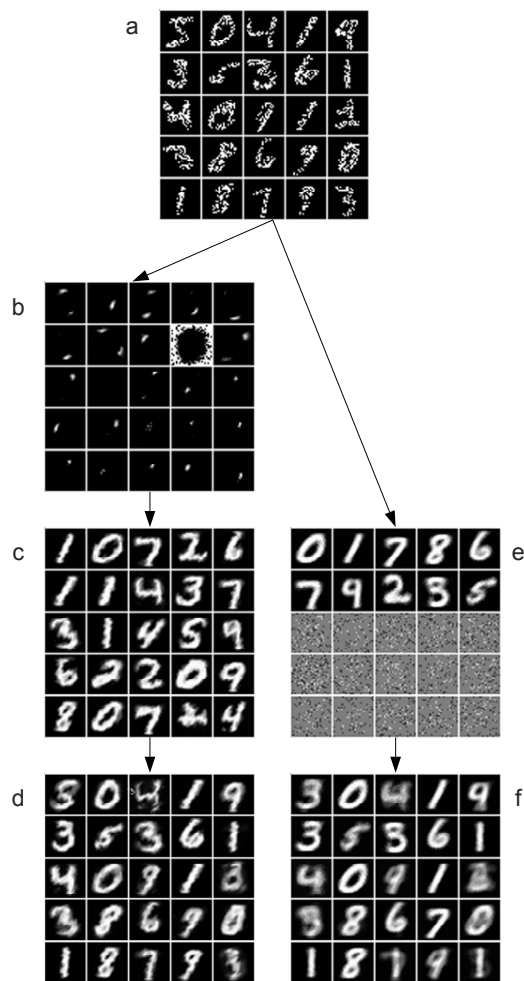


Figure 6.7: a) Original training data b) Bi-LDA strokes c) Bi-LDA image-groups d) Bi-LDA reconstruction e) URP-LDA image-groups f) URP-LDA reconstruction. The white frame in image b) is a pixel group for the black pixels which surround most digits

then the probability that another movie chooses the same group is

$$P(z_{m'u}^m = a | z^m, J, \alpha^m) = \frac{(N_{m'a}^{-(m'u)} + \alpha^m / J)}{J + \alpha^m} \rightarrow 0 \quad (6.15)$$

as $J \rightarrow \infty$

HDP models this by introducing a *root* pool of groups. Each movie then draws a distribution over groups, π_m^{movie} using the *root* distribution as a prior. Starting

with the finite version of this extended model with J movie groups and K user groups we replace the distribution for π^m, π^u in the Bi-LDA model with the following: $\tau^m \sim \text{Dirichlet}(\gamma/J, \dots, \gamma/J)$, $\pi_m^m \sim \text{Dirichlet}(\alpha^m \tau^m)$, $\tau^u \sim \text{Dirichlet}(\gamma/K, \dots, \gamma/K)$, $\pi_u^u \sim \text{Dirichlet}(\alpha^u \tau^u)$. The rest of the model remains the same as in Bi-LDA (section 6.1.1). The full generative model before taking the limit as $J, K \rightarrow \infty$ thus becomes,

1. Choose $J \times K$ distributions over ratings $\phi_{jk}^m \sim \text{Dirichlet}(\beta)$
2. Choose a *root* distribution for movies $\tau^m \sim \text{Dirichlet}(\gamma^m/J, \dots, \gamma^m/J)$
3. Choose a *root* distribution for users $\tau^u \sim \text{Dirichlet}(\gamma^u/K, \dots, \gamma^u/K)$
4. Choose a distribution over K user groups for each user $\pi_u^u \sim \text{Dirichlet}(\alpha^u \tau^u)$
5. Choose a distribution over J movie groups for each movie $\pi_m^m \sim \text{Dirichlet}(\alpha^m \tau^m)$
6. For each movie-user pair (mu)
 - (a) Choose a user group $z_{mu}^u \sim \text{Multinomial}(\pi_u^u)$
 - (b) Choose a movie group $z_{mu}^m \sim \text{Multinomial}(\pi_m^m)$
 - (c) Choose a rating $r_{mu} \sim p(r_{mu} | z_{mu}^u, z_{mu}^m, \phi_{z_{mu}^m, z_{mu}^u}^m)$

Next, we use the results from [43] to take the limit as $J, K \rightarrow \infty$ and get the nonparametric version of Bi-LDA, Bi-HDP. The graphical model for Bi-HDP is in figure 6.10. The result is that each movie can belong to any number of groups and the groups will be shared across different movies. Separate movies can share some, none, or all groups. The same is true for users.

Describing the graphical model for Multi-HDP can be a little bit complicated, but in terms mixture blocks it is relatively simple, figure 6.8 and 6.9. First, there is a mixture model for the ratings W . The variables W are partitioned as a function of Z^M, Z^U , or

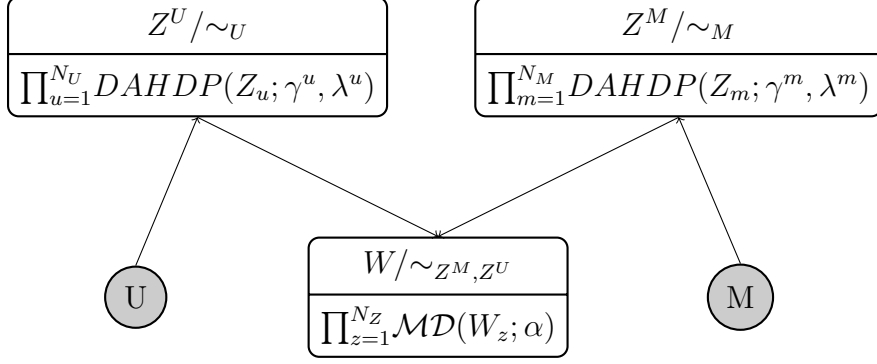


Figure 6.8: NMB for Multi-HDP using direct assignment HDP blocks

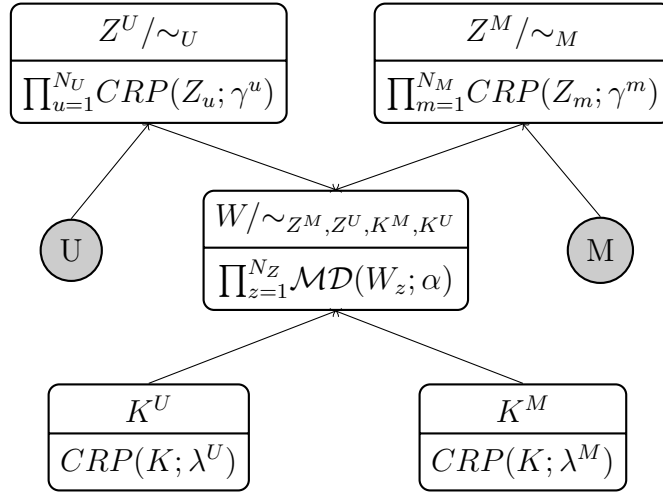


Figure 6.9: NMB for Multi-HDP using Hierarchy of CRP.

in other words w_{mu} is assigned to mixture group z_{mu}^U, z_{mu}^M . Second, Z^M is partitioned into N_m groups, one group for each movie, and Z^M is distributed according to a hierarchical Dirichlet process. One graphical representation of the NMB model for Multi-HDP where the HDP structure has been encapsulated in DAHDP blocks is in figure 6.8. Another version where the HDP structure has been built using CRP blocks is in figure 6.9.

For inference we use the direct assignment method of Gibbs sampling for a HDP distribution (i.e. the NMB model in figure 6.8). Unlike in Bi-LDA and Multi-LDA where all variables other than \mathbf{z}, X were marginalized over, we keep τ^u, τ^m .

The equations for the conditional probability of \mathbf{z} (the user and movie group assignments for a rating) are the following:

$$P(z_{um}^m = j | \mathbf{z} \setminus z_{um}^m, \boldsymbol{\tau}) \propto (\alpha_m \tau_j^m + N_{mj}^{-\text{(um)}}) \times \left(\frac{N_{jk}^{v, \text{(um)}} + \beta^v}{N_{jk}^{-\text{(um)}} + \beta} \right) \quad (6.16)$$

$$P(z_{um}^u = k | \mathbf{z} \setminus z_{um}^u, \boldsymbol{\tau}) \propto (\alpha_u \tau_k^u + N_{uk}^{-\text{(um)}}) \times \left(\frac{N_{jk}^{v, \text{(um)}} + \beta^v}{N_{jk}^{-\text{(um)}} + \beta} \right) \quad (6.17)$$

where N_{mj}, N_{uk}, N_{jk}^v are defined in (6.4),(6.3),(6.2). Again, as in section 6.1.1 assignment variables corresponding to unobserved entries are simply skipped in the Gibbs sampler. The key difference between these equations and the finite case (6.13), (6.6) is that τ^m and τ^u are distributions over $J+1$ and $K+1$ possible groups. If there currently exist K user groups, then $\alpha_u \tau_{(K+1)}^u$ is proportional to the accumulated probability of the infinite pool of “empty” clusters. The same holds true for the movies. In this way at every sampling step there is the possibility of choosing a new group from the countably infinite pool of empty groups. If the count for the number of ratings assigned to a group ever goes to zero, the group is returned to the pool of empty groups. In this way the Gibbs sampler samples over the number of groups or if we make the connection to matrix factorization again, the rank of the matrix decomposition. We must also sample τ^m, τ^u the details of which are in section 6.1.12.

Although with the transition to Bi-HDP we have eliminated the need to choose the number of user and movie groups, we still have the parameters $\alpha^m, \gamma^m, \alpha^u, \gamma^u$ to choose. Fortunately we can also sample these parameters using the auxiliary variable trick explained in [43]. Again we postpone the details of this to section 6.1.13. The Gibbs sampler thus alternates sampling the assignment variables $\{z^m, z^u\}$ with $\alpha^m, \alpha^u, \gamma^m, \gamma^u, \tau^m, \tau^u$. This is guaranteed to converge to the equilibrium distribution

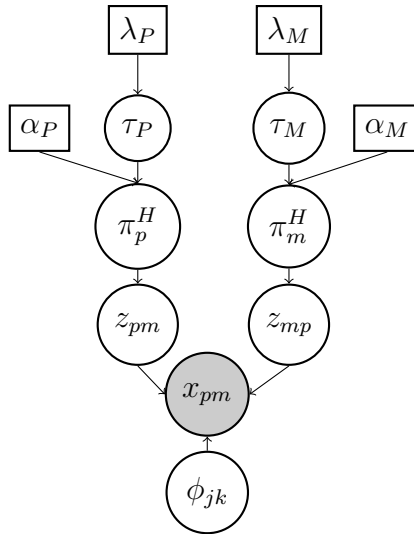


Figure 6.10: Graphical model for Bi-HDP

of the Bi-HDP model.

Finally, we can again extend Bi-HDP to Multi-HDP in the same way that we extended Bi-LDA to Multi-LDA.

6.1.10 Bi-HDP Experiments

NIPS papers

We use a corpus of papers from 13 NIPS conferences (<http://nips.cc/>), 1987 through 1999. NIPS conferences deal with topics that cover human and machine intelligence, including machine learning, computer vision, neuroscience, etc. We use the pre-processed version of the data available at <http://www.cs.utoronto.ca/roweis/nips>. In order to speed up experiments, we additionally reduce the vocabulary size to the 500 most frequently occurring words.

Two modalities are used to model this data. The first modality corresponded to

the year in which the paper was published. The second modality corresponded to the author of the paper. For each pair of year and author, all words from all papers written by that author in that year are used as *iid* data points. For multi-author papers, all words in the paper are used multiple times, once for each author. For a given pair of author and year groups, a multinomial distribution over the vocabulary is used to model the probability of each word (this is the standard bag-of-words representation). Thus, a pair of author and year groups specifies a topic or distribution over words.

A collapsed Gibbs sampler is used to fit Bi-HDP as described in section 6.1.9. After 3000 epochs the system converged to 204 author groups and 11 year groups.

Figures 6.11 and 6.12 illustrate the Bi-HDP model learned on NIPS papers. Figure 6.11 illustrates the year groups. For each year group we show the five years that contributed most to that group. In addition, we show the three most popular topics within that year group. For each topic, we show the 10 most popular words from that topic. As one might expect, mostly consecutive years are grouped together. In addition, it can be seen how the most popular topics shift from various aspects of neural networks for years 1989–1992 to various machine learning topics for years 1994–1999.

Figure 6.12 illustrates the author groups. For each author group, we show the 10 authors that contributed the most data points to that group. Additionally, we show the four most popular topics for that author group. For each topic, we show the 10 most popular words from that topic. The results show how Bi-HDP is able to find authors grouped by their research interests, with the topics in each group corresponding to the research interests of these authors.

Years in this year group: 1991 (34%), 1988 (22%), 1989 (20%), 1990(17%), 1992 (7%)

network training set units networks time figure pattern trained performance
weight weights learning back propagation unit generalization hidden shown gradient
system neural response order based high function research single theory

Years in this year group: 1997 (32%), 1998 (28%), 1999 (19%), 1996 (15%), 1994 (5%)

likelihood parameters data posterior mixture probability prior structure values space
distribution noise functions training approximation linear Gaussian figure generalization fixed
data test space results experiments pattern performance section machine optimization

Figure 6.11: Year groups on NIPS data. Two groups are shown. Top: years that contributed the most data points to the group shown. For each year the percentage of points accounted for in the year group is shown. Below, the three most popular topics for that year group are shown. For each topic the 10 most popular words are shown within a box.

Market basket data

In this experiment we investigate the performance of Bi-HDP on a synthetic market basket dataset. We used the IBM Almaden association and sequential patterns generator to create this dataset [1]. This is a standard synthetic transaction dataset generator often used by the association research community. The generated data consists of purchases from simulated groups of customers who have similar buying habits. Similarity of buying habits refers to the fact that customers within a group buy similar groups of items. For example, items like strawberries and cream are likely to be in the same item group and thus are likely to be purchased together in the same market basket. The following parameters were used to generate data for our experiments: 1M transactions, 10K customers, 1K different items, 4 items per transaction on average, 4 item groups per customer group on average, 50 market basket patterns, 50 customer patterns. Default values were used for the remaining parameters.

Two modalities were used to model this data. The first modality corresponded to customers, and the second modality corresponded to items. For a given pair of customer and item groups, a binomial distribution was used to model the probability of a customer group making a purchase from that item group.

A collapsed Gibbs sampler was used to fit the model. After 1000 epochs the system converged to 278 customer groups and 39 item groups.

Figure 6.13 illustrates the learned model. Within each table, the top row shows an item group learned by the model. For each item group, the items most frequently assigned to that group are shown. The bottom row shows the ground truth item group that most closely resembles the learned item group. (The ground truth data comes from the item groups used to generate the data.) Note that item groups were selected by how popular they are (the 10 most popular item groups are shown) rather than by how easy it was to interpret them in terms of ground truth data.

As can be seen, most item groups correspond directly to the hidden ground truth data. The conclusion is that the model can learn successfully the hidden structure in the data.

Next, we evaluate how well our Bi-HDP model can predict whether a particular customer will buy a particular item. The performance is measured in terms of RMSE (root mean squared error). We compare performance to Non-negative matrix Factorization (NMF) [23], which has been applied in the past for market basket analysis. Figure 6.14 compares the performance of our model to that of NMF. As can be seen, Bi-HDP achieves a significantly better value of RMSE compared to NMF.

6.1.11 Multi-LDA/Multi-HDP Discussion

We have introduced a novel model for nonparametric Bayesian tensor factorization. The model has several advantages. First, it provides a full distribution over predictions for missing data. In collaborative filtering experiments we show we can use the variance for a prediction’s distribution to reliably order the predictions by their accuracy. Having an estimate for the accuracy of predictions allows one to discard predictions below a certain threshold. Second, we show that because the model infers structure in multiple modalities concurrently, structure in the data is revealed that might not be apparent when considering only one modality of the data at a time. We also derive the nonparametric version of the model which infers the rank of the matrix/tensor decomposition and only requires a few hyper-parameters. Finally, we demonstrate that the model can find interesting structure in a variety of applications with multiple data modalities, including collaborative filtering, text mining, and market basket analysis.

Throughout our description of the model, we use a multinomial distribution, $\text{Multinomial}(\phi_{j_1..j_m})$, for a data value. However, the model works with other distributions with a conjugate prior, such as normal, with little modification. In the NMB formulation, it is simply a matter of changing the distribution family of the mixture block for the observed data items X .

6.1.12 Sampling Augmented Variables

Gibbs sampling for Bi-HDP requires sampling variables not present in the finite versions of the model. Given the assignment variables $\{z^m, z^u\}$ we sample τ^m, τ^u as follows. We only discuss this for the movie-branch, but the procedure is identical for

user-branch. First, we generate vectors

$$v_{tmj} = \frac{\alpha^m \tau_j^m}{t - 1 + \alpha^m \tau_j^m} \quad t = 1, \dots, N_{mj} \quad (6.18)$$

Then, we sample N_{mj} Bernoulli random variables $s_{tmj} \sim \text{Bern}[v_{tmj}]$ (with probability of success v_{tmj}) and compute $S_j = \sum_{tm} s_{tmj}$. Finally, we sample $\tau^m \sim \text{Dirichlet}(\gamma^m, [S_1, \dots, S_K])$ (note τ is a distribution over $K + 1$ states). In the language of the Dirichlet Process, this is equivalent to running a Chinese restaurant process with N_{mj} customers and strength parameter $\alpha^m \tau_j^m$ and counting how many tables become occupied (see [43] for details).

In the NMB framework, we assume a mixture block object has a function which samples variables that are not collapsed.

6.1.13 Sampling Hyper-parameters

For Bi-HDP we sample the model parameters instead of holding them fixed. The parameters $\alpha^m, \gamma^m, \alpha^u, \gamma^u$ are sampled using the auxiliary variable trick explained in [43]. We only write the equation for the movies-branch, but the procedure is identical

for the user branch

$$\begin{aligned}
P(\alpha^m | w, y) &= \\
&\text{Gamma} \left(a^m + S - \sum_i y_i; b^m - \sum_i \log(w_i) \right) \\
P(y_i | \alpha^m) &= \text{Bernoulli} \left(\frac{N_i}{\alpha^m + N_i} \right) \\
P(w_i | \alpha^m) &= \text{Beta} (\alpha^m + 1; N_i) \\
P(\gamma^m | w_0^m, y_0^m) &= \text{Gamma} (a_0^m + K - y_0^m, b_0^m - \log(w_0^m)) \\
P(w_0^m | \gamma^m) &= \text{Beta} (\gamma^m + 1, S) \\
P(y_0^m | \gamma^m) &= \text{Bernoulli} \left(\frac{S}{S + \gamma^m} \right)
\end{aligned} \tag{6.19}$$

where $S = \sum_j S_j$, $N_m = \sum_j N_{mj}$ and where w_i, w_0, y_i, y_0 are auxiliary variables and a^m, a_0^m, b^m, b_0^m are hyper-parameters.

6.2 Bayesian Matrix Factorization with Side Information and Dirichlet Process Mixtures

In section 6.1 we introduced a matrix factorization model for collaborative filtering inspired by topic models such as LDA. It can be viewed as a form of non-negative matrix factorization. In this section we will look at a model without the non-negative constraint that can easily incorporate known side information about users, movies, or ratings.

Unlike Multi-LDA and Multi-HDP, Bayesian matrix Factorization with Side Information (BMFSI) is not composed solely of mixture blocks. Instead of the data points being distributed according to a mixture model, each data point is separately dis-

tributed according to a density whose parameters are a function of latent factors. The latent factors are in turn distributed according to a mixture model and we therefore introduce mixture blocks for the latent factors. Thus BMFSI is an example of a hybrid model, where part of the full Bayesian model is factored into mixture block objects. We will first introduce the novel features of BMFSI independent of the mixture block framework and then introduce the “hybrid” NMB construction.

A drawback of standard matrix factorization algorithms is that they are susceptible to overfitting on the training data and require careful tuning of the regularization parameters and the number of optimization steps. Bayesian approaches to matrix factorization [38] [36] [7] attempt to address this weakness by integrating over model parameters and hyper-parameters, thus allowing complex models to be learned without requiring much parameter tuning.

Recent progress on the Netflix Prize [20] has shown that combining latent factor models with other models, such as neighborhood models, can improve performance. Although for the anonymized Netflix Prize data set there is not much additional information available for the customers, for many applications it is expected that additional side information about the customers or movies/products would be available and beneficial to incorporate into a model. Our approach significantly improves prediction performance by performing regression on additional side information while simultaneously learning the factorization.

We also introduce a Dirichlet process mixture of normal distributions as a prior for the matrix decomposition. Besides being a more flexible prior, we find that examining the posterior distribution of the mixtures reveals interesting latent structure in the data. We evaluate our approach on the Netflix Prize problem of predicting movie ratings and we find that our approach is able to outperform other Bayesian and non-Bayesian matrix factorization techniques.

The main contributions of our model are the following: (1) We describe a general scheme for seamlessly incorporating known information; (2) We introduce a fully Bayesian nonparametric model which can be learned via a scalable Gibbs sampling algorithm; (3) We apply our approach to the Netflix prize and are able to outperform many other factorization techniques without requiring tuning of parameters.

In the next section, we discuss related factorization approaches. We then describe our model in steps, first introducing the simpler related Bayesian probabilistic matrix factorization [38], then adding side information and Dirichlet process mixture extensions to the model. Next, we present experiments showing the efficacy of our approach on the Netflix Prize problem. Finally, we conclude with a discussion of the applicability of our approach to more general machine learning problems.

For ease of explanation we will use movie ratings as an example throughout the thesis and refer to movies and users. However, the model is general in form and not specialized for movie ratings or the Netflix competition.

6.2.1 Related work

Bayesian matrix factorization is a technique of growing interest. The work most closely related to our own is the Bayesian probabilistic matrix factorization (BPMF) model of [38] which features a Gaussian bi-linear factor model complete with Gaussian-Wishart priors. This model was applied to the Netflix problem and learned via Gibbs sampling. The Matchbox Bayesian recommendation system [40] is another bi-linear model featuring feedback and dynamics models and a similar mechanism to incorporate known information, with expectation propagation as the inference algorithm. Variational Bayesian factorization methods have also been applied to the Netflix problem [25].

On a more general level, nonparametric Bayesian factorization models such as those based on Indian Buffet Processes have been developed [11]. While these models adaptively adjust the inner dimensionality of the matrix factorization, our model is nonparametric in the sense that the number of underlying data clusters can increase. Thus these techniques are potentially complementary to our approach.

Finally, there exist numerous non-Bayesian matrix factorization techniques, including variants of singular value decomposition, and these techniques have been successfully applied to the Netflix problem [20] [41]. Later in the thesis, we will show that our approach is competitive to both Bayesian and non-Bayesian techniques.

First we will review BPMF, which is a special case of our model and will help to make our contributions clear. Next we will extend the model to include side information about movies, users or ratings. Finally, we introduce non-parametric prior distributions as mixture blocks over the latent feature vectors to complete the full model.

6.2.2 Bayesian probabilistic matrix factorization

Latent factor matrix factorization models for collaborative filtering assume that users and movies can be represented by vectors of latent factors U_i, V_j , where i and j designate the particular user and movie. Given the latent factor vectors for users and movies, a user's expected rating for a movie is predicted by the inner product of those vectors, $r_{ij} = U_i^T V_j$. In this way the matrix of all ratings R is factored into $U^T V = R$. The parameters of the model are learned given the sparsely observed ratings matrix R .

BPMF [38] puts matrix factorization in a Bayesian framework by assuming a genera-

tive probabilistic model for ratings with prior distributions over parameters. The full joint distribution of BPFM, $p(R, U, V, \Theta_u, \Theta_v | \sigma, \Theta_0)$ can be written as the product of the following conditional distributions (Figure 6.15):

$$p(R|U, V, \sigma) = \prod_i \prod_j [\mathcal{N}(R_{ij}; U_i^T V_j, \sigma)]^{I_{ij}}; \quad (6.20)$$

$$p(U|\Theta_u) = \prod_i \mathcal{N}(U_i | \mu_u, \Lambda_u); \quad p(V|\Theta_v) = \prod_j \mathcal{N}(V_j | \mu_v, \Lambda_v) \quad (6.21)$$

$$p(\Theta_\kappa | \Theta_0) = \mathcal{N}(\mu_\kappa | \mu_0, \Lambda_\kappa / \beta_0) \mathcal{W}(\Lambda_\kappa | \Lambda_0, \nu_0) \quad \kappa = u, v \quad (6.22)$$

where $\Theta_\kappa = \{\Lambda_\kappa, \mu_\kappa\}$, $\kappa = u, v$, $\Theta_0 = \{\Lambda_0, \nu_0, \mu_0, \beta_0\}$. In words, BPFM has the following generative process:

1. For each user i sample a vector of parameters $U_i \sim \mathcal{N}(U_i | \mu_u, \Lambda_u)$
2. For each movie j sample a vector of parameters $V_j \sim \mathcal{N}(V_j | \mu_v, \Lambda_v)$
3. For each movie j rated by user i sample a rating $r_{ij} \sim \mathcal{N}(R_{ij}; U_i^T V_j, \sigma)$

where the parameters of the multivariate normal distributions for parameter vectors, (Θ_u, Θ_v) , are given a normal-Wishart prior. The posterior predictive distribution of a rating r_{ij} is found by marginalizing over model parameters and hyperparameters:

$$p(r_{ij} | R^{-ij}, \Theta_0) = \iint p(r_{ij} | U_i, V_j) p(U, V | R^{-ij}, \Theta_u, \Theta_v) p(\Theta_u, \Theta_v | \Theta_0)$$

Since marginalizing over model parameters is analytically intractable, approximate inference using MCMC is performed.

6.2.3 Bayesian matrix factorization with side information

BPMF performs matrix factorization and prediction of new ratings based solely on the existing ratings. However, we would prefer a model that included extra information if it was available. For example, if the user has explicitly told us they have a preference for a certain type of movie we would want to incorporate that information into the model. As another motivating example outside movie recommendation, a bank may want to offer new products to its customers. However, besides monitoring the responses of costumers to products in the past (similar to ratings) it also has an enormous amount of side information about those customers to exploit.

Next we show how it is possible to extend BPMF to include side information about movies, users or ratings. In the extended version, BPMF with side information (BMFSI), instead of just generating a rating from the product of latent factor vectors $U_i^T V_j$ we augment U, V with additional terms that contain information about the movie, user or rating. The augmented version of V_j is now specific to a rating, $V_{ji}^p = \{V_j, X_{ji}^v\}$. V_j contains the free parameters that will be learned for movie j and X_{ji}^v contains additional side information about the rating that user i can regress against.

We still calculate the predicted mean of each rating by taking the inner product of the augmented vectors U_{ij}^p and V_{ji}^p . To understand the consequences of this change we further segment U_{ij}^p and V_{ji}^p into three parts and examine how each part plays a role in calculating r_{ij} . The parts are depicted in table 6.1, and described below.

- The mean estimate of a rating is determined by the sum-product of the parts of the vectors U_{ij}^{pT} and V_{ji}^p

$$\mu_{ij} = U_{ai}^T V_{aj} + U_{bi}^T X_{ji}^v + X_{ij}^{uT} V_{bj}$$

- The first term, $U_{ai}^T V_{ai}$, is the matrix factorization term. If this is the only term, the model is BPFM.
- The second term, $U_{bi}^T X_{ji}^v$ is the result of user i 's linear regression against the features of the movies they have rated or features of the rating itself. For example, if X_{ij}^v contains a flag indicating whether or not it is an action movie then the corresponding variable in U_{bi}^T indicates the users bias towards action movies. X_{ji}^v can also contain ratings specific information, such as, the date of the rating. In this case the corresponding variable in U_{bi}^T indicates the users trend in ratings over time.
- The third term, $X_{ij}^{uT} V_{bj}$, is the complement to the second term and is the result of the movies linear regression against features of the user or the rating specific information. For example, just as in the second term, X_{ij}^{uT} could contain the date of the rating. The corresponding variable in the movies vector V_{bj} indicates how the movies ratings have trended over time. X_{ij}^{uT} could also contain information about the user who made the rating, such as whether or not they are married. The movie can then learn a bias about how married users rate the movie.

The model is very flexible as to what features are put into X_{ij}^u and X_{ji}^v . The features can be user specific, movie specific, or rating specific. The only limitation is that they will be linearly combined in the final prediction. X_{ij}^u and X_{ji}^v can be the same size, or different sizes. The feature vectors can be symmetric or different for users and movies.

The only change in the model specification is that U^p and V^p now replace U, V in the

U_{ij}^p	U_{ai}	U_{bi}	X_{ij}^u
V_{ji}^p	V_{aj}	X_{ji}^v	V_{bj}

Table 6.1: The rating for user i of movie j , r_{ij} , comes from the product of the augmented feature vectors U_{ij}^p, V_{ji}^p . In this table the feature vectors are arranged so that the sections of vectors that are multiplied together are adjacent to each other.

analogous BPFM equation 6.20, to give the following distribution for R

$$p(R|U^p, V^p, \sigma) = \prod_i \prod_j \left[\mathcal{N}(r_{ij} | U_i^{pT} V_j^p, \sigma) \right]^{I_{ij}} \quad (6.23)$$

The prior distributions for U_i, V_j remain the same as in BPFM equations 6.21 and 6.21 and are not replaced by U_i^p, V_j^p . However, the conditional distribution of $p(U_i | \Theta_u, V^p, R)$ is different. In order to find the posterior distribution of a rating r_{ij} we again want to marginalize over the model parameters and hyperparameters:

$$p(r_{ij} | R^{-ij}, X, \Theta_0) = \iint p(r_{ij} | U_i^p, V_j^p) p(U, V | R^{-ij}, \Theta_u, \Theta_v) p(\Theta_u, \Theta_v | \Theta_0)$$

6.2.4 Inference

Although the posterior distribution of r_{ij} is intractable as it is for BPFM, we can still derive an efficient Gibbs sampler to calculate an approximation. The Gibbs sampler works by cycling through the latent variables U, V and the parameters Θ_u, Θ_v , sampling each conditioned on the current values of all the other variables. The use of conjugate priors provides us with a convenient form for the conditional distributions of the latent variables. The details are provided in section 6.2.8

6.2.5 BMFSI with Dirichlet process mixture prior

The BMFSI model assumes that every user and movie draws their vector of free parameters, U_i and V_j from a single common multivariate normal distribution with a full covariance matrix. However, we expect that there are clusters of movies or users that are more similar to each other than to the population in general. Consequently, a better generative model might be one where there are groups of users or movies and they draw their vector of latent factors from group specific distributions. In the generative process for this new model instead of drawing a factor vector from a single common distribution, each user or movie first picks a group and then draws a vector from that group's distribution. So a user might first pick an action film group and then draw a vector of factors from the action group's distribution. To summarize, the model would have the following generative process:

1. For each user i sample a group assignment $z_i^u \sim \pi_u$
2. For each movie j sample a group assignment $z_j^v \sim \pi_v$
3. For each user i sample a vector of parameters $U_i \sim \mathcal{N}(U_i | \mu_{z_i^u}, \Lambda_{z_i^u})$
4. For each movie j sample a vector of parameters $V_j \sim \mathcal{N}(V_j | \mu_{z_j^v}, \Lambda_{z_j^v})$
5. For each movie j rated by user i sample a rating $r_{ij} \sim \mathcal{N}(r_{ij}; U_i^{pT} V_j^p, \sigma)$

Since a priori we have no knowledge of the number of groups, we would like to use a non-parametric distribution that does not require us to specify the number of groups. To this end we use a Dirichlet process mixture [3] [10] to model the user and movie latent feature vectors. The Dirichlet process mixture model has support for a countably infinite number of mixture components but only a few will dominate in the posterior, providing us with a convenient non-parametric distribution. We will

again want to marginalize over the parameters to find the posterior distribution of the rating predictions r_{ij} . Conditioned on the group assignment variables the Gibbs sampling algorithm is the same as the one for BMFSI, with the exception that there are per group Θ_{ug}, Θ_{vg} parameters to sample. Conditioned on the sampled value for U, V , sampling of the z^u, z^v is according to a Dirichlet process mixture model with U, V acting as data vectors. We use Algorithm 2 in [33]. Details of the sampling algorithm are provided in section 6.2.9.

We would like to build the model using just mixture blocks, but we can not because R is not distributed according to a mixture model. However, we can still build a hybrid NMB. By hybrid we mean that the model is composed of normal mixture block objects and modified mixture block objects specific to this model (see figure 6.17). Mixture blocks for Z^u and Z^v are normal mixture blocks but the mixture blocks for U and V are specific to the model. U and V must be specific to this model because their interface with their children is not through a partition. Special mixture blocks are built which partition the latent factors U, V based on the assignment $\rho(\pi(U_i)) = z_i^u$ as in a normal mixture block. The details of the resulting sampling algorithm for inference using the hybrid NMB model is in section 6.2.9.

Building a hybrid NMB may seem like a waste of effort, but there is still an advantage because it allows one to easily experiment with alternative models. For example if one knew there were distinct groups of users, one could use an HDP mixture block for Z^u instead of a DP mixture block. In the resulting model, each separate group of users would be distributed according to its own DP, but the groups would still share “topics”.

If we look at the samples of assignment variables from the posterior distribution of the model when applied to the Netflix prize data set we find support for our expectation that there are multiple distinct groups. In particular we find clusters of movies that

are easily recognized as having common characteristics. We analyzed the assignment variables z from one sample of the Gibbs sampler after 200 iterations to inspect the clusters. For each of the 14 clusters found, we picked the top 5 largest movies by number of ratings and looked up their titles. The results from a qualitatively representative sample of the clusters are found in table 6.3. The clusters of movies found by the model contain groups of movies for which we would expect the distribution of latent factors to be similar. There are also clusters of users, but they are more difficult to interpret because we lack labels.

6.2.6 Experiments

We evaluate our approach on the well-known Netflix Prize competition, an ideal collaborative filtering problem with a well-defined system for objectively comparing different solutions. The Netflix Prize data set is a sparse user-movie matrix of over 100,000,000 ratings where there are 480,000 users and 17,700 movies. Each rating is an integer from 1 to 5, and the date when the user rated the movie is also provided.

The objective of the competition is to predict ratings for a held-out portion of the data matrix, known as the Quiz set. Netflix also identifies a Probe set, a subset of the ratings in the given data matrix, which was generated using the same process that generated the Quiz set; thus, the Probe set is useful for internal evaluation. The evaluation metric used is the root mean squared error (RMSE) between the predicted ratings and the actual held-out ratings. To perform internal RMSE calculations for our experiments, we create a new Probe set consisting of 10% of the original Probe set, and we train on the rest of the given data (including the other 90% of the original Probe set). We have also submitted our Quiz set predictions to the Netflix web site in order to obtain Quiz set RMSEs.

We regress against the following side information in our experiments: the date (normalized from 0 to 1), a date flag (which indicates whether the rating’s date was before March 12, 2004, where a change in the average rating was observed), the users previous two ratings, and if available, the rating the user gave the K most similar movies measured by Pearson’s correlation coefficients. We typically set $K = 5$. We also tried using other movie metadata which we extracted from Wikipedia (e.g. movie director, actors, languages) but we found that these Wikipedia-extracted features do not improve performance measured by RMSE.

The parameters that need to be manually set in our model are the number of user and movie dimensions D_u, D_v, σ, α and Θ_0 parameters (when $D_u = D_v$ we just use D to specify both). For all our experiments we set $\sigma = 0.8$. Λ_0 is set to the identity for all runs, $\mu_0 = 0$, $\beta_0 = .8$ and $\alpha = .01$. For runs with side information and DP mixtures and $\alpha = .0000001$ for DP mixture runs without side information.

We ran our collapsed Gibbs samplers on a heterogeneous collection of machines, ranging from dual-core machines with 8GB RAM to 16-core machines with 128GB RAM. Parallelization across cores was achieved through OpenMP.

Since our approach is based on Gibbs sampling, it is important to average over many different samples in order to achieve good results. Figure 6.18(a) shows the Probe RMSEs for individual samples as well as the online average of samples during the sampler burn-in period, for various D . While each individual sample gives a fairly high RMSE (e.g. 0.93), averaging these samples (as one would usually do for MCMC techniques) gives a substantially better RMSE (e.g. 0.89). In Figure 6.18(b), we run the sampler starting from the burn-in position and show the effect of averaging over multiple samples. As the number of samples approaches one hundred, there is little improvement from adding more samples.

D	Probe RMSE	Quiz RMSE
45	0.8970	–
45 (DP)	0.8957	0.8988
45 (SI)	0.8866	0.8909
45 (DP, SI)	0.8865	0.8907

Table 6.2: RMSEs for the model with/without side information (SI) and the Dirichlet process (DP).

The Twilight Zone: Vol. 16	Star Trek II: The Wrath of Khan
The Twilight Zone: Vol. 22	Star Trek: Nemesis
The Twilight Zone: Vol. 2	Star Trek: First Contact
The Twilight Zone: Vol. 25	Planet of the Apes
The Twilight Zone: Vol. 1	Star Trek: Insurrection
The Sopranos: Season 1	Indiana Jones and the Last Crusade
The Sopranos: Season 2	The Matrix
The Sopranos: Season 3	Raiders of the Lost Ark
South Park: Bigger, Longer and Uncut	Harry Potter and the Chamber of Secrets
The Sopranos: Season 4	The Matrix: Reloaded

Table 6.3: Largest movies by number of ratings for four different movie clusters.

We investigate the benefits of incorporating additional side information as well as activating the full nonparametric model. We perform runs with all combinations of w/wo Dirichlet process (DP) or w/wo side information (SI). As shown in table 6.2, a significant improvement is achieved when side information is included.² In Figure 6.19, we examine the mean and variance of the coefficients (diagonal of Λ_u, Λ_v) for the side information dimensions and the collaborative filtering dimensions. In particular, the large coefficients in user dimensions 40-44 correspond to the five nearest neighbors. The spike in the user variance plot corresponds with the date side information, indicating the amount users ratings depend on date, varies significantly.

Our results in table 6.2 suggest that the multi group model with Dirichlet process mixtures only marginally improves upon the single group model with side information for the Netflix prize data set. While this RMSE gain is insignificant for this data set,

²To ensure a fair comparison, whenever we include side information dimensions, we remove the equivalent number of free dimensions

Method	Quiz RMSE	% Improvement
Cinematch Baseline	0.9514	0%
Variational Bayes [25]	0.9141	3.73%
Matchbox [40]	0.9100	4.14%
BPMF, D=60 [38]	0.8989	5.25%
BPMF, D=300 [38]	0.8954	5.60%
SVD++, D=50 [20]	0.8952	5.62%
SVD++, D=200 [20]	0.8911	6.03%
BRISMF D=250 [41]	0.8954	5.60%
BRISMF, D=1000 [41]	0.8904	6.10%
BMFSI (our model), D=45	0.8907	6.07%
BMFSI (our model), D=100	0.8875	6.39%

Table 6.4: Comparison between our model and other factorization techniques

there are other benefits to having a Dirichlet process mixture model, such as the creation of clusters which are interpretable. After performing a hierarchical D=45 run, the sampler was able to find interesting movie clusters: a “Star Trek” cluster, an action movie cluster, a “Lord of the Rings” cluster, among many others. Table 6.3 shows the movie titles for four different clusters.

In summary, Table 6.4 shows the best RMSEs for competing factorization approaches. We find that our approach generally outperforms both Bayesian and non-Bayesian factorization techniques. We note that integrated models which combine matrix factorization with neighborhood models or time models can generally perform better than the factorization techniques below [20]. In addition, bagging results over many different methods can yield significantly better RMSEs, as evidenced by the Netflix Leaderboard. However, we restrict our attention to matrix factorization and find that our approach is competitive with the state-of-the-art in matrix factorization.

6.2.7 BMFSI Discussion

In experiments we have shown our approach achieves better accuracy than other Bayesian and non-Bayesian factorization techniques. A benefit to our approach is that it can seamlessly handle additional information within the framework of a Bayesian factorization model, without requiring the tuning of many different parameters. Our results suggest that additional side information can act as a useful informative prior that can significantly improve results.

The addition of Dirichlet process mixtures to the model provides a more flexible prior for the latent feature vectors, but more importantly it discovers interpretable latent structure in the data. When used in combination with the side information it can provide a novel means of exploring groups in the data. For example, if the users are made up of married and un-married people with different preferences, then the model will likely form at least two groups for married and un-married users. Based on the mean latent factors of these groups, μ_{ug} , we could then examine group wide preferences for movies (or products) by plotting the inner-product of μ_{ug} with all the individual product vectors $\forall_j V_j$. When you combine the ability to discover groups with the ability to add side information, further possibilities are enabled. For example, using the married/un-married example, we could add other covariates, such as income, and see how this effects the groups discovered by the model.

Possible extensions of our approach include using nonparametric techniques for adjusting the inner dimensionality D in tandem with our nonparametric approach over user clusters. Another facet of our approach that can be improved is the selection of side information to include. Perhaps techniques such as Indian Buffet Processes can be used to infer binary features on a small subset of the data, and these inferred features can then be used to automatically select which side information features to

incorporate.

6.2.8 BMFSI Gibbs sampling algorithm

The Gibbs sampler works by cycling through the latent variables $U_i, V_j, \Theta_u, \Theta_v$, sampling each conditioned on the current values of all the other variables. The following are the necessary conditional distributions:

The conditional distribution of U_i (and equivalently V_j) is:

$$U_i := (U_{ai}, U_{ab}) \quad W_{ij}^v := (V_{aj}, X_{ij}^v)$$

$$p(U_i | V^p, R, \Theta_u) \propto p(U_i | \Theta_u) \prod_{ij} \left[N(R; U_i^{pT} V_j^p, \sigma) \right]^{I_{ij}} \quad (6.24)$$

$$p(U_i | V^p, R, \Theta_u) = N(U_i | \mu_n, \Lambda_n)$$

$$\Lambda_n^{-1} = \frac{1}{\sigma^2} \sum_j W_{ij}^v W_{ij}^{vT} + \Lambda_u^{-1}$$

$$\mu_n = \Lambda_n [\Lambda_u^{-1} \mu_u + \frac{1}{\sigma^2} \sum_j (R_{ij} W_{ij}^v - W_{ij}^v [X_{ij}^{uT} V_{bj}])] \quad (6.25)$$

The conditional distribution over the user hyperparameters Θ_u (and equivalently Θ_v) is independent of ratings and only depends on the user factor vectors U . The conditional distribution is simply the posterior distribution of a multivariate normal

model with a Normal-Wishart prior and observations U .

$$p(\mu_u, \Lambda_u | U, \Theta_0) = \mathcal{N}(\mu_u | \mu_k, \Lambda_u / \beta_k) \mathcal{W}^{-1}(\Lambda_u | \Sigma_k, \nu_k) \quad (6.26)$$

$$\mu_k = \frac{\beta_0}{\beta_0 + N_u} \mu_0 + \frac{N_u}{\beta_0 + N_u} \bar{U}$$

$$\beta_k = \beta_0 + N_u$$

$$\nu_k = \nu_0 + N_u \quad (6.27)$$

$$\Sigma_k = \Lambda_0 + S + \frac{\beta_0 N_u}{\beta_0 + N_u} (\bar{U} - \mu_0)(\bar{U} - \mu_0)^T$$

$$S = \sum_{i=1}^{N_u} (U_i - \bar{U})(U_i - \bar{U})^T \quad (6.28)$$

where N_u is the number of users.

BMFSI Gibbs sampling algorithm

1. Initialize model parameters U, V, X^u, X^v
2. for $t=1, \dots, T$
 - (a) Sample the hyperparameters according to equation 6.26
 - $\Theta_u^t \sim p(\Theta_u | U, \Theta_0)$
 - $\Theta_v^t \sim p(\Theta_v | V, \Theta_0)$
 - (b) Foreach user i sample U_i in parallel according to equation 6.24
 - $U^{t+1} \sim p(U_i | V^p, R, \Theta_u)$
 - (c) Foreach movie j sample V_j in parallel according to equation 6.24 (swapping the role of U and V)
 - $V^{t+1} \sim p(V_j | U^p, R, \Theta_v)$

6.2.9 Gibbs sampling BMFSI with Dirichlet process mixture sampling algorithm

With the addition of the Dirichlet process mixtures, the sampling algorithm is the same as in sec 6.2.8, with the exception that during each iteration there are per group parameters to sample, and z^u, z^v must be sampled. Θ_{ug}, Θ_{vg} , are still sampled according to (6.26,6.27,6.28) with the modification that instead of depending on U_i for all users, only U_i for users assigned to group g , (i.e. $z_i^u = g$).

z^u, z^v are sampled according to the following:

$$\begin{aligned}
 p(z_i^u = g | z_{-i}^u, U_i, \Theta_u) &= b \frac{N_{ug}^{-i}}{N_u - 1 + \alpha} \mathcal{N}(U_i | \Theta_{ug}) \\
 p(z_i^u \neq z_j^u \text{ for all } j \neq i | z_{-i}^u, U_i, \Theta_u) &= b \frac{\alpha}{N_u - 1 + \alpha} \times \\
 &\int \mathcal{N}(U_i | \Theta_u^*) \mathcal{N}(\mu_u^* | \mu_0, \Lambda_v / \beta_0) \mathcal{W}^{-1}(\Lambda_v^* | \Lambda_0, \nu_0) d\Theta_u^*
 \end{aligned} \tag{6.29}$$

b is the normalizing factor that makes the probabilities sum to one, separately for each equation. N_u is the number of users, N_{ug} is the number of users assigned to group g and $N_{ug} = \sum_i [z_i^u = g]$. $\Theta_u^* = \mu_u^*, \Lambda_v^*$.

Authors in this author group: Tresp, V., Williams, C., Bishop, C., Hinton, G., Jordan, M., Smyth, P., Ghahramani, Z., Bengio, Y., Barber, D., Tenenbaum, J.

likelihood parameters data posterior mixture probability prior structure values space
data distribution model set variables algorithm em log methods maximum
Bayesian Gaussian distribution matrix approach distributions log estimation variable estimate
model models density Bayesian mixture probabilities variance procedure Gaussian distributions

Authors in this author group: Koch, C., Sejnowski, T., Bower, J., Wilson, M., Pouget, A., Li, Z., Mel, B., Kammen, D., Baird, B., Zemel, R.

cells visual model orientation cortex stimulus cortical cell center processing
response spatial model responses neurons frequency receptive stimuli complex field
visual figure spatial temporal left receptive responses computation cortex pattern
phase figure networks models excitatory synaptic cells spatial simulations cell

Authors in this author group: Smola, A., Scholkopf, B., Vapnik, V., Simard, P., Denker, J., Platt, J., Burges, C., Shawe-Taylor, J., Muller, K., LeCun, Y.

data test space results experiments pattern performance section machine optimization
training set data space points classification solution optimal size parameter
set training learning function test performance values algorithms procedure shown
set training vector feature space vectors support problem method error

Authors in this author group: Sejnowski, T., Oja, E., Hyvarinen, A., Bell, A., Viola, P., Moody, J., Bialek, W., Hoyer, P., Schraudolph, N., Mayhew, J.

independent component components algorithm processing results matrix based basis statistical
system neural response order based high function research single theory
neural analysis rule form function data rules found figure make
order method algorithms analysis signal obtained set figure show single

Figure 6.12: Author groups on NIPS data. Four groups are shown. Top: authors that contributed the most data points to the group shown. Below, the four most popular topics for that author group are shown. For each topic, the 10 most popular words are shown within a box.

Learned	223, 619 , 271, 448, 39, 390
True	223, 271, 448, 39, 427, 677
Learned	364, 250 , 718, 952, 326, 802
True	364, 718, 952, 326, 542, 98
Learned	159, 563, 780, 995, 103, 216, 598 , 72
True	159, 563, 780, 995, 103, 216, 542, 72
Learned	227, 130, 862, 991, 904, 213
True	227, 130, 862, 991, 904, 213
Learned	953, 175, 956, 385, 269, 14, 64
True	953, 175, 956, 385, 269, 14, 956
Learned	49, 657, 906, 604, 229
True	49, 657, 906, 604, 229
Learned	295, 129, 662, 922, 705, 210
True	295, 129, 662, 922, 705, 68
Learned	886, 460, 471, 933, 544
True	886, 460, 471, 933, 917
Learned	489, 818, 927, 378, 64, 710
True	489, 818, 927, 378, 64, 247
Learned	776, 224, 139, 379
True	776, 224, 139, 379

Figure 6.13: Item groups learned by the Bi-HDP model compared to ground truth item groups. Each table corresponds to a single learned item group. The 10 most popular item groups learned by Bi-HDP are shown. In each table, the top row shows the most popular items for that item group, in order of decreasing popularity. The bottom row shows the ground truth item group corresponding to the learned item group. The items in a ground truth group have no associated weight; therefore, they were ordered to facilitate comparison with the top row. Non-matching items are shown in boldface. As can be seen, in most cases a learned item group corresponds accurately to a true item group.

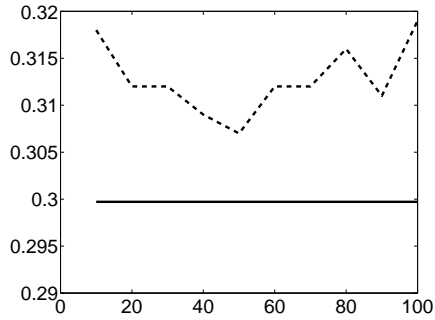


Figure 6.14: Performance of Bi-HDP and NMF on the market basket data. Y axis: RMSE (smaller values are better). X axis: the number of factors for NMF. Dashed curve: NMF. Solid curve: Bi-HDP. Since Bi-HDP determines the number of factors automatically, a horizontal curve is shown.

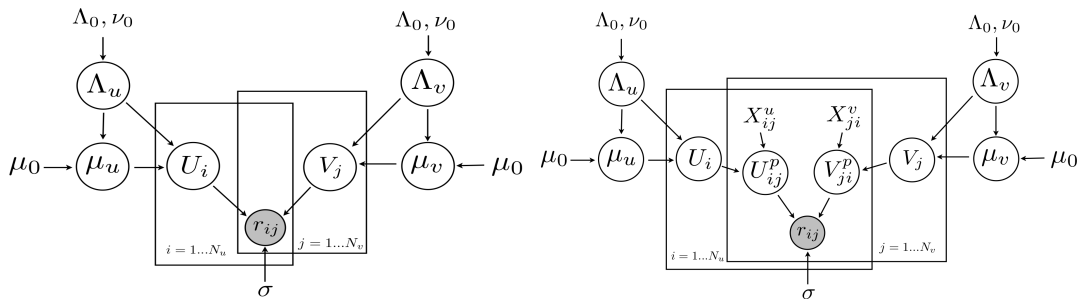


Figure 6.15: LEFT: Graphical model for Bayesian probabilistic matrix factorization (BPMF). RIGHT: Graphical model for Bayesian matrix factorization with side information (BMFSI).

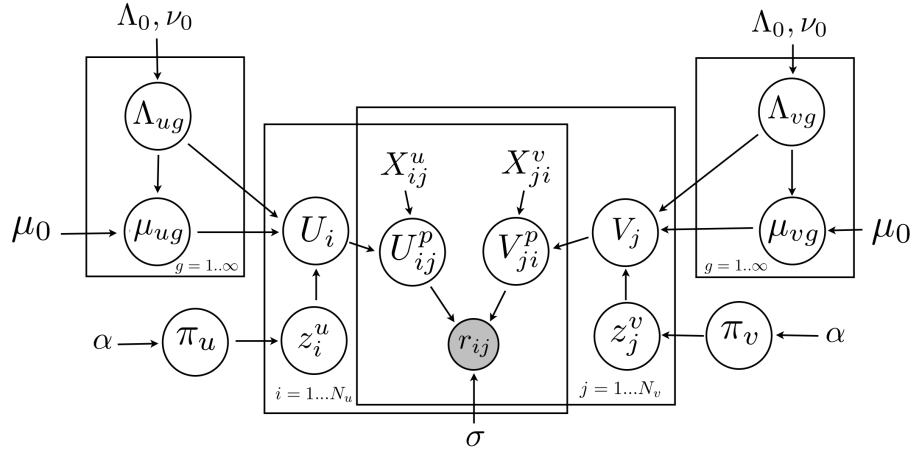


Figure 6.16: BMFSI with Dirichlet process mixture

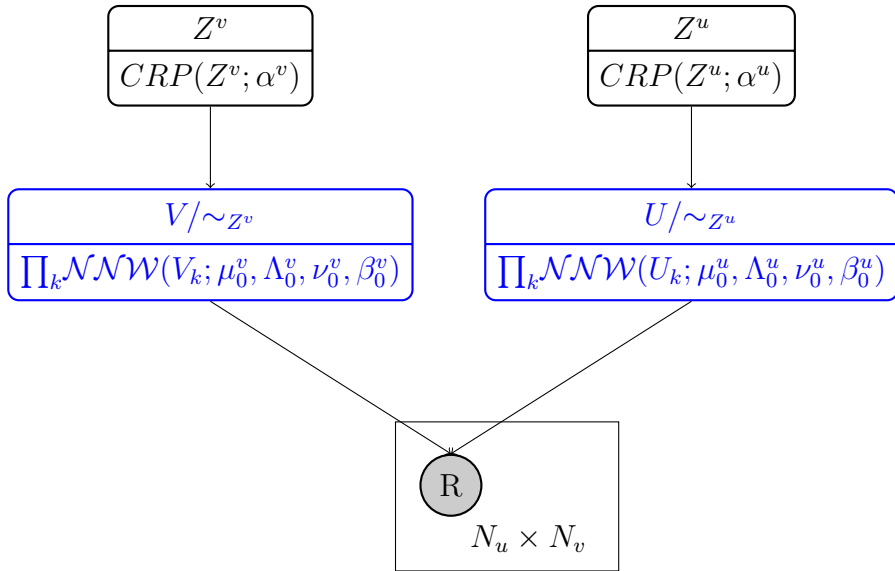


Figure 6.17: BMFSI with DP as NMB. The U and V mixture blocks are hybrid mixture objects because their child R is not a mixture block.

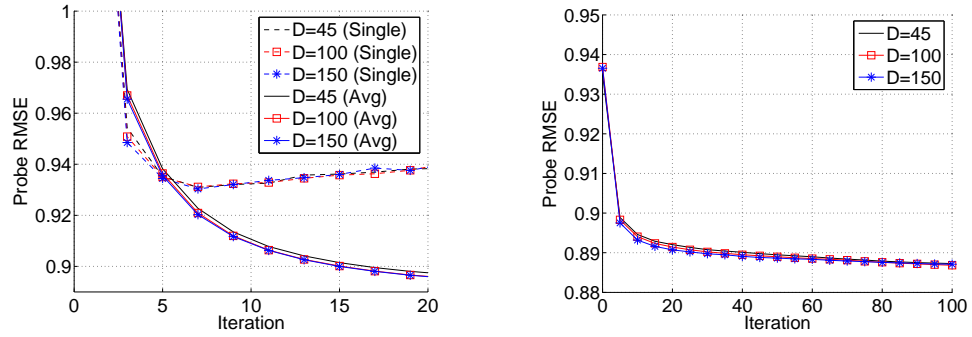


Figure 6.18: (a) Left: Probe RMSEs during burn-in, for various D . (b) Right: Probe RMSE after burn-in, with online averaging of samples.

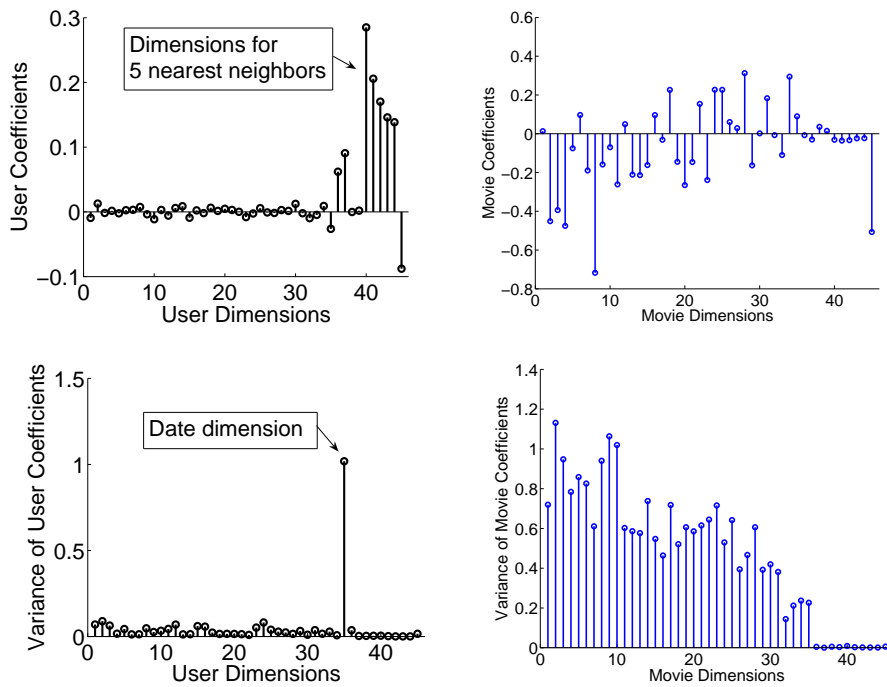


Figure 6.19: Coefficients learned on 45D run.

Chapter 7

Flexible Nonparametric Priors

Because the interface between blocks in the NMB framework is a partition, given a new distribution over partitions we could simply create a new mixture block for that distribution and use it as a replacement in NMB models. Consequently, it is natural to ask what other distributions we could use to form partitions besides multinomial and Dirichlet process. We will first consider some desirable properties of a distribution over partitions.

1. An efficient inference procedure. In particular if we plan to build a mixture block, we want to be able to perform MCMC sampling over possible partitions.
2. Support for all possible partitions of n data-items.
3. *Exchangeability*: The probability distribution is invariant under permuting the labels of the data-items.

The DP, and more generally the Pitman Yor Process (PYP), satisfy these conditions [26, 33], [34, 12], and have proven to be a very useful model for applications related

to clustering and density estimation. The first two properties are clearly desirable in the context of the NMB framework. Additionally, for many cases there is no prior knowledge that depends on the label of a data item, so exchangeability is also a desirable property. However, there is one more condition that PYP satisfies which is more constraining.

4. *Consistency (Heritability)*: The probability distribution for n data-items is the same as the distribution for $N > n$ data items and subsequently marginalizing down to n data-items.

Consistency requires that the probability over partitions of n items is identical to first specifying the probability over $N > n$ items and then marginalizing out an arbitrary subset of size $N - n$. Taking the limit $N \rightarrow \infty$ this implies for instance that all partitions/groups will grow arbitrary large as we sample more and more items from our prior. However, it's easy to imagine cases where a more appropriate model is an arbitrarily large number of finite sized groups instead of finite number of arbitrarily large groups. In order to consider the former family of models we must reconsider the process of marginalization.

If we marginalize by randomly picking items to delete we are forced to consider only groups that contain an infinite number of items in the $N = \infty$ limit. However, if we allow ourselves a different model for marginalization, for instance by first picking a group and then deleting all members of that group, even as $N \rightarrow \infty$ we can be consistent yet have groups of finite size. To distinguish the two notions we will speak of “*consistency under structured marginalization*”.

To explain the DP a species sampling metaphor is often employed. For the DP, we expect that we discover members of a certain species at a rate which matches their frequencies in the world. We can also use an animal sampling metaphor to explain

our model which relaxes consistency to consistency under structured marginalization. However, instead of sampling species, we will consider sampling the pack-label of animals. Unlike the species sampling process, when sampling packs one is likely to observe every member of a pack simultaneously. For example, one could sample animals by flying a airplane over an area and taking photographs. In this example structured marginalization would delete one photograph at a time. Each time a photograph is deleted, it's likely that all the members of a pack captured in that photograph are removed from the data. Hence, if we are interested in partitioning animals by their pack-labels rather than their species-labels, structured marginalization is more appropriate.

Analogously we also believe that it is not appropriate to assume that a document is a random sample of words from an infinitely long document. It is more appropriate to think of a document as a self-contained entity and in particular that we have seen all the words that were written on a particular topic.

We propose to use a family of flexible priors (FP) [45] which allows for this more general view of consistency. In order to guarantee exchangeability FP works with variables $\mathbf{m} = \{m_i\}$ which correspond to the number of clusters of size i , i.e. $\sum_{i=1}^N im_i = N$ and $\sum_{i=1}^N m_i = K$ with K the total number of clusters. Flexible priors are log-linear models over features which can only depend on \mathbf{m} , i.e. the log-probability is proportional to $\sum_i \lambda_i^n \phi_i(\mathbf{m})$. Note that the parameters λ_i^n depend on the the number of data-items n . We will learn the parameters λ_i^n in a couple of ways. In the first approach we will express our prior belief that a fraction q_i of the data are in groups of size i as an expected average statistic $\mathbb{E}[m_i] = nq_i/i$. We can then use this prior information to learn the parameters λ_i^n . In the second approach we will learn λ_i^n directly from the data.

Although learning the parameters for a dataset of size n works fine for applications where the size of the dataset does not change, such as clustering, it is problematic when we are trying to make predictions for data not in the original set. For example, what prior for $N = n + M$ data items would be consistent with the choice of prior for n . If we know beforehand that the largest number of data-items will be $N = n + M$, we can deal with this issue by just learning parameters $\{\lambda_i^N\}$, for $n+M$ items and marginalizing out M items to get samples for our training set of size n . This marginalization should follow the *structured marginalization* process appropriate for the problem at hand. (Random deletion of M points would correspond to the ordinary marginalization assumed for the DP.) One can then use the samples at level n to compute average sufficient statistics and pre-learn the parameters $\{\lambda_i^n\}$ that will be consistent with the parameters $\{\lambda_i^N\}$ at level N . Alternatively, if we are learning hyper-parameters from data directly, we can add M unobserved data items to the dataset. Although we would like to identify an entire hierarchy of consistent priors from $n = 1 : \infty$, it seems hard to parametrize these families analytically (or even prove they exist for a given marginalization procedure). Our approach is thus to work with partially consistent hierarchies over levels. It is important to appreciate that inference in FPs can be done efficient as for DPs through Gibbs sampling or related methods.

To demonstrate how flexible priors fit in the NMB frameworks we will incorporate FP into two NMB models:

1. The basic mixture model, with a Z mixture block which determines the partition and a X mixture block which represents the distribution over observed data.
2. A hierarchical ‘‘HDP’’ model where we replace a hierarchy of DP mixture blocks with a hierarchy of FP mixture blocks.

7.1 FLEXIBLE PRIORS OVER PARTITIONS

As previously noted, the Dirichlet process can be thought of as a prior distribution over all possible partitions of N data-items [37]. One derivation of the DP prior over partitions is achieved by taking the infinite limit of a finite mixture model [13] resulting in,

$$p(g) = \frac{\Gamma(\alpha)\alpha^K}{\Gamma(N + \alpha)} \prod_{i=1}^N \Gamma(i)^{m_i} \propto e^{\sum_{i=1}^N (\log \alpha + \log \Gamma(i)) m_i} \quad (7.1)$$

with the additional constraints that $K = \sum_i m_i$ and $N = \sum_i i m_i$. $\Gamma(\cdot)$ is the Gamma function. The second term expresses the DP prior on partitions as a maximum entropy distribution with parameters $\lambda_i = \log \alpha + \log \Gamma(i)$. Note that the variables $\{m_i\}$ are not independent due to the constraint $\sum_i i m_i = N$.

Distribution 7.1 is completely specified by the variables $\{m_i\}$ which represent the number of clusters with exactly i data-items in them. Consequently, the probability does not change if we exchange the labels of either the clusters or the data-items and many partitions will have the same probability. For example, if we express the partition in terms of the size of the subsets, i.e. as the partition of an integer, then $\{2, 1, 1\}$ has the same probability as $\{1, 2, 1\}$. So that we can talk about the set of partitions that have the same probability we will use the ordered group sizes as a signature, e.g. $\{2, 1, 1\}$. Signatures are exactly specified by the variables $\{m_i\}$ and they represent the maximal invariants under the permutation group [12]. One can count the number of partitions in a signature to be $N! / \prod_{i=1}^N (i!)^{m_i} m_i!$ [3]. Multiplying equation 7.1 with this number results in the Ewens sampling distribution.

By introducing general features $\phi_a(\mathbf{m})$ which are linearly combined in a log-linear model (or maximum entropy model), we generalize the DP expression equation 7.1

to the FP expression.

$$\text{FP}(g|N, \boldsymbol{\lambda}) \propto e^{\sum_a \lambda_a \phi_a(\mathbf{m})} \mathbb{I}\left(\sum_i i m_i = N\right) \quad (7.2)$$

Note that equation 7.1 is a special case of equation 7.2 with $\phi_i(\mathbf{m}) = m_i$ and $\lambda_i = \log(\alpha) + \log \Gamma(i)$. One can also show that the PYP is special case where again $\phi_i(\mathbf{m}) = m_i$ but $\lambda_i = \log \Gamma(i - \gamma) - \log \Gamma(1 - \gamma)$ plus one additional nonlinear feature $\phi_0(\mathbf{m}) = \sum_{k=0}^{K(\mathbf{m})-1} \log(\alpha + k\gamma)$ with $K(\mathbf{m}) = \sum_i m_i$, $\lambda_0 = 1$, $\gamma \in [0, 1]$ and $\alpha > -\gamma$.

7.2 Learning Flexible Priors

As previously noted the first approach to learning the parameters of FP, λ_i^n , uses an expected average statistic $\mathbb{E}[m_i] = nq_i/i$ to express our belief that a fraction q_i of the data is in groups of size i .

$$\mathbb{E}[\phi_a(\mathbf{m})] \approx c_a \quad a = 1..F \quad (7.3)$$

where $\phi_i(\mathbf{m}) = m_i$ and we have the constraint that $\sum_i i \mathbb{E}[m_i] = N$. Not all choices of c_a are possible due to the constraint on \mathbf{m} . The FP formulation allows one to choose features $\phi_i(\mathbf{m})$ other than m_i , but in this work we restrict ourselves to m_i .

There is a danger that due to the discrete nature of \mathbf{m} and the constraint $\sum_i i m_i = N$ that the specified constraints can not be met exactly. To solve this problem one could state the constraints using a potential function [9] allowing small violations.

If we have an efficient sampler for FP, then we can learn the parameters by running this sampler and interrupting it at regular intervals to change the parameters $\boldsymbol{\lambda}$ as

follows

$$\lambda_a^{\text{new}} = \lambda_a^{\text{old}} + \eta(c_a - \mathbb{E}[\phi_a(\mathbf{m})]_{\text{FP}} - \alpha\lambda_a^{\text{old}}) \quad (7.4)$$

where η is a learning rate and the average is computed as a sample average from the Gibbs sampler. In theory this method works as long as the rate of change in the parameters is slower than the rate of convergence of the sampler. This algorithm is theoretically analyzed in [47] and empirically tested in [44].

Fortunately, collapsed Gibbs sampling for the FP in equation 7.2 is easy. We will use the parameters $Z = \{z_1, \dots, z_n\}$ to represent the partition, where $z_i = k$ indicates that item i is in group k . Sampling the z -variables instead of the m -variables ensures that the constraint $\sum_i im_i = N$ is satisfied. Analogous to the collapsed Gibbs sampler [26], we sample one assignment variable at a time. In each Gibbs step we evaluate the probability of assigning z_i to any of the existing groups or a new group, $p(z_i = k | Z_{-i}) \propto \text{FP}(z_i = k, z_{-i})$, which can be conveniently expressed as follows. Let a be the size of the cluster to which item i is currently assigned and b be the current size of the cluster to which item i will be assigned; then

$$\text{FP}(z_n = k, z_{-n}) \propto e^{-\lambda_a + \lambda_{a-1} - \lambda_b + \lambda_{b+1}} \quad (7.5)$$

The second approach to learning the variables λ_i^n is to take an empirical Bayesian point of view and learn hyper-parameters directly from data. In this case one obviously runs a greater risk of over-fitting. The procedure we use is based on contrastive divergence [16]. We sample assignment variables Z from the posterior. At regular intervals we interrupt the sampler, record Z and continue sampling with the likelihood terms removed (i.e. we sample directly from the prior but initialize at the last iteration from the posterior). After just a few steps of sampling from the prior the

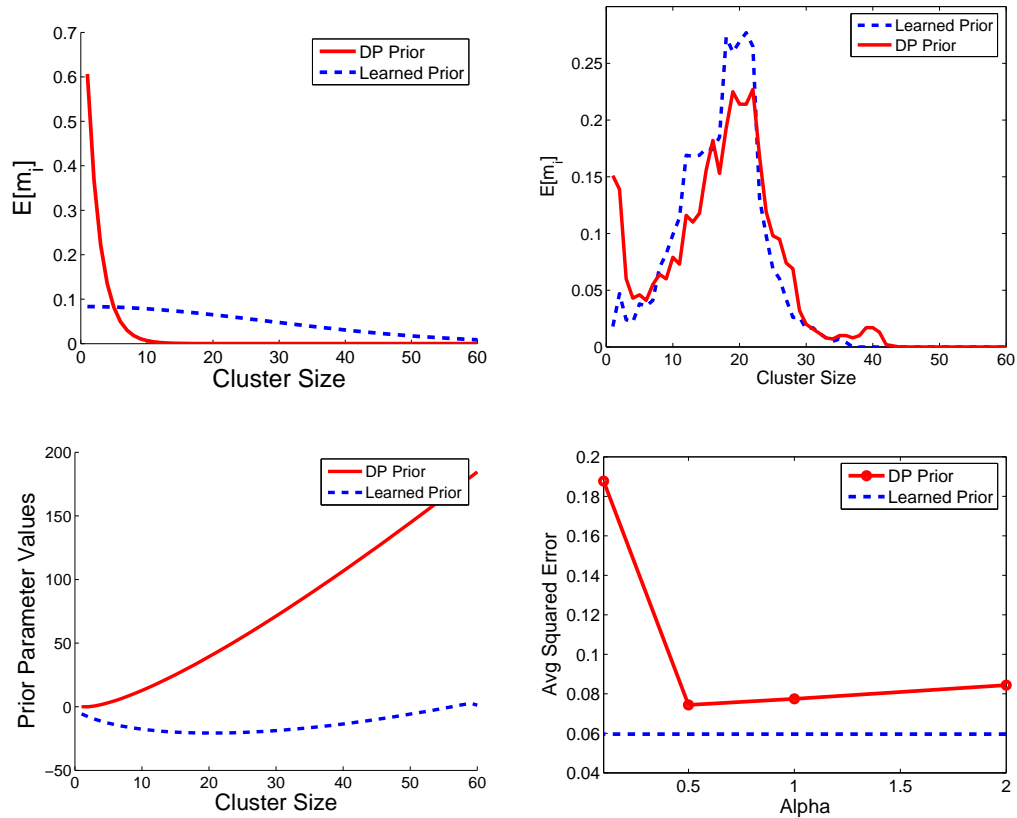


Figure 7.1: (a) Prior distribution over \mathbf{m} for DP prior (solid) and learned prior (dashed). (b) Posterior distribution over \mathbf{m} for both models. Note the peak at very small clusters for the DP. (c) Parameter values for λ for both models. (d) Error in association between datapoints as a function of α . Note that even for optimal α performance is worse than for the learned prior because small clusters have been suppressed.

samples will start to diverge from the posterior. We then use the difference to perform a learning update on the hyper-parameters, replacing c_a in Equation 7.4 with the posterior statistics. This will move the prior towards the posterior. After a learning update we recall the last posterior sample and continue sampling from the posterior.

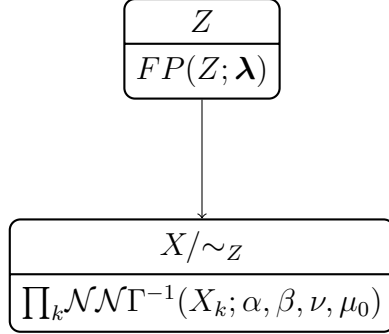


Figure 7.2: We replace the CRP mixture block in the NMB for the Dirichlet process mixture model with a FP mixture block.

7.3 ILLUSTRATIVE EXAMPLE

In order to demonstrate learning and applying flexible priors we compare it to the Dirichlet process on a simple clustering problem. One potential drawback of the Dirichlet process is that it can result in many small clusters. Therefore, we learn and then apply a flexible prior that discourages small clusters relative to the Dirichlet process.

We specify a new mixture block which uses the FP distribution over Z . The NMB model is in figure 7.2. First, we specify our prior knowledge through the variables \mathbf{m} . In particular, we specify the values for $\mathbb{E}[\mathbf{m}]$ using a truncated normal distribution with $\mu = 0$ and $\sigma = 40$. Relative to the $\mathbb{E}[\mathbf{m}]$ in the Dirichlet process, our distribution for \mathbf{m} puts less strength on the small clusters and more strength on the large clusters (Figure 7.3-a). Next, using just the FP mixture block, we learn the parameters λ using the Gibbs sampler described in Section 7.4. The learned parameters and the parameters for the Dirichlet process are plotted in Figure 7.3-c.

To compare clustering results, we generate a total of 60 points from three normal distributions, 20 points each, in 2 dimensions. All clusters have $\sigma=1$ and means $\mu_1 = [3, 3], \mu_2 = [9, 3], \mu_3 = [6, 6]$. We then cluster these points using a normal-

Wishart prior, keeping all hyper parameters the same but varying the prior over partitions. First, four runs of the Gibbs sampler are done with the Dirichlet process prior and four different settings of the α parameter [2,1,.5,.1]. Next, one run is made with the learned prior. For each run, 1000 iterations are used for burn-in and another 1000 iterations are used to calculate the sample mean.

To compare the results we examine two quantities. First, we examine the posterior $\mathbb{E}[\mathbf{m}]$ for the best run using Dirichlet process prior ($\alpha = .5$) versus the run using the learned prior. Figure 7.3-b shows that (as expected) the Dirichlet process generates more small clusters than the learned prior. Next, we examine the clustering performance by comparing the average squared difference between the true association between each pair of points and the sampled association. Where the true association between a pair of points equals 1 if they were generated by the same cluster and the sampled association equals the proportion of samples where the pair of points are assigned to the same cluster. Figure 7.3-d shows the association error using the Dirichlet process with four different settings for α versus the association error using the learned prior. When α is small (e.g. $\alpha = .1$) the error is large because the Dirichlet prior encourages only two clusters. As α increases, the Dirichlet process results in more than two clusters, but there are often small clusters sampled with only a few points. Consequently, getting the best results from the Dirichlet process requires careful tuning of the α parameter. It should be noted that if there are enough data, or the data are well separated, the Dirichlet process prior has little effect versus the learned prior on the posterior results.

7.4 HIERARCHICAL FLEXIBLE PRIORS

For clarity we will describe the hierarchical flexible priors (HFP) model in terms of words, documents, and topics. However, in general a word is an observed data item, documents are sets of words known to be grouped together and topics are partitions of words that the model infers.

The HDP [43] extends the DP as a prior for jointly partitioning multiple structured objects (e.g. documents, images) in such a way that words of different documents may be assigned to the same global topic, i.e. “topics are shared across documents”. This means that the HDP can be considered as a prior on partitions for all words jointly, but one which is exchangeable only under document-label permutations and permutations of words within documents, but not under permutations of words between documents. We have seen that the HDP can be defined in terms of CRP mixture blocks shown in figure 2.11 (left). We will now describe the analogous HFP model by replacing the CRP mixture blocks (figure 2.11) with FP mixture blocks (figure 7.3).

We will call the groups of words in each document document-topics. We want these document-topics to be grouped into larger entities which we call super-topics. The set $Z = \{z_{11}, \dots, z_{N_d, D}\}$ of variables will be used to indicate the assignment of a word to a document-topic, i.e. z_{id} is the assignment of word i in document d to a document-topic. The set $R = \{r_{11}, \dots, r_{N_d, D}\}$ of variables will be used to indicate the assignment of a document-topic to a super-topic. That is, r_{kd} indicates the super-topic that document-topic k in document d is assigned.

In terms of NMB we have an FP mixture block for $Z = \{Z_1, \dots, Z_k\}$, where Z has been partitioned according to observed document assignments D (i.e. $\rho^Z(\pi^Z(z_{id})) = d_{id}$). We have another FP mixture block for R . Finally, we have a \mathcal{MD} mixture block for the words X , where X is partitioned as a function of both Z and R . The

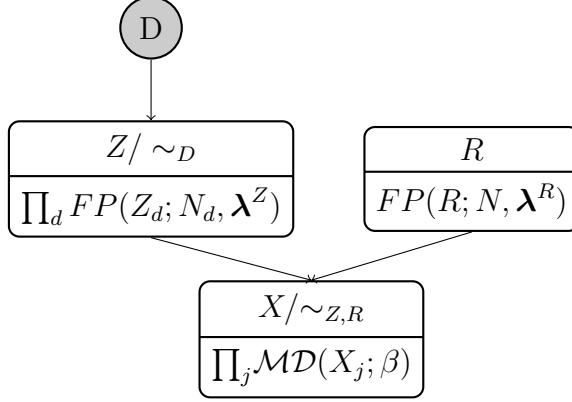


Figure 7.3: Two FP mixture blocks are used to define the assignment of X variables to super-topics in HFP.

parents $\pi^X(x_{id}) = \{z_{id}, R\}$, and the equivalence classes $\rho^X(\{z_{id}, R\}) = r_{z_{id},d}$. In terms of mixture blocks the definition is straightforward. Furthermore, if we have mixture block objects that implement the FP family of priors then a Gibbs sampling implementation can be done in terms of the interface described in section 5.3. Basically, we ask the R and Z mixture block objects to sample these variables.

We will now delve further into the details of the model that are hidden by the description in terms of mixture blocks. The generative process for the model is:

1. For each document d draw a partition of words into document-topics

$$Z_d \sim \text{FP}_d(\boldsymbol{\lambda}_d^{N_d})$$

2. draw a partition of document-topics into super-topics

$$R \sim \text{FP}_0(\boldsymbol{\lambda}_0^N)$$

3. Partition the words into super-topics

$$\text{super-topic for word } i \text{ in document } d = R_{z_{id},d}$$

4. For each super-topic, k , choose a distribution over words

$$\pi_k \sim \text{Dirichlet}(\alpha)$$

5. Choose N_d words for each document according to the super-topic of its group

$$x_{id} \sim \text{Multinomial}(\pi_k, k = R_{z_{id},d})$$

N_d is the number of words in document d and $N = \sum_d N_d$ is the total number of words.

Another way to describe the process is to use a metaphor similar to the Chinese restaurant franchise, see figure 7.4. There are N_d customers in restaurant d represented by the variables z_{id} . The customers sit down at the tables jointly. Where the probability of any particular seating arrangement is given by $Z_d \sim \text{FP}(N_d, \boldsymbol{\lambda}_d)$. For each table in each restaurant there is a representative r_{kd} . There are N_d possible tables in each restaurant because that is the maximum number of groups N_d customers can be partitioned into. Therefore there are N representatives. The representatives sit down at the tables in franchise restaurant, where the probability of a seating arrangement is given by a FP. Now that all of the customers Z and their table representatives R are sitting, a dish (super-topic) is assigned to each word by first looking at the table the customer is sitting at $z_{id} = k$, and then looking for their representative $r_{kd} = t$.

The HFP thus specifies FPs at the document level and at the topic level. Using an inconsistent FP at the document level is not an issue because we do not expect a document to represent a random sub-sample from a much larger document. At the topic level we specify our FP over N elements and marginalize out the unnecessary (empty) ones by simply keeping them around as dummies in the sampler. The top level FP could now be used to express our prior expectations over the total number of topics in the corpus.

Gibbs sampling proceeds by sequentially reassigning $\{z_{id}\}$ and $\{r_{kd}\}$. If we reassign z_{id} we can choose an existing document-topic or create a new document-topic in

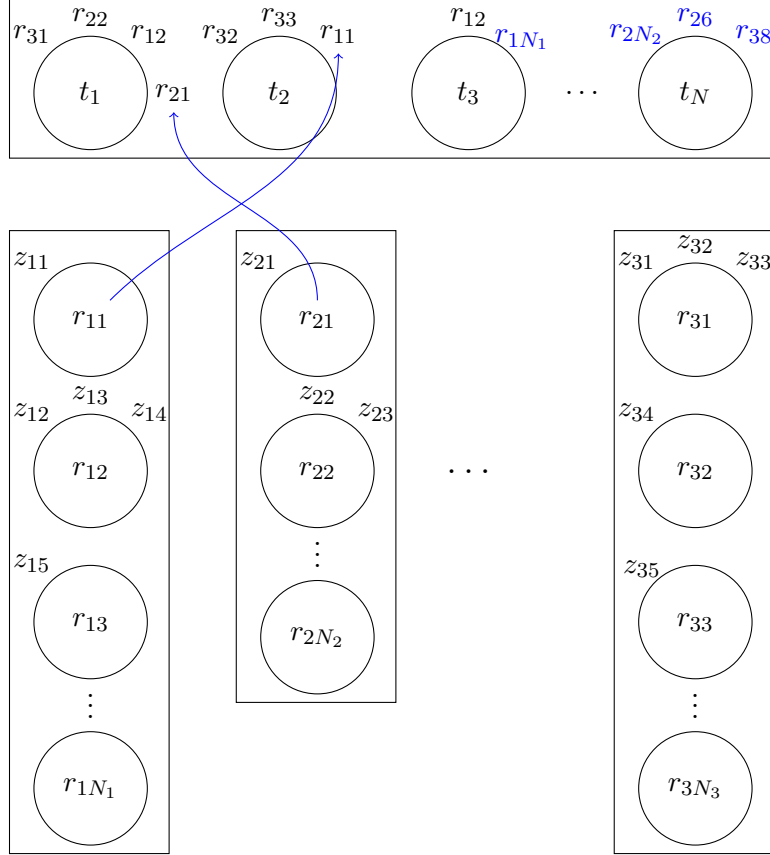


Figure 7.4: Graphical representation of HFP model. Each box represents a restaurant (FP). Customers jointly sit down at tables according to the FP distribution. The vertically oriented restaurants each represent a document with N_d customers and N_d tables. Each table represents a document-topic. The horizontal restaurant represents the franchise restaurant and each table represents a super-topic. There are $N = \sum_d N_d$ table representatives in the franchise restaurant, one for each possible table. A unique super-topic is assigned to each z_{id} by asking the representative of their table what super-topic ($t_{r_{z_{id}d}}$) is assigned to their table ($r_{z_{id}d}$) in the franchise restaurant. The blue customers in the horizontal franchise restaurant are representatives for tables in the vertical restaurants that have no customers (not all representatives are shown). If there are L current occupied tables, then there are $N - L$ blue representatives.

document d . However, there is no difference in the choice between a new or an

existing document-topic because there is already a super-topic representative.

$$p(z_{id} = k | -) \propto P_{r_{z_{id}d}}^{-x_{id}}(x_{id}) \text{FP}_d(z_{id} = k, \mathbf{z}_{-id} | N_d, \boldsymbol{\beta}_j) \quad (7.6)$$

$$P_t^{-x_{id}}(x_{id}) \propto \int d\theta p(x_{id} | \theta) \prod_{\substack{i', d' \neq i, d \\ z_{i'd'} = k, r_{kd'} = t}} p(x_{i'd'} | \theta) p(\theta) \quad (7.7)$$

We use a collapsed Gibbs sampler which avoids resampling parameters $\boldsymbol{\theta}$.

We will also reassign the super-topic assignments, r_{kd} . This will only change the probabilities $\text{FP}_0(N, \boldsymbol{\alpha})$ but not affect $\text{FP}_d(N_d, \boldsymbol{\beta})$. We define $\mathbf{x}_{\tilde{c}}$ to be the subset of X whose assignment variable is equal to k . That is, $\mathbf{x}_{\tilde{c}(kd)}$ is the set of all data-items assigned to document-topic k in document d . The conditionals are then given by,

$$p(r_l = t | -) \propto \quad (7.8)$$

$$P_t^{-\mathbf{x}_{\tilde{c}(kd)}}(\mathbf{x}_{\tilde{c}(kd)}) \text{FP}_0(r_{kd} = t, \mathbf{r}_{-kd} | N, \boldsymbol{\alpha})$$

$$P_t^{-\mathbf{x}_k}(\mathbf{x}_k) \propto \int d\theta \prod_{i: z_{id}=k} p(x_{id} | \theta) \prod_{\substack{i', d' \neq i, j \\ z_{i'j'} = k, r_{kj'} = t}} p(x_{i'j'} | \theta) p(\theta)$$

Note that in some cases r_{kd} is not associated with data, the set $\tilde{c}(kd) = \emptyset$. Hence, there will be no likelihood term in Eq.7.8 implying that dummies are sampled directly from the flexible prior $\text{FP}_0(r_{kd} = t, \mathbf{r}_{-kd} | N, \boldsymbol{\alpha})$.

Iterating these two sampling updates in succession constitutes our Gibbs sampler for the HFP.

7.5 AN EXPERIMENT WITH TEXT DATA

In this experiment we compare HFP vs HDP on the KOS text corpus. Our conjecture is that consistency under marginalization is not necessary at the document level when

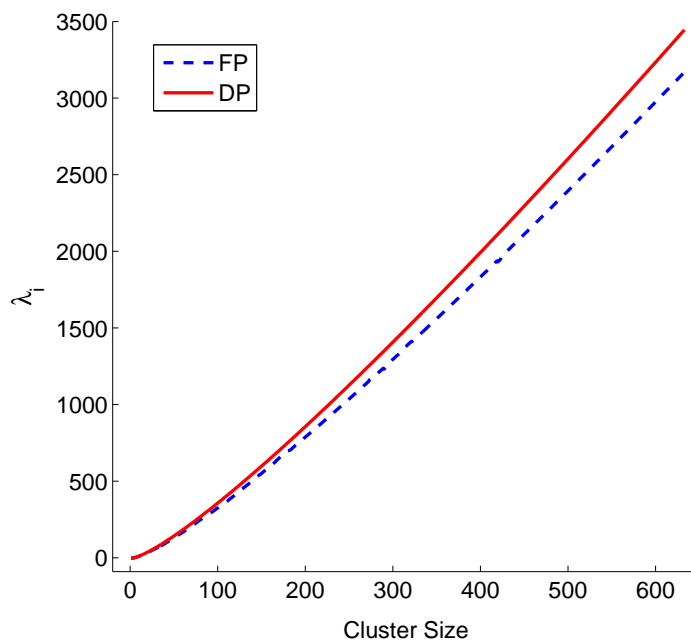


Figure 7.5: Resulting λ from HFP learning a, b, c in equation 7.9, and HDP learning α choosing how many words in a document are assigned to a topic, but still makes sense when choosing the topics. That is, we still assume that the topic assigned to each group of words in a document is drawn from an infinite pool of super-topics we do not believe that a document is a random subset of an infinitely long document (and

if it were a subset of a longer article we do not believe that the unobserved remainder is well represented by the same topic distribution). The result is a model where the parameters α of the top level FP_0 are set such that the FP is equivalent to a DP, i.e. $\alpha_i = \log(\xi) + \log \Gamma(i)$ (ξ is DP parameter). Inference in the model is done using Gibbs sampling as described in section 7.4, using the conditional distributions equations 7.6,7.8.

Instead of specifying our prior knowledge in terms of c_a we take an empirical Bayesian approach where we learn c_a given the data. Specifically, we are using contrastive divergence as described at the end of section 7.1. We use a single set of λ 's for all FP_j with the following functional form:

$$\lambda_i = a + b \log \Gamma(i + c) \tag{7.9}$$

We use the KOS data set [4] for evaluation. We do 10 runs, each with $\frac{1}{10}$ of the full data set. For each run we use half of the words in one-quarter of the documents as held out test data and the rest as training data. For each data set we run both HDP and HFP and compare the models in terms of test data perplexity. We use contrastive divergence to learn the parameters of equation 7.9 for HFP and α for HDP.

To calculate the perplexity we sample the distribution for test words given the current state of the Gibbs chain $\theta_{stj} = \sum_k p(z_{tj} = k | -) P_{r_{c_k}}(x_{tj})$ (tj indexes test word t in document j), where z_{tj} is a latent variable and is sampled as part of the inference procedure, but x_{tj} is unobserved. After allowing the chain to burn-in for 300 iterations, we average over 400 samples $s = 1 \dots S$ to get $\theta_{tj} = \frac{1}{S} \sum_{s=1}^S \theta_{stj}$. Perplexity is then $\exp(-1/T \sum_{tj} \log p(x_{tj} | \theta_{tj}))$. For HDP we make the equivalent calculation using samples from the Chinese restaurant franchise sampling scheme [42].

Figure 7.5 compares the perplexity of HFP and HDP on 10 subsets of KOS. We find

a small but significant improvement in terms of perplexity between HDP and HFP. The average perplexity for HDP and HFP is 1737 and 1675, respectively. Using a paired t-test we can conclude with a p-value of .01 that the differences are significant. We can also examine the difference in learned λ between HFP and HDP in figure 7.5. In particular we see that HFP learns a distribution with a different shape than a DP. These experiments represent evidence that we can learn better nonparametric models by relaxing consistency.

7.6 Flexible Priors Discussion

Flexible priors are a natural addition to the NMB framework, because they are essentially a large family of distributions over partitions and mixture blocks interface with each other through partitions. Because FP priors fit within the NMB framework, there are many models that where multinomial, CRP or HDP mixture blocks could be replaced with FP mixture blocks. In this work we have given one example, the HDP model, where we found a benefit from replacing a CRP block.

One difficulty when working with the FP family of priors is how to learn the parameters without overfitting. However, in preliminary experiments we have been able to overcome this difficulty by taking an empirical Bayesian approach, moving the prior towards the posterior.

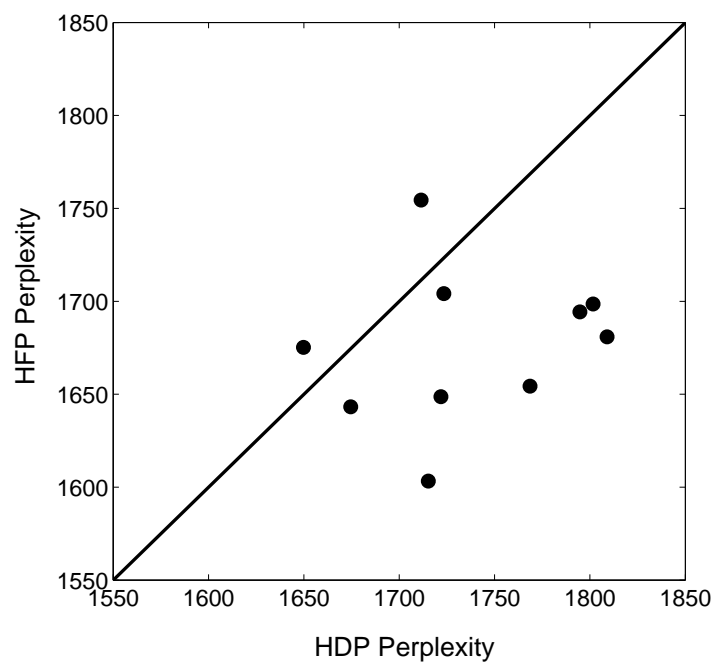


Figure 7.6: Comparison of the test perplexity of HDP vs HFP for the KOS text corpus. The x and y coordinates of each point are the perplexity of HDP (x) and HFP (y) on one subset of the KOS text corpus. Points below the diagonal are tests where HFP had lower perplexity than HDP

Chapter 8

Conclusions

In this dissertation we introduced a general framework, NMB, for (non)parametric hierarchical structured mixture models. The NMB framework allows us to factor these models in a way that simplifies their description and implementation. Once expressed in terms of the NMB framework several dimensions of model experimentation become easy. First, the distribution family of the mixture block can be easily exchanged, for example, we can replace a parametric DCM distribution with a nonparametric HDP distribution. Second, the generative process that partitions the observations can be easily modified, for example, converting LDA to Multi-LDA creating clusters based on multiple latent factors. Third, new types of distributions over partitions, such as flexible priors can be easily introduced into existing models.

Using the NMB framework we introduce several models that make their own contributions as well as demonstrate experimentation within framework.

Multi-LDA and Multi-HDP are a new family of models for tensor factorization. We show in collaborative filtering experiments that not only do they produce good models for the data, they also produce a good estimate of the uncertainty in the prediction.

We also show that in a number of different domains, including collaborative filtering, market basket analysis, text analysis, they discover interesting hidden structure in the data. Using the NMB framework, it is easy to see how to extend the standard LDA model to the Multi-LDA model by simply adding a new DCM mixture block and changing the function that assigns observations to mixture groups. The transition from Multi-LDA to Multi-HDP is also just a matter of replacing the mixture block distribution family.

BMFSI is a new model for matrix factorization which makes it easy to integrate available side information into a Bayesian collaborative filtering model. BMFSI produces state-of-the-art results for matrix factorization methods. However, BMFSI does not trivially fit into the NMB framework. Instead, by creating a “hybrid” NMB we allow the non-mixture block part of the model to interface with standard mixture blocks. The joining of BMFSI with a Dirichlet process model combines accurate predictions with discovery of latent structure in the data.

Finally, we introduce a new class of nonparametric Flexible priors into the framework. The new family of distributions allows us to learn better nonparametric priors over partitions when the data come from a generative process not consistent under marginalization. We replace CRP mixture blocks with FP mixture blocks in the HDP to create a new HFP model. In experiments we show that it is possible for HFP to learn a better prior over partitions than the HDP for modeling text.

Bibliography

- [1] R. Agrawal, T. Imielinski, and A. Swami. Mining associations between sets of items in massive databases. In *Proc. of the ACM-SIGMOD 1993 Intl Conf on Management of Data*, pages 207–216, 1993.
- [2] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 2008. in press.
- [3] C. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian non-parametric problems. *The Annals of Statistics*, 2:1152–1174, 1974.
- [4] A. Asuncion and D. Newman. UCI machine learning repository, 2007.
- [5] D. Barry and J. A. Hartigan. Product partition models for change point problems. *The Annals of Statistics*, 20(1):260–279, 1992.
- [6] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation, 2003.
- [7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [8] W. L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 2:159–225, 1994.
- [9] M. Dudík and R. E. Schapire. Maximum entropy distribution estimation with generalized regularization. In *COLT*, pages 123–138, 2006.
- [10] T. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1:209–230, 1973.
- [11] Z. Ghahramani, T. Griffiths, and P. Sollich. Bayesian nonparametric latent feature models. *Bayesian Statistics*, 8, 2007.
- [12] P. Green and S. Richardson. Modelling heterogeneity with and without the Dirichlet process. *Scandinavian Journal of Statistics*, 28:355377, 2001.
- [13] T. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. In *Advances in Neural Information Processing Systems 18*, pages 475–482, 2006.

- [14] T. Griffiths and M. Steyvers. A probabilistic approach to semantic representation. In *Proceedings of the 24th Annual Conference of the Cognitive Science Society*, 2002.
- [15] T. Griffiths and M. Steyvers. Finding scientific topics. In *Proceedings of the National Academy of Sciences (PNAS)*, pages 5228–5235, 2004.
- [16] G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.
- [17] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.
- [18] H. D. III. Hbc:hierarchical bayes compiler. <http://hal3.name/HBC>, 2007.
- [19] C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI '06)*, Menlo Park, CA, 2006.
- [20] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, New York, NY, USA, 2008. ACM.
- [21] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 50(2):157–224, 1988.
- [22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.
- [23] D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.
- [24] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*, pages 577–584, 2006.
- [25] Y. Lim and Y. Teh. Variational Bayesian approach to movie rating prediction. In *Proceedings of KDD Cup and Workshop*, 2007.
- [26] S. MacEachern and P. Müller. Estimating mixture of Dirichlet process models. *Communications in Statistics*, 7:223–238, 1998.
- [27] R. E. Madsen, D. Kauchak, and C. Elkan. Modeling word burstiness using the dirichlet distribution. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 545–552, New York, NY, USA, 2005. ACM.

- [28] V. Mansinghka, C. Kemp, J. B. Tenenbaum, and T. L. Griffiths. Structured priors for structure learning. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence (UAI 2006)*, 2006.
- [29] B. Marlin. Modeling user rating profiles for collaborative filtering. In *Neural Information Processing Systems (NIPS-03)*, Vancouver, CA, 2003.
- [30] B. Marlin. Modeling user rating profiles for collaborative filtering. In *Advances in Neural Information Processing Systems 16*, 2004.
- [31] E. Meeds, Z. Ghahramani, R. M. Neal, and S. T. Roweis. Modeling dyadic data with binary latent factors. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 977–984. MIT Press, Cambridge, MA, 2007.
- [32] B. Milch, B. Marthi, S. Russell, D. Sontag, D. L. Ong, and A. Kolobov. Blog: Probabilistic models with unknown objects. In *In IJCAI*, pages 1352–1359, 2005.
- [33] R. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:283–297, 2000.
- [34] J. Pitman. Combinatorial stochastic processes. Technical report, Dept. Statistics, U.C. Berkeley, 2002. Lecture notes for St. Flour course, Technical Report no.621.
- [35] I. Porteous, A. Asuncion, and M. Welling. Bayesian matrix factorization with side information and dirichlet process mixtures. In *AAAI*, 2010.
- [36] I. Porteous, E. Bart, and M. Welling. Multi-hdp: A non parametric bayesian model for tensor factorization. In *AAAI*, pages 1487–1490, 2008.
- [37] F. A. Quintana and P. L. Iglesias. Bayesian clustering and product partition models. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 65(2):557–574, 2003.
- [38] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 880–887, New York, NY, USA, 2008. ACM.
- [39] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.
- [40] D. Stern, R. Herbrich, and T. Graepel. Matchbox: Large scale online bayesian recommendations. In *18th International World Wide Web Conference*, pages 111–111, April 2009.
- [41] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Scalable collaborative filtering approaches for large recommender systems. *Journal of Machine Learning Research*, 10:623–656, 2009.

- [42] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. In *NIPS*, volume 17, 2004.
- [43] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.
- [44] T. Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the International Conference on Machine Learning*, volume 25, pages 1064–1071, 2008.
- [45] M. Welling. Flexible priors for infinite mixture models. In *ICML Workshop on Nonparametric Bayesian Methods*, 2006.
- [46] M. Welling, I. Porteous, and E. Bart. Infinite state Bayesian networks. In *Advances in Neural Processing Systems – NIPS*, 2007.
- [47] L. Younes. Parametric inference for imperfectly observed gibbsian fields. *Probability Theory and Related Fields*, 82:625–645, 1989.