

Copyright  
by  
Ayan Acharya  
2015

The Dissertation Committee for Ayan Acharya  
certifies that this is the approved version of the following dissertation:

**Knowledge Transfer Using Latent Variable Models**

Committee:

---

Joydeep Ghosh, Supervisor

---

Raymond J. Mooney, Co-Supervisor

---

Sanjay Shakkottai

---

Sujay Sanghavi

---

Suju Rajan

# **Knowledge Transfer Using Latent Variable Models**

**by**

**Ayan Acharya, B.E., M.S.E.**

## **DISSERTATION**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

## **DOCTOR OF PHILOSOPHY**

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2015

Dedicated to my parents, grandparents, wife and friends.

## Acknowledgments

First of all, I would like to convey my sincere gratitude to Dr. Joydeep Ghosh and Dr. Raymond J. Mooney who guided and motivated me through all the tough times. I would also like to thank the other members in my thesis committee: Dr. Sanjay Shakkottai, Dr. Sujay Sanghavi, and Dr. Suju Rajan, for their insightful comments and suggestions. This thesis came out as part of my collaboration with Dr. Eduardo R. Hruschka and Dr. Mingyuan Zhou. I am indebted to Dr. Zhou for introducing me to the challenging, outlandish, yet extremely intriguing domain of Bayesian nonparametrics and steering me through all the impediments. I thank my fellow labmates: Sreangsu Acharyya, Sanmi Koyejo, Priyank Patel, Suriya Guansekar, Joyce Ho, Yubin Park, Neeraj Gaur, Nikita Sudan, Dean Teffer, Rajiv Khanna, Shalmali Joshi, Avradeep Bhowmik, Dylan Anderson, and Marcus Tyler, for the stimulating discussions and all the fun we have had in the last few years. Last but not the least, I would like to thank my parents, grand parents, wife and all other members of my family for supporting and encouraging me all through these years, even when I was not able to spend much time with them. This thesis would not have been possible without the encouragement from some of my close friends: Soumya Dutta, Abhishek Ghosh, Protul Moitra, Anand Seetharam, Sharad Banka, Swagata Das, Arindam Sanyal, Somsubhra Barik, Abhik Bhattacharya, and Debarati Kundu. Time and again you have provided an ambience that never made me feel uncomfortable for being away from home for so long.

# Knowledge Transfer Using Latent Variable Models

Ayan Acharya, Ph.D.

The University of Texas at Austin, 2015

Supervisors: Joydeep Ghosh  
Raymond J. Mooney

In several applications, scarcity of labeled data is a challenging problem that hinders the predictive capabilities of machine learning algorithms. Additionally, the distribution of the data changes over time, rendering models trained with older data less capable of discovering useful structure from the newly available data. Transfer learning is a convenient framework to overcome such problems where the learning of a model specific to a domain can benefit the learning of other models in other domains through *either* simultaneous training of domains *or* sequential transfer of knowledge from one domain to the others. This thesis explores the opportunities of knowledge transfer in the context of a few applications pertaining to object recognition from images, text analysis, network modeling and recommender systems, using probabilistic latent variable models as building blocks. Both simultaneous and sequential knowledge transfer are achieved through the latent variables, *either* by sharing these across multiple related domains (for simultaneous learning) *or* by adapting their distributions to fit data from a new domain (for sequential learning).

# Table of Contents

<b>Acknowledgments</b>	<b>v</b>
<b>Abstract</b>	<b>vi</b>
<b>List of Tables</b>	<b>xiii</b>
<b>List of Figures</b>	<b>xiv</b>
<b>Chapter 1. Introduction</b>	<b>1</b>
<b>Chapter 2. Background</b>	<b>7</b>
2.1 Distributions . . . . .	7
2.1.1 SumLog Distribution . . . . .	7
2.1.2 Poisson Distribution . . . . .	8
2.1.3 Poisson Bernoulli Distribution . . . . .	8
2.1.4 Gamma Distribution . . . . .	9
2.1.5 Dirichlet Distribution . . . . .	10
2.1.6 Negative Binomial Distribution . . . . .	11
2.2 Random Processes . . . . .	11
2.2.1 Random Process . . . . .	11
2.2.2 Gamma Process (GP) . . . . .	12
2.2.3 Dirichlet Process (DP) . . . . .	13
2.2.4 Hierarchical Dirichlet Process (HDP) . . . . .	16
2.3 Matrix Factorization . . . . .	16
2.4 Statistical Topic Models . . . . .	19
2.4.1 Parametric Topic Models . . . . .	19
2.4.2 Nonparametric Topic Models . . . . .	21
2.4.2.1 Traditional Stick Breaking Construction of HDP Topic Model . . . . .	21

2.4.2.2	Modified Stick Breaking Construction of HDP Topic Model . . . . .	22
2.4.3	Inference in Topic Models . . . . .	23
2.4.3.1	Batch Variational Inference in Topic Models . .	24
2.4.3.2	Incremental EM Algorithm . . . . .	24
2.5	Transfer Learning . . . . .	26
2.6	Active Learning <i>via</i> Expected Error Reduction . . . . .	29

### **Chapter 3. Multitask Learning using Both Supervised and Shared Latent Topics 31**

3.1	Background . . . . .	33
3.1.1	Labeled Latent Dirichlet Allocation (LLDA) . . . . .	34
3.1.2	Maximum Entropy Discriminant Latent Dirichlet Allocation (MedLDA) . . . . .	34
3.2	Doubly Supervised LDA (DSLDA) . . . . .	36
3.2.1	Model Description . . . . .	37
3.2.2	Inference and Learning . . . . .	38
3.3	Non-parametric DSLDA . . . . .	41
3.4	Experimental Evaluation . . . . .	47
3.4.1	Data Description . . . . .	47
3.4.1.1	aYahoo Data . . . . .	47
3.4.1.2	ACM Conference Data . . . . .	48
3.4.2	Methodology for Experiments with Multitask Learning . .	49
3.4.3	Multitask Learning Results . . . . .	52
3.5	Conclusion . . . . .	55

### **Chapter 4. Active Multitask Learning using Both Supervised and Shared Latent Topics 57**

4.1	Background . . . . .	58
4.1.1	Active Knowledge Transfer . . . . .	58
4.1.2	Online Support Vector Machines . . . . .	59
4.2	Active Doubly Supervised Latent Dirichlet Allocation (Act-DSLDA) 60	
4.2.1	Inference and Learning . . . . .	63
4.2.1.1	Learning in Batch Mode . . . . .	63

4.2.1.2	Incremental Learning in Active Selection . . . .	66
4.3	Active Non-parametric DSLDA (Act-NPDSLDA) . . . . .	67
4.3.1	Inference and Learning . . . . .	68
4.3.1.1	Learning in Batch Mode . . . . .	68
4.3.1.2	Incremental Learning in Active Selection . . . .	70
4.4	Experimental Results . . . . .	71
4.4.1	Methodology for Experiments with Active Multitask Learning . . . . .	71
4.4.2	Active Multitask Learning Results . . . . .	73
4.4.3	Discussion . . . . .	74
4.5	Conclusion . . . . .	77
<b>Chapter 5.</b>	<b>Active Multitask Learning Using Annotators' Rationale</b>	<b>79</b>
5.1	Related Work . . . . .	80
5.2	Active Learning with Annotators' Rationale in Doubly Supervised Latent Dirichlet Allocation (Act-Rationale-DSLDA) . . . .	82
5.2.1	Inference and Learning . . . . .	86
5.2.1.1	Learning in Batch Mode . . . . .	86
5.2.1.2	Incremental Learning in Active Selection . . . .	89
5.3	Experimental Results . . . . .	90
5.3.1	Datasets . . . . .	90
5.3.2	Methodology for Experiments . . . . .	92
5.3.3	Results . . . . .	92
5.4	Conclusion . . . . .	95
<b>Chapter 6.</b>	<b>Gamma Process Poisson Factorization for Joint Modeling of Network and Documents</b>	<b>96</b>
6.1	Related Work . . . . .	98
6.2	Gamma Process Poisson Factorization for Networks (N-GPPF) . .	102
6.2.1	Gibbs Sampling for N-GPPF . . . . .	104
6.2.2	Gibbs Sampling for N-GPPF with Missing Entries . . . .	106
6.3	Gamma Process Poisson Factorization for Count Matrices (C-GPPF) . . . . .	106
6.3.1	Gibbs Sampling for C-GPPF . . . . .	107

6.3.2	Gibbs Sampling for C-GPPF with Missing Entries . . . . .	109
6.4	Joint Gamma Process Poisson Factorization (J-GPPF) . . . . .	110
6.4.1	Inference <i>via</i> Gibbs Sampling . . . . .	112
6.4.2	Gibbs Sampling for J-GPPF with Missing Entries . . . . .	115
6.4.3	Special cases: Network Only GPPF (N-GPPF) and Cor- pus Only GPPF (C-GPPF) . . . . .	115
6.4.4	Computation Complexity . . . . .	116
6.5	Experimental Results . . . . .	117
6.5.1	Experiments with Synthetic Data . . . . .	117
6.5.2	Experiments with Real World Data . . . . .	119
6.5.2.1	NIPS Authorship Network . . . . .	119
6.5.2.2	GoodReads Data . . . . .	121
6.5.2.3	Twitter Data . . . . .	121
6.5.2.4	Experimental Setup and Results . . . . .	123
6.6	Conclusion . . . . .	124

## **Chapter 7. Nonparametric Dynamic Models 125**

7.1	Nonparametric Bayesian Dynamic Count and Binary Matrix Fac- torization . . . . .	125
7.1.1	Gamma Process Dynamic Poisson Factor Analysis . . . . .	128
7.1.1.1	Inference <i>via</i> Gibbs Sampling . . . . .	130
7.1.1.2	Modeling Binary Observations . . . . .	133
7.1.2	Experimental Results . . . . .	135
7.1.2.1	Results with Synthetic Datasets . . . . .	135
7.1.2.2	Results with Real World Datasets . . . . .	137
7.2	Nonparametric Dynamic Network Modeling . . . . .	142
7.2.1	Related Work . . . . .	143
7.2.2	Dynamic Gamma Process Poisson Factorization for Net- works (D-NGPPF) . . . . .	145
7.2.2.1	Gibbs Sampling for D-NGPPF . . . . .	147
7.2.2.2	Gibbs Sampling for D-NGPPF with Missing Entries . . . . .	149
7.2.3	Experiments . . . . .	150
7.2.3.1	Synthetic Data . . . . .	150

7.2.3.2	Real World Data . . . . .	154
7.3	Nonparametric Dynamic Count Matrix Factorization . . . . .	158
7.3.1	Background and Related Work . . . . .	160
7.3.2	Dynamic Gamma Process Poisson Factorization for Count Matrices (D-CGPPF) . . . . .	161
7.3.2.1	Gibbs Sampling for Dynamic Poisson Matrix Factor Model . . . . .	163
7.3.2.2	Gibbs Sampling for Dynamic Poisson Matrix Factor Model with Missing Entries . . . . .	165
7.3.3	Experiments . . . . .	168
7.3.3.1	Synthetic Data . . . . .	168
7.3.3.2	Real World Data . . . . .	169
7.4	Conclusion . . . . .	171
<b>Chapter 8. Bayesian Combination of Classification and Clustering En- sembles</b>		<b>172</b>
8.1	Related Work . . . . .	174
8.2	Probabilistic Model . . . . .	175
8.2.1	Approximate Inference and Estimation: . . . . .	180
8.2.1.1	Inference: . . . . .	180
8.2.1.2	Estimation: . . . . .	183
8.3	Privacy Preserving Learning . . . . .	184
8.3.1	Row Distributed Ensemble: . . . . .	185
8.3.2	Column Distributed Ensemble: . . . . .	186
8.3.3	Arbitrarily Distributed Ensemble: . . . . .	188
8.4	Experiments . . . . .	191
8.4.1	Transfer Learning: . . . . .	191
8.4.2	Semi-supervised Learning: . . . . .	195
8.5	Conclusion . . . . .	196
<b>Chapter 9. Future Work</b>		<b>198</b>
9.1	Dynamic Count Tensor Factorization . . . . .	198
9.2	Distributed Count Matrix Factorization . . . . .	199

<b>Bibliography</b>	<b>200</b>
<b>Vita</b>	<b>218</b>

## List of Tables

3.1	Illustration of Latent and Supervised Topics . . . . .	54
4.1	Distributions in Act-NPDSLDA . . . . .	69
4.2	Latent and Supervised Topics Discovered by Act-DSLDA . . . . .	77
6.1	Generative Process of N-GPPF . . . . .	104
6.2	Generative Process of C-GPPF . . . . .	107
6.3	Gibbs sampling updates in J-GPPF . . . . .	114
6.4	Sampling of $\phi_{nk_B}, \psi_{wk_B}, \rho_{k_B}$ in J-GPPF with missing entries . . . .	115
6.5	Sampling of $\theta_{dk_Y}, \beta_{wk_Y}, r_{k_Y}$ in J-GPPF with missing entries . . . .	116
7.1	AUC Results on Real World Data . . . . .	154
7.2	MAP Results on Real World Data . . . . .	170
7.3	NDCG Results on Real World Data . . . . .	170
8.1	Equations for update of variational and model parameters in <b>BC<sup>3</sup>E</b> .	182
8.2	Performance of <b>BC<sup>3</sup>E</b> on text classification data — Avg. Accuracies $\pm$ (Standard Deviations). . . . .	194
8.3	Comparison of <b>BC<sup>3</sup>E</b> with <b>C<sup>3</sup>E</b> and <b>BGCM</b> — Avg. Accuracies $\pm$ (Standard Deviations). . . . .	197

## List of Figures

2.1	Gamma-Poisson Construction of NB Distribution . . . . .	15
2.2	Compound Poisson Construction of NB Distribution . . . . .	15
2.3	LDA . . . . .	20
2.4	Smoothened LDA . . . . .	20
2.5	Illustration of Active Learning [Settles, 2009] . . . . .	30
3.1	MTL with Shared Attributes . . . . .	32
3.2	MTL with Multi-layer Perceptron . . . . .	32
3.3	LLDA . . . . .	33
3.4	MedLDA . . . . .	33
3.5	Graphical Model of DSLDA . . . . .	39
3.6	Illustration of DSLDA . . . . .	39
3.7	Graphical Model of NP-DSLDA . . . . .	42
3.8	Illustration of NP-DSLDA . . . . .	42
3.9	Illustration of MedLDA-OVA . . . . .	48
3.10	Illustration of MedLDA-MTL . . . . .	48
3.11	Illustration of DSLDA-OSST . . . . .	50
3.12	Illustration of DSLDA-NSLT . . . . .	50
3.13	$p_1 = 0.5$ (aYahoo) . . . . .	51
3.14	$p_1 = 0.7$ (aYahoo) . . . . .	52
3.15	$p_1 = 0.5$ (Conference) . . . . .	53
3.16	$p_1 = 0.7$ (Conference) . . . . .	54
4.1	Graphical Model of Act-DSLDA . . . . .	61
4.2	Illustration of Act-DSLDA . . . . .	61
4.3	Graphical Model of Act-NPDSLDA . . . . .	69
4.4	Illustration of Act-NPDSLDA . . . . .	69
4.5	Illustration of Act-MedLDA-OVA . . . . .	73

4.6	Illustration of Act-MedLDA-MTL . . . . .	73
4.7	Illustration of Act-DSLDA-OSST . . . . .	73
4.8	Illustration of Act-DSLDA-NSLT . . . . .	73
4.9	Illustration of MedLDA-MTL-Random . . . . .	73
4.10	Illustration of DSLDA-Random . . . . .	73
4.11	aYahoo Learning Curves . . . . .	74
4.12	ACM Conference Learning Curves . . . . .	74
4.13	aYahoo Query Distribution . . . . .	75
4.14	ACM Conference Query Distribution . . . . .	76
5.1	Illustration of Rationale for Images . . . . .	80
5.2	Distribution of query time for rationales – ACM Conference . . . . .	91
5.3	Distribution of query time for rationales – aYahoo . . . . .	91
5.4	ACM Conference Results . . . . .	93
5.5	aYahoo Results . . . . .	93
5.6	Distribution of query types – ACM Conference . . . . .	94
5.7	Distribution of query types – aYahoo . . . . .	94
6.1	(a) Time to generate a million of samples, (b) $\mathbf{B}$ with held-out data, (c) $\mathbf{Y}$ . . . . .	118
6.2	Performance of J-GPPF: $\epsilon = 10^{-3}$ (top row), $\epsilon = 1$ (middle row), $\epsilon = 10$ (bottom row) . . . . .	120
6.3	(a) AUC with $\epsilon = 0.001$ , (b) AUC with $\epsilon = 1.0$ , (c) AUC with $\epsilon = 10.0$ . . . . .	120
6.4	(a) NIPS Data, (b) GoodReads Data . . . . .	122
6.5	Twitter Data . . . . .	122
6.6	MAP NIPS . . . . .	123
6.7	MAP GoodReads . . . . .	123
7.1	Illustration of GP-DPFA . . . . .	130
7.2	(a) correlation across topics over time, (b) latent factors dominant over time for GP-DPFA, (c) latent factors dominant over time for the baseline. . . . .	136

7.3	(a) correlation of the observed data across time, (b) correlation discovered in the latent space, (c) correlation between the observation and latent counts, (d) correlation between the ten most prominent latent factors for GP-DPFA, (e) correlation between the ten most prominent latent factors in the baseline model. . . . .	138
7.4	Top Row: Correlation plots for JSB chorales, Middle Row: Correlation plots for Piano.midi, Bottom Row: Correlation plots for Musedata. In each row, figures from left to right are plots that are analogous to Figs. 5 (a)-(c). . . . .	140
7.5	Illustration of Dynamic Network Model . . . . .	146
7.6	Results from D-NGPPF . . . . .	151
7.7	Results from N-GPPF . . . . .	152
7.8	Infocom: Hour 5 <sup>th</sup> to 8 <sup>th</sup> . . . . .	155
7.9	Infocom: Hour 9 <sup>th</sup> to 12 <sup>th</sup> . . . . .	156
7.10	Infocom: Hour 13 <sup>th</sup> to 16 <sup>th</sup> . . . . .	157
7.11	Illustration of D-CGPPF . . . . .	163
7.12	Results from D-CGPPF . . . . .	167
7.13	Results from C-GPPF . . . . .	167
8.1	Combining Classifiers and Clusterers. . . . .	177
8.2	Graphical Model for <b>BC<sup>3</sup>E</b> . . . . .	177
8.3	Arbitrarily Distributed Ensemble . . . . .	189
8.4	Parameter Update for Arbitrarily Distributed Ensemble . . . . .	190

# Chapter 1

## Introduction

In several practical applications of machine learning, scarcity of labeled data is a challenging problem that adversely affects the predictive capabilities. Additionally, the distribution of the data changes over time, rendering models trained with older data less capable of discovering useful structure from the newly available data. Such problems can conveniently be addressed by a mechanism that can exploit the labeled information from one domain, such as labeled images belonging to a given object category, and use the same in analyzing other related domains, such as images consisting of visually similar objects. The learning of models specific to a given domain can potentially benefit from other models from other related domains through *either* simultaneous *or* sequential training. Such mechanism is defined as “knowledge transfer” in this thesis. Simultaneous knowledge transfer is more popularly known as multitask learning in the machine learning literature [Caruana, 1997; Pan and Yang, 2010]. In such a framework, labeled data from two or more closely related domains are collected and the models corresponding to all these related domains are trained jointly, with the assumption that the labeled data from these domains provide helpful inductive bias to one another. Sequential knowledge transfer is more popularly known as transfer learning [Bollacker and Ghosh, 2000; Pan and Yang, 2010] in the machine learning literature. In the premise of transfer

learning, a model learnt from a “source” domain is adapted for a “target” domain, and solving a supervised or unsupervised learning problem in a target domain is the primary objective. Interestingly, Pan and Yang [2010] categorizes multitask learning as a type of “inductive” transfer learning and hence the distinction between multitask learning and transfer learning is not so clear. However, herein, multitask learning always refers to simultaneous knowledge transfer.

This thesis explores the effectiveness of knowledge transfer in the context of the following applications pertaining to object recognition from images, text analysis, network modeling and recommender systems, with probabilistic latent variable models used as the building blocks.

- Recognizing objects from images is a challenging problem simply because of the abundance of too many object categories in nature. Annotation by human experts is time consuming and expensive, for which isolated training of models for each individual object category is practically infeasible. However, the fact that many different object categories are visually similar and share some set of features can potentially be utilized for joint training of multiple models where the labeled data from one object category can provide useful information for the learning of models corresponding to other categories. For example, a model capable of identifying donkeys can also be utilized to identify zebras as both these object categories share some common set of features, such as four legs, two ears and a tail.
- Analysis and prediction of the category of text articles based on the contents only is often a hard problem to solve when the number of training examples is limited.

However, analogous to the application of object recognition from images, one may use documents from multiple related categories and try to learn predictive models for all these categories jointly. As an example, models to identify research papers from conferences like International Conference on Machine Learning (ICML) and Neural Information Processing Systems (NIPS) can be trained jointly, as the sets of training documents are pretty similar and together they might enhance the models learnt, of course in presence of other “negative” categories, such as papers from a completely different domain like physical design and integrated circuits.

- For applications of object recognition from images and text analysis, often the human annotators can provide high-level semantic features for objects or documents belonging to a given category. Such semantic features can potentially span multiple categories, implying that they might describe some characteristics of multiple related categories. When such high-level semantic features are included in the modeling assumptions, one can potentially improve the performance of the predictive models.
- Often, relational data, such as a graph of users in a social network, comes with auxiliary side information, such as a list of books or movies rated. Being able to learn and impute the graph better using the auxiliary information is yet another example where models are learnt simultaneously, one for analyzing the graph and the other for predicting the preference of the users for the movies or books.
- Relational data changes with time. For example, in a social network, users can get acquainted with new users. Among a group of authors publishing scientific ar-

ticles, the pattern of collaboration can change. At each different time slice, there might not be enough information available, in which case, one might benefit from learning from the older interactions. Similarly, dyadic data, which records the interaction between two different sets of entities, also changes over time. Such temporal evolution is very common in recommender systems where the pattern of interaction between users and items adapt with time.

- For applications in e-commerce, the language models built from older data for identifying product category often lose their utility with time, particularly because of the natural changes in language that describes the product or the query that is supposed to retrieve the product. Instead of retraining the models from scratch, a process that is time consuming and expensive, one may adapt the old model to fit new descriptions.

The first four problems: recognizing objects from images, identifying category of a text document, incorporating high-level semantic features, and network modeling using side information, benefit from simultaneous knowledge transfer. The latent variable models proposed herein for solving these problems have a common theme. All of them use some low-dimensional representation of the original features/observation that is shared across multiple domains. The key idea here is to project data from two or more domains onto a common low-dimensional space, so that the mapping from the original feature space to the low-dimensional space is learnt better given data from all these domains. The mapping from this low-dimensional space to the target variables can then be learnt much more efficiently

given only a few labeled information associated with the target variables. To further clarify the utility of a shared low-dimensional space, one can consider a “baseline” scheme where data from different domains are projected onto disjoint low-dimensional subspaces and the mapping to the target variables are learnt in an isolated manner. In such a scheme, if some domain has less labeled information, the mapping from the original feature space to the low-dimensional space is not learnt well, which might result in poor predictive performance. Chapter 3 deals with simultaneous knowledge transfer which, in Chapter 4, is further integrated with active learning, another mechanism that lowers the computation cost of annotation by querying for most useful information from annotators. Chapter 5 explores further how human annotators can be engaged more to improve the performance of simultaneous knowledge transfer framework, by providing specific evidence for the labels they provide. The application of simultaneous knowledge transfer in Chapter 6 is little different from those proposed in Chapter 3, 4 and 5. In this chapter, observations from two different domains (a network of users and an auxiliary count matrix) are projected onto a shared latent space which helps better imputation in both the domains.

The last two problems mentioned in the list above avail of sequential knowledge transfer. The common underlying theme here is the learning of the parameters of the associated latent variable models with data from an older domain and careful modification of these parameters for fitting the new distribution from the new domain. In Chapter 7, several temporal models are proposed that can analyze the evolution of count-valued vectors and matrices obtained from analysis of text doc-

uments, networks, and dyadic data that change over time. Chapter 8 demonstrates how the prediction from a model built from older data can be adapted, without re-training the old model from scratch, to serve the need for modeling new data with a slightly different distribution. To make the thesis self-contained, all the relevant mathematical tools are presented in Chapter 2. Conclusion and future work are listed further in Chapter 9.

## Chapter 2

### Background

In this chapter, the background mathematical tools and some of the related works are presented. For an easy perusal, whenever convenient, different sections point to different chapters where the concerned tools, algorithms and models are used. Vectors and matrices are denoted by bold-faced lowercase and capital letters, respectively. Scalar variables are written in italic font, and sets are denoted by calligraphic uppercase letters.  $\text{Dir}()$ ,  $\text{Gam}()$ ,  $\text{Beta}()$ ,  $\text{Pois}()$ , and  $\text{Mult}()$  stand for Dirichlet, Gamma, Beta, Poisson and multinomial distribution respectively.

#### 2.1 Distributions

This section describes several distributions used throughout this thesis. Some of the relevant lemmas are also listed which we use quite frequently in multiple chapters.

##### 2.1.1 SumLog Distribution

**SumLog Distribution:**  $m$  is defined to have a SumLog distribution with parameters  $(l, p)$  when  $m = \sum_{t=1}^l u_t, u_t \sim \text{Log}(p)$ .  $u \sim \text{Log}(p)$  is the logarithmic distribu-

tion [Johnson et al., 2005] with PMF:

$$f(u = k) = -p^k / (k \ln(1 - p)).$$

The mean and variance of a logarithmic distribution are  $\frac{-1}{\ln(1-p)} \frac{p}{1-p}$  and  $-p \frac{p + \ln(1-p)}{(1-p)^2 \ln^2(1-p)}$  respectively. For conciseness, we represent a SumLog distribution as  $m \sim \sum \text{Log}(p)$ .

### 2.1.2 Poisson Distribution

**Poisson Distribution:** A discrete random variable  $X$  is said to have a Poisson distribution with parameter  $\lambda > 0$ , if for  $k = 0, 1, 2, \dots$ , the probability mass function of  $X$  is given by:

$$f(k; \lambda) = \Pr(X=k) = \frac{\lambda^k \exp(-\lambda)}{k!},$$

where  $k!$  is the factorial of  $k$ . For Poisson distribution:  $\mathbb{E}[X] = \text{Var}[X] = \lambda$ .

### 2.1.3 Poisson Bernoulli Distribution

**Poisson Bernoulli Distribution:** When  $b = 1$  ( $n \geq 1$ ),  $n \sim \text{Pois}(\lambda)$ , the distribution of  $b$  given  $\lambda$  is called the Poisson Bernoulli distribution, with PMF

$$f(b|\lambda) = \exp(-\lambda(1-b))(1 - \exp(-\lambda))^b, \quad b \in \{0, 1\}.$$

The conditional posterior of the latent count  $n$  is simply  $(n|b, \lambda) \sim b \cdot \text{Pois}_+(\lambda)$ , where  $k \sim \text{Pois}_+(\lambda)$  is the truncated Poisson distribution with PMF:

$$f(k) = \frac{1}{1 - \exp(-\lambda)} \frac{\lambda^k \exp(-\lambda)}{k!}, \quad k = 1, 2, \dots$$

Thus if  $b = 0$ , then  $n = 0$  almost surely (a.s.), and if  $b = 1$ , then  $n$  is drawn from a truncated Poisson distribution.

**Sampling from Truncated Poisson distribution:** To simulate the truncated Poisson random variable  $x \sim \text{Pois}_+(\lambda)$ , we use rejection sampling: if  $\lambda \geq 1$ , we draw  $x \sim \text{Pois}(\lambda)$  until  $x \geq 1$ ; if  $\lambda < 1$ , we draw  $y \sim \text{Pois}(\lambda)$  and  $u \sim \text{Unif}(0, 1)$ , and let  $x = y + 1$  if  $u < 1/(y + 1)$ . The acceptance rate is  $1 - e^{-\lambda}$  if  $\lambda \geq 1$  and  $\lambda^{-1}(1 - e^{-\lambda})$  if  $\lambda < 1$ . Thus the minimum acceptance rate is 63.2% (when  $\lambda = 1$ ).

#### 2.1.4 Gamma Distribution

**Gamma Distribution:** A random variable  $X \sim \text{Gamma}(a, b)$  has probability density function  $f(X) = \frac{1}{\Gamma(a)b^a} x^{a-1} \exp(-\frac{x}{b})$ . This is the shape-scale parameterization of the Gamma distribution with shape  $a > 0$  and scale  $b > 0$ . For Gamma distribution, we have:  $\mathbb{E}[X] = ab$ ,  $\text{Var}[X] = ab^2$ .

For computational convenience, many of the modeling assumptions are designed using conjugate prior distributions. Some results are presented here in the form of lemmas for ease of deriving the conditional posterior equations. The proofs of these Lemmas follow from the definitions of the respective distributions.

**Lemma 2.1.1.** *If  $\lambda \sim \text{Gam}(r, 1/c)$ ,  $x_i \sim \text{Poisson}(m_i \lambda)$ , then*

$$\lambda | \{x_i\} \sim \text{Gam}(r + \sum_i x_i, 1/(c + \sum_i m_i)).$$

**Lemma 2.1.2.** *If  $r_i \sim \text{Gam}(a_i, 1/b) \forall i \in \{1, 2, \dots, K\}$ ,  $b \sim \text{Gam}(c, 1/d)$ , then*

$$b | \{r_i\} \sim \text{Gam}\left(\sum_{i=1}^K a_i + c, 1/(\sum_{i=1}^K r_i + d)\right).$$

**Lemma 2.1.3.** *If  $z_i \sim \mathcal{N}(\mu_i, \sigma^{-1}) \forall i \in \{1, 2, \dots, K\}$ ,  $\sigma \sim \text{Gam}(a, 1/b)$ , then*

$$\sigma | \{z_i\} \sim \text{Gam}\left(a + K/2, 1/(b + \sum_{i=1}^K (z_i - \mu_i)^2/2)\right).$$

**Lemma 2.1.4.** *Let  $x_k \sim \text{Pois}(\zeta_k) \forall k$ ,  $X = \sum_{k=1}^K x_k$ ,  $\zeta = \sum_{k=1}^K \zeta_k$ . If  $(y_1, \dots, y_K) \sim \text{mult}(X; \zeta_1/\zeta, \dots, \zeta_K/\zeta)$ , then the following holds:*

$$p(x_1, \dots, x_K) = p(y_1, \dots, y_K; X).$$

### 2.1.5 Dirichlet Distribution

**Dirichlet Distribution:** The Dirichlet distribution of order  $K \geq 2$  with parameters  $\alpha = (\alpha_1, \dots, \alpha_K) > 0$  has a probability density function with respect to Lebesgue measure on the Euclidean space  $\mathbb{R}^{(K-1)}$  given by:

$$f(x_1, \dots, x_{K-1}; \alpha) = \frac{1}{B(\alpha)} \prod_{k=1}^K x_k^{\alpha_k - 1},$$

on the open  $(K - 1)$ -dimensional simplex, which is defined by:

$$\sum_{k=1}^K x_k = 1.0, x_k > 0 \forall k.$$

The normalizing constant is the multinomial Beta function, which can be expressed in terms of the gamma function:  $B(\alpha) = \frac{\prod_{k=1}^K \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^K \alpha_k)}$ . Additionally, we have:  $E[X_k] = \frac{\alpha_k}{\alpha_0}$ ,  $\text{Var}[X_k] = \frac{\alpha_k(\alpha_0 - \alpha_k)}{\alpha_0^2(\alpha_0 + 1)}$ , where  $\alpha_0 = \sum_{k=1}^K \alpha_k$ .

**Relation with Gamma Distribution:** For  $K$  independently distributed Gamma distributions:  $Y_1 \sim \text{Gam}(\alpha_1, \theta), \dots, Y_K \sim \text{Gam}(\alpha_K, \theta)$ , one has the following:  $V = \sum_{k=1}^K Y_k \sim \text{Gam}(\sum_{k=1}^K \alpha_k, \theta)$ ,  $X = (X_1, \dots, X_K) = (\frac{Y_1}{V}, \dots, \frac{Y_K}{V}) \sim \text{Dir}(\alpha_1, \dots, \alpha_K)$ .

### 2.1.6 Negative Binomial Distribution

**Negative Binomial Distribution:** The negative binomial (NB) distribution  $m \sim \text{NB}(r, p)$  has probability mass function:

$$f(m) = \frac{\Gamma(m+r)}{m!\Gamma(r)} p^m (1-p)^r \text{ for } m \in \mathbb{Z}.$$

NB variables can be constructed *via* augmentation into a gamma-Poisson construction as  $m \sim \text{Pois}(\lambda)$ ,  $\lambda \sim \text{Gam}(r, p/(1-p))$ , where the gamma distribution is parameterized by its shape  $r$  and scale  $p/(1-p)$ .

This construction can be extended *via* the following lemma:

**Lemma 2.1.5.** *If  $\lambda \sim \text{Gam}(r, 1/c)$ ,  $x_i \sim \text{Poisson}(m_i \lambda)$ , then  $x = \sum_i x_i \sim \text{NB}(r, p)$ , where  $p = \frac{\sum_i m_i}{c + \sum_i m_i}$ .*

## 2.2 Random Processes

In this section, we introduce a few random processes which are used in the nonparametric models proposed herein. We start with the formal definition of a random process and then explain Gamma Process, Dirichlet Process, Chinese Restaurant Process and Hierarchical Dirichlet Process in sequence.

### 2.2.1 Random Process

**Random Process:** Let  $(\Omega, \mathcal{F}, P)$  be a probability space, and  $(E, \mathcal{E})$  a measurable space. A random element with values in  $E$  is a function  $X : \Omega \rightarrow E$  which is  $(\mathcal{F}, \mathcal{E})$ -measurable. That is, a function  $X$  such that for any  $B \in \mathcal{E}$  the pre-image

of  $B$  lies in  $\mathcal{F} : \{\omega : X(\omega) \in B\} \in \mathcal{F}$ . Given a probability space  $(\Omega, \mathcal{F}, P)$  and a measurable space  $(E, \mathcal{E})$ , an  $E$ -valued *random process* is a collection of  $E$ -valued random elements on  $\Omega$ .

### 2.2.2 Gamma Process (GP)

**Gamma Process:** Following [Wolpert et al., 2011], for any  $\nu^+ \geq 0$  and any probability distribution  $\pi(dp d\omega)$  on the product space  $\mathbb{R} \times \Omega$ , let  $K^+ \sim \text{Pois}(\nu^+)$  and  $(p_k, \omega_k) \stackrel{iid}{\sim} \pi(dp d\omega)$  for  $k = 1, \dots, K^+$ . Defining  $\mathbf{1}_A(\omega_k)$  as being one if  $\omega_k \in A$  and zero otherwise, the random measure  $\mathcal{L}(A) \equiv \sum_{k=1}^{K^+} \mathbf{1}_A(\omega_k) p_k$  assigns independent infinitely divisible random variables  $\mathcal{L}(A_i)$  to disjoint Borel sets  $A_i \subset \Omega$ , with characteristic functions:

$$\mathbb{E}[e^{it\mathcal{L}(A)}] = \exp\left(\int \int_{\mathbb{R} \times A} (\exp^{itp} - 1) \nu(dp d\omega)\right), \quad (2.1)$$

where  $\nu(dp d\omega) \equiv \nu^+ \pi(dp d\omega)$ . A random signed measure  $\mathcal{L}$  satisfying the above characteristic function is called a Lévy random measure. More generally, if the Lévy measure  $\nu(dp d\omega)$  satisfies  $\int \int_{\mathbb{R} \times S} \min\{1, |p|\} \nu(dp d\omega) < \infty$  for each compact  $S \subset \Omega$ , the Lévy random measure  $\mathcal{L}$  is well defined, even if the Poisson intensity  $\nu^+$  is infinite. A nonnegative Lévy random measure  $\mathcal{L}$  satisfying the integration condition is called a completely random measure [Kingman, 1967, 1993] which was introduced to machine learning in [Jordan, 2010; Thibaux and Jordan, 2007]. The Gamma Process [Ferguson, 1973; Wolpert et al., 2011]  $G \sim \Gamma P(c, H)$  is a completely random measure defined on the product space  $\mathbb{R}_+ \times \Omega$ , with concentration parameter  $c$  and a finite and continuous base measure  $H$  over a complete

separable metric space  $\Omega$ , such that  $G(A_i) \sim \text{Gam}(H(A_i), 1/c)$  are independent gamma random variables for disjoint partition  $\{A_i\}_i$  of  $\Omega$ .

The Lévy measure of the Gamma Process can be expressed as  $\nu(drd\omega) = r^{-1}e^{-cr}drH(d\omega)$ . Since the Poisson intensity  $\nu^+ = \nu(\mathbb{R}_+ \times \Omega) = \infty$  and the value of  $\int_{\mathbb{R}_+ \times \Omega} r\nu(drd\omega)$  is finite, a draw from the Gamma Process consists of countably infinite atoms, which can be expressed as follows:

$$G = \sum_{k=1}^{\infty} r_k \delta_{\omega_k}, \quad (r_k, \omega_k) \stackrel{iid}{\sim} \pi(drd\omega), \quad \pi(drd\omega)\nu^+ \equiv \nu(drd\omega). \quad (2.2)$$

A gamma process based model has an inherent shrinkage mechanism, as in the prior the number of atoms with weights greater than  $\tau \in \mathbb{R}_+$  follows a Poisson distribution with parameter  $H(\Omega) \int_{\tau}^{\infty} r^{-1} \exp(-cr) dr$ , the value of which decreases as  $\tau$  increases.

### 2.2.3 Dirichlet Process (DP)

**Dirichlet Process:** [Antoniak, 1974] Denote  $\tilde{G} = G/G(\Omega)$ , where  $G \sim \Gamma P(c, G_0)$ , then for any measurable disjoint partition  $\mathcal{A}_1, \dots, \mathcal{A}_Q$  of  $\Omega$ , we have

$$[\tilde{G}(\mathcal{A}_1), \dots, \tilde{G}(\mathcal{A}_Q)] \sim \text{Dir}(\gamma_0 \tilde{G}_0(\mathcal{A}_1), \dots, \tilde{G}_0(\mathcal{A}_Q)), \quad (2.3)$$

where  $\gamma_0 = G_0(\Omega)$  and  $\tilde{G}_0 = G_0/\gamma_0$ . With a space invariant concentration parameter, the normalized gamma process  $\tilde{G} = G/G(\Omega)$  is a Dirichlet process with concentration parameter  $\gamma_0$  and base probability measure  $\tilde{G}_0$ , expressed as  $\tilde{G} \sim DP(\gamma_0, \tilde{G}_0)$ .

**Chinese Restaurant Process (CRP): [Pitman, 2006]** In a Dirichlet process  $\tilde{G} \sim \text{DP}(\gamma_0, \tilde{G}_0)$ , if one assumes that  $\mathbf{X}_i \sim \tilde{G}$ , then  $\{\mathbf{X}_i\}$ 's are independent given  $\tilde{G}$  and hence exchangeable. The predictive distribution of a new data point  $\mathbf{X}_{(m+1)}$ , conditioned on  $\mathbf{X}_1, \dots, \mathbf{X}_m$ , with  $\tilde{G}$  marginalized out, can be expressed as:

$$\mathbf{X}_{(m+1)} \sim \mathbf{X}_1, \dots, \mathbf{X}_m = \mathbb{E}[\tilde{G} | \mathbf{X}_1, \dots, \mathbf{X}_m] = \sum_{k=1}^K \frac{n_k}{m + \gamma_0} \delta_{\omega_k} + \frac{\gamma_0}{m + \gamma_0} \tilde{G}_0, \quad (2.4)$$

where  $\{\omega_k\}_{k=1}^K$  are discrete atoms in  $\Omega$  observed in  $\mathbf{X}_1, \dots, \mathbf{X}_m$ .  $n_k = \sum_{i=1}^m \mathbf{X}_i(\omega_k)$  is the number of data points associated with  $\omega_k$ . The stochastic process described in Eq. (2.4) is known as the Chinese restaurant process. We now introduce a relevant distribution.

**Chinese Restaurant Table Distribution (CRT):** Under the Chinese restaurant process metaphor, the number of data points  $m$  is assumed to be known whereas the number of distinct atoms  $K$  is treated as a random variable dependent on  $m$  and  $\gamma_0$ . Let  $s(m, l)$  be the Stirling number of the first kind. Then it is shown in [Antoniak, 1974] that the random number of distinct atoms  $K$  has the following PMF:

$$\Pr(K = l | m, \gamma_0) = \frac{\Gamma(\gamma_0)}{\Gamma(m + \gamma_0)} |s(m, l)| \gamma_0^l, \quad l = 0, 1, \dots, m. \quad (2.5)$$

This distribution is referred to as the Chinese restaurant table (CRT) distribution and denoted by  $l \sim \text{CRT}(m, \gamma_0)$ , a CRT random variable.

We explained in Section 2.1.6 how the NB distribution can be augmented using a Gamma-Poisson construction. Interestingly, it can also be augmented under a compound Poisson representation [Zhou and Carin, 2012, 2015] as  $m =$

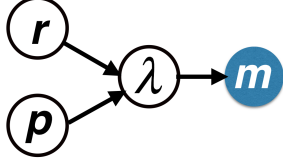


Figure 2.1: Gamma-Poisson Construction of NB Distribution

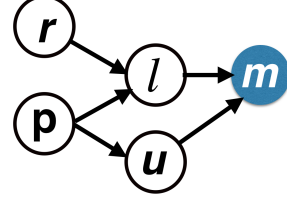


Figure 2.2: Compound Poisson Construction of NB Distribution

$\sum_{t=1}^l u_t, u_t \stackrel{iid}{\sim} \text{Log}(p), l \sim \text{Pois}(-r \ln(1 - p))$ , where  $u \sim \text{Log}(p)$  is the logarithmic distribution [Johnson et al., 2005]. These two different constructions are shown graphically in Fig. 2.1 and Fig. 2.2, and together they lead to the following lemma:

**Lemma 2.2.1.** [Zhou and Carin, 2012, 2015] *If  $m \sim \text{NB}(r, p)$  is represented under its compound Poisson representation, then the conditional posterior of  $l$  given  $m$  and  $r$  has PMF:*

$$\Pr(l = j | m, r) = \frac{\Gamma(r)}{\Gamma(m + r)} |s(m, j)| r^j, \quad j = 0, 1, \dots, m,$$

where  $|s(m, j)|$  are unsigned Stirling numbers of the first kind. The conditional posterior as  $l | m, r \sim \text{CRT}(m, r)$ , a Chinese restaurant table (CRT) count random variable, which can be generated via  $l = \sum_{n=1}^m z_n, z_n \sim \text{Bernoulli}(r / (n - 1 + r))$ .

This lemma leads to the next lemma, which provides closed form sampling of the gamma shape parameter *via* CRT data augmentation in the gamma-gamma-Poisson framework. We explain this lemma more thoroughly in Section 7.1.1 as this is the key to deriving closed form inference for the dynamic models introduced in Chapter 7.

**Lemma 2.2.2.** *If  $r_1 \sim \text{Gam}(a, 1/b)$ ,  $r_2 \sim \text{Gam}(r_1, 1/d)$ ,  $x_i \sim \text{Poisson}(m_i r_2) \forall i$ , then  $r_1 | \{x_i\} \sim \text{Gam}(a + \ell, 1/(b - \log(1 - p)))$  where  $\ell \sim \text{CRT}(\sum_i x_i, r_1), p = \sum_i m_i / (d + \sum_i m_i) \forall i$ .*

#### 2.2.4 Hierarchical Dirichlet Process (HDP)

Hierarchical Dirichlet Process (HDP), a convenient tool for sharing clusters among multiple related groups, follows the generative process mentioned below:

$$G_0 | \gamma_0, H \sim \text{DP}(\gamma_0, H), G_j | \alpha_0, G_0 \sim \text{DP}(\alpha_0, G_0) \forall j, \quad (2.6)$$

where  $j$  indexes the group. There are multiple explanations available for HDP, namely the Chinese restaurant franchise, infinite limit of a finite hierarchical mixture model [Teh et al., 2006] and the stick breaking process [Sethuraman, 1994; Teh et al., 2006]. A detailed discussion of HDP is beyond the scope of this thesis and the interested readers can check [Teh et al., 2006] for a thorough understanding of its theoretical properties. In Section 2.4.2, we describe HDP in the context of non-parametric topic models in more details.

### 2.3 Matrix Factorization

Matrix factorization [Gopalan et al., 2014a; Koren et al., 2009; Salakhutdinov and Mnih, 2007, 2008] has been widely applied for solving numerous problems related to analysis of dyadic data, such as in topic modeling [Arora et al., 2012], recommender systems [Koren et al., 2009] and network analysis [Zhou, 2015]. Tensor factorization [Ho et al., 2014a,b] is an extension of matrix factorization where the

data is represented as a three dimensional array, signifying interactions among three different sets of variables. Typically, matrix factorization is used to recover a low-rank latent structure of a matrix by approximating it as a product of two low rank matrices, whose elements are real numbers. A given matrix, such as a document-by-word matrix  $\mathbf{X} \in \mathbb{R}^{N \times D}$ , is usually decomposed into two low rank matrices  $\mathbf{U} \in \mathbb{R}^{N \times K}$  and  $\mathbf{V} \in \mathbb{R}^{D \times K}$  such that  $\mathbf{X} \sim \mathbf{UV}^\dagger$ . Many specialized factorization models have further been proposed to deal with non-categorical variables. For example, SVD++ uses neighborhood of a user for analysis of user-movie rating data [Koren, 2008], TimeSVD++ [Koren, 2009] and Bayesian Temporal collaborative filtering with Bayesian probabilistic tensor factorization (BPTF) [Xiong et al., 2010] discretize time which is a continuous variable, and Factorizing Personalized Markov Chains for Next-Basket Recommendation (FPMC) [Rendle et al., 2010] considers the purchase history of users to recommend items. Numerous learning techniques have also been proposed for factorization models which include SGD [Koren et al., 2009], alternating least-squares [Zhou et al., 2008], variational Bayes [Kim and Choi, 2014; Lim and Teh, 2007; Silva and Carin, 2012] and MCMC based inference [Salakhutdinov and Mnih, 2008].

In many practical applications, the negative values obtained in the factors from the matrix factorization are difficult to interpret. Positive matrix factorization [Paatero and Tapper, 1994], non-negative matrix factorization [Lee and Seung, 2001], and non-negative independent component analysis [Plumbley and Oja, 2004] are techniques that perform factorization in positively constrained components. The factors herein are interpretable as they represent the assignment score of certain en-

tities to corresponding latent factors. Although these methods are fast and stable under relatively mild assumptions, they lack clear probabilistic generative semantics. Bayesian formulations of similar ideas have also been proposed [Højén-Sørensen et al., 2002; Miskin, 2000] in which the positivity is imposed using rectified Gaussian distribution [Socci et al., 1998], exponential distribution or a mixture of both [Hoffman et al., 2010]. A large number of discrete latent variable models for count matrix factorization can be united under Poisson factor analysis (PFA) [Zhou et al., 2012], which factorizes a count matrix  $\mathbf{Y} \in \mathbb{Z}^{D \times V}$  under the Poisson likelihood as  $\mathbf{Y} \sim \text{Pois}(\Phi\Theta)$ , where  $\Phi \in \mathbb{R}_+^{D \times K}$  is the factor loading matrix or dictionary,  $\Theta \in \mathbb{R}_+^{K \times V}$  is the factor score matrix and hence is an example of non-negative matrix factorization. In Chapter 6 and 7, we will see many applications of PFA and hence a brief discussion of the related works is presented below.

A wide variety of algorithms, although constructed with different motivations and for distinct problems, can all be viewed as PFA with different prior distributions imposed on  $\Phi$  and  $\Theta$ . For example, non-negative matrix factorization [Cemgil, 2009; Lee and Seung, 2001], with the objective to minimize the Kullback-Leibler divergence between  $\mathbf{N}$  and its factorization  $\Phi\Theta$ , is essentially PFA solved with maximum likelihood estimation. Latent Dirichlet Allocation (LDA) [Blei et al., 2003] is equivalent to PFA, in terms of both block Gibbs sampling and variational inference, if Dirichlet distribution priors are imposed on both  $\phi_k \in \mathbb{R}_+^D$ , the columns of  $\Phi$ , and  $\theta_k \in \mathbb{R}_+^V$ , the columns of  $\Theta$  [Zhou et al., 2012]. The gamma-Poisson model [Canny, 2004; Titsias, 2008] is PFA with gamma priors on  $\Phi$  and  $\Theta$ . A family of negative binomial (NB) processes, such as the beta-NB [Broderick

et al., 2013; Zhou et al., 2012] and gamma-NB processes [Zhou and Carin, 2012, 2015], impose different gamma priors on  $\{\theta_{vk}\}$ , the marginalization of which leads to differently parameterized NB distributions to explain the latent counts. For example, the beta-NB process imposes  $\theta_{vk} \sim \text{Gamma}(r_v, p_k/(1 - p_k))$ , where  $\{p_k\}_{1,\infty}$  are the weights of the countably infinite atoms of the beta process [Hjort, 1990], and the gamma-NB process imposes  $\theta_{vk} \sim \text{Gamma}(r_k, p_v/(1 - p_v))$ , where  $\{r_k\}_{1,\infty}$  are the weights of the countably infinite atoms of the gamma process. Both the beta-NB and gamma-NB process PFAs are nonparametric Bayesian models that allow  $K$  to grow without limits [Hjort, 1990].

## 2.4 Statistical Topic Models

In this section, we introduce statistical topic models which are used as the building blocks for models used in Chapter 3, 4, and 5. We first introduce parametric topic model in Section 2.4.1, in which one needs to set the value of  $K$ , the number of latent topics in advance. In Section 2.4.2, we describe generalizations of parametric topic model that are able to handle potentially infinite number of latent topics and can even infer the ideal number of latent topics from the data.

### 2.4.1 Parametric Topic Models

Latent Dirichlet Allocation (LDA) [Blei et al., 2003] treats documents as a mixture of topics, which in turn are defined by a distribution over a set of words. The words in a document are assumed to be sampled from multiple topics. In its original formulation, unsmoothed LDA can be viewed as a purely-unsupervised

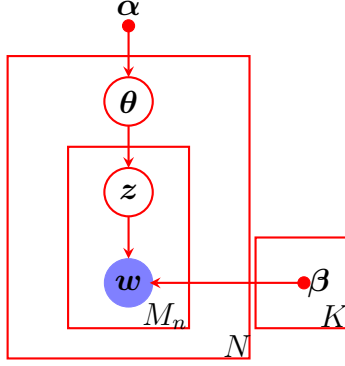


Figure 2.3: LDA

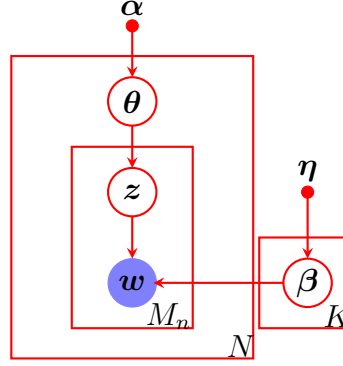


Figure 2.4: Smoothed LDA

form of dimensionality reduction and clustering of documents in the topic space. On a high level, LDA can be thought of as performing a non-negative matrix factorization [Buntine, 2002; Zhou et al., 2012]. The graphical model of LDA is shown in Fig 2.3. The generative process of LDA is described below:

- For the  $n^{\text{th}}$  document, sample a topic selection probability vector  $\theta_n \sim \text{Dir}(\alpha)$ , where  $\alpha$  is the parameter of a Dirichlet distribution of dimension  $K$ , which is the total number of topics.
- For the  $m^{\text{th}}$  word in the  $n^{\text{th}}$  document, sample a topic  $z_{nm} \sim \text{multinomial}(\theta_n)$ .
- Sample the word  $w_{nm} \sim \text{multinomial}(\beta_{z_{nm}})$ , where  $\beta_k$  is a multinomial distribution over the vocabulary of words corresponding to the  $k^{\text{th}}$  topic.
- For a smoothed LDA model, shown in Fig. 2.4, one additionally samples  $\beta_k \sim \text{Dir}(\eta)$ , where  $\eta$  is the parameter of a Dirichlet distribution of dimension  $V$ , which is the size of the vocabulary.

## 2.4.2 Nonparametric Topic Models

Nonparametric topic models are built on HDP. As explained in Section 2.2.4, there are multiple explanations available for HDP, namely the Chinese restaurant franchise, infinite limit of a finite hierarchical mixture model [Teh et al., 2006] and the stick breaking process [Sethuraman, 1994; Teh et al., 2006]. In what follows, we describe two different stick breaking constructions of HDP.

### 2.4.2.1 Traditional Stick Breaking Construction of HDP Topic Model

An explicit representation of a draw from a DP was given by Sethuraman [1994], who showed that if  $G \sim \text{DP}(\alpha_0; G_0)$ , then with probability one:  $G = \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k}$  where the  $\phi_k$ 's are independent random variables distributed according to  $G_0$  and  $\delta_{\phi_k}$  is an atom at  $\phi_k$ . The “stick-breaking weights”  $\beta_k$  are random and depend on the parameter  $\alpha_0$ . The representation above shows that draws from a DP are discrete with probability one. This discrete nature of the DP makes it suitable for the problem of placing priors on mixture components in mixture modeling where a mixture component can be associated with each atom in  $G$ .

To force  $G_0$  to be discrete and yet have broad support,  $G_0$  itself is drawn from a Dirichlet process  $\text{DP}(\gamma_0, H)$ . The atoms in  $\phi_k$  are shared among the multiple DPs, yielding the desired sharing of atoms among groups [Teh et al., 2006]. For HDP topic model, the lower level measures correspond to each document. The generative process for the first stage of HDP topic model goes as follows:

- $\forall k \in \{1, 2, \dots, \infty\}$ , generate  $\beta'_k \sim \text{Beta}(1, \gamma_0)$ . Set  $\beta_k = \beta'_k \prod_{j=1}^{k-1} (1 - \beta'_j)$ .

- Generate  $\phi_k \sim H$ , where  $H$  is some distribution.
- Create the base distribution  $G_0$  following the summation:  $G_0 = \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k}$ .

The generative process for the second stage is illustrated below:

- $\forall t \in \{1, 2, \dots, \infty\}$ , generate  $\pi'_{nt} \sim \text{Beta}(\alpha_0 \beta_t, \alpha_0(1 - \sum_{l=1}^t \beta_l))$ . Set  $\pi_{nt} = \pi'_{nt} \prod_{l=1}^{t-1} (1 - \pi'_{nl})$ .
- Create document specific distribution  $G_n = \sum_{t=1}^{\infty} \pi_{nt} \delta_{\phi_t}$ . Note that here  $\pi'_{nt}$ 's are generated in a way so that there is some sharing of atom weights across the DPs corresponding to different documents. This sharing of parameters is essential for maintaining the hierarchical structure assumed in the parametric LDA model.

Any non-parametric topic model that uses this particular view of HDP uses either Gibbs sampling [Teh et al., 2006] or collapsed variational inference [Teh et al., 2007] for inference.

#### 2.4.2.2 Modified Stick Breaking Construction of HDP Topic Model

Wang et al. [2011a] proposed the following modification to the second stage of the stick breaking construction of HDP as shown below:

- $\forall t \in \{1, 2, \dots, \infty\}$ , generate  $\psi_{nt} \sim G_0$ .
- $\forall t \in \{1, 2, \dots, \infty\}$ , generate  $\pi'_{nt} \sim \text{Beta}(1, \alpha_0)$ . Set  $\pi_{nt} = \pi'_{nt} \prod_{l=1}^{t-1} (1 - \pi'_{nl})$ .
- Create data specific distribution:  $G_n = \sum_{t=1}^{\infty} \pi_{nt} \delta_{\psi_{nt}}$ .

Essentially, this sampling process avoids the complicated sharing of atom weights in lower level measures and suggests a new method which is amenable for variational inference without compromising the hierarchical structure. Interested reader can explore [Wang et al., 2011a; Wang and Carin, 2012] for a more comprehensive discussion of this modification. We use this view of HDP in the nonparametric models proposed in Section 3.3 and 4.3.

### 2.4.3 Inference in Topic Models

Inference of the latent variables in the topic models is usually solved using two different techniques: variational approximation [Blei et al., 2003; Wang et al., 2011a] and Gibbs sampling [Porteous et al., 2008]. We do not use Gibbs sampling for LDA type models in this thesis. Therefore, we describe only the variational methods for inference in LDA. The joint distribution over hidden and observed variables in unsupervised (unsmoothed) LDA, shown in Fig. 2.3, can be written as follows:

$$p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\kappa}) = \prod_{n=1}^N p(\boldsymbol{\theta}_n | \boldsymbol{\alpha}) \prod_{m=1}^{M_n} p(z_{nm} | \boldsymbol{\theta}_n) p(w_{nm} | \boldsymbol{\beta}_{z_{nm}}). \quad (2.7)$$

The exact inference using EM is intractable due to the coupling between  $\boldsymbol{\beta}$  and  $\mathbf{Z}$  and hence variational EM is utilized. Here, the posterior distribution over the hidden variables is approximated by a completely factorized one as given below:

$$q(\mathbf{Z} | \boldsymbol{\kappa}^v) = \prod_{n=1}^N q(\boldsymbol{\theta}_n | \boldsymbol{\gamma}_n) \prod_{m=1}^{M_n} q(z_{nm} | \boldsymbol{\phi}_n), \quad (2.8)$$

where  $\boldsymbol{\kappa}^v$  is the set of free variational parameters.

### 2.4.3.1 Batch Variational Inference in Topic Models

In batch variational inference for LDA, the following **Evidence Lower Bound** (ELBO) on the log-likelihood of the observed data is maximized w.r.t the model parameters  $\kappa$  and the variational parameters  $\kappa^v$ :

$$\mathcal{L}(\kappa^v, \kappa) = \mathbb{E}_q[\log p(\mathbf{X}, \mathbf{Z}|\kappa)] - \mathbb{E}_q[\log q(\mathbf{Z}|\kappa^v)] \leq \log p(\mathbf{X}|\kappa). \quad (2.9)$$

Using the factorization structure of both  $p(\cdot)$  and  $q(\cdot)$ , one can see that the ELBO decomposes as follows:

$$\begin{aligned} \mathcal{L}(\kappa^v, \kappa) &= \sum_{n=1}^N \mathbb{E}_q[\log p(\mathbf{w}_n|\mathbf{z}_n, \beta)] + \mathbb{E}_q[\log p(\theta_n|\alpha)] \\ &+ \mathbb{E}_q[\log p(\mathbf{z}_n|\theta_n)] - \mathbb{E}_q[\log q(\theta_n|\gamma_n)] - \mathbb{E}_q[\log q(\mathbf{z}_n|\phi_n)] = \sum_{n=1}^N \ell(\kappa_n^v, \kappa). \end{aligned} \quad (2.10)$$

Here  $\ell(\kappa_n^v, \kappa)$  denotes the contribution in the ELBO by the  $n^{\text{th}}$  document.  $\mathcal{L}$  is optimized using coordinate ascent over each set of model and variational parameters. In the E-step,  $\forall n, \ell(\kappa_n^v, \kappa)$  is maximized w.r.t  $\gamma_n$  and  $\phi_n$ . In the M step, the ELBO is maximized w.r.t the model parameters  $\kappa$ . The algorithm, presented in 1, has constant memory requirements and empirically converges faster than batch collapsed Gibbs sampling [Asuncion et al., 2009].

### 2.4.3.2 Incremental EM Algorithm

The EM algorithm proposed by Dempster et al. [1977] can be viewed as a joint maximization problem over  $q(\cdot)$ , the conditional distribution of the hidden variables  $\mathbf{Z}$  given the model parameters  $\kappa$  and the observed variables  $\mathbf{X}$ . The rele-

---

**Algorithm 1** Batch Variational Bayes for LDA
 

---

**Input:**  $\mathbf{X}$ .

**Output:**  $\kappa$ .

Initialize  $\{\gamma^n\}_{n=1}^N, \{\phi_n\}_{n=1}^N$  randomly.

Until Convergence

**E-Step**

**for**  $n = 1 : N$

**for**  $m = 1 : M_n$

**for**  $k = 1 : K$

$$\phi_{nmk} \propto \beta_{kw_{nm}} \exp(\Psi(\gamma_{nk})).$$

**end**

    Normalize  $\phi_{nm}$ .

$$\gamma_{nk} = \alpha_k + \sum_{m=1}^{M_n} \phi_{nmk}.$$

**end**

**end**

**M-Step**

Update  $\alpha$  using Newton-Raphson method.

$$\beta_{kv} \propto \sum_{n=1}^N \sum_{m=1}^{M_n} \phi_{nmk} \mathbb{I}_{\{w_{nm}=v\}}.$$

Normalize  $\beta_k \forall k$ .

---

vant objective function is given as follows:

$$F(q, \kappa) = \mathbb{E}_q[\log(p(\mathbf{X}, \mathcal{Z}|\kappa))] + H(q), \quad (2.11)$$

where  $H(q)$  is the entropy of the distribution  $q(\cdot)$ . Often,  $q(\cdot)$  is restricted to a family of distributions  $\mathcal{Q}$ . It can be shown that if  $\kappa^*$  is the maximizer of the above objective  $F$  then it also maximizes the likelihood of the observed data. Therefore, another representation of the  $t^{\text{th}}$  step of the EM algorithm is as follows:

- **E step:**  $q^{(t)} = \underset{q}{\operatorname{argmax}} F(q, \kappa^{(t-1)}).$

- **M step:**  $\kappa^{(t)} = \underset{\kappa}{\operatorname{argmax}} F(q^{(t)}, \kappa)$ .

In most of the models used in practice, the joint distribution is assumed to factorize over the instances implying that  $p(\mathbf{X}, \mathbf{Z}|\kappa) = \prod_{n=1}^N p(\mathbf{x}_n, \mathbf{z}_n|\kappa)$ . One can further restrict the family of distributions  $\mathcal{Q}$  to maximize over in Eq. (2.11) to the factorized form:  $q(\mathbf{Z}) = \prod_{n=1}^N q(\mathbf{z}_n|\mathbf{x}_n) = \prod_{n=1}^N q_n$ .

An incremental variant of the EM algorithm that exploits such separability structure in both  $p(\cdot)$  and  $q(\cdot)$  was first proposed by Neal and Hinton [1999]. Under such structure, the objective function in Eq. (2.11) decomposes over the observations  $F(q, \theta) = \sum_{n=1}^N F_n(q_n, \kappa)$ , and the following incremental algorithm can instead be used to maximize  $F$ :

- **E step:** Choose some observation  $n$  to be updated over, set  $q_{n'}^{(t)} = q_{n'}^{(t-1)}$  for  $n' \neq n$  (no update) and set  $q_n^{(t)} = \underset{q_n}{\operatorname{argmax}} F_n(q_n, \kappa^{(t-1)})$ .
- **M step:**  $\kappa^{(t)} = \underset{\kappa}{\operatorname{argmax}} F(q^{(t)}, \kappa)$ .

Such an incremental view of the EM algorithm is useful for updating parameters in the proposed models in Section 4.2, 4.3 and 5.2 for active query selection.

## 2.5 Transfer Learning

Transfer learning [Pan and Yang, 2010] allows the learning of models specific to some domains to benefit from the learning of other models from other domains through either simultaneous [Caruana, 1997] or sequential [Bollacker and

Ghosh, 2000] training. Pan and Yang [2010] categorizes transfer learning into three primary groups – 1. inductive transfer learning, 2. transductive transfer learning, and 3. unsupervised transfer learning. In inductive transfer learning, the labels/tasks are different for different domains and the knowledge from one domain is shared with the learning problems in other related domains simultaneously or sequentially. In multitask learning (MTL [Caruana, 1997]), a type of inductive transfer learning, a single model is simultaneously trained to perform multiple related tasks. MTL has emerged as a very promising research direction for various applications including biomedical informatics [Bickel et al., 2008], marketing [Evgeniou et al., 2007], natural language processing [Ando, 2006], and computer vision [Torrallba et al., 2007]. Many different MTL approaches have been proposed over the past 15 years (*e.g.*, see [Pan and Yang, 2010; Passos et al., 2012; Weinberger et al., 2009] and references therein). These include different learning methods, such as empirical risk minimization using group-sparse regularizers [Jenatton et al., 2011; Kim and Xing, 2010], hierarchical Bayesian models [Low et al., 2011; Zhang et al., 2008] and hidden conditional random fields [Quattoni et al., 2007]. Evgeniou et al. [2005] proposed the regularized MTL which constrained the models of all tasks to be close to each other. The task relatedness in MTL has also been modeled by constraining multiple tasks to share a common underlying structure [Argyriou et al., 2007; Ben-David and Schuller, 2003; Caruana, 1997]. Ando and Zhang [2005] proposed a structural learning formulation, which assumed multiple predictors for different tasks shared a common structure on the underlying predictor space. In all of the MTL formulations mentioned above, the basic assumption is that all tasks are re-

lated. In practical applications, these might not be the case and the tasks might exhibit a more sophisticated group structure. Such structure is handled using clustered multi-task learning (CMTL). In [Bakker and Heskes, 2003] CMTL is implemented by considering a mixture of Gaussians instead of single Gaussian priors. Xue et al. [2007] introduced the Dirichlet process prior that automatically identifies subgroups of related tasks. In [Jacob et al., 2008], a clustered MTL framework was proposed that simultaneously identified clusters and performed multi-task inference. In Chapter 3, 4, 5, we explore the applications of multitask learning.

In another formulation of inductive transfer learning framework, models learnt from a “source” domain are utilized in the “target” domain and improving the learning capabilities in the target domain is the primary objective. This formulation had traditionally been known as transfer learning before Pan and Yang [2010] introduced a categorization of several existing transfer learning problems. In Chapter 8, we explore an application of this type of transfer learning where model learnt from a source domain is carefully adapted to fit the data from the new domain.

In unsupervised transfer learning framework, the domains involved do not contain any labeled information at all. Rather, the unsupervised learning problems in multiple domains are solved either sequentially or simultaneously. In Chapter 6 and 7 we explore applications of unsupervised transfer learning frameworks. In Chapter 6, the observations from two different domains are utilized for individual clustering of the data from these domains *simultaneously*. Chapter 7 deals with time-series data, where the learning problems for the individual time instances (or domains) affect the learning problems in other time instances (or domains) *sequen-*

tially.

Due to the existing ambiguity regarding some of the terminologies in the transfer learning literature, we adopt the term “knowledge transfer” to indicate the diverse categories of learning algorithms proposed herein. Whenever the domains are learnt simultaneously, we address such framework as “simultaneous knowledge transfer” and when the domains are learnt in sequence, we address that framework as “sequential knowledge transfer”. Chapter 3, 4, 5, and 6 are examples of simultaneous knowledge transfer frameworks. Chapter 7 and 8 illustrate how sequential knowledge transfer can be utilized to solve real-world problems.

## **2.6 Active Learning *via* Expected Error Reduction**

Active learning is a subfield of machine learning where the key hypothesis is that if the learning algorithm is allowed to choose the data from which it learns, it will perform better with less training [Settles, 2009]. This is especially pertinent in supervised learning systems where labeled data is often expensive and/or hard to come by. Active learning systems attempt to overcome this labeling bottleneck by asking “informative” queries in the form of unlabeled instances to be labeled by an oracle (e.g., a human annotator). Fig. 2.5 shows an illustration of an active learning system.

Of the several measures for selecting labels in active learning algorithms, a decision-theoretic approach called Expected Error Reduction [Roy and McCallum, 2001] has been used quite extensively in practice [Kovashka et al., 2011; Settles, 2009]. This approach aims to measure how much the generalization error of a

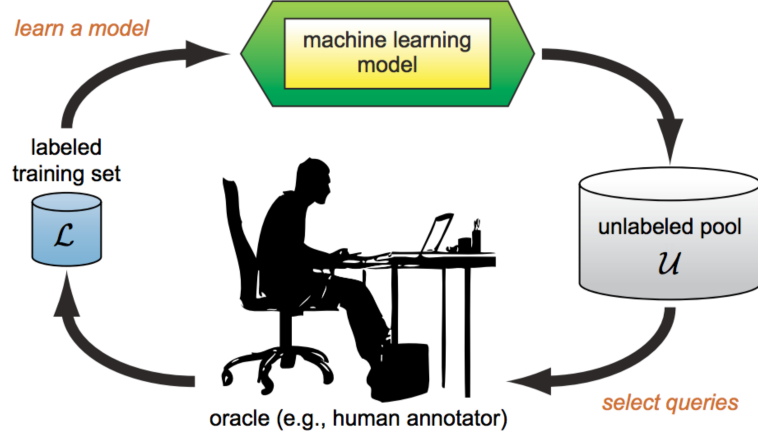


Figure 2.5: Illustration of Active Learning [Settles, 2009]

model is likely to be reduced based on some labeled information  $y$  of an instance  $\mathbf{x}$  taken from the unlabeled pool  $\mathcal{U}$ . The idea is to estimate the expected future error of a model trained using  $\mathcal{L} \cup \langle \mathbf{x}, y \rangle$  on the remaining unlabeled instances in  $\mathcal{U}$ , and query the instance with minimal expected future error. Here  $\mathcal{L}$  denotes the labeled pool of data. One approach is to minimize the expected 0/1 loss:

$$\mathbf{x}_{0/1}^* = \underset{\mathbf{x}}{\operatorname{argmax}} \sum_n P_{\kappa}(y_n | \mathbf{x}) \left( \sum_{u=1}^U 1 - P_{\kappa + \langle \mathbf{x}, y_n \rangle}(\hat{y}, \mathbf{x}^{(u)}) \right). \quad (2.12)$$

where  $\kappa + \langle \mathbf{x}, y_n \rangle$  refers to the new model after it has been re-trained with the training set  $\mathcal{L} \cup \langle \mathbf{x}, y_n \rangle$ . Note that we do not know the true label for each query instance, so we approximate using expectation over all possible labels under the current model. The objective is to reduce the expected number of incorrect predictions.

## Chapter 3

### Multitask Learning using Both Supervised and Shared Latent Topics

Humans can distinguish as many as 30,000 relevant object classes [Biederman, 1987]. Training an isolated object detector for each of these different classes would require millions of training examples in aggregate. Computer vision researchers have proposed a more efficient learning mechanism in which object categories are learned via *shared* attributes, abstract descriptors of object properties such as “striped” or “has four legs” [Farhadi et al., 2009; Kovashka et al., 2011; Lampert et al., 2009]. The attributes serve as an intermediate layer in a classifier cascade. The classifier in the first stage is trained to predict the attributes from the raw features and that in the second stage is trained to predict the categories from the attributes. During testing, only the raw features are observed and the attributes must be inferred. This approach is inspired by human perception and learning from high-level object descriptions. For example, from the phrase “eight-sided red traffic sign with white writing”, humans can detect stop signs [Lampert et al., 2009]. Similarly, from the description “large gray animals with long trunks”, human can identify elephants. If the *shared* attributes transcend object class boundaries, such a classifier cascade is beneficial for *transfer learning* [Pan and Yang, 2010] where fewer labeled examples are available for some object categories compared to others

[Lampert et al., 2009]. This representation is illustrated in Fig. 3.1.

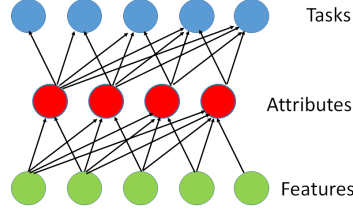


Figure 3.1: MTL with Shared Attributes

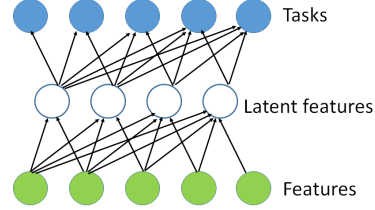


Figure 3.2: MTL with Multi-layer Perceptron

Multitask learning (MTL) is a form of transfer learning in which simultaneously learning multiple related “tasks” allows each one to benefit from the learning of all of the others. If the tasks are related, training one task should provide helpful “inductive bias” for learning the other tasks. To enable the reuse of training information across multiple related tasks, all tasks might utilize the same latent shared intermediate representation – for example, a shared hidden layer in a multi-layer perceptron [Caruana, 1997] (as shown in Fig. 3.2). In this case, the training examples for all tasks provide good estimates of the weights connecting the input layer to the hidden layer, and hence only a small number of examples per task is sufficient to achieve high accuracy. This approach is in contrast to “isolated” training of tasks where each task is learned independently using a separate classifier.

In this chapter, our objective is to combine these two approaches to build an MTL framework that can use *both* attributes *and* class labels. The multiple tasks in such setting correspond to different object categories (classes), and *both* observable attributes and latent properties are shared across the tasks. We want to emphasize that the proposed frameworks support general MTL; however, the datasets we use happen to be multiclass, where each class is treated as a separate

“task” (as typical in multi-class learning based on binary classifiers). But, in no way are the frameworks restricted to multiclass MTL. Since attribute-based learning has been shown to support effective transfer learning in computer vision, the tasks here naturally correspond to object classes.

The rest of the chapter is organized as follows. We present related literature in Section 3.1, followed by the descriptions of DSLDA and NP-DSLDA, two MTL frameworks in Section 3.2 and 3.3 respectively. Experimental results on both multi-class image and document categorization are presented in Section 3.4, demonstrating the value of integrating MTL, supervised topics and latent shared topics. Finally, future directions and conclusions are presented in Section 3.5.

### 3.1 Background

In this section, a brief introduction to supervised topic models is provided which are the building blocks for DSLDA and NP-DSLDA.

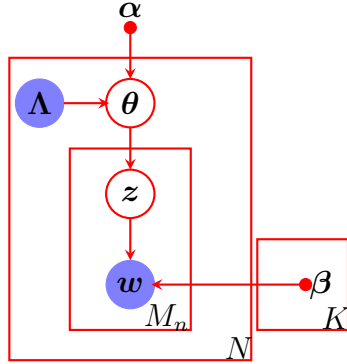


Figure 3.3: LLDA

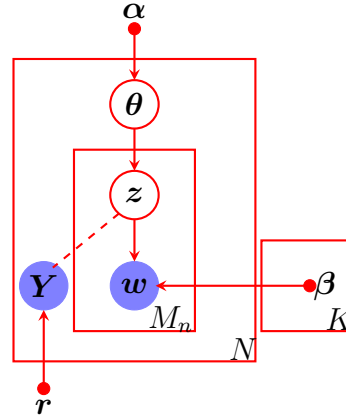


Figure 3.4: MedLDA

### 3.1.1 Labeled Latent Dirichlet Allocation (LLDA)

Several extensions of LDA have incorporated some sort of supervision. Some approaches provide supervision by labeling each document with its set of topics [Ramage et al., 2009; Rubin et al., 2011]. In particular, in *Labeled LDA* (LLDA [Ramage et al., 2009]), the primary objective is to build a model of the words that indicate the presence of certain topic labels. For example, when a user explores a webpage based on certain tags, LLDA can be used to highlight interesting portions of the page or build a summary of the text from multiple webpages that share the same set of tags. The words in a given training document are assumed to be sampled *only* from the supervised topics, which the document has been labeled as covering. The graphical model of LLDA is shown in Fig. 3.3.

### 3.1.2 Maximum Entropy Discriminant Latent Dirichlet Allocation (MedLDA)

Some other researchers [Blei and Mcauliffe, 2007; Chang and Blei, 2009; Zhu et al., 2009] assume that supervision is provided for a single *response variable* to be predicted for a given document. The response variable might be real-valued or categorical, and modeled by a normal, Poisson, Bernoulli, multinomial or other distribution (see Chang and Blei [2009] for details). Some examples of documents with response variables are essays with their grades, movie reviews with their numerical ratings, web pages with their number of hits over a certain period of time, and documents with category labels.

In *Maximum Entropy Discriminative LDA* (MedLDA) [Zhu et al., 2009], the objective is to infer some low-dimensional (topic-based) representation of doc-

uments which is predictive of the response variable. Essentially, MedLDA solves two problems jointly – dimensionality reduction and max-margin classification using the features in the dimensionally-reduced space. In earlier versions of supervised topic models [Blei and McAuliffe, 2007; Chang and Blei, 2009], categorical response variables were difficult to model since the resulting inference equations were complex. In particular, the use of Taylor’s approximations breaks the guarantee that the likelihood lower bound increases after each update. Compared to earlier versions of supervised topic models [Blei and McAuliffe, 2007; Chang and Blei, 2009], MedLDA has simpler update equations and produces superior experimental results.

In MedLDA, the generative process of the words in the documents are same as the unsupervised LDA. However, the topic space representation of the documents is treated as features for an SVM learning framework. In particular, for the  $n^{\text{th}}$  document, we generate  $Y_n = \arg \max_y \mathbf{r}_y^T \mathbb{E}(\bar{\mathbf{z}}_n)$  where  $Y_n$  is the class label associated with the  $n^{\text{th}}$  document,  $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm} / M_n$ . Here,  $\mathbf{z}_{nm}$  is an indicator vector of dimension  $K$ .  $\mathbf{r}_y$  is a  $K$ -dimensional real vector corresponding to the  $y^{\text{th}}$  class, and it is assumed to have a prior distribution  $\mathcal{N}(0, 1/C)$ .  $M_n$  is the number of words in the  $n^{\text{th}}$  document. The maximization problem to generate  $Y_n$  (or the classification problem) is carried out using a max-margin principle – the exact formulation of which will be discussed later using variational approximation. Since MedLDA includes discriminative modeling of the class labels, it is not possible to draw a plate model. However, for the ease of understanding, in Fig. 3.4, we show a representative plate model with the discriminative part denoted by dotted lines. MedLDA

has empirically shown very good performance, and is generally considered state of the art for class prediction in supervised topic modeling. Further development of MedLDA has led to the so-called Gibbs MedLDA [Zhu et al., 2013]. Gibbs MedLDA employs a Gibbs sampling based inference framework on a completely generative model instead of using variational approximations by using various ideas from Gibbs-based classifiers. However, it does this at the cost of native multiclass classification, requiring a one-versus-all framework to extend binary predictions to the multiclass setting.

### 3.2 Doubly Supervised LDA (DSLDA)

Assume we are given a training corpus consisting of  $N$  documents belonging to  $Y$  different classes (where each document belongs to exactly one class and each class corresponds to a different task). Further assume that each of these training documents is also annotated with a set of  $K_2$  different topic “tags” (henceforth referred to as “supervised topics”). For computer vision data, the supervised topics correspond to the attributes provided by human experts. The objective is to train a model using the words in a data, as well as the associated supervised topic tags and class labels, and then use this model to classify completely unlabeled test data for which no topic tags nor class labels are provided. The human-provided supervised topics are presumed to provide abstract information that is helpful in predicting the class labels of test documents.

### 3.2.1 Model Description

In order to include both types of supervision (class and topic labels), a combination of the approaches described at the beginning of this Chapter is proposed. Note that LLDA uses *only* supervised topics and does not have any mechanism for generating class labels. On the other hand, MedLDA has only *latent* topics but learns a discriminative model for predicting classes from these topics. To the best of our knowledge, ours is the first LDA approach to integrate both types of supervision in a single framework. The generative process of DSLDA is described below.

- For the  $n^{\text{th}}$  document, sample a topic selection probability vector  $\theta_n \sim \text{Dir}(\alpha_n)$ , where  $\alpha_n = \Lambda_n \alpha$  and  $\alpha$  is the parameter of a Dirichlet distribution of dimension  $K$ , which is the total number of topics. The topics are assumed to be of two types – latent and supervised, and there are  $K_1$  latent topics and  $K_2$  supervised topics. Therefore,  $K = K_1 + K_2$ . Latent topics are never observed, while supervised topics are observed in training but not in test data. Henceforth, in each vector or matrix with  $K$  components, it is assumed that the first  $K_1$  components correspond to the latent topics and the next  $K_2$  components to the supervised topics.  $\Lambda_n$  is a diagonal binary matrix of dimension  $K \times K$ . The  $k^{\text{th}}$  diagonal entry is unity if *either*  $1 \leq k \leq K_1$  *or*  $K_1 < k \leq K$  and the  $n^{\text{th}}$  document is tagged with the  $k^{\text{th}}$  topic. Also,  $\alpha = (\alpha_1, \alpha_2)$  where  $\alpha_1$  is a parameter of a Dirichlet distribution of dimension  $K_1$  and  $\alpha_2$  is a parameter of a Dirichlet distribution of dimension  $K_2$ .

- For the  $m^{\text{th}}$  word in the  $n^{\text{th}}$  document, sample a topic  $z_{nm} \sim \text{Mult}(\theta'_n)$ , where  $\theta'_n = (1 - \epsilon)\{\theta_{nk}\}_{k=1}^{k_1} \epsilon\{\Lambda_{n,kk}\theta_{nk}\}_{k=1+k_1}^K$ . This implies that the supervised topics

are weighted by  $\epsilon$  and the latent topics are weighted by  $(1 - \epsilon)$ . Sample the word  $w_{nm} \sim \text{Mult}(\beta_{z_{nm}})$ , where  $\beta_k$  is a multinomial distribution over the vocabulary of words corresponding to the  $k^{\text{th}}$  topic.

- For the  $n^{\text{th}}$  document, generate  $Y_n = \arg \max_y \mathbf{r}_y^T \mathbb{E}(\bar{\mathbf{z}}_n)$  where  $Y_n$  is the class label associated with the  $n^{\text{th}}$  document,  $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm} / M_n$ . Here,  $\mathbf{z}_{nm}$  is an indicator vector of dimension  $K$ .  $\mathbf{r}_y$  is a  $K$ -dimensional real vector corresponding to the  $y^{\text{th}}$  class, and it is assumed to have a prior distribution  $\mathcal{N}(0, 1/C)$ .  $M_n$  is the number of words in the  $n^{\text{th}}$  document. The maximization problem to generate  $Y_n$  (or the classification problem) is carried out using a max-margin principle.

Note that predicting each class is effectively treated as a separate task, and that the shared topics are useful for generalizing the performance of the model across classes. In particular, when all classes have few training examples, knowledge transfer between classes can occur through the shared topics. So, the mapping from the original feature space to the topic space is effectively learned using examples from all classes, and a few examples from each class are sufficient to learn the mapping from the reduced topic space to the class labels. The corresponding graphical model is shown in Fig. 3.5.

### 3.2.2 Inference and Learning

Let us denote the hidden variables by  $\mathbf{Z} = \{\{z_{nm}\}, \{\boldsymbol{\theta}_n\}\}$ , the observed variables by  $\mathbf{X} = \{w_{nm}\}$  and the model parameters by  $\boldsymbol{\kappa}_0$ . The joint distribution

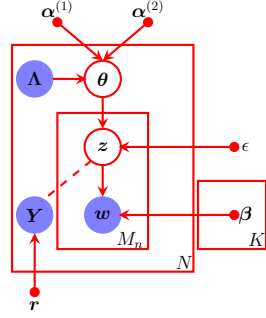


Figure 3.5: Graphical Model of DSLDA

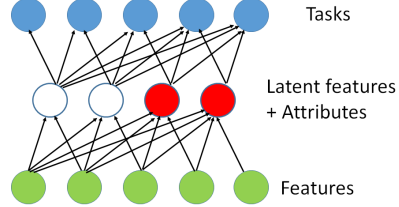


Figure 3.6: Illustration of DSLDA

of the hidden and observed variables is:

$$p(\mathbf{X}, \mathbf{Z} | \kappa_0) = \prod_{n=1}^N p(\theta_n | \alpha_n) \prod_{m=1}^{M_n} p(z_{nm} | \theta'_n) p(w_{nm} | \beta_{z_{nm}}). \quad (3.1)$$

To avoid computational intractability, inference and estimation are performed using Variational **EM**. The factorized approximation to the posterior distribution on hidden variables  $\mathbf{Z}$  is given by:

$$q(\mathbf{Z} | \{\kappa_n\}_{n=1}^N) = \prod_{n=1}^N q(\theta_n | \gamma_n) \prod_{m=1}^{M_n} q(z_{nm} | \phi_{nm}), \quad (3.2)$$

where  $\theta_n \sim \text{Dir}(\gamma_n) \forall n$ ,  $z_{nm} \sim \text{Mult}(\phi_{nm}) \forall n, m$ , and  $\kappa_n = \{\gamma_n, \{\phi_{nm}\}\}$ , which is the set of variational parameters corresponding to the  $n^{\text{th}}$  instance. Further,  $\gamma_n = (\gamma_{nk})_{k=1}^K \forall n$ , and  $\phi_{nm} = (\phi_{nmk})_{k=1}^K \forall n, m$ . With the use of the lower bound obtained by the factorized approximation, followed by Jensen's inequality, DSLDA reduces to solving the following optimization problem<sup>1</sup>:

$$\begin{aligned} \min_{q, \kappa_0, \{\xi_n\}} \quad & \frac{1}{2} \|\mathbf{r}\|^2 - \mathcal{L}(q(\mathbf{Z})) + C \sum_{n=1}^N \xi_n, \\ \text{s.t. } \forall n, y \neq Y_n : \quad & \mathbb{E}[\mathbf{r}^T \Delta f_n(y)] \geq 1 - \xi_n; \xi_n \geq 0. \end{aligned} \quad (3.3)$$

<sup>1</sup>Please see [Zhu et al., 2009] for further details.

Here,  $\Delta f_n(y) = f(Y_n, \bar{z}_n) - f(y, \bar{z}_n)$  and  $\{\xi_n\}_{n=1}^N$  are the slack variables, and  $f(y, \bar{z}_n)$  is a feature vector whose components from  $(y-1)K+1$  to  $yK$  are those of the vector  $\bar{z}_n$  and all the others are 0.  $\mathbb{E}[\mathbf{r}^T \Delta f_n(y)]$  is the “expected margin” over which the true label  $Y_n$  is preferred over a prediction  $y$ . From this viewpoint, DSLDA projects the documents onto a combined topic space and then uses a max-margin approach to predict the class label. The parameter  $C$  penalizes the margin violation of the training data.

$$\phi_{nmk}^* \propto \Lambda_{n,kk} \exp \left[ \psi(\gamma_{nk}) + \log(\beta_{kw_{nm}}) + \log(\epsilon') + 1/M_n \sum_{y \neq Y_n} \mu_n(y) \mathbb{E}[r_{Y_n k} - r_{y k}] \right] \quad \forall n, m, k. \quad (3.4)$$

$$\gamma_{nk}^* = \Lambda_{n,kk} \left[ \alpha_k + \sum_{m=1}^{M_n} \phi_{nmk} \right] \quad \forall n, k. \quad (3.5)$$

$$\beta_{kv}^* \propto \sum_{n=1}^N \sum_{m=1}^{M_n} \phi_{nmk} \mathbb{I}_{\{w_{nm}=v\}} \quad \forall k, v. \quad (3.6)$$

$$\mathcal{L}_{[\alpha_1/\alpha_2]} = \left[ \sum_{n=1}^N \log(\Gamma(\sum_{k=1}^K \alpha_{nk})) - \sum_{n=1}^N \sum_{k=1}^K \log(\Gamma(\alpha_{nk})) \right] + \sum_{n=1}^N \sum_{k=1}^K \left[ \psi(\gamma_{nk}) - \psi(\sum_{k=1}^K \gamma_{nk}) \right] (\alpha_{nk} - 1). \quad (3.7)$$

Let  $\mathcal{Q}$  be the set of all distributions having a fully factorized form as given in (8.2). Let the distribution  $q^*$  from the set  $\mathcal{Q}$  optimize the objective in Eq. (3.3). The optimal values of corresponding variational parameters are given in Eqs. (3) and (3.5). In Eq. (3),  $\epsilon' = (1 - \epsilon)$  if  $k \leq K_1$  and  $\epsilon' = \epsilon$  otherwise. Since  $\phi_{nm}$  is a multinomial distribution, the updated values of the  $K$  components should be normalized to unity. The optimal values of  $\phi_{nm}$  depend on  $\gamma_n$  and vice-versa. Therefore, iterative optimization is adopted to maximize the lower bound until convergence is achieved.

During testing, one does not observe a document’s supervised topics and, in principle, has to explore  $2^{K_2}$  possible combinations of supervised tags – an expensive process. A simple approximate solution, as employed in LLDA [Ramage et al., 2009], is to assume the absence of the variables  $\{\Lambda_n\}$  altogether in the test phase, and just treat the problem as inference in MedLDA with  $K$  latent topics. One can then threshold over the last  $K_2$  topics if the tags of a test document need to be inferred. Equivalently, one can also assume  $\Lambda_n$  to be an identity matrix of dimension  $K \times K \forall n$ . This representation ensures that the expressions for update equations (3) and (3.5) do not change in the test phase.

In the M step, the objective in Eq. (3.3) is maximized w.r.t  $\kappa_0$ . The optimal value of  $\beta_{kv}$  is given in Eq. (3.6). Since  $\beta_k$  is a multinomial distribution, the updated values of the  $V$  components should be normalized. However, numerical methods for optimization are required to update  $\alpha_1$  or  $\alpha_2$ . The part of the objective function that depends on  $\alpha_1$  and  $\alpha_2$  is given in Eq. (3.7). The update for the parameter  $\mathbf{r}$  is carried out using a multi-class SVM solver [Fan et al., 2008]. With all other model and variational parameters held fixed (*i.e.* with  $\mathcal{L}(q)$  held constant), the objective in Eq. (3.3) is optimized w.r.t.  $\mathbf{r}$ . A reader familiar with the updates in unsupervised LDA can see the subtle (but non-trivial) changes in the update equations for DSLDA.

### 3.3 Non-parametric DSLDA

We now propose a non-parametric extension of DSLDA (NP-DSLDA) that solves the model selection problem and automatically determines the best number

of latent topics for modeling the given data. A modified stick breaking construction of Hierarchical Dirichlet Process (HDP) [Teh et al., 2006], recently introduced in [Wang et al., 2011a] is used here which makes variational inference feasible. The idea in such representation is to share the corpus level atoms across documents by sampling atoms with replacement for each document and modifying the weights of these samples according to some other GEM distribution [Teh et al., 2006] whose parameter does not depend on the weights of the corpus-level atoms.

Figure 3.7: Graphical Model of NP-DSLDA

Figure 3.8: Illustration of NP-DSLDA

modeling of the HDP. This will be evident in the generative process of NP-DSLDA presented below:

- Sample  $\phi_{k_1} \sim \text{Dir}(\eta_1) \forall k_1 \in \{1, 2, \dots, \infty\}$ , where  $\eta_1$  is the parameter of Dirichlet distribution of dimension  $V$ .
- Sample  $\phi_{k_2} \sim \text{Dir}(\eta_2) \forall k_2 \in \{1, 2, \dots, K_2\}$ , where  $\eta_2$  is the parameter of Dirichlet distribution of dimension  $V$ .
- Sample  $\beta'_{k_1} \sim \text{Beta}(1, \delta_0) \forall k_1 \in \{1, 2, \dots, \infty\}$ .
- For the  $n^{\text{th}}$  document, sample  $\pi_n^{(2)} \sim \text{Dir}(\Lambda_n \alpha_2)$ .  $\alpha_2$  is the parameter of Dirichlet of dimension  $K_2$ .  $\Lambda_n$  is a diagonal binary matrix of dimension  $K_2 \times K_2$ . The  $k^{\text{th}}$  diagonal entry is unity if the  $n^{\text{th}}$  word is tagged with the  $k^{\text{th}}$  supervised topic.
- $\forall n, \forall t \in \{1, 2, \dots, \infty\}$ , sample  $\pi'_{nt} \sim \text{Beta}(1, \alpha_0)$ . Assume  $\pi_n^{(1)} = (\pi_{nt})_t$  where  $\pi_{nt} = \pi'_{nt} \prod_{l < t} (1 - \pi'_{nl})$ .
- $\forall n, \forall t$ , sample  $c_{nt} \sim \text{Mult}(\beta)$  where  $\beta_{k_1} = \beta'_{k_1} \prod_{l < k_1} (1 - \beta'_l)$ .  $\pi_n^{(1)}$  represents the probability of selecting the sampled atoms in  $c_n$ . Due to sampling with replacement,  $c_n$  can contain multiple atoms of the same index from the corpus level DP.
- For the  $m^{\text{th}}$  word in the  $n^{\text{th}}$  document, sample  $z_{nm} \sim \text{Mult}((1 - \epsilon)\pi_n^{(1)}, \epsilon\pi_n^{(2)})$ . This implies that w.p.  $\epsilon$ , a topic is selected from the set of supervised topics and w.p.  $(1 - \epsilon)$ , a topic is chosen from the set of (infinite number of) unsupervised topics. Note that by weighting the  $\pi$ 's appropriately, the need for additional hidden “switching” variable is avoided.

- Sample  $w_{nm}$  from a multinomial given by the following equation:

$$\prod_{k_1=1}^{\infty} \prod_{v=1}^V \phi_{k_1 v}^{\mathbb{I}_{\{w_{nm}=v\}} \mathbb{I}_{\{c_n z_{nm}=k_1 \in \{1, \dots, \infty\}\}}} \prod_{k_2=1}^{K_2} \prod_{v=1}^V \phi_{k_2 v}^{\mathbb{I}_{\{w_{nm}=v\}} \mathbb{I}_{\{z_{nm}=k_2 \in \{1, \dots, K_2\}\}}}. \quad (3.8)$$

The corresponding graphical model is shown in Fig. 3.7. The joint distribution of NP-DSLDA is given as follows:

$$\begin{aligned} p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\kappa}_0) &= \prod_{k_1=1}^{\infty} p(\phi_{k_1} | \boldsymbol{\eta}_1) p(\beta'_{k_1} | \boldsymbol{\delta}_0) \prod_{k_2=1}^{K_2} p(\phi_{k_2} | \boldsymbol{\eta}_2) \prod_{n=1}^N p(\boldsymbol{\pi}_n^{(2)} | \boldsymbol{\alpha}_2) \\ &\prod_{t=1}^{\infty} p(\boldsymbol{\pi}_{nt}'^{(1)} | \boldsymbol{\alpha}_0) p(c_{nt} | \boldsymbol{\beta}') \prod_{m=1}^{M_n} p(z_{nm} | \boldsymbol{\pi}_n^{(1)}, \boldsymbol{\pi}_n^{(2)}, \epsilon) p(w_{nm} | \boldsymbol{\phi}, c_n z_{nm}, z_{nm}). \end{aligned} \quad (3.9)$$

As an approximation to the posterior distribution over the hidden variables, we use the following factorized distribution:

$$\begin{aligned} q(\mathbf{Z} | \boldsymbol{\kappa}) &= \prod_{k_1=1}^{K_1} q(\phi_{k_1} | \boldsymbol{\lambda}_{k_1}) \prod_{k_2=1}^{K_2} q(\phi_{k_2} | \boldsymbol{\lambda}_{k_2}) \prod_{k_1=1}^{K_1-1} q(\beta'_{k_1} | u_{k_1}, v_{k_1}) \\ &\prod_{n=1}^N q(\boldsymbol{\pi}_n^{(2)} | \boldsymbol{\gamma}_n) \prod_{t=1}^{T-1} q(\boldsymbol{\pi}_{nt}'^{(1)} | a_{nt}, b_{nt}) \prod_{t=1}^T q(c_{nt} | \boldsymbol{\varphi}_{nt}) \prod_{m=1}^{M_n} q(z_{nm} | \boldsymbol{\zeta}_{nm}). \end{aligned} \quad (3.10)$$

Here,  $\boldsymbol{\kappa}_0$  and  $\boldsymbol{\kappa}$  denote the sets of model and variational parameters, respectively.  $K_1$  is the truncation limit of the corpus-level Dirichlet Process and  $T$  is the truncation limit of the document-level Dirichlet Process.  $\{\boldsymbol{\lambda}_k\}$  are the parameters of Dirichlet each of dimension  $V$ .  $\{u_{k_1}, v_{k_1}\}$  and  $\{a_{nt}, b_{nt}\}$  are the parameters of variational Beta distribution corresponding to corpus level and document level sticks respectively.  $\{\boldsymbol{\varphi}_{nt}\}$  are Mult parameters of dimension  $K_1$  and  $\{\boldsymbol{\zeta}_{nm}\}$  are Mults of dimension  $(T + K_2)$ .  $\{\boldsymbol{\gamma}_n\}_n$  are parameters of Dirichlet distribution of dimension  $K_2$ .

The underlying optimization problem takes the same form as in Eq. (3.3). The only difference lies in the calculation of  $\Delta f_n(y) = f(Y_n, \bar{s}_n) - f(y, \bar{s}_n)$ . The first set of dimensions of  $\bar{s}_n$  (corresponding to the unsupervised topics) is given by  $1/M_n \sum_{m=1}^{M_n} \mathbf{c}_{nz_{nm}}$ , where  $\mathbf{c}_{nt}$  is an indicator vector over the set of unsupervised topics. The following  $K_2$  dimensions (corresponding to the supervised topics) are given by  $1/M_n \sum_{m=1}^{M_n} \mathbf{z}_{nm}$ . After the variational approximation with  $K_1$  number of corpus level sticks,  $\bar{s}_n$  turns out to be of dimension  $(K_1 + K_2)$  and the feature vector  $f(y, \bar{s}_n)$  constitutes  $Y(K_1 + K_2)$  elements. The components of  $f(y, \bar{s}_n)$  from  $(y - 1)(K_1 + K_2) + 1$  to  $y(K_1 + K_2)$  are those of the vector  $\bar{s}_n$  and all the others are 0. Essentially, due to the variational approximation, NP-DSLDA projects each document on to a combined topic space of dimension  $(K_1 + K_2)$  and learns the mapping from this space to the classes.

$$\zeta_{nmt}^* \propto \exp \left[ [\psi(a_{nt}) - \psi(a_{nt} + b_{nt})] \mathbb{I}_{\{t < T\}} + \sum_{t'=1}^{t-1} [\psi(b_{nt'}) - \psi(a_{nt'} + b_{nt'})] \right. \\ \left. + \sum_{k_1=1}^{K_1} \varphi_{ntk_1} \left[ \psi(\lambda_{k_1 w_{nm}}) - \psi\left(\sum_{v=1}^V \lambda_{k_1 v}\right) \right] + \sum_{y \neq Y_n} \mu_n(y) \sum_{k_1=1}^{K_1} \mathbb{E}[r_{Y_n k_1} - r_{y k_1}] \varphi_{ntk_1} \right], \quad (3.11)$$

$$\zeta_{nm(T+k_2)}^* \propto \Lambda_{nk_2 k_2} \exp \left[ \psi(\gamma_{nk_2}) - \psi\left(\sum_{k_2=1}^{K_2} \gamma_{nk_2}\right) + \psi(\lambda_{(K_1+k_2)w_{nm}}) \right. \\ \left. - \psi\left(\sum_{v=1}^V \lambda_{(K_1+k_2)v}\right) + 1/M_n \sum_{y \neq Y_n} \mu_n(y) \mathbb{E}[r_{Y_n(K_1+k_2)} - r_{y(K_1+k_2)}] \right], \quad (3.12)$$

$$\begin{aligned}
\varphi_{ntk_1}^* &\propto \exp \left[ [\psi(u_{k_1}) - \psi(u_{k_1} + v_{k_1})] \mathbb{I}_{\{k_1 < K_1\}} \right. \\
&+ \sum_{k'=1}^{k_1-1} [\psi(v_{k'}) - \psi(u_{k'} + v_{k'})] + \sum_{m=1}^{M_n} \zeta_{nmt} \left[ \psi(\lambda_{k_1 w_{nm}}) - \psi\left(\sum_{v=1}^V \lambda_{k_1 v}\right) \right. \\
&\left. \left. + 1/M_n \sum_{y \neq Y_n} \mu_n(y) \mathbb{E}[r_{Y_n k_1} - r_{y k_1}] \left( \sum_{m=1}^{M_n} \zeta_{nmt} \right) \right] \right]. \quad (3.13)
\end{aligned}$$

Some of the update equations of NP-DSLDA are given in the above equations, where  $\{\varphi_{ntk_1}\}$  are the set of variational parameters that characterize the assignment of the documents to the global set of  $(K_1 + K_2)$  topics. One can see how the effect of the class labels is included in the update equation of  $\{\varphi_{ntk_1}\}$  via the average value of the parameters  $\{\zeta_{nmt}\}$ . This follows intuitively from the generative assumption. update exists for the model parameters and hence numerical optimization has to be used. Other updates are either similar to DSLDA or the model in [Wang et al., 2011a] and are omitted due to space constraints.  $\{\zeta_{nm}\}$ , corresponding to supervised and unsupervised topics, should be individually normalized and then scaled by  $\epsilon$  and  $(1 - \epsilon)$  respectively. Otherwise, the effect of the Dirichlet prior on supervised topics will get compared to that of the GEM prior on the unsupervised topics which does not follow the generative assumptions. The variational parameters  $\{\lambda_k\}$  and  $\{\varphi_{nt}\}$  are also normalized.

Note that NP-DSLDA offers some flexibility with respect to the latent topics that could be dominant for a specific task. One could therefore postulate that NP-DSLDA can learn the clustering of tasks from the data itself by making a subset of latent topics to be dominant for a set of tasks. Though we do not have supporting experiments, NP-DSLDA is, in principle, capable of performing clustered multi-

task learning without any prior assumption on the relatedness of the tasks.

## **3.4 Experimental Evaluation**

### **3.4.1 Data Description**

Our evaluation used two datasets, a text corpus and a multi-class image database, as described below.

#### **3.4.1.1 aYahoo Data**

The first set of experiments was conducted with the aYahoo image dataset from Farhadi et al. [2009] which has 12 classes – carriage, centaur, bag, building, donkey, goat, jetski, monkey, mug, statue, wolf, and zebra.<sup>2</sup> Each image is annotated with relevant visual attributes such as “has head”, “has wheel”, “has torso” and 61 others, which we use as the supervised topics. Using such intermediate “attributes” to aid visual classification has become a popular approach in computer vision [Kovashka et al., 2011; Lampert et al., 2009]. After extracting SIFT features [Lowe, 2004] from the raw images, quantization into 250 clusters is performed, defining the vocabulary for the bag of visual words [Gabriella et al., 2004]. Images with less than two attributes were discarded. The resulting dataset of size 2275 was equally split into training and test data.

---

<sup>2</sup><http://vision.cs.uiuc.edu/attributes/>

### 3.4.1.2 ACM Conference Data

The text corpus consists of conference chapter abstracts from two groups of conferences. The first group has four conferences related to data mining – WWW, SIGIR, KDD, and ICML, and the second group consists of two VLSI conferences – ISPD and DAC. The classification task is to determine the conference at which the abstract was published. As supervised topics, we use keywords provided by the authors, which are presumably useful in determining the conference venue. Since authors usually take great care in choosing keywords so that their chapter is retrieved by relevant searches, we believed that such keywords made a good choice of supervised topics. Part of the data, crawled from ACM’s website, was used in Wang et al. [2009]. A total of 2300 abstracts were collected each of which had at least three keywords and an average of 78 ( $\pm 33.5$ ) words. After stop-word removal, the vocabulary size for the assembled data is 13412 words. The final number of supervised topics, after some standard pre-processing of keywords, is 55. The resulting dataset was equally split into training and test data.

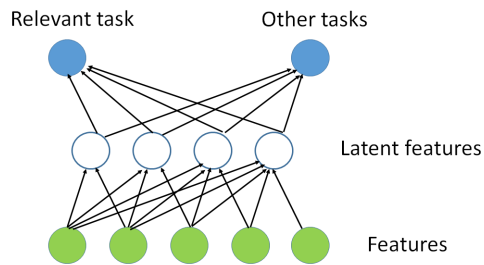


Figure 3.9: Illustration of MedLDA-OVA

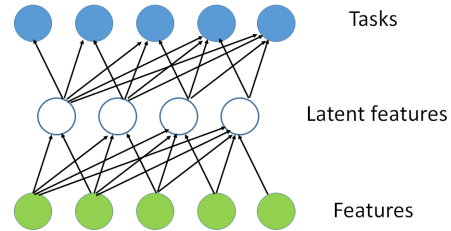


Figure 3.10: Illustration of MedLDA-MTL

### 3.4.2 Methodology for Experiments with Multitask Learning

In order to demonstrate the contribution of each aspect of the overall model, DSLDA and NP-DSLDA are compared against the following simplified models:

- MedLDA with **one-vs-all** classification (MedLDA-OVA) (shown in Fig. 3.9): A separate model is trained for each class using a one-vs-all approach leaving no possibility of transfer across classes.
- MedLDA with **multitask learning** (MedLDA-MTL) (shown in Fig. 3.10): A single model is learned for all classes where the latent topics are shared across classes.
- DSLDA with **only shared supervised topics** (DSLDA-OSST) (shown in Fig. 3.11): A model in which supervised topics are used and shared across classes but there are no latent topics.
- DSLDA with **no shared latent topics** (DSLDA-NSLT) (shown in Fig. 3.12): A model in which only supervised topics are shared across classes and a separate set of latent topics is maintained for each class.
- **Majority class method** (MCM): A simple baseline which always picks the most common class in the training data.

There could be one more baseline where LDA is learnt in an unsupervised way and an SVM is trained with the topic level assignment used as the features. Since MedLDA already outperforms this baseline, we did not plot the performances of

such baseline. The plot of the majority class method is there only to highlight the imbalance of the labeled data among the multiple categories.

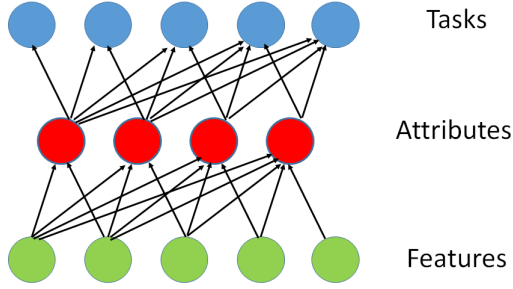


Figure 3.11: Illustration of DSLDA-OSST

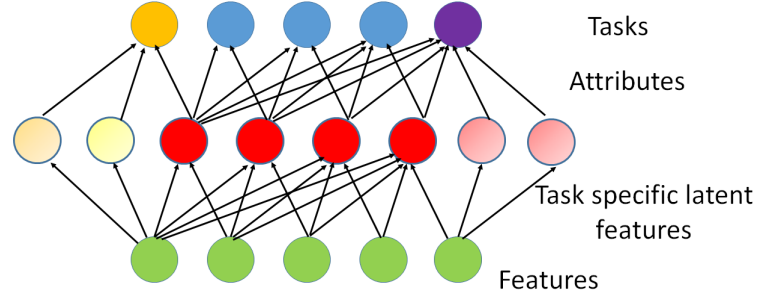


Figure 3.12: Illustration of DSLDA-NSLT

These baselines are useful for demonstrating the utility of *both* supervised and latent shared topics for multitask learning in DSLDA. MedLDA-OVA is a non-transfer method, where a separate model is learned for each of the classes, *i.e.* one of the many classes is considered as the positive class and the union of the remaining ones is treated as the negative class. Since the models for each class are trained separately, there is no possibility of sharing inductive information across classes. MedLDA-MTL trains on examples from all classes simultaneously, and thus allows for sharing of inductive information *only* through a common set of latent topics. In DSLDA-OSST, only supervised topics are maintained and knowledge transfer can *only* take place *via* these supervised topics. DSLDA-NSLT uses shared supervised topics but also includes latent topics which are *not* shared across classes. This model provides for transfer *only* through shared supervised topics but provides extra modeling capacity compared to DSLDA-OSST through the use of latent topics that are not shared. DSLDA and NP-DSLDA are MTL frameworks

where both supervised *and* latent topics are shared across all classes. Note that, all of the baselines can be implemented using DSLDA with a proper choice of  $\Lambda$  and  $\epsilon$ . For example, DSLDA-OSST is just a special case of DSLDA with  $\epsilon$  fixed at 1.

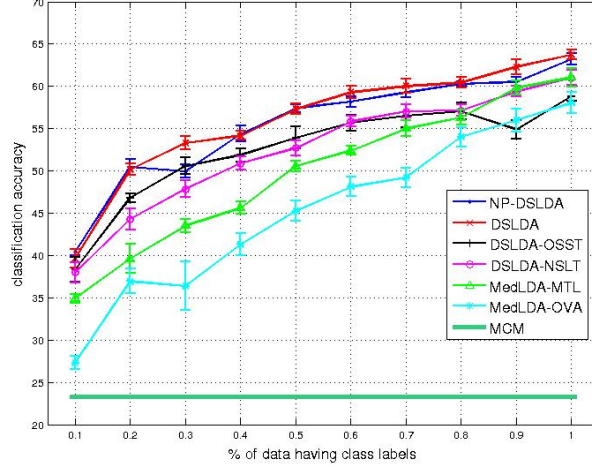


Figure 3.13:  $p_1 = 0.5$  (aYahoo)

In order to explore the effect of different amounts of both types of supervision, we varied the amount of both topic-level and class-level supervision. Specifically, we provided topic supervision for a fraction,  $p_1$ , of the overall training set, and then provided class supervision for only a further fraction  $p_2$  of this data. Therefore, only  $p_1 * p_2$  of the overall training data has class supervision. By varying the number of latent topics from 20 to 200 in steps of 10, we found that  $K_1 = 100$  generally worked the best for all the parametric models. Therefore, we show parametric results for 100 latent topics. For each combination of  $(p_1, p_2)$ , 50 random trials were performed with  $C = 10$ . To maintain equal representational capacity, the total number of topics  $K$  is held the same across all parametric models (ex-

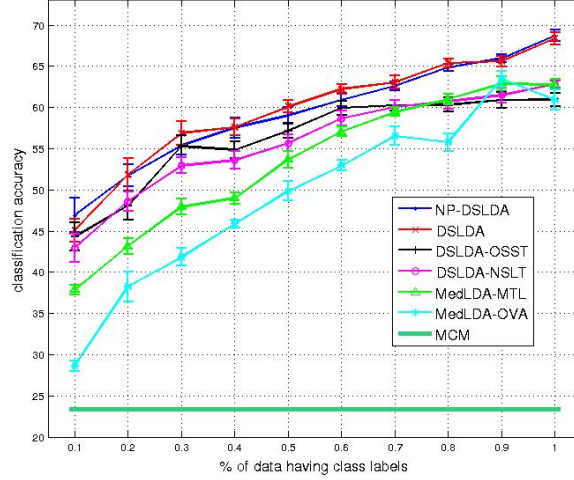


Figure 3.14:  $p_1 = 0.7$  (aYahoo)

cept for DSLDA-OSST where the total number of topics is  $K_2$ ). For NP-DSLDA, following the suggestion of [Wang et al., 2011a], we set  $K_1 = 150$  and  $T = 40$ , which produced uniformly good results. When required,  $\epsilon$  was chosen using 5-fold internal cross-validation using the training data.

### 3.4.3 Multitask Learning Results

Figs. 3.15 and 3.16 present representative learning curves for the image data, showing how classification accuracy improves as the amount of class supervision ( $p_2$ ) is increased. Results are shown for two different amounts of topic supervision ( $p_1 = 0.5$  and  $p_1 = 0.7$ ). Figs. 4.3 and 4.4 present similar learning curves for the text data. The error bars in the curves show standard deviations across the 50 trials. The results demonstrate that DSLDA and NP-DSLDA quite consistently outperform all of the baselines, clearly demonstrating the advantage of

combining both types of topics. NP-DSLDA performs about as well as DSLDA, for which the optimal number of latent topics has been chosen using an expensive model-selection search. This demonstrates that NP-DSLDA is doing a good job of automatically selecting an appropriate number of latent topics. Overall, DSLDA-OSST and MedLDA-MTL perform about the same, showing that, individually, both latent and supervised shared topics each support multitask learning about equally well when used alone. However, combining both types of topics provides a clear improvement.

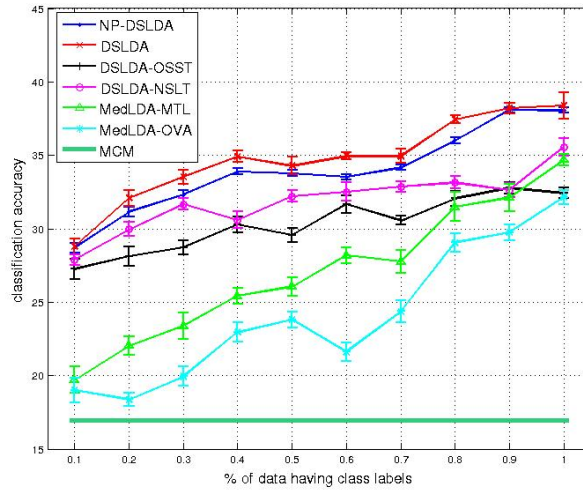


Figure 3.15:  $p_1 = 0.5$  (Conference)

MedLDA-OVA performs quite poorly when there is only a small amount of class supervision (note that this baseline uses *only* class labels). However, the performance approaches the others as the amount of class supervision increases. This is consistent with the intuition that multitask learning is most beneficial when each task has limited supervision and therefore has more to gain by sharing information

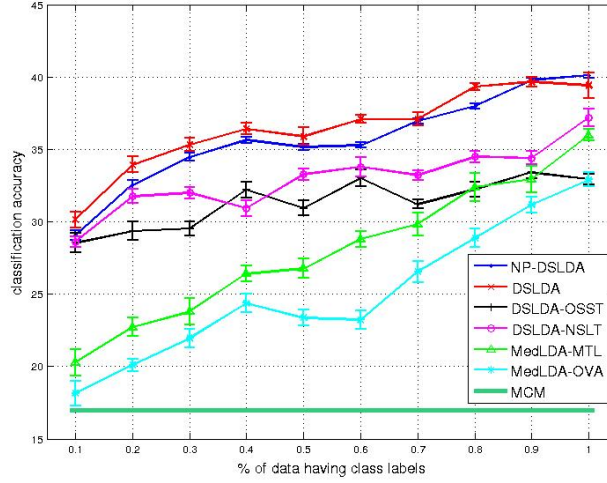


Figure 3.16:  $p_1 = 0.7$  (Conference)

LT1	function, label, graph, classification, database, propagation, algorithm, accuracy, minimization, transduction
LT2	performance, design, processor, layer, technology, device, bandwidth, architecture, stack, system
CAD	design, optimization, mapping, pin, simulation, cache, programming, routing, biochip, electrode
VLSI	design, physical, lithography, optimization, interdependence, global, robust, cells, layout, growth
IR	algorithm, web, linear, query, precision, document, repair, site, search, semantics
Ranking	integration, catalog, hierarchical, dragpushing, structure, source, sequence, alignment, transfer, flattened, speedup
Learning	model, information, trajectory, bandit, mixture, autonomous, hierarchical, feedback, supervised, task

Table 3.1: Illustration of Latent and Supervised Topics

with other tasks. Shared supervised topics clearly increase classification accuracy when class supervision is limited (i.e. small values of  $p_2$ ), as shown by the performance of both DSLDA-NSLT and DSLDA-OSST. When  $p_2 = 1$  (equal amounts of topic and class supervision), DSLDA-OSST, MedLDA-MTL and MedLDA-OVA all perform similarly; however, by exploiting *both* types of supervision, DSLDA and NP-DSLDA still maintain a performance advantage.

**Topic Illustration:** In Table 3.1, we show the most indicative words for several topics discovered by DSLDA from the text data (with  $p_1 = 0.8$  and  $p_2 = 1$ ). LT1

and LT2 correspond to the most frequent latent topics assigned to documents in the two broad categories of conferences (data mining and VLSI, respectively). The other five topics are supervised ones. CAD and IR stand for Computer Aided Design and Information Retrieval respectively. The illustrated topics are particularly discriminative when classifying documents.

### **3.5 Conclusion**

This chapter has introduced approaches that combine the following – generative and discriminative models, latent and supervised topics, and class and topic level supervision, in a principled probabilistic manner. Different ablations of the proposed models are also evaluated in order to understand the individual effects of latent/supervised topics, active learning and multitask learning on the overall model performance. The general idea of “double supervision” could be applied to many other models, for example, in multi-layer perceptrons, latent SVMs [Yu and Joachims, 2009] or in deep belief networks [Hinton and Osindero, 2006]. In MTL, sharing tasks blindly is not always a good approach and further extension with clustered MTL [Zhou et al., 2011] is possible.

We would like to emphasize that the use of variational approximations does hurt the empirical performance of the proposed models, for which the models cannot yet beat the performance of other discriminative alternatives for solving similar problems [Farhadi et al., 2009; Lampert et al., 2009]. However, the interpretability of the proposed models is also an added benefit which is not available from the discriminative models. Further, sampling based algorithm could also be developed for

solving the inference, possibly leading to even better performance without any factorized approximations. On the other hand, the MTL framework proposed herein can be combined with active learning. In the next Chapter, we propose two such models, Act-DSLDA and ACT-NPDSLDA, which are extensions of DSLDA and NP-DSLDA with the flexibility of active learning incorporated.

## Chapter 4

### Active Multitask Learning using Both Supervised and Shared Latent Topics

In Chapter 3, we explored how MTL can leverage from sharing of information across multiple tasks *via* a shared intermediate layer and can reduce the requirement for labeled information. Another well-known approach to reducing supervision is *active learning*, where a system can request labels for the most informative training examples [Jain and Kapoor, 2009; Joshi et al., 2009; Kovashka et al., 2011; Qi et al., 2008]. In this chapter, our objective is to combine these two orthogonal approaches in order to leverage the benefits of both – learning from a shared abstract feature space and making active queries. In particular, we build on the approach proposed in [Acharya et al., 2013b] and also described in Chapter 3 where multitask learning (MTL) [Caruana, 1997] is accomplished using both shared supervised attributes and a shared latent (*i.e.* unsupervised) set of features.

The chapter is organized as follows. We present related work in Section 4.1, followed by the descriptions of two of our models Active Doubly Supervised Latent Dirichlet Allocation (Act-DSLDA) and a non-parametric variation of the same (Act-NPDSLDA) in Sections 4.2 and 4.3 respectively. Experimental results on both multi-class image and document categorization are presented in Section

4.4. Finally, future directions and conclusions are presented in Section 4.5.

## **4.1 Background**

### **4.1.1 Active Knowledge Transfer**

There has been some effort to integrate active and transfer learning in the same framework. [Jun and Ghosh, 2008] utilized a maximum likelihood classifier to learn parameters from the source domain and use these parameters to seed the EM algorithm that explains the unlabeled data in the target domain. The example which contributed to maximum expected KL divergence of the posterior distribution with the prior distribution was selected in the active step. In [Rai et al., 2010], the source data is first used to train a classifier, the parameters of which are later updated in an online manner with new examples actively selected from the target domain. The active selection criterion is based on uncertainty sampling [Settles, 2009]. Similarly, in [Chan and Ng, 2007], a naïve Bayes classifier is first trained with examples from the source domain and then incrementally updated with data from the target domain selected using uncertainty sampling. The method proposed in [Shi et al., 2008] maintains a classifier trained on the source domain(s) and the prediction of this classifier is trusted only when the likelihood of the data in the target domain is sufficiently high. In case of lower likelihood, domain experts are asked to label the example. Harpale and Yang [2010] proposed active multitask learning for adaptive filtering [Robertson and Soboroff, 2002] where the underlying classifier is logistic regression with Dirichlet process priors. Any feedback provided in the active selection phase improves both the task-specific and the global performance *via* a measure

called *utility gain* [Harpale and Yang, 2010]. Saha et al. [2011] formulated an online active multitask learning framework where the information provided for one task is utilized for other tasks through a task correlation matrix. The updates are similar to perceptron updates. For active selection, they use a margin based sampling scheme which is a modified version of the sampling scheme used in [Cesa-Bianchi et al., 2006].

In contrast to this previous work, our approach employs a topic-modeling framework and uses expected error reduction for active selection. Such an active selection mechanism necessitates fast incremental update of model parameters, and hence the inference and estimation problems become challenging. This approach to active selection is more immune to noisy observations compared to simpler methods such as uncertainty sampling [Settles, 2009]. Additionally, our approach can query both class labels *and* supervised topics (i.e. attributes), which has not previously been explored in the context of MTL.

#### **4.1.2 Online Support Vector Machines**

The online SVM proposed by Bordes et al. [2005, 2007] has three distinct modules that work in unison to provide a scalable learning mechanism. These modules are named “ProcessNew”, “ProcessOld” and “Optimize”. All of these modules use a common operation called “SMOStep” and the memory footprint is limited to the support vectors and associated gradient information. The module “Process-New” operates on a pattern that is not a support pattern. In such an update, one of the classes is chosen as the label of the support pattern and the other class is chosen

such that it defines a feasible direction with the highest gradient. It then performs an SMO step with the example and the selected classes. The module “ProcessOld” randomly picks a support pattern and chooses two classes that define the feasible direction with the highest gradient for that support pattern. “Optimize” resembles “ProcessOld” but picks two classes among those that correspond to existing support vectors.

## 4.2 Active Doubly Supervised Latent Dirichlet Allocation (Act-DSLDA)

We will treat examples as “documents” which consist of a “bag of words” for text or a “bag of visual words” for images. Assume we are given an initial training corpus  $\mathcal{L}$  with  $N$  documents belonging to  $Y$  different classes. Further assume that each of these training documents is also annotated with a set of  $K_2$  different “supervised topics”. The objective is to train a model using the words in a document, as well as the associated supervised topics and class labels, and then use this model to classify completely unlabeled test documents for which no topics or class labels are provided.

When the learning starts,  $\mathcal{L}$  is assumed to have fully labeled documents. However, as the learning progresses more documents are added to the pool  $\mathcal{L}$  with class and/or a subset of supervised topics labeled. Therefore, at any intermediate point of the learning process,  $\mathcal{L}$  can be assumed to contain several sets:  $\mathcal{L} = \{\mathcal{T} \cup \mathcal{T}_C \cup \mathcal{T}_{A_1} \cup \mathcal{T}_{A_2} \cup \dots \cup \mathcal{T}_{A_{K_2}}\}$ , where  $\mathcal{T}$  contains fully labeled documents (*i.e.* with class and all supervised topics labeled),  $\mathcal{T}_C$  are the documents that have

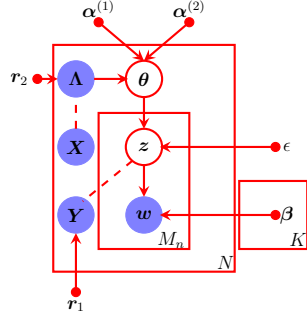


Figure 4.1: Graphical Model of Act-DSLDA

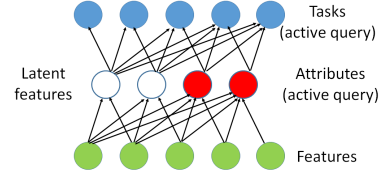


Figure 4.2: Illustration of Act-DSLDA

class labels, and  $1 \leq k \leq K_2$ ,  $\mathcal{T}_{A_k}$  are the documents that have the  $k^{\text{th}}$  supervised topic labeled. Since, human-provided labels are expensive to obtain, we design an active learning framework where the model can query over an unlabeled pool  $\mathcal{U}$  and request either class labels or a subset of the supervised topics. The Act-DSLDA generative model is defined as follows. The generative process is very similar to the generative process of DSLDA described in Section 3.2, but provided here in details to make the notations easy to understand.

- For the  $n^{\text{th}}$  document, sample a topic selection probability vector  $\theta_n \sim \text{Dir}(\alpha_n)$ , where  $\alpha_n = \Lambda_n \alpha$  and  $\alpha$  is the parameter of a Dirichlet distribution of dimension  $K$ , the total number of topics. The topics are assumed to be of two types – latent and supervised, and there are  $K_1$  latent topics and  $K_2$  supervised topics ( $K = K_1 + K_2$ ). Latent topics are never observed, while supervised topics are observed in the training data but not in the test data. Henceforth, in each vector or matrix with  $K$  components, it is assumed that the first  $K_1$  components correspond to the latent topics and the next  $K_2$  components to the supervised topics.  $\Lambda_n$  is a diagonal binary matrix of dimension  $K \times K$ . The  $k^{\text{th}}$  diagonal entry is unity if *either*  $1 \leq$

$k \leq K_1$  or  $K_1 < k \leq K$  and the  $n^{\text{th}}$  document is tagged with the  $k^{\text{th}}$  topic. Also,  $\alpha = (\alpha^{(1)}, \alpha^{(2)})$  where  $\alpha^{(1)}$  is a parameter of a Dirichlet distribution of dimension  $K_1$  and  $\alpha^{(2)}$  is a parameter of a Dirichlet distribution of dimension  $K_2$ .

- In the test data, the supervised topics are not observed and one has to infer them from either the parameters of the model or use some other auxiliary information. Since one of our objectives is to query over the supervised topics as well as the final category, we train a set of binary SVM classifiers that can predict the individual attributes from the features of the data. We denote the parameters of such classifiers by  $\{\mathbf{r}_{2k}\}_{1 \leq k \leq K_2}$ . This is important to get an uncertainty measure over the supervised topics. To further clarify the issue, let us consider that only one supervised topic has to be labeled by the annotator for the  $n^{\text{th}}$  document from the set of supervised topics of size  $K_2$ . To select the most uncertain topic, one needs to compare the uncertainty of predicting the presence or absence of the individual topics. This uncertainty is different from that calculated from the conditional distribution calculated from the posterior over  $\theta_n$ .

- For the  $m^{\text{th}}$  word in the  $n^{\text{th}}$  document, sample a topic  $z_{nm} \sim \text{Mult}(\theta'_n)$ , where  $\theta'_n = (1 - \epsilon)\{\theta_{nk}\}_{k=1}^{k_1} \epsilon\{\Lambda_{n,kk}\theta_{nk}\}_{k=1+k_1}^K$ . This implies that the supervised topics are weighted by  $\epsilon$  and the latent topics are weighted by  $(1 - \epsilon)$ . Sample the word  $w_{nm} \sim \text{Mult}(\beta_{z_{nm}})$ , where  $\beta_k$  is a multinomial distribution over the vocabulary of words corresponding to the  $k^{\text{th}}$  topic.

- For the  $n^{\text{th}}$  document, generate  $Y_n = \arg \max_y \mathbf{r}_{1y}^T \mathbb{E}(\bar{\mathbf{z}}_n)$  where  $Y_n$  is the class label associated with the  $n^{\text{th}}$  document,  $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm}/M_n$ . Here,  $\mathbf{z}_{nm}$  is an indi-

cator vector of dimension  $K$ .  $\mathbf{r}_{1y}$  is a  $K$ -dimensional real vector corresponding to the  $y^{\text{th}}$  class, and it is assumed to have a prior distribution  $\mathcal{N}(0, 1/C)$ .  $M_n$  is the number of words in the  $n^{\text{th}}$  document. The maximization problem to generate  $Y_n$  (i.e. the classification problem) is carried out using the max-margin principle and we use online SVMs [Bordes et al., 2005, 2007] for such updates. Since the model has to be updated incrementally in the active selection step, a batch SVM solver is not applicable, while an online SVM allows one to update the learned weights incrementally given each new example. Note that predicting each class is treated as a separate task, and that the shared topics are useful for generalizing the performance of the model across classes.

#### 4.2.1 Inference and Learning

Inference and parameter estimation have two phases – one for the batch case when the model is trained with fully labeled data, and the other for the active selection step where the model has to be incrementally updated to observe the effect of any labeled information that is queried from the oracle.

##### 4.2.1.1 Learning in Batch Mode

Let us denote the hidden variables by  $\mathbf{Z} = \{\{z_{nm}\}, \{\boldsymbol{\theta}_n\}\}$ , the observed variables by  $\mathbf{X} = \{w_{nm}\}$  and the model parameters by  $\boldsymbol{\kappa}_0$ . The joint distribution of the hidden and observed variables is:

$$p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\kappa}_0) = \prod_{n=1}^N p(\boldsymbol{\theta}_n | \boldsymbol{\alpha}_n) \prod_{m=1}^{M_n} p(z_{nm} | \boldsymbol{\theta}'_n) p(w_{nm} | \boldsymbol{\beta}_{z_{nm}}). \quad (4.1)$$

To avoid computational intractability, inference and estimation are performed using variational **EM**. The factorized approximation of the posterior distribution with hidden variables  $\mathbf{Z}$  is given by:

$$q(\mathbf{Z}|\{\boldsymbol{\kappa}_n\}_{n=1}^N) = \prod_{n=1}^N q(\boldsymbol{\theta}_n|\boldsymbol{\gamma}_n) \prod_{m=1}^{M_n} q(z_{nm}|\boldsymbol{\phi}_{nm}), \quad (4.2)$$

where  $\boldsymbol{\theta}_n \sim \text{Dir}(\boldsymbol{\gamma}_n)$ ,  $z_{nm} \sim \text{multinomial}(\boldsymbol{\phi}_{nm}) \forall n \in \{1, 2, \dots, N\}$  and  $\forall m \in \{1, 2, \dots, M_n\}$ , and  $\boldsymbol{\kappa}_n = \{\boldsymbol{\gamma}_n, \{\boldsymbol{\phi}_{nm}\}\}$ , which is the set of variational parameters corresponding to the  $n^{\text{th}}$  instance. Further,  $\boldsymbol{\gamma}_n = (\gamma_{nk})_{k=1}^K \forall n$ , and  $\boldsymbol{\phi}_{nm} = (\phi_{nmk})_{k=1}^K \forall n, m$ . With the use of the lower bound obtained by the factorized approximation, followed by Jensen's inequality, Act-DSLDA reduces to solving the following optimization problem<sup>1</sup>:

$$\begin{aligned} \min_{q, \boldsymbol{\kappa}_0, \{\xi_n\}} \quad & \frac{1}{2} \|\mathbf{r}_1\|^2 - \mathcal{L}(q(\mathbf{Z})) + C \sum_{n=1}^N \xi_n \mathbb{I}_{\mathcal{T}_C, n}, \\ \text{s.t. } \quad & \forall n \in \mathcal{T}_C, y \neq Y_n : \mathbb{E}[\mathbf{r}_1^T \Delta f_n(y)] \geq 1 - \xi_n; \xi_n \geq 0. \end{aligned} \quad (4.3)$$

Here,  $\Delta f_n(y) = f(Y_n, \bar{\mathbf{z}}_n) - f(y, \bar{\mathbf{z}}_n)$  and  $\{\xi_n\}_{n=1}^N$  are the slack variables, and  $f(y, \bar{\mathbf{z}}_n)$  is a feature vector whose components from  $(y-1)K+1$  to  $yK$  are those of the vector  $\bar{\mathbf{z}}_n$  and all the others are 0.  $\mathbb{E}[\mathbf{r}_1^T \Delta f_n(y)]$  is the ‘‘expected margin’’ over which the true label  $Y_n$  is preferred over a prediction  $y$ . From this viewpoint, Act-DSLDA projects the documents onto a combined topic space and then uses a max-margin approach to predict the class label. The parameter  $C$  penalizes the margin violation of the training data. The indicator variable  $\mathbb{I}_{\mathcal{T}_C, n}$  is unity if the  $n^{\text{th}}$

---

<sup>1</sup>Please see [Zhu et al., 2009] for further details.

document has a class label (*i.e.*  $n \in \mathcal{T}_C$ ) and 0 otherwise. This implies that only the documents that have class labels are used to update the parameters of the online SVM.

Let  $\mathcal{Q}$  be the set of all distributions having a fully factorized form as given in (8.2). Note that such a factorized approximation makes the use of incremental variation of EM possible in the active selection step following the discussion in Section 2.4.3.2. Let the distribution  $q^*$  from the set  $\mathcal{Q}$  optimize the objective in Eq. (4.3). The optimal values of the corresponding variational parameters are same as those of DSLDA [Acharya et al., 2013b]. The optimal values of  $\phi_{nm}$  depend on  $\gamma_n$  and vice-versa. Therefore, iterative optimization is adopted to maximize the lower bound until convergence is achieved.

During testing, one does not observe a document's supervised topics and instead an approximate solution, as also used in [Acharya et al., 2013b; Ramage et al., 2009], is employed where the variables  $\{\Lambda_n\}$  are assumed to be absent altogether in the test phase, and the problem is treated as inference in MedLDA with  $K$  latent topics. In the M step, the objective in Eq. (4.3) is maximized w.r.t  $\kappa_0$ . The optimal value of  $\beta_{kv}$  is again similar to that of DSLDA [Acharya et al., 2013b]. However, numerical methods for optimization are required to update  $\alpha_1$  or  $\alpha_2$ . The update for the parameters  $\{\mathbf{r}_{1y}\}_{y=1}^Y$  is carried out using online SVM [Bordes et al., 2005, 2007] following Eq. (4.3).

#### 4.2.1.2 Incremental Learning in Active Selection

The method of Expected Entropy Reduction requires one to take an example from the unlabeled pool and one of its possible labels, update the model, and observe the generalized error on the unlabeled pool. This process is computationally expensive unless there is an efficient way to update the model incrementally. The incremental view of EM and the online SVM framework are appropriate for such updates.

Consider that a completely unlabeled or partially labeled document, indexed by  $n'$ , is to be included in the labeled pool with one of the  $(K_2 + 1)$  labels (one for the class label and each different supervised topic), indexed by  $k'$ . In the E step, variational parameters corresponding to all other documents except for the  $n'$ th one are kept fixed and the variational parameters for only the  $n'$ th document are updated. In the M-step, we keep the priors  $\{\alpha^{(1)}, \alpha^{(2)}\}$  over the topics and the SVM parameters  $r_2$  fixed as there is no easy way to update such parameters incrementally. From the empirical point of view, these parameters do not change much w.r.t. the variational parameters (or features in topic space representation) of a single document. However, the update of the parameters  $\{\beta, r_1\}$  is easier. Updating  $\beta$  is accomplished by a simple update of the sufficient statistics. Updating  $r_1$  is done using the “ProcessNew” operation of online SVM followed by a few iterations of “ProcessOld”. The selection of the document-label pair is guided by the measure given in Eq. (2.12). Note that since SVM uses hinge loss which, in turn, upper bounds the 0–1 loss in classification, use of the measure from Eq. (2.12) for active query selection is justified.

From the modeling perspective, the difference between DSLDA [Acharya et al., 2013b] and Act-DSLDA lies in maintaining attribute classifiers and ignoring documents in the max-margin learning that do not have any class label. Online SVM for max-margin learning is essential in the batch mode just to maintain the support vectors and incrementally update them in the active selection step. One could also use incremental EM for batch mode training. However, that is computationally more complex when the labeled dataset is large, as the E step for each document is followed by an M-step in incremental EM.

### 4.3 Active Non-parametric DSLDA (Act-NPDSLDA)

A non-parametric extension of Act-DSLDA (Act-NPDSLDA) automatically determines the best number of latent topics for modeling the given data. It uses a modified stick breaking construction of Hierarchical Dirichlet Process (HDP), recently introduced in [Wang et al., 2011a], to make variational inference feasible. The Act-NPDSLDA generative model is presented below.

- Sample  $\phi_{k_1} \sim \text{Dir}(\eta_1) \forall k_1 \in \{1, 2, \dots, \infty\}$ . Also, sample  $\beta'_{k_1} \sim \text{Beta}(1, \delta_0) \forall k_1 \in \{1, 2, \dots, \infty\}$ .  $\eta_1$  is the parameters of Dirichlet distribution of dimension  $V$ .
- Sample  $\phi_{k_2} \sim \text{Dir}(\eta_2) \forall k_2 \in \{1, 2, \dots, K_2\}$ .  $\eta_2$  is the parameters of Dirichlet distribution of dimension  $V$ .
- For the  $n^{\text{th}}$  document, sample  $\pi_n^{(2)} \sim \text{Dir}(\Lambda_n \alpha^{(2)})$ .  $\alpha^{(2)}$  is the parameter of Dirichlet of dimension  $K_2$ .  $\Lambda_n$  is a diagonal binary matrix of dimension  $K_2 \times K_2$ . The  $k^{\text{th}}$  diagonal entry is unity if the  $n^{\text{th}}$  word is tagged with the  $k^{\text{th}}$  supervised

topic. Similar to the case of Act-DSLDA, in the test data, the supervised topics are not observed and the set of binary SVM classifiers, trained with document-attribute pair data, are used to predict the individual attributes from the input features. The parameters of such classifiers are denoted by  $\{\mathbf{r}_{2k}\}_{1 \leq k \leq K_2}$ .

- $\forall n, \forall t \in \{1, 2, \dots, \infty\}$ , sample  $\pi'_{nt} \sim \text{Beta}(1, \alpha_0)$ . Assume  $\boldsymbol{\pi}_n^{(1)} = (\pi_{nt})_t$  where  $\pi_{nt} = \pi'_{nt} \prod_{l < t} (1 - \pi'_{nl})$ .  $\forall n, \forall t$ , sample  $c_{nt} \sim \text{Mult}(\boldsymbol{\beta})$  where  $\beta_{k_1} = \beta'_{k_1} \prod_{l < k_1} (1 - \beta'_l)$ .  $\boldsymbol{\pi}_n^{(1)}$  represents the probability of selecting the sampled atoms in  $\mathbf{c}_n$ .
- For the  $m^{\text{th}}$  word in the  $n^{\text{th}}$  document, sample  $z_{nm} \sim \text{Mult}((1 - \epsilon)\boldsymbol{\pi}_n^{(1)}, \epsilon\boldsymbol{\pi}_n^{(2)})$ . This implies that with probability  $\epsilon$ , a topic is selected from the set of supervised topics and with probability  $(1 - \epsilon)$ , a topic is chosen from the set of unsupervised topics. Sample  $w_{nm}$  from a multinomial given by Eq. (3).
- For the  $n^{\text{th}}$  document, generate  $Y_n = \arg \max_y \mathbf{r}_{1y}^T \mathbb{E}(\bar{\mathbf{z}}_n)$  where  $Y_n$  is the class label associated with the  $n^{\text{th}}$  document,  $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm} / M_n$ . The maximization problem to generate  $Y_n$  (i.e. the classification problem) is carried out using an online support vector machine. The joint distribution of the hidden and observed variables is given in Eq. (1).

### 4.3.1 Inference and Learning

#### 4.3.1.1 Learning in Batch Mode

As an approximation to the posterior distribution over the hidden variables, we use the factorized distribution given in Eq. (2).  $\boldsymbol{\kappa}_0$  and  $\boldsymbol{\kappa}$  denote the sets of model and variational parameters, respectively.  $\bar{K}_1$  is the truncation limit of the

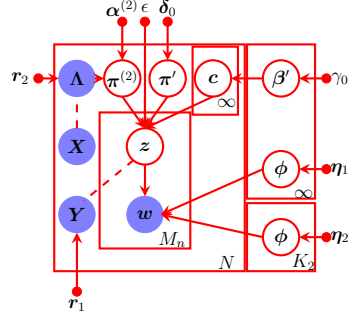


Figure 4.3: Graphical Model of Act-NPDSLDA

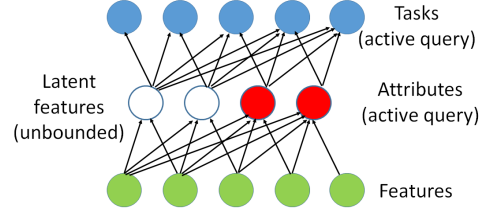


Figure 4.4: Illustration of Act-NPDSLDA

<b>Joint Distribution of Act-NPDSLDA</b>	
$p(\mathbf{X}, \mathbf{Z}   \kappa_0) = \prod_{k_1=1}^{\infty} p(\phi_{k_1}   \eta_1) p(\beta'_{k_1}   \delta_0) \prod_{k_2=1}^{K_2} p(\phi_{k_2}   \eta_2) \prod_{n=1}^N p(\pi_n^{(2)}   \alpha_2) \prod_{t=1}^{\infty} p(\pi_{nt}^{(1)}   \alpha_0) p(c_{nt}   \beta') \prod_{m=1}^{M_n} p(z_{nm}   \pi_n^{(1)}, \pi_n^{(2)}, \epsilon) p(w_{nm}   \phi, c_{nz_{nm}}, z_{nm}). \quad (1)$	
<b>Variational Distribution of Act-NPDSLDA</b>	
$q(\mathbf{Z}   \kappa) = \prod_{k_1=1}^{\bar{K}_1} q(\phi_{k_1}   \lambda_{k_1}) \prod_{k_2=1}^{K_2} q(\phi_{k_2}   \lambda_{k_2}) \prod_{k_1=1}^{\bar{K}_1-1} q(\beta'_{k_1}   u_{k_1}, v_{k_1}) \prod_{n=1}^N q(\pi_n^{(2)}   \gamma_n) \prod_{t=1}^{T-1} q(\pi_{nt}^{(1)}   a_{nt}, b_{nt}) \prod_{t=1}^T q(c_{nt}   \varphi_{nt}) \prod_{m=1}^{M_n} q(z_{nm}   \zeta_{nm}). \quad (2)$	
<b>Multinomial Distribution for Sampling Words in Act-NPDSLDA</b>	
$\prod_{k_1=1}^{\infty} \prod_{v=1}^V \phi_{k_1 v}^{\mathbb{I}\{w_{nm}=v\}} \mathbb{I}\{c_{nz_{nm}}=k_1 \in \{1, \dots, \infty\}\} \prod_{k_2=1}^{K_2} \prod_{v=1}^V \phi_{k_2 v}^{\mathbb{I}\{w_{nm}=v\}} \mathbb{I}\{z_{nm}=k_2 \in \{1, \dots, K_2\}\}. \quad (3)$	

Table 4.1: Distributions in Act-NPDSLDA

corpus-level Dirichlet Process and  $T$  is the truncation limit of the document-level Dirichlet Process.  $\{\lambda_k\}$  are the parameters of the Dirichlet, each of dimension  $V$ .  $\{u_{k_1}, v_{k_1}\}$  and  $\{a_{nt}, b_{nt}\}$  are the parameters of Beta distribution corresponding to corpus level and document level sticks respectively.  $\{\varphi_{nt}\}$  are multinomial parameters of dimension  $\bar{K}_1$  and  $\{\zeta_{nm}\}$  are multinomials of dimension  $(T + K_2)$ .  $\{\gamma_n\}_n$  are parameters of the Dirichlet distribution of dimension  $K_2$ .

The underlying optimization problem takes the same form as in Eq. (4.3). The only difference lies in the calculation of  $\Delta f_n(y) = f(Y_n, \bar{s}_n) - f(y, \bar{s}_n)$ . The first set of dimensions of  $\bar{s}_n$  (corresponding to the unsupervised topics) is given by  $1/M_n \sum_{m=1}^{M_n} c_{nz_{nm}}$ , where  $c_{nt}$  is an indicator vector over the set of unsupervised topics. The following  $K_2$  dimensions (corresponding to the supervised topics) are given by  $1/M_n \sum_{m=1}^{M_n} z_{nm}$ . After the variational approximation with  $\bar{K}_1$  number of corpus level sticks,  $\bar{s}_n$  turns out to be of dimension  $(\bar{K}_1 + K_2)$  and the feature vector  $f(y, \bar{s}_n)$  constitutes  $Y(\bar{K}_1 + K_2)$  elements. The components of  $f(y, \bar{s}_n)$  from  $(y - 1)(\bar{K}_1 + K_2) + 1$  to  $y(\bar{K}_1 + K_2)$  are those of the vector  $\bar{s}_n$  and all the others are 0. The E-step update equations of Act-NPDSLDA are similar to NP-DSLDA [Acharya et al., 2013b]. The M-step updates are similar to Act-DSLDA and are omitted here due to space constraints.

#### 4.3.1.2 Incremental Learning in Active Selection

Assume that a completely unlabeled or partially labeled document, indexed by  $n'$ , is to be included in the labeled pool with the  $k'$ th label. In the E step, variational parameters corresponding to all other documents except for the  $n'$ th one is

kept fixed and the variational parameters for only the  $n$ 'th document are updated. The incremental update of the “global” variational parameters  $\{u_{k_1}, v_{k_1}\}_{k_1=1}^{K_1}$  is also straightforward following the equations given in [Acharya et al., 2013b]. In the M-step, we keep the priors  $\{\eta_1, \eta_2, \alpha^{(2)}\}$  and the SVM parameters  $r_2$  fixed but the parameters  $r_1$  are updated using online SVM.

## 4.4 Experimental Results

### 4.4.1 Methodology for Experiments with Active Multitask Learning

We evaluate Act-DSLDA and Act-NPDSLDA on two real world datasets, aYahoo and ACM Conference, described in Section 3.4.1.1 and Section 3.4.1.2 respectively. Act-DSLDA and Act-NPDSLDA are compared against the following simplified models:

- Active Learning in MedLDA with **one-vs-all** classification (Act-MedLDA-OVA) (shown in Fig. 4.5): A separate MedLDA model is trained for each class using a one-vs-all approach leaving no possibility of transfer across classes. Supervised topics are not included in such modeling and the class labels are also obtained using active learning.
- Active Learning in MedLDA with **multitask learning** (Act-MedLDA-MTL) (shown in Fig. 4.6): A single MedLDA model is learned for all classes where the latent topics are shared across classes. Again, supervised topics are not used and the class labels are obtained using active learning. This baseline is intended to be stronger than Act-MedLDA-OVA where the latent topics are not shared.

- Act-DSLDA with **only shared supervised topics** (Act-DSLDA-OSST) (shown in Fig. 4.7): A model in which supervised topics are used and shared across classes but there are no latent topics. Both the supervised topics and the class labels are queried using active selection strategy.
- Act-DSLDA with **no shared latent topics** (Act-DSLDA-NSLT) (shown in Fig. 4.8): A model in which only supervised topics are shared across classes and a separate set of latent topics is maintained for each class. Both the supervised topics and the class labels are queried using active selection strategy. This model has richer representational capacity compared to Act-DSLDA-OSST which does not use any latent topics at all.
- Random selection of only class labels (MedLDA-MTL-Random) (shown in Fig. 4.9): A MedLDA-MTL model where examples with only class labels are selected at random but supervised topics are not used at all. Note that designing a DSLDA based model where only class labels are selected at random is tricky as one needs to balance the number of supervised topics queried and the number of class labels selected at random. This baseline shows the utility of active selection of classes in the MedLDA-MTL framework.
- Random selection of class and attribute labels (DSLDA-Random) (shown in Fig. 4.10): A DSLDA model where both queries for class and the supervised topics are selected at random. This baseline is weaker than RSC since the supervised topics are generally less informative compared to class labels. Both MedLDA-MTL-Random and DSLDA-Random are used to exhibit the utility of active learning

for both class and supervised topic selection.

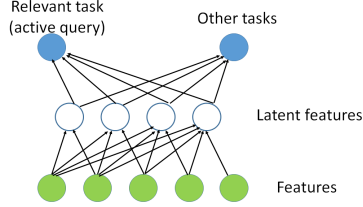


Figure 4.5: Illustration of Act-MedLDA-OVA

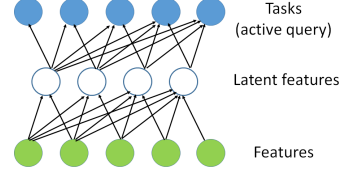


Figure 4.6: Illustration of Act-MedLDA-MTL

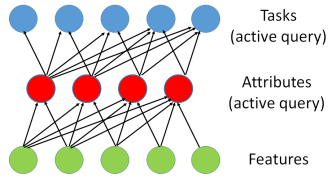


Figure 4.7: Illustration of Act-DSLDA-OSST

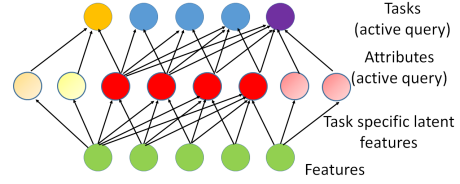


Figure 4.8: Illustration of Act-DSLDA-NSLT

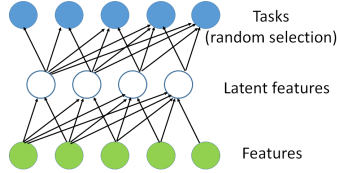


Figure 4.9: Illustration of MedLDA-MTL-Random

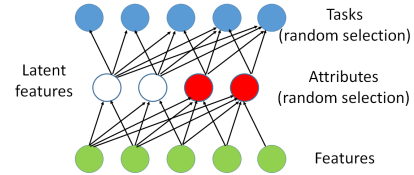


Figure 4.10: Illustration of DSLDA-Random

#### 4.4.2 Active Multitask Learning Results

For the experiments with active multitask learning, we start with a completely labeled dataset  $\mathcal{L}$  consisting of 300 documents. In every active iteration, we query for 50 labels (class labels or supervised topics). Figs. 4.3 and 4.4 present representative learning curves for the image and the text data respectively, show-

ing how classification accuracy improves as the amount of supervision is increased.

The error bars in the curves show standard deviations across 20 trials.

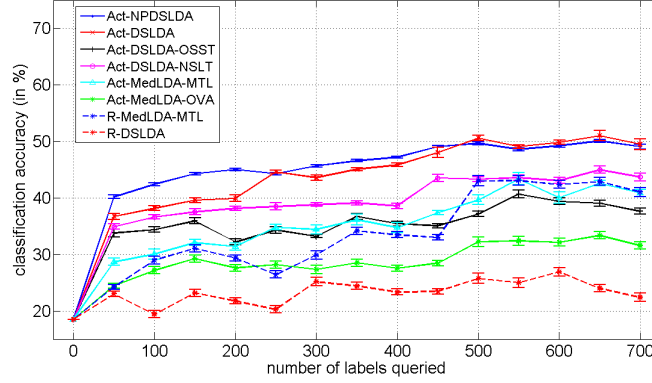


Figure 4.11: aYahoo Learning Curves

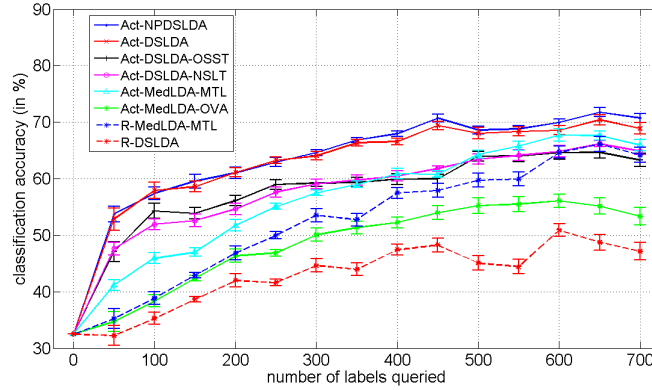


Figure 4.12: ACM Conference Learning Curves

#### 4.4.3 Discussion

DSLDA-NSLT only allows sharing of supervised topics and its implementation is not straightforward. Since MedLDA-OVA, MedLDA-MTL and DSLDA use

$K$  topics (latent or a combination of supervised and latent), to make the comparison fair, it is necessary to maintain the same number of topics for DSLDA-NSLT. This ensures that the models compared have the same representational capacity. Therefore, for each class in DSLDA-NSLT,  $k_2/Y$  latent topics are maintained. While training DSLDA-NSLT with examples from the  $y^{\text{th}}$  class, only a subset of the first  $k_1$  topics (or a subset of the supervised ones based on which of them are present in the training documents) and the next  $\left(\frac{(y-1)k_2}{Y} + 1\right)^{\text{th}}$  to  $\left(\frac{yk_2}{Y}\right)^{\text{th}}$  topics are considered to be “active” among the latent topics. The other latent topics are assumed to have zero contribution, implying that the parameters associated with these topics are not updated based on observations of documents belonging to class  $y$ . During testing, however, one needs to project a document onto the entire  $K$ -dimensional space, and the class label is predicted based on this representation and the parameters  $\mathbf{r}$ .

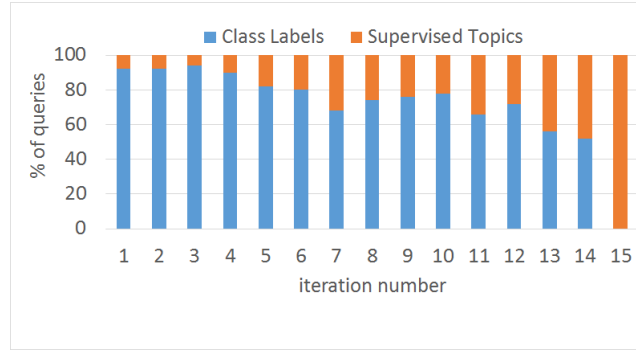


Figure 4.13: aYahoo Query Distribution

Act-DSLDA and Act-NPDSLDA quite consistently outperform all of the baselines, clearly demonstrating the advantage of combining both types of topics and integrating active learning and transfer learning in the same framework. Act-NPDSLDA performs about as well or better as Act-DSLDA, for which the optimal

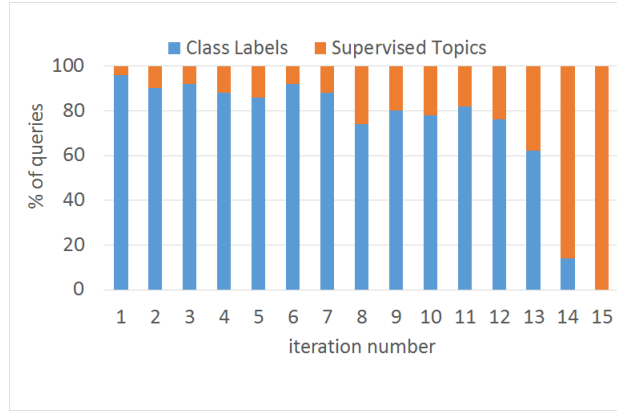


Figure 4.14: ACM Conference Query Distribution

number of latent topics has been chosen using an expensive model-selection search.

As to be expected, the active DSLDA methods' advantage over their random selection counterpart (RSC) is greatest at the lower end of the learning curve. Act-MedLDA-OVA does a little better than RSCA showing that the active selection of class labels helps even if there is no transfer across classes. Act-MedLDA-MTL consistently outperforms Act-MedLDA-OVA as well as RSC showing that active transfer learning is beneficial for MedLDA-MTL. Act-DSLDA-OSST does better than both Act-MedLDA-MTL and RSC towards the lower end of the learning curve but with more labeled information this model does not perform that well since it does not use latent topics. Act-DSLDA-NSLT also performs better than Act-DSLDA-OSST because the former utilizes latent topics.

Figs. 4.13 and 4.14 show the percentage (out of 50 queries) of class labels and supervised topics queried by Act-DSLDA at each iteration step in the vision and text data, respectively. Initially, the model queries for more class labels but towards

Latent Topic 1	function, transduction, label, graph, algorithm, accuracy
Latent Topic 2	architecture, stack, performance, processor, layer, system
VLSI	global, robust, design, physical, cells, layout, growth
IR	repair, site, search, semantics, algorithm, web
SVD	matrix, decomposition, rank, performance, completion
Clustering	model, information, hierarchical, mixture, task

Table 4.2: Latent and Supervised Topics Discovered by Act-DSLDA

the end of the learning curve, more supervised topics are queried. By the 14<sup>th</sup> iteration, the class labels of all the documents in the training set are queried. From the 15<sup>th</sup> iteration onwards, only supervised topics are queried. This observation is not that surprising since the class labels are more discriminative compared to the supervised topics and hence are queried more. However, queries of supervised topics are also helpful and allow continued improvement later in the learning curve.

**Topic Illustration:** In Table 4.2, we show the most indicative words for several topics discovered by Act-DSLDA from the text data after all the class labels are queried. We emphasize that such interpretability is one of the key artifacts of the proposed models.

## 4.5 Conclusion

This paper has introduced two new models, Act-DSLDA and Act-NPDSDLA, for active multitask learning. Experimental results comparing to six different ablations of these models demonstrate the utility of integrating active and multitask learning in one framework that also unifies latent and supervised shared topics. The computational complexity of the proposed models largely depends on the active se-

lection mechanism adopted. For large scale applications, one needs to use better approximation techniques for active selection as suggested in [Jain and Kapoor, 2009; Vijayanarasimhan et al., 2014]. In the next Chapter, we discuss how one could additionally actively query for rationales [Donahue and Grauman, 2011; Zaidan et al., 2008] and further improve the predictive performance of Act-DSLDA.

## **Chapter 5**

### **Active Multitask Learning Using Annotators’ Rationale**

In traditional supervised learning framework, a human annotator is typically asked for labels, of classes or supervised topics. We have seen in the previous chapters how these labels can be incorporated into a knowledge transfer framework and queried over in a smarter way using active learning. Arguably, this is a rather restricted form of engagement of the human annotators in a supervised learning process as the annotators do not only have ideas about the labels, but also why a given label is associated with a given image or text document. Hence, in this chapter, our objective is to receive deeper cues from the annotators and include them in the learning process. For example, if an annotator believes that a given image belongs to a particular class or contains a particular supervised topic, then he can also annotate part of the image that lead him to that belief [Donahue and Grauman, 2011]. One can see Fig. 5.1) for examples of the annotation process for images. We develop this annotation framework using Google app engine where an image is presented with either the category label or one of its attributes and the annotator draws a bounding box which might correspond to the specific label provided. Likewise, if an annotator thinks that a document should belong to some class, then he/she can highlight parts of the text or the set of words that influenced

him/her to make that decision [Zaidan et al., 2008].

**Annotation for Rationale**

**Instructions:**

- Please click and drag on the image within the black border to select a region (as a bounding box) that you believe accounts for the following label:

"Snout"

- To deselect a bounding box, just click once outside the bounding box on the image.
- You can specify only one region. If there are multiple regions, please select one randomly.
- Just press the submit button if you don't find any relevant region or leave a feedback.
- Annotate as many images as possible. :-)

**Position of the bounding box:**

X <sub>1</sub> :	<input type="text" value="71"/>	X <sub>2</sub> :	<input type="text" value="182"/>	Width:	<input type="text" value="111"/>
Y <sub>1</sub> :	<input type="text" value="369"/>	Y <sub>2</sub> :	<input type="text" value="484"/>	Height:	<input type="text" value="115"/>

[Any feedback? \(optional\)](#)

Please click here if you don't want to continue further.

Submit Your Response

**Annotation for Rationale**

**Instructions:**

- Please click and drag on the image within the black border to select a region (as a bounding box) that you believe accounts for the following label:

"Face"

- To deselect a bounding box, just click once outside the bounding box on the image.
- You can specify only one region. If there are multiple regions, please select one randomly.
- Just press the submit button if you don't find any relevant region or leave a feedback.
- Annotate as many images as possible. :-)

**Position of the bounding box:**

X <sub>1</sub> :	<input type="text" value="192"/>	X <sub>2</sub> :	<input type="text" value="262"/>	Width:	<input type="text" value="70"/>
Y <sub>1</sub> :	<input type="text" value="34"/>	Y <sub>2</sub> :	<input type="text" value="110"/>	Height:	<input type="text" value="76"/>

[Any feedback? \(optional\)](#)

Please click here if you don't want to continue further.

Submit Your Response

Figure 5.1: Illustration of Rationale for Images

The rest of the chapter is organized as follows. We present related work in Section 5.1. The probabilistic model for active multitask learning with annotators' rationale is presented in Section 5.2. The experimental results are reported in Section 5.3 which is then followed by conclusion in Section 5.4.

## 5.1 Related Work

Emergence of online services such as Mechanical Turk has facilitated annotations of large amount of images quickly and efficiently. However, to enhance the quality of feedback and improve the learning of algorithms, researchers are exploring the impact of requesting deeper, more complete annotations. This includes gathering fully segmented [Russell et al., 2008], pose-annotated [Bourdev and Malik, 2009], or attribute-labeled images [Acharya et al., 2013b; Farhadi et al., 2009; Siddiquie et al., 2011]. Attribute labels increase the labeling cost, but often reveal

useful mid-level cues [Acharya et al., 2013a, 2014b; Kumar et al., 2011], or enable novel tasks like zero-shot learning [Lampert et al., 2009]. Human describable properties offer a new way for the annotator to communicate to the learning algorithm, and better teach it to recognize a complex visual category. Interestingly, in both language and vision, researchers have studied how to capture elements most important to a human. The information can be explicitly gathered through classic iterative relevance feedback [Chang et al., 2005]. However, more implicit measures are also possible, such as by learning what people mention first in a natural scene [Hwang and Grauman, 2012; Spain and Perona, 2008], or what they deem a foreground object [Spain and Perona, 2008]. Whereas such methods use these cues to predict important regions in novel images, our goal is to use what a human deems influential so as to better predict the category label for novel images. Work in natural language processing explores whether humans can pick out words relevant for a given document category as a form of human feature selection [Druck et al., 2009; Raghavan et al., 2005]. In particular, the NLP method of [Zaidan et al., 2008] proposes rationales to better predict sentiment in written movie reviews, and inspires our approach; we adapt the authors’ basic idea to create two new forms of contrast examples for the visual domain.

The basic framework proposed in [Donahue and Grauman, 2011; Zaidan et al., 2008] for learning with rationales entails an SVM learner. For simple binary classification problems, a positive example is forced to maintain a gap with a negative example as well as the example derived from masking features of the positive example that an annotator found relevant for labeling the example positive.

Formally, the SVM objective takes the following form:

$$\begin{aligned}
& \frac{1}{2} \|\mathbf{w}\|^2 + C_1 \sum_n \xi_n + C_2 \sum_{n,m} \xi_{nm}, \quad (5.1) \\
& \text{s.t. } \forall n, y_n \langle \mathbf{w}, \mathbf{x}_n \rangle \geq (1 - \xi_n), \xi_n \geq 0, \\
& \forall (n, m) \in \chi, y_n \langle \mathbf{w}, (\mathbf{x}_n - \mathbf{v}_{nm}) \rangle \geq \mu(1 - \xi_{nm}), \xi_{nm} \geq 0.
\end{aligned}$$

where,  $\mathbf{x}_n$  is the  $n^{\text{th}}$  example,  $y_n \in \{-1, +1\}$  is the label of the  $n^{\text{th}}$  example and  $\mathbf{v}_{nm}$  is the example obtained from the positive example  $\mathbf{x}_n$  by removing features found relevant for the example being positive by the  $m^{\text{th}}$  annotator, conveniently addressed as a “contrast example”. A collection of positive and associated contrast examples is denoted by  $\chi$ .  $\xi_n$  and  $\xi_{nm}$  are the slack variables corresponding to the two different margin constraints, one of which is scaled by the parameter  $\mu$ , thereby allowing more flexibility in the modeling.  $C_1$  and  $C_2$  determine how the margin violations would be penalized w.r.t the regularization term. The bias terms are omitted above, but accounted for by appending a 1-element to each training example.

## 5.2 Active Learning with Annotators’ Rationale in Doubly Supervised Latent Dirichlet Allocation (Act-Rationale-DSLDA)

The existing formulation of learning with rationales has two major drawbacks. Firstly, the formulation is valid only for binary classification problems and cannot be readily extended to solve multi-class problems. Secondly, the annotation is usually provided at the category level, and not for the supervised topics. Our application involves a multi-class problem. Additionally, to exploit the annotators’

knowledge more comprehensively, we require them to give feedback for selecting both the class label and the supervised topics. Below, we propose a framework, built on Act-DSLDA, which can accept annotators' rationale and potentially improve the predictive performance when labeled data is sparse.

Assume we are given an initial training corpus  $\mathcal{L}$  with  $N$  documents belonging to  $Y$  different classes (where each document belongs to exactly one class and each class corresponds to a different task). We overload the word "document" to imply an image and the word "word" to indicate "visual words", obtained from a quantization of SIFT features, for image data. Further assume that each of these training documents is also annotated with a set of  $K_2$  different "supervised topics". The objective here is to train a model using the words in a document, as well as the associated supervised topics and class labels, and then use this model to classify completely unlabeled test data for which no topic or class label is provided. When the learning starts,  $\mathcal{L}$  is assumed to have fully labeled documents. However, as the learning progresses more documents are added to the pool  $\mathcal{L}$  with class and/or a subset of supervised topics labeled. Therefore, at any intermediate point of the learning process,  $\mathcal{L}$  can be assumed to contain several sets:  $\mathcal{L} = \{\mathcal{T} \cup \mathcal{T}_C \cup \mathcal{T}_{A_1} \cup \dots \cup \mathcal{T}_{A_{K_2}}\}$ , where  $\mathcal{T}$  contains fully labeled documents (*i.e.* with both class and all of supervised topics labeled) and  $\mathcal{T}_C$  represents documents that have class labels. For  $1 \leq k \leq K_2$ ,  $\mathcal{T}_{A_k}$  represents the documents that have the  $k^{\text{th}}$  supervised topic labeled. Since, human provided annotations and class labels are expensive to obtain, we design an active learning framework where the model can query over an unlabeled pool  $\mathcal{U}$  and request either class labels or a subset of the

supervised topics.

Let the  $n^{\text{th}}$  document be annotated by  $L_n$  annotators. The  $l_n^{\text{th}}$  annotation comes with an explanation of the form of highlighting a set of words  $\{w'_{nl_n}\}_{l_n}$  which the corresponding annotator found relevant for identifying the document's class label or supervised topic. In case of an image, the annotator draws a bounding box around certain relevant regions and the SIFT features, that appear in such regions are considered relevant for the label, which in turn affect the bag-of-words representation. For text data, this annotation is little more straightforward as one just needs to highlight some set of words. Once these set of words  $\{w'_{nl}\}_{l_n}$  is removed from the document, it no longer belongs to the class that the annotator identified. However, it does not also mean that the derived document can belong to some other class. To avoid such ambiguity for such derived documents, we define one extra class which we call a “negative” class, index it by  $(Y + 1)$ , and assign the derived documents to this particular class. In case of rationales for supervised topics, only the set of words can be removed and the derived document is not assumed to be annotated by that supervised topic anymore. The corresponding attribute classifier can also be retrained with this derived attribute as done in normal annotator rationale work. The rest of the generative process goes as follows:

- For the  $n^{\text{th}}$  document, sample a topic selection probability vector  $\theta_n \sim \text{Dir}(\alpha_n)$ , where  $\alpha_n = \Lambda_n \alpha$  and  $\alpha$  is the parameter of a Dirichlet distribution of dimension  $K$ , which is the total number of topics. The topics are assumed to be of two types – latent and supervised, and there are  $K_1$  latent topics and  $K_2$  supervised topics.

Therefore,  $K = K_1 + K_2$ . Latent topics are never observed, while supervised topics are observed in training but not in test data. Henceforth, in each vector or matrix with  $K$  components, it is assumed that the first  $K_1$  components correspond to the latent topics and the next  $K_2$  components to the supervised topics.  $\Lambda_n$  is a diagonal binary matrix of dimension  $K \times K$ . The  $k^{\text{th}}$  diagonal entry is unity if *either*  $1 \leq k \leq K_1$  *or*  $K_1 < k \leq K$  and the  $n^{\text{th}}$  document is tagged with the  $k^{\text{th}}$  topic. Also,  $\alpha = (\alpha_1, \alpha_2)$  where  $\alpha_1$  is a parameter of a Dirichlet distribution of dimension  $K_1$  and  $\alpha_2$  is a parameter of a Dirichlet distribution of dimension  $K_2$ .

- For the  $m^{\text{th}}$  word in the  $n^{\text{th}}$  document, sample a topic  $z_{nm} \sim \text{multinomial}(\theta'_n)$ , where  $\theta'_n = (1 - \epsilon)\{\theta_{nk}\}_{k=1}^{K_1} + \epsilon\{\Lambda_{n,kk}\theta_{nk}\}_{k=1+K_1}^K$ . This implies that the supervised topics are weighted by  $\epsilon$  and the latent topics are weighted by  $(1 - \epsilon)$ . Sample the word  $w_{nm} \sim \text{multinomial}(\beta_{z_{nm}})$ , where  $\beta_k$  is a multinomial distribution over the vocabulary of words corresponding to the  $k^{\text{th}}$  topic.

- For the  $n^{\text{th}}$  document, generate  $Y_n = \arg \max_y \mathbf{r}_y^T \mathbb{E}(\bar{\mathbf{z}}_n)$  where  $Y_n$  is the class label associated with the  $n^{\text{th}}$  document,  $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm}/M_n$ . Here,  $\mathbf{z}_{nm}$  is an indicator vector of dimension  $K$ .  $\mathbf{r}_y$  is a  $K$ -dimensional real vector corresponding to the  $y^{\text{th}}$  class, and it is assumed to have a prior distribution  $\mathcal{N}(0, 1/C)$ .  $M_n$  is the number of words in the  $n^{\text{th}}$  document. The maximization problem to generate  $Y_n$  (or the classification problem) is carried out using a max-margin principle similar to MedLDA.

- For the  $l_n^{\text{th}}$  document derived by removing words from the  $n^{\text{th}}$  document, the generative process is same and limited to the set of words contained in that derived

document. If the annotator’s feedback on this derived document is for the class label, the derived document is included in the set  $\mathcal{C}_n$ . Any document that belongs to this set is assigned to the  $(Y + 1)^{\text{th}}$  class during training.

### 5.2.1 Inference and Learning

Inference and parameter estimation have two phases – one for the batch case when the model is trained with completely labeled data, and the other for the active selection step where the model has to be incrementally updated to observe the effect of label or rationale that is queried from the oracle. Conceptually, the learning and inference are very similar to those of Act-DSLDA and Act-NPDSLDA described in the previous Chapter. However, Act-DSLDA and Act-NPDSLDA do not deal with rationales and hence, to make the description of learning and inference unambiguous, in what follows, we explain them rather meticulously.

#### 5.2.1.1 Learning in Batch Mode

Let us denote the hidden variables by  $\mathbf{Z} = \{\{z_{nm}\}, \{\boldsymbol{\theta}_n\}\}$ , the observed variables by  $\mathbf{X} = \{w_{nm}\}$  and the model parameters by  $\boldsymbol{\kappa}_0$ . The joint distribution of the hidden and observed variables is:

$$p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\kappa}_0) = \prod_{n=1}^N p(\boldsymbol{\theta}_n | \boldsymbol{\alpha}_n) \prod_{m=1}^{M_n} p(z_{nm} | \boldsymbol{\theta}'_n) p(w_{nm} | \boldsymbol{\beta}_{z_{nm}}). \quad (5.2)$$

To avoid computational intractability, inference and estimation are performed using variational **EM**. The factorized approximation of the posterior distribution with

hidden variables  $\mathbf{Z}$  is given by:

$$q(\mathbf{Z}|\{\boldsymbol{\kappa}_n\}_{n=1}^N) = \prod_{n=1}^N q(\boldsymbol{\theta}_n|\gamma_n) \prod_{m=1}^{M_n} q(z_{nm}|\phi_{nm}), \quad (5.3)$$

where  $\boldsymbol{\theta}_n \sim \text{Dir}(\gamma_n)$ ,  $z_{nm} \sim \text{multinomial}(\phi_{nm}) \forall n \in \{1, 2, \dots, N\}$  and  $\forall m \in \{1, 2, \dots, M_n\}$ , and  $\boldsymbol{\kappa}_n = \{\gamma_n, \{\phi_{nm}\}\}$ , which is the set of variational parameters corresponding to the  $n^{\text{th}}$  instance. Further,  $\gamma_n = (\gamma_{nk})_{k=1}^K \forall n$ , and  $\phi_{nm} = (\phi_{nmk})_{k=1}^K \forall n, m$ . With the use of the lower bound obtained by the factorized approximation, followed by Jensen's inequality, Act-Rationale-DSLDA reduces to solving the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\zeta}} \sum_{y=1}^Y \frac{1}{2} \|\mathbf{r}_{1y}\|^2 + \sum_{k_2=1}^{K_2} \frac{1}{2} \|\mathbf{r}_{2k_2}\|^2 - \sum_{n=1}^N \ell(\boldsymbol{\kappa}_n, \boldsymbol{\kappa}) + \sum_{n=1}^N (C_1 \xi_{1n} + C_2 \xi_{2n} + C_3 \xi_{3n} + C_4 \xi_{4n}), \quad (5.4) \\ \forall y \in \{1, 2, \dots, Y\} \setminus \{Y_n\}, \forall n \in \{\mathcal{T}_C \setminus \chi_C\} : \mathbb{E}[\mathbf{r}_{1y}^T \Delta f_n(y)] \geq (1 - \xi_{1n}), \xi_{1n} \geq 0, \\ \forall y \in \{1, 2, \dots, Y\} \setminus \{Y_n\}, \forall l_n \in \chi_C : \mathbb{E}[\mathbf{r}_{1y}^T \Delta f'_n(y)] \geq \mu(1 - \xi_{2n}), \xi_{2n} \geq 0, \\ \forall k_2 \in \{1, 2, \dots, K_2\}, \forall n \in \mathcal{T}_{A_{k_2}} \setminus \chi_{A_{k_2}} : y_{nk_2} \langle \mathbf{r}_{2k_2}, \mathbf{w}_n \rangle \geq (1 - \xi_{3n}), \xi_{3n} \geq 0, \\ \forall k_2 \in \{1, 2, \dots, K_2\}, \forall n, l_n \in \chi_{A_{k_2}} : y_{nk_2} \langle \mathbf{r}_{2k_2}, (\mathbf{w}_n - \mathbf{w}'_{l_n}) \rangle \geq \mu(1 - \xi_{4n}), \xi_{4n} \geq 0. \end{aligned}$$

where  $\boldsymbol{\zeta} = \{\{\mathbf{r}_{1y}\}, \{\mathbf{r}_{2k_2}\}, \{\boldsymbol{\kappa}_n\}, \boldsymbol{\kappa}\}$ ,  $\Delta f_n(y) = f(Y_n, \bar{\mathbf{z}}_n) - f(y, \bar{\mathbf{z}}_n)$ , and  $\Delta f'_n(y) = f(Y_n, \bar{\mathbf{z}}'_n) - f(y, \bar{\mathbf{z}}'_n)$ .  $\{\xi_{1n}, \xi_{2n}, \xi_{3n}, \xi_{4n}\}_{n=1}^N$  are the slack variables, and  $f(y, \bar{\mathbf{z}}_n)$  is a feature vector whose components from  $(y-1)K+1$  to  $yK$  are those of the vector  $\bar{\mathbf{z}}_n$  and all the others are 0.  $\mathbb{E}[\mathbf{r}_{1y}^T \Delta f_n(y)]$  is the “expected margin” over which the true label  $Y_n$  is preferred over a prediction  $y$ .  $\bar{\mathbf{z}}'_n$  is the feature vector obtained from the variational approximation with  $\{w_{l_n}\}_{l_n=1}^{L_n}$  removed. The parameter  $\mu$  adds some flexibility in the modeling by maintaining separate margins for “negative” examples and “contrast” examples. From this viewpoint, Act-Rationale-DSLDA projects the

documents onto a combined topic space and then uses a max-margin approach to predict the class label. The parameters  $C_1, C_2, C_3, C_4$  penalize the margin violation of the training data.

Let  $\mathcal{Q}$  be the set of all distributions having a fully factorized form as given in (8.2). Note that such a factorized approximation makes the use of incremental variation of EM possible in the active selection step following the discussion in Section 2.4.3.2. Let the distribution  $q^*$  from the set  $\mathcal{Q}$  optimize the objective in Eq. (5.4). The optimal values of the corresponding variational parameters are same as those of DSLDA [Acharya et al., 2013b]. The optimal values of  $\phi_{nm}$  depend on  $\gamma_n$  and vice-versa. Therefore, iterative optimization is adopted to maximize the lower bound until convergence is achieved.

During testing, one does not observe a document's supervised topics and instead an approximate solution, as also used in [Acharya et al., 2013b; Ramage et al., 2009], is employed where the variables  $\{\Lambda_n\}$  are assumed to be absent altogether in the test phase, and the problem is treated as inference in MedLDA with  $K$  latent topics. In the M step, the objective in Eq. (5.4) is maximized w.r.t  $\kappa_0$ . The optimal value of  $\beta_{kv}$  is again similar to that of DSLDA [Acharya et al., 2013b]. However, numerical methods for optimization are required to update  $\alpha_1$  or  $\alpha_2$ . The update for the parameters  $\{\mathbf{r}_{1y}\}_{y=1}^Y$  is carried out using online SVM [Bordes et al., 2005, 2007] following Eq. (5.4).

### 5.2.1.2 Incremental Learning in Active Selection

The method of Expected Entropy Reduction requires one to take an example from the unlabeled pool and one of its possible labels, update the model, and observe the generalized error on the unlabeled pool. This process is computationally expensive unless there is an efficient way to update the model incrementally. The incremental view of EM and the online SVM framework are appropriate for such updates. Consider that a completely unlabeled or partially labeled document, indexed by  $n'$ , is to be included in the labeled pool with one of the  $(K_2 + 1)$  labels (one for the class label and each different supervised topic) only, indexed by  $k'$  or the label with its corresponding rationale. In the E-step, variational parameters corresponding to all other documents except for the  $n'$ th one are kept fixed and the variational parameters for only the  $n'$ th document are updated, based on the label or the label with its rationale. In the M-step, we keep the priors  $\{\alpha^{(1)}, \alpha^{(2)}\}$  over the topics and the SVM parameters  $\{r_{2k_2}\}$  fixed as there is no easy way to update such parameters incrementally. From the empirical point of view, these parameters do not change much w.r.t. the variational parameters (or features in topic space representation) of a single document. However, the update of the parameters  $\{\beta, \{r_{1y}\}\}$  is easier. Updating  $\beta$  is accomplished by a simple update of the sufficient statistics. Updates of  $\{r_{1y}\}$  are performed using the “ProcessNew” operation of online SVM followed by a few iterations of “ProcessOld”. The selection of the document-label pair is guided by the measure given in Eq. (2.12). Note that since SVM uses hinge loss which, in turn, upper bounds the 0 – 1 loss in classification, use of the measure from Eq. (2.12) for active query selection is justified.

## 5.3 Experimental Results

### 5.3.1 Datasets

We explore the utility of our approach with two different datasets. The first one is the ACM conference abstract dataset, described in detail in Section 3.4.1.2, and the second one is the aYahoo dataset, explained in detail in Section 3.4.1.1. For both these datasets, we annotate the examples with their classes, supervised topics and the associated rationales. From our experiments, we found out that there is significant mismatch among the processes of identifying class labels, supervised topics, and annotating rationale for class labels and supervised topics. This is illustrated in Figs. 5.2 and 5.3. Note that annotating a rationale in a text document usually takes longer time than annotating rationale in an image. This is primarily due to the fact that selecting a part of an image is relatively easy and can be done just by visual inspection. However, for text documents, one needs to look for the representative set of words, a harder cognitive problem. For ACM conference data, according to Fig. 5.2, we observe that the mean query time for rationale corresponding to the class labels is approximately 4.0 times larger than that corresponding to identifying the class labels or supervised topics. Similarly, the mean query time for rationale of supervised topics is approximately 6.0 times larger than that of identifying the class labels or supervised topics. For aYahoo dataset, these factor are approximately 1.6 and 3.0 respectively (see Fig. 5.3 for the distribution in query times). The reason for the difference in query times for two types of annotations is that the supervised topics are much more specific compared to the class labels. Hence we had to spend extra time in finding the cues from the images and texts

for the supervised topics. For large scale implementation, this difference in annotation cost should be taken into account while designing the algorithm, otherwise one could spend lot more time and money in retrieving annotations from online engines like Mechanical Turk. Therefore, in our implementation, for querying for either the class labels or supervised topics, we use unit cost. However, for annotation for class labels we use a cost of 4.0 for the ACM conference dataset and 1.6 for the aYahoo dataset. For annotation for supervised topics, these costs are 6.0 for the ACM conference dataset and 3.0 for the aYahoo dataset. We incorporate these costs in Eq. 2.12 while making decision about which query to make. The information gain is reduced by these costs when a label or a label with its rationale is included in the training pool and the model is incrementally updated.

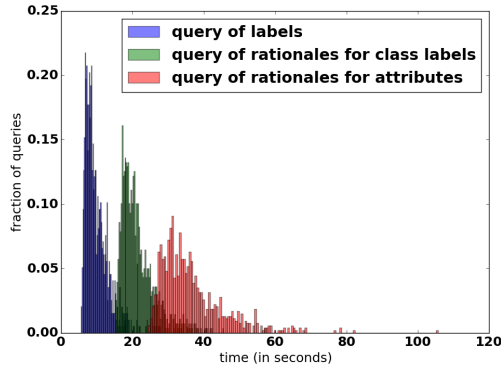


Figure 5.2: Distribution of query time for rationales – ACM Conference

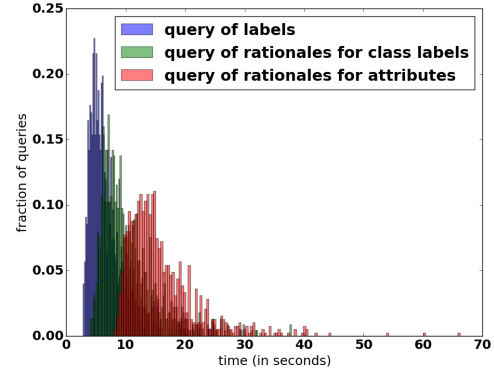


Figure 5.3: Distribution of query time for rationales – aYahoo

### 5.3.2 Methodology for Experiments

We evaluate the performance of Act-Rationale-DSLDA against the following baselines:

- **Act-DSLDA:** This model is described in detail in Section 4.2. Act-DSLDA queries only for class labels or supervised topics, but does not use any rationale at all. We have seen in Chapter 4, that Act-DSLDA outperforms some of the relevant baselines that do not use active learning or some other components of Act-DSLDA. Hence, this is a strong baseline to beat.
- **Act-Rationale-Class-DSLDA:** This is Act-Rationale-DSLDA where the rationales for the supervised topics are not queried, but the rationales for the class labels can be queried.
- **Act-Rationale-Topics-DSLDA:** This is Act-Rationale-DSLDA where the rationales for the class labels are not queried, but the rationales for the supervised topics can be queried.

### 5.3.3 Results

For the experiments with active multitask learning, we start with a completely labeled dataset  $\mathcal{L}$  consisting of 300 documents. In every active iteration, we query for 50 labels (class labels or supervised topics). Figs. 5.4 and 5.5 present representative learning curves for the ACM conference data and the aYahoo data respectively, showing how classification accuracy improves as the amount of super-

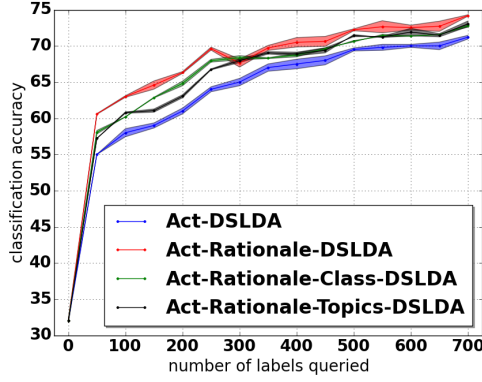


Figure 5.4: ACM Conference Results

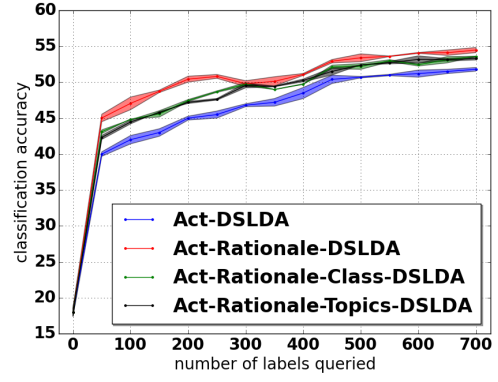


Figure 5.5: aYahoo Results

vision is increased. The error bars in the curves show standard deviations across 20 trials.

Figs. 5.6 and 5.7 show the distribution of the class labels, supervised topics and annotations queried by Act-Rationale-DSLDA at each iteration in the ACM conference and aYahoo data, respectively. Note that a query for a class label or supervised topic may or may not get augmented with the corresponding rationale. The fact that a label can get augmented with its corresponding rationale when queried for leads to four different types of answer for a query: only class labels, class labels with rationale, only supervised topics, and supervised topics with rationale. From Figs. 5.6 and 5.7, one can see that initially the model queries for more labels along with their annotations, but towards the end of the learning curve, this trend drops. To make the illustration more clear and interpretable, we present the distribution of query types for every alternate active learning epoch. These plots clearly demonstrate the utility of learning with rationales, of both class labels and supervised topics. The performances of two baselines Act-Rationale-Class-DSLDA and

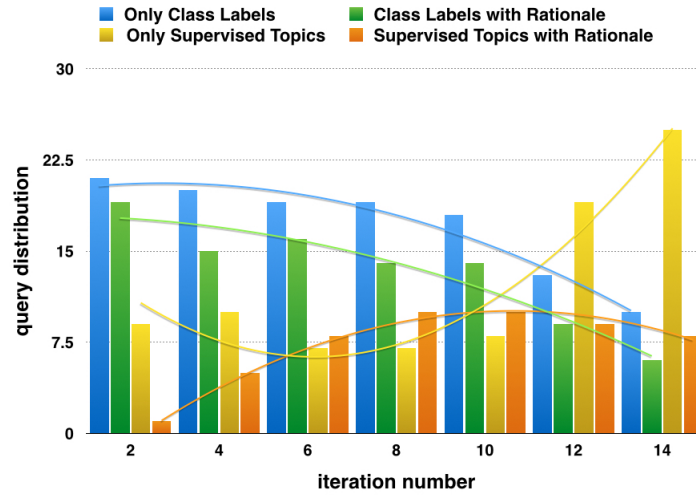


Figure 5.6: Distribution of query types – ACM Conference

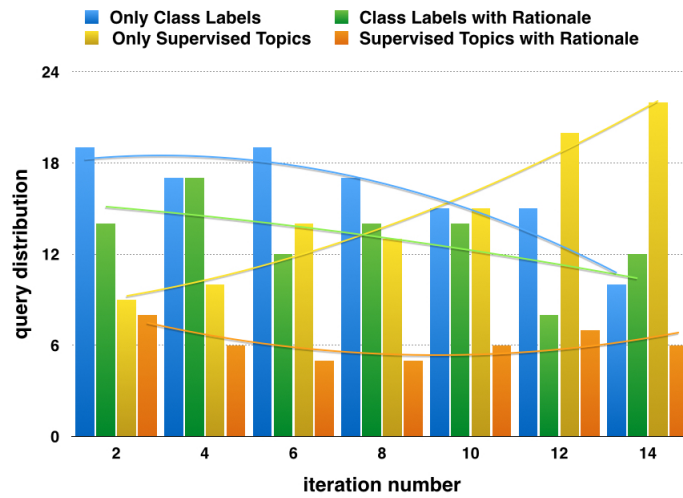


Figure 5.7: Distribution of query types – aYahoo

Act-Rationale-Topics-DSLDA are very similar, and both of them gain from the extra information provided by the rationales. However, the performance of these two baselines is inferior compared to Act-Rationale-DSLDA which uses rationale for both class labels and supervised topics. Act-DSLDA performs the worst among all the models as it does not take information about rationales into account. Note that we plot the learning curves till we incorporate all the class labels from the unlabeled pool. Therefore, unlike in traditional active learning setup, we don't get to see a "banana" curve as one can still incorporate supervised topics and their rationales once the class labels are exhausted. Also, another reasonable comparison among the proposed method and the baselines could be performed by using the cost of annotation, instead of the number of labels queried, as the independent variable. We leave that as an interesting future work.

## 5.4 Conclusion

This chapter has introduced Act-Rationale-DSLDA, a framework that unifies active learning, multitask learning, latent topics, supervised topics, and learning with rationales. Extensive experiments with both text and vision datasets reveal that the proposed method can exploit the rationales provided by the annotators efficiently to improve the predictive performance. In the next Chapter, we explore a different type of simultaneous knowledge transfer for network imputation using auxiliary information.

## Chapter 6

### **Gamma Process Poisson Factorization for Joint Modeling of Network and Documents**

In the last three chapters, we have seen the successful application of simultaneous knowledge transfer for object recognition from images and analysis of text documents. In this chapter, we explore a different type of simultaneous knowledge transfer in the context of social network analysis.

Social networks and other relational datasets often involve a large number of nodes  $N$  with sparse connections between them. If the relationship is symmetric, it can be represented compactly using a binary symmetric adjacency matrix  $\mathbf{B} \in \{0, 1\}^{N \times N}$ , where  $b_{ij} = b_{ji} = 1$  if and only if nodes  $i$  and  $j$  are linked. Often, the nodes in such datasets are also associated with “side information,” such as documents read or written, movies rated, or messages sent by these nodes. Integer-valued side information are commonly observed and can be naturally represented by a count matrix  $\mathbf{Y} \in \mathbb{Z}^{D \times V}$ , where  $\mathbb{Z} = \{0, 1, \dots\}$ . For example,  $\mathbf{B}$  may represent a coauthor network and  $\mathbf{Y}$  may correspond to a document-by-word count matrix representing the documents written by all these authors. In another example,  $\mathbf{B}$  may represent a user-by-user social network and  $\mathbf{Y}$  may represent a user-by-item rating matrix that adds nuance and support to the network data. Incorporating such side

information can result in better community identification and superior link prediction performance as compared to modeling only the network adjacency matrix  $\mathbf{B}$ , especially for sparse networks.

Many of the popular network models [Airoldi et al., 2008; Gopalan et al., 2012; Kemp et al., 2006; Miller et al., 2009; Palla et al., 2012] are demonstrated to work well for small size networks. However, small networks are often dense, while larger real-world networks tend to be much sparser and hence challenge existing modeling approaches. Incorporating auxiliary information associated with the nodes has the potential to address such challenges, as it may help better identify latent communities and predict missing links. A model that takes advantage of such side information has the potential to outperform network-only models. However, the side information may not necessarily suggest the same community structure as the existing links. Thus a network latent factor model that allows separate factors for side information and network interactions, but at the same time is equipped with a mechanism to capture dependencies between the two types of factors, is desirable.

This chapter proposes **Joint Gamma Process Poisson Factorization (J-GPPF)** to jointly factorize  $\mathbf{B}$  and  $\mathbf{Y}$  in a nonparametric Bayesian manner. The paper makes the following contributions: 1) we present a fast and effective model that uses side information to help discover latent network structures, 2) we perform nonparametric Bayesian modeling for discovering latent structures in both  $\mathbf{B}$  and  $\mathbf{Y}$ , and 3) our model scales with the number of non-zero entries in the network  $S_{\mathbf{B}}$  as  $O(S_{\mathbf{B}}K_{\mathbf{B}})$ , where  $K_{\mathbf{B}}$  is the number of network groups inferred from the data.

The remainder of the chapter is organized as follows. We present back-

ground material and related work in Section 6.1. We introduce two new models, N-GPPF (in Section 6.2) for network analysis and C-GPPF (in Section 6.3) for analysis of count matrix, which are combined in an intuitive way in J-GPPF for joint analysis of network and documents. J-GPPF and its inference algorithm are explained in Section 6.4. Experimental results are reported in Section 6.5, followed by conclusions in Section 6.6.

## 6.1 Related Work

The Infinite Relational Model (IRM) [Kemp et al., 2006] allows for multiple types of relations between entities in a network and an infinite number of clusters, but restricts these entities to belong to only one cluster. The Mixed Membership Stochastic Blockmodel (MMSB) [Airoldi et al., 2008] assumes that each node in the network can exhibit a mixture of communities. Though the MMSB has been applied successfully to discover complex network structure in a variety of applications, the computational complexity of the underlying inference mechanism is in the order of  $N^2$ , which limits its use to small networks. Computation complexity is also a problem with many other existing latent variable network models, such as the latent feature relational model [Miller et al., 2009] and its max margin version [Zhu, 2012], and the infinite latent attribute model [Palla et al., 2012]. The Assortative Mixed-Membership Stochastic Blockmodel (a-MMSB) [Gopalan et al., 2012] bypasses the quadratic complexity of the MMSB by making certain assumptions about the network structure that might not be true in general. The hierarchical Dirichlet process relational model [Kim et al., 2013] allows mixed membership with an un-

bounded number of latent communities; however, it is built on the a-MMSB whose assumptions could be restrictive.

Some of the existing approaches handle sparsity in real-world networks by using some auxiliary information [Leskovec and Julian, 2012; Menon and Elkan, 2011; Yoshida, 2013]. For example, in a protein-protein interaction network, the features describing the biological properties of each protein can be used [Menon and Elkan, 2011]. In an extremely sparse social network, information about each user’s profile can be used to better recommend friends [Leskovec and Julian, 2012]. Recommender system and text mining researchers, in contrast, tend to take an orthogonal approach. In recommender systems [Chaney et al., 2013; Ma et al., 2008],  $\mathbf{Y}$  may represent a user-by-item rating matrix and the objective in this setting is to predict the missing entries in  $\mathbf{Y}$ , and the social network matrix  $\mathbf{B}$  plays a secondary role in providing auxiliary information to facilitate this task [Ma et al., 2008]. Similarly, in the text mining community, many existing models [Balasubramanyan and Cohen, 2011; McCallum et al., 2007; Nallapati et al., 2008; Wen and Lin, 2010] use the network information or other forms of side information to improve the discovery of “topics” from the document-by-word matrix  $\mathbf{Y}$ . The matrix  $\mathbf{B}$  can represent, for example, the interaction network of authors participating in writing the documents. The Relational Topic Model [Chang and Blei, 2009] discovers links between documents based on their topic distributions, obtained through unsupervised exploration. The Author-Topic framework [Rosen-Zvi et al., 2004] and the Author-Recipient-Topic model [McCallum et al., 2007] jointly model documents along with the authors of the documents. Block-LDA [Balasubramanyan and Co-

hen, 2011], on the other hand, provides a generative model for the links between authors and recipients in addition to documents. The Nubbi model [Chang et al., 2009] discovers entity relations from the text data by relying on words that appear in the context of entities and entity pairs in the text. The Group-Topic model [Wang et al., 2006] addresses the task of modeling events pertaining to pairs of entities with textual attributes that annotate the event. Wen and Lin [2010] describe an approach that uses both content and network information to analyse enterprise data. J-GPPF differs from these existing approaches in mathematical formulation, including more effective modeling of both sparsity and the dependence between network interactions and side information.

J-GPPF models both  $\mathbf{Y}$  and  $\mathbf{B}$  using Poisson factorization. As discussed in [Acharya et al., 2015], Poisson factorization has several practical advantages over other factorization methods that use Gaussian assumptions (*e.g.* in [Ma et al., 2008]). First, zero-valued observations could be efficiently processed during inference, so the model can readily accommodate large, sparse datasets. Second, Poisson factorization is a natural representation of count data. Additionally, the model allows mixed membership across an unbounded number of latent communities using the gamma Process as a prior. The authors in [Ball et al., 2011] also use Poisson factorization to model a binary interaction matrix. However, this is a parametric model and a KL-divergence based objective is optimized w.r.t. the latent factors without using any prior information. To model the binary observations of the network matrix  $\mathbf{B}$ , J-GPPF additionally uses a novel Poisson-Bernoulli (PoBe) link, discussed in detail in Section 6.4, that transforms the count values from the

Poisson factorization to binary values. Similar transformation has also been used in the BigCLAM model [Yang and Leskovec, 2013] which builds on the works of [Ball et al., 2011]. This model was later extended to include non-network information in the form of binary attributes [Yang et al., 2013]. Neither BigCLAM nor its extension allows non-parametric modeling or imposing prior structure on the latent factors, thereby limiting the flexibility of the models and making the obtained solutions more sensitive to initialization. The collaborative topic Poisson factorization (CTPF) framework proposed in [Gopalan et al., 2014b] solves a different problem where the objective is to recommend articles to users of similar interest. CTPF is a parametric model and variational approximation is adopted to solve the inference.

Joint matrix factorization is also addressed by Factorization machines (FM) [Rendle, 2010], which combines high prediction quality of factorization models with the flexibility of feature engineering. In challenging prediction problems, where additional side-information is available and/or higher order features may be needed, new challenges due to feature engineering needs arise. While interaction terms are typically desired in such scenarios, the number of such terms grows very quickly. This dilemma is cleverly addressed by the FM which represents data as real-valued features like standard machine learning approaches, such as SVMs, and uses interactions between each pair of variables. However, by restricting the interaction to a latent space, the number of parameters needed to be determined is kept manageable. Interestingly, the framework of FM subsumes many successful factorization models like matrix factorization [Koren et al., 2009], SVD++ [Koren, 2008], TimeSVD++ [Koren, 2009], PITF [Rendle and Schmidt-Thieme, 2010] and

FPMC [Rendle et al., 2010]. This generalization ability of FM is perhaps its main drawback. The more popular learning algorithms for FM are stochastic gradient descent (SGD) [Koren et al., 2009] and MCMC sampling [C. Freudenthaler, 2011] both of which require lot of parameter tuning. Additionally, the data is still assumed to be normally distributed, and hence, unlike in Poisson factorization (to be discussed in detail later), the sparsity of the matrices is not utilized. Neither does any of the proposed FM algorithms discover the ideal number of latent factors from the data. We plan for a thorough empirical analysis with FM methods in future though.

## 6.2 Gamma Process Poisson Factorization for Networks (N-GPPF)

Let there be a network of  $N$  users encoded as an  $N \times N$  binary matrix  $B$ . To model the latent factors in a network, a Gamma process  $G \sim \Gamma\mathcal{P}(c, G_0)$  is maintained, a draw from which is expressed as  $G = \sum_{k=1}^{\infty} r_k \delta_{\phi_k}$ , where  $\phi_k \in \Omega$  is an atom drawn from an  $N$ -dimensional base distribution as  $\phi_k \sim \prod_{n=1}^N \text{Gamma}(e_0, 1/c_n)$  and  $r_k = G(\phi_k)$  is the associated weight. Also,  $\gamma_0 = G_0(\Omega)$  is defined as the mass parameter corresponding to the base measure  $G_0$ . The  $(n, m)^{\text{th}}$  entry in the matrix  $B$  is assumed to be derived from a latent count as:

$$b_{nm} = \mathbb{I}_{\{x_{nm} \geq 1\}}, \quad x_{nm} \sim \text{Pois}(\lambda_{nm}), \quad \lambda_{nm} = \sum_k \lambda_{nmk}, \quad (6.1)$$

where  $\lambda_{nmk} = r_k \phi_{nk} \phi_{mk}$ . This is called as the Poisson-Bernoulli (PoBe) link in [Acharya et al., 2015; Zhou, 2015]. The distribution of  $b_{nm}$  given  $\lambda_{nm}$  is named as the Poisson-Bernoulli distribution, explained in Section 2.1.3. One may consider

$\lambda_{nmk}$  as the strength of mutual latent community membership between nodes  $n$  and  $m$  in the network for latent community  $k$ , and  $\lambda_{nm}$  as the interaction strength aggregating all possible community membership. For example, consider professional and recreational interactions between people  $n, m$ , and  $m'$  who all work together. Person  $n$  has about the same level of professional interactions with both persons  $m$  and  $m'$ . Yet if we add the condition that person  $n$  and  $m'$  go fishing together during the weekend,  $n$  and  $m'$  will have membership in the “fishing together” latent community while  $n$  and  $m$  will not. The strength of interactions between any two persons could be considered as the aggregation of a possibly infinite kinds of latent community memberships. Using Lemma 2.1.4, one may augment the above representation as  $x_{nm} = \sum_k x_{nmk}$ ,  $x_{nmk} \sim \text{Pois}(\lambda_{nmk})$ . Thus each interaction pattern contributes a count and the total latent count aggregates the countably infinite interaction patterns.

Unlike the usual approach that links the binary observations to latent Gaussian random variables with a logistic or probit function, the above approach links the binary observations to Poisson random variables. Thus, this approach transforms the problem of modeling binary network interaction into a count modeling problem, providing several potential advantages. First, it is more interpretable because  $r_k$  and  $\phi_k$  are non-negative and the aggregation of different interaction patterns increases the probability of establishing a link between two nodes. Second, the computational benefit is significant since the computational complexity is approximately linear in the number of non-zeros  $S$  in the observed binary adjacency matrix  $\mathbf{B}$ . This benefit is especially pertinent in many real-world datasets where  $S$  is significantly smaller

than  $N^2$ . To complete the generative process, we put Gamma priors over  $c$  and  $c_n$  as:

$$c \sim \text{Gamma}(c_0, 1/d_0), c_n \sim \text{Gamma}(f_0, 1/g_0). \quad (6.2)$$

### 6.2.1 Gibbs Sampling for N-GPPF

Though N-GPPF supports countably infinite number of latent communities for network modeling, in practice it is impossible to instantiate all of them. Instead of marginalizing out the underlying stochastic process [Blackwell and MacQueen, 1973; Neal, 2000] or using slice sampling [Walker, 2007] for non-parametric modeling, for simplicity, a finite approximation of the infinite model is considered by truncating the number of graph communities  $K$ . Such an approximation approaches the original infinite model as  $K$  approaches infinity. With such finite approximation, the generative process of N-GPPF is further summarized in Table 6.1.

$b_{nm} = I_{\{x_{nm} \geq 1\}}, x_{nm} \sim \text{Pois} \left( \sum_k r_k \phi_{nk} \phi_{mk} \right),$ $r_k \sim \text{Gamma}(\gamma_k, 1/c), \phi_{nk} \sim \text{Gamma}(e_0, 1/c_n), c_n \sim \text{Gamma}(f_0, 1/g_0),$ $\gamma_k \sim \text{Gamma}(a_0, 1/b_0), c \sim \text{Gamma}(c_0, 1/d_0).$
---

Table 6.1: Generative Process of N-GPPF

**Sampling of  $x_{nmk}$  :**  $x_{nm}$ 's are sampled only corresponding to the following entries:

$$(n, m) : n = \{1, \dots, (N-1)\}, m = \{(n+1), \dots, N\}.$$

For the above entries the sampling goes as follows:

$$x_{nm} | \sim b_{nm} \text{Pois}_+ \left( \sum_{k=1}^K r_k \phi_{nk} \phi_{mk} \right), \quad (6.3)$$

where  $\text{Pois}_+(\cdot)$  is the truncated Poisson distribution, the sampling from which is detailed in Section 2.1.3. Since, one can augment  $x_{nm} \sim \text{Pois}\left(\sum_{k=1}^K \lambda_{nmk}\right)$  as  $x_{nm} = \sum_{k=1}^K x_{nmk}$ , where  $x_{nmk} \sim \text{Pois}(\lambda_{nmk})$ , equivalently, one obtains the following according to Lemma 2.1.4:

$$(x_{nmk})_{k=1}^K | \sim \text{Mult} \left( (r_k \phi_{nk} \phi_{mk})_{k=1}^K / \sum_{k=1}^K r_k \phi_{nk} \phi_{mk}; x_{nm} \right). \quad (6.4)$$

**Sampling of  $\phi_{nk}$  and  $r_k$  :** Sampling of these parameters follow from Lemma 2.1.1 and are given as follows:

$$\phi_{nk} | \sim \text{Gamma} \left( e_0 + \sum_{m=1}^{(n-1)} x_{mnk} + \sum_{m=(n+1)}^N x_{nmk}, 1 / \left( c_n + r_k \sum_{\substack{m=1 \\ m \neq n}}^N \phi_{mk} \right) \right), \quad (6.5)$$

$$r_k | \sim \text{Gamma} \left( \gamma_k + \sum_{n=1, n < m}^{(N-1), N} x_{nmk}, 1 / \left( c + \sum_{n=1, n < m}^{(N-1), N} \phi_{nk} \phi_{mk} \right) \right). \quad (6.6)$$

**Sampling of  $c_n$  and  $c$  :** Sampling of these parameters follow from Lemma 2.1.2 and are given as follows:

$$c_n | \sim \text{Gamma} \left( f_0 + K e_0, 1 / \left( g_0 + \sum_{k=1}^K \phi_{nk} \right) \right), \quad (6.7)$$

$$c | \sim \text{Gamma} \left( c_0 + \sum_{k=1}^K \gamma_k, 1 / \left( d_0 + \sum_{k=1}^K r_k \right) \right). \quad (6.8)$$

**Sampling of  $\gamma_k$  :** Using Lemma 2.1.4, one can show that  $x_{..k} \sim \text{Pois}(r_k s_k)$ , where  $x_{..k} = \sum_{n=1}^{(N-1)} \sum_{m=(n+1)}^N x_{nmk}$ , and  $s_k = \sum_{n=1}^{(N-1)} \sum_{m=(n+1)}^N \phi_{nk} \phi_{mk}$ . Since  $r_k \sim \text{Gam}(\gamma_k, 1/c)$  and one can augment  $\ell_k \sim \text{CRT}(x_{..k}, \gamma_k)$ , following Lemma 2.2.2 one can sample:

$$\gamma_k | \sim \text{Gamma}(a_0 + \ell_k, 1/(b_0 - \log(1 - p_k))), \quad (6.9)$$

where  $p_k = s_k/(c + s_k)$ .

### 6.2.2 Gibbs Sampling for N-GPPF with Missing Entries

Variables whose update get affected in presence of missing entries  $\mathcal{M}$  are  $\phi_{nk}$ 's and  $r_k$ 's. Sampling of these parameters follow from Lemma 2.1.1 and are given as follows:

$$\phi_{nk} | \sim \text{Gamma} \left( e_0 + \sum_{\substack{m=1 \\ (m,n) \notin \mathcal{M}}}^{(n-1)} x_{mnk} + \sum_{\substack{m=(n+1) \\ (n,m) \notin \mathcal{M}}}^N x_{nmk}, 1 / \left( c_n + r_k \sum_{\substack{m=1 \\ m \neq n; (n,m) \notin \mathcal{M}}}^N \phi_{mk} \right) \right), \quad (6.10)$$

$$r_k | \sim \text{Gamma} \left( \gamma_k + \sum_{\substack{n=1, n < m \\ (n,m) \notin \mathcal{M}}}^{(N-1), N} x_{nmk}, 1 / \left( c + \sum_{\substack{n=1, n < m \\ (n,m) \notin \mathcal{M}}}^{(N-1), N} \phi_{nk} \phi_{mk} \right) \right). \quad (6.11)$$

## 6.3 Gamma Process Poisson Factorization for Count Matrices (C-GPPF)

Let there be a count matrix of  $\mathbf{Y}$  of dimension  $D \times V$ . We introduce a Gamma process  $G \sim \Gamma P(c, G_0)$ , a draw from which is expressed as  $G = \sum_{k=1}^{\infty} r_k \delta_{\theta_k}$ , where  $\theta_k \in \Omega$  is an atom drawn from a  $D$ -dimensional base distribution as  $\theta_k \sim \prod_{d=1}^D \text{Gam}(g_0, 1/c_d)$  and  $r_k = G(\phi_k)$  is the associated weight. Also,  $\gamma_0 = G_0(\Omega)$  is defined as the mass parameter corresponding to the base measure  $G_0$ . Each atom  $\theta_k$  is marked with an atom  $\beta_k$ , drawn from a  $V$ -dimensional base distribution as  $\beta_k \sim \prod_{w=1}^V \text{Gam}(h_0, 1/s_w)$ . The  $(d, w)^{\text{th}}$  entry in the matrix  $\mathbf{Y}$  is assumed to be derived from a sum of latent counts as  $y_{dw} \sim \text{Pois}(\sum_k \lambda_{dwk})$  where  $\lambda_{dwk} = r_k \theta_{dk} \beta_{wk}$ .

One may consider  $\lambda_{dwk}$  as the strength of the latent factor that dictates the relation between the  $d^{\text{th}}$  user and the  $w^{\text{th}}$  item. Each latent factor contributes such a count and the total count aggregates the countably infinite latent factors. Each of these latent counts is composed of three parts. The parameter  $r_k$  models the global popularity of the latent factor  $k$ ,  $\theta_{dk}$  models the affinity of the  $d^{\text{th}}$  user to the  $k^{\text{th}}$  latent factor and  $\beta_{wk}$  models the popularity of the  $w^{\text{th}}$  word among the  $k^{\text{th}}$  latent factor. As described in Section 2.3, such modeling assumption is one instance of Poisson factor analysis. To complete the generative process, we put Gamma priors over  $c$ ,  $c_d$  and  $s_w$  as:

$$c \sim \text{Gam}(c_0, 1/d_0), c_d \sim \text{Gam}(e_0, 1/f_0), s_w \sim \text{Gam}(t_0, 1/u_0). \quad (6.12)$$

$y_{dw} \sim \text{Pois} \left( \sum_{k_Y=1}^{\infty} r_k \theta_{dk} \beta_{wk} \right), \boldsymbol{\theta}_k \sim \prod_{d=1}^D \text{Gam}(g_0, 1/c_d),$ $\boldsymbol{\beta}_k \sim \prod_{w=1}^V \text{Gam}(h_0, 1/s_w), c_d \sim \text{Gam}(e_0, 1/f_0), s_w \sim \text{Gam}(t_0, 1/u_0),$ $r_k \sim \text{Gamma}(\gamma_k, 1/c), \gamma_k \sim \text{Gamma}(a_0, 1/b_0).$
---

Table 6.2: Generative Process of C-GPPF

### 6.3.1 Gibbs Sampling for C-GPPF

A finite approximation of the infinite model is considered by truncating the number of factors to  $K$  which approaches the original infinite model as  $K \rightarrow \infty$ . The sampling proceeds as follows:

**Sampling of  $x_{dwk}$**  : This follows from the relation between Poisson and multinomial distribution, given in Lemma 2.1.4, and can be derived as:

$$(x_{dwk})_{k=1}^K | \sim \text{mult} \left( r_k \theta_{dk} \beta_{wk} / \sum_{k=1}^K r_k \theta_{dk} \beta_{wk}; y_{dw} \right). \quad (6.13)$$

**Sampling of  $r_k, \theta_{dk}$  and  $\beta_{wk}$  :** Sampling of these variables can be derived according to Lemma 2.1.1 as:

$$r_k | \sim \text{Gam}(\gamma_k + x_{..k}, 1/(c + \theta_{.k} \beta_{.k})), \quad (6.14)$$

$$\theta_{dk} | \sim \text{Gam}(g_0 + x_{d.k}, 1/(c_d + r_k \beta_{.k})), \quad (6.15)$$

$$\beta_{wk} | \sim \text{Gam}(h_0 + x_{.wk}, 1/(s_w + r_k \theta_{.k})). \quad (6.16)$$

**Sampling of  $c_d, s_w$  and  $c$  :** Sampling of these variables can be derived according to Lemma 2.1.2 and are given as:

$$c_d | \sim \text{Gam}(e_0 + K g_0, 1/(f_0 + \theta_d.)), \quad (6.17)$$

$$s_w | \sim \text{Gam}(t_0 + K h_0, 1/(u_0 + \beta_w.)), \quad (6.18)$$

$$c | \sim \text{Gam} \left( \sum_{k=1}^K \left( \gamma_k + \sum_{t=0}^{(T-1)} r_{tk} \right) + c_0, 1 / \left( \sum_{k=1}^K \sum_{t=0}^T r_{tk} + d_0 \right) \right). \quad (6.19)$$

**Sampling of  $\gamma_k$  :** Using Lemma 2.1.4, one can show that  $x_{..k} \sim \text{Pois}(r_k s_k)$ , where  $x_{..k} = \sum_{d,w} x_{dwk}$ , and  $s_k = \sum_{d,w} \theta_{dk} \beta_{wk}$ . Since  $r_k \sim \text{Gam}(\gamma_k, 1/c)$  and one can augment  $\ell_k \sim \text{CRT}(x_{..k}, \gamma_k)$ , following Lemma 2.2.2 one can sample:

$$\gamma_k | \sim \text{Gamma}(a_0 + \ell_k, 1/(b_0 - \log(1 - p_k))), \quad (6.20)$$

where  $p_k = s_k/(c + s_k)$ . A consequence of closed form updates for Gibbs sampling is that the computation per iteration for CGPPF is  $O((S + D + V)K)$  where  $S$  is the number of number of non-zero entries, which is a huge saving for sparse matrices compared to Probabilistic matrix factorization (PMF) [Salakhutdinov and Mnih,

2007] and Bayesian probabilistic matrix factorization (BPMF) [Salakhutdinov and Mnih, 2008] whose computation cost per iteration is  $O(DVK^2)$ . This follows from the underlying assumptions of Poisson distribution. When the observation is zero, the corresponding latent counts  $\{x_{dwk}\}_{k=1}^K$  are zero with probability 1, and hence one needs to sample latent counts corresponding to non-zero entries only.

### 6.3.2 Gibbs Sampling for C-GPPF with Missing Entries

Variables whose update get affected in presence of missing values are  $r_k$ 's and  $\theta_{dk}$ 's and  $\beta_{wk}$ 's. Rest of the update equations are same as in the C-GPPF without any missing value. Below, the updates are enlisted, where  $\mathcal{M}$  denotes the set of missing entries.

**Sampling of  $r_k$ ,  $\theta_{dk}$  and  $\beta_{wk}$  :** Sampling of  $r_k$ ,  $\theta_{dk}$  and  $\beta_{wk}$  follow from Lemma 2.1.1:

$$r_k | \sim \text{Gam} \left( \gamma_k + \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_t}}^{D,V} x_{dwk}, 1/(c + \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_t}}^{D,V} \theta_{dk} \beta_{wk}) \right), \quad (6.21)$$

$$\theta_{dk} | \sim \text{Gam} \left( d_0 + \sum_{\substack{w=1 \\ (d,w) \notin \mathcal{M}_t}}^V x_{dwk}, 1/(c_d + \sum_{\substack{w=1 \\ (d,w) \notin \mathcal{M}}}^V r_k \beta_{wk}) \right), \quad (6.22)$$

$$\beta_{wk} | \sim \text{Gam} \left( h_0 + \sum_{\substack{d=1 \\ (d,w) \notin \mathcal{M}}}^D x_{dwk}, 1/(s_w + \sum_{\substack{d=1 \\ (d,w) \notin \mathcal{M}}}^D r_k \theta_{dk}) \right). \quad (6.23)$$

## 6.4 Joint Gamma Process Poisson Factorization (J-GPPF)

Let there be a network of  $N$  users encoded in an  $N \times N$  binary matrix  $\mathbf{B}$ . The users in the network participate in writing  $D$  documents summarized in a  $D \times V$  count matrix  $\mathbf{Y}$ , where  $V$  is the size of the vocabulary. Additionally, a binary matrix  $\mathbf{Z}$  of dimension  $D \times N$  can also be maintained, where the unity entries in each column indicate the set of documents in which the corresponding user contributes. In applications where  $\mathbf{B}$  represents a user-by-user social network and  $\mathbf{Y}$  represents a user-by-item rating matrix,  $\mathbf{Z}$  turns out to be an  $N$ -dimensional identity matrix. However, in the following model description we consider the more general document-author framework. Also, to make the notations more explicit, the variables associated with the side information have  $\mathbf{Y}$  as a subscript (e.g.,  $G_{\mathbf{Y}}$ ) and those associated with the network make similar use of the subscript  $\mathbf{B}$  (e.g.,  $G_{\mathbf{B}}$ ).

We employ two separate Gamma Processes. The first one models the latent factors in the network. A draw from this Gamma Process  $G_{\mathbf{B}} \sim \Gamma\mathbf{P}(c_{\mathbf{B}}, H_{\mathbf{B}})$  is expressed as  $G_{\mathbf{B}} = \sum_{k_{\mathbf{B}}=1}^{\infty} \rho_{k_{\mathbf{B}}} \delta_{\phi_{k_{\mathbf{B}}}}$ , where  $\phi_{k_{\mathbf{B}}} \in \Omega_{\mathbf{B}}$  is an atom drawn from an  $N$ -dimensional base distribution as  $\phi_{k_{\mathbf{B}}} \sim \prod_{n=1}^N \text{Gam}(a_{\mathbf{B}}, 1/\sigma_n)$  and  $\rho_{k_{\mathbf{B}}} = G_{\mathbf{B}}(\phi_{k_{\mathbf{B}}})$  is the associated weight. The second Gamma Process models the latent groups of side information. A draw from this gamma process  $G_{\mathbf{Y}} \sim \Gamma\mathbf{P}(c_{\mathbf{Y}}, H_{\mathbf{Y}})$  is expressed as  $G_{\mathbf{Y}} = \sum_{k_{\mathbf{Y}}=1}^{\infty} r_{k_{\mathbf{Y}}} \delta_{\beta_{k_{\mathbf{Y}}}}$ , where  $\beta_{k_{\mathbf{Y}}} \in \Omega_{\mathbf{Y}}$  is an atom drawn from a  $V$ -dimensional base distribution as  $\beta_{k_{\mathbf{Y}}} \sim \prod_{w=1}^V \text{Gam}(\xi_{\mathbf{Y}}, 1/\zeta_w)$  and  $r_{k_{\mathbf{Y}}} = G_{\mathbf{Y}}(\beta_{k_{\mathbf{Y}}})$  is the associated weight. Also,  $\gamma_{\mathbf{B}} = H_{\mathbf{B}}(\Omega_{\mathbf{B}})$  is defined as the mass parameter corresponding to the base measure  $H_{\mathbf{B}}$  and  $\gamma_{\mathbf{Y}} = H_{\mathbf{Y}}(\Omega_{\mathbf{Y}})$  is defined as the mass parameter corresponding to the base measure  $H_{\mathbf{Y}}$ . The  $(n, m)^{\text{th}}$

entry in the matrix  $\mathbf{B}$  is assumed to be derived from a latent count as:

$$b_{nm} = \mathbb{I}_{\{x_{nm} \geq 1\}}, x_{nm} \sim \text{Pois}(\lambda_{nm}), \lambda_{nm} = \sum_{k_B} \lambda_{nmk_B},$$

where  $\lambda_{nmk_B} = \rho_{k_B} \phi_{nk_B} \phi_{mk_B}$ . This is called as the Poisson-Bernoulli (PoBe) link in [Acharya et al., 2015; Zhou, 2015]. To model the matrix  $\mathbf{Y}$ , its  $(d, w)^{\text{th}}$  entry  $y_{dw}$  is generated as:

$$y_{dw} \sim \text{Pois}(\zeta_{dw}), \zeta_{dw} = \left( \sum_{k_Y} \zeta_{Ydwk_Y} + \sum_{k_B} \zeta_{Bdwk_B} \right),$$

where  $\zeta_{Ydwk_Y} = r_{k_Y} \theta_{dk_Y} \beta_{wk_Y}$ ,  $Z_{nd} \in \{0, 1\}$  and  $Z_{nd} = 1$  if and only if author  $n$  is one of the authors of paper  $d$  and  $\zeta_{Bdwk_B} = \epsilon \rho_{k_B} (\sum_n Z_{nd} \phi_{nk_B}) \psi_{wk_B}$ . One can consider  $\zeta_{dw}$  as the affinity of document  $d$  for word  $w$ , This affinity is influenced by two different components, one of which comes from the network modeling. Without the contribution from network modeling, the joint model reduces to a gamma process Poisson matrix factorization model, in which the matrix  $\mathbf{Y}$  is factorized in such a way that  $y_{dw} \sim \text{Pois}(\sum_{k_Y} r_{k_Y} \theta_{dk_Y} \beta_{wk_Y})$ . Here,  $\Theta \in \mathbb{R}_+^{D \times \infty}$  is the factor score matrix,  $\beta \in \mathbb{R}_+^{V \times \infty}$  is the factor loading matrix (or dictionary) and  $r_{k_Y}$  signifies the weight of the  $k_Y^{\text{th}}$  factor. The number of latent factors, possibly smaller than both  $D$  and  $V$ , would be inferred from the data.

In the proposed joint model,  $\mathbf{Y}$  is also determined by the users participating in writing the  $d^{\text{th}}$  document. We assume that the distribution over word counts for a document is a function of *both* its topic distribution *as well as* the characteristics of the users associated with it. In the author-document framework, the authors employ different writing styles and have expertise in different domains. In the user-rating

framework, the entries in  $\mathbf{Y}$  are also believed to be influenced by the interaction network of the users. Such influence of the authors is modeled by the interaction of the authors in the latent communities *via* the latent factors  $\phi \in \mathbb{R}_+^{N \times \infty}$  and  $\psi \in \mathbb{R}_+^{V \times \infty}$ , which encodes the writing style of the authors belonging to different latent communities. Since an infinite number of network communities is maintained, each entry  $y_{dw}$  is assumed to come from an infinite dimensional interaction.  $\rho_{k_B}$  signifies the interaction strength corresponding to the  $k_B^{\text{th}}$  network community. The contributions of the interaction from all the authors participating in a given document are accumulated to produce the total contribution from the networks in generating  $y_{dw}$ . Since  $\mathbf{B}$  and  $\mathbf{Y}$  might have different levels of sparsity and the range of integers in  $\mathbf{Y}$  can be quite large, a parameter  $\epsilon$  is required to balance the contribution of the network communities in dictating the structure of  $\mathbf{Y}$ . A low value of  $\epsilon$  forces disjoint modeling of  $\mathbf{B}$  and  $\mathbf{Y}$ , while a higher value implies joint modeling of  $\mathbf{B}$  and  $\mathbf{Y}$  where information can flow both ways, from network discovery to topic discovery and vice-versa. To complete the generative process, we put Gamma priors over  $c_B$ ,  $c_Y$ ,  $\sigma_n$ ,  $\varsigma_d$  and  $\epsilon$  as:

$$c_B \sim \text{Gam}(g_B, 1/h_B), c_Y \sim \text{Gam}(g_Y, 1/h_Y), \epsilon \sim \text{Gam}(g_0, 1/f_0),$$

$$\sigma_n \sim \text{Gam}(\alpha_B, 1/\varepsilon_B), \varsigma_d \sim \text{Gam}(\alpha_Y, 1/\varepsilon_Y).$$

#### 6.4.1 Inference *via* Gibbs Sampling

Though J-GPPF supports countably infinite number of latent communities for network modeling and infinite number of latent factors for topic modeling, in practice it is impossible to instantiate all of them. We consider a finite approxima-

tion of the infinite model by truncating the number of graph communities and the latent topics to  $K_B$  and  $K_Y$  respectively, by letting  $\rho_{k_B} \sim \text{Gam}(\gamma_B/K_B, 1/c_B)$  and  $r_{k_Y} \sim \text{Gam}(\gamma_Y/K_Y, 1/c_Y)$ . Such approximation approaches the original infinite model as both  $K_B$  and  $K_Y$  approach infinity.

**Sampling of  $(x_{nmk_B})_{k_B=1}^{K_B}$**  : First, the total latent count corresponding to the non-zero entries can be derived as:

$$(x_{nm}|-) \sim b_{nm} \text{Pois}_+ \left( \sum_{k_B=1}^{K_B} \lambda_{nmk_B} \right). \quad (6.24)$$

After which, following Lemma 2.1.4 one can derive:

$$\left( (x_{nmk_B})_{k_B=1}^{K_B} | - \right) \sim \text{Mult} \left( x_{nm}, \left( \frac{\lambda_{nmk_B}}{\sum_{k_B=1}^{K_B} \lambda_{nmk_B}} \right)_{k_B=1}^{K_B} \right). \quad (6.25)$$

**Sampling of  $(y_{dwk})_k$**  : Again, following Lemma 2.1.4, we have:

$$\begin{aligned} & \left( (y_{dwk_Y})_{k_Y=1}^{K_Y}, (y_{dnwk_B})_{k_B=1, n \in \mathcal{Z}_d}^{K_B} | - \right) \sim \\ & \text{Mult} \left( y_{dw}, \frac{\{\zeta_{dwk_Y}\}_{k_Y}, \{\zeta_{dnwk_B}\}_{n \in \mathcal{Z}_d, k_B}}{\sum_{k_Y} \zeta_{dwk_Y} + \sum_{n \in \mathcal{Z}_d} \sum_{k_B} \zeta_{dnwk_B}} \right). \end{aligned} \quad (6.26)$$

**Sampling of  $\phi_{nk_B}$ ,  $\psi_{wk_B}$ ,  $\rho_{k_B}$ ,  $\theta_{dk_Y}$ ,  $\beta_{wk_Y}$ ,  $r_{k_Y}$  and  $\epsilon$**  : Sampling of these parameters follow from Lemma 2.1.1 and are given in Table 6.3. The sampling of parameters  $\phi_{nk_B}$  and  $\rho_{k_B}$  exhibits how information from the count matrix  $\mathbf{Y}$  influences the discovery of the latent network structure. The latent counts from  $\mathbf{Y}$  impact the shape parameters for both the posterior gamma distribution of  $\phi_{nk_B}$  and  $\rho_{k_B}$ , while  $\mathbf{Z}$  influences the corresponding scale parameters.

**Sampling of  $\sigma_n$ ,  $\varsigma_d$ ,  $\epsilon$ ,  $\zeta_w$ ,  $\eta_w$ ,  $c_B$  and  $c_Y$**  : Sampling of these parameters follow from Lemma 2.1.2 and are given in Table 6.3.

$(\phi_{nk_B} -) \sim \text{Gam} \left( a_B + \sum_{m=1}^{(n-1)} x_{mnk_B} + \sum_{m=(n+1)}^N x_{nmk_B} + y_{..k_B}, \left( \sigma_n + \rho_{k_B} \left( \sum_{m=1}^N \phi_{mk_B} + \epsilon \sum_{d,w} Z_{nd} \psi_{wk_B} \right) \right)^{-1} \right),$
$(\psi_{wk_B} -) \sim \text{Gam} \left( \xi_B + y_{..wk_B}, \frac{1}{\zeta_w + \epsilon \rho_{k_B} \sum_{d,n} Z_{nd} \phi_{nk_B}} \right), \quad (\theta_{dk_Y} -) \sim \text{Gam} \left( a_Y + y_{d \cdot k_Y}, \frac{1}{\zeta_d + r_{k_Y} \beta_{\cdot k_Y}} \right),$
$(\rho_{k_B} -) \sim \text{Gam} \left( \frac{\gamma_B}{K_B} + \sum_{\substack{(n,m) \\ n < m}} x_{nmk_B} + y_{...k_B}, \left( c_B + \sum_{\substack{(n,m) \\ n < m}} \phi_{nk_B} \phi_{mk_B} + \epsilon \sum_{n,d,w} Z_{nd} \phi_{nk_B} \psi_{wk_B} \right)^{-1} \right),$
$(r_{k_Y} -) \sim \text{Gam} \left( \frac{\gamma_Y}{K_Y} + y_{..k_Y}, \frac{1}{c_Y + \theta_{\cdot k_Y} \beta_{\cdot k_Y}} \right), \quad (\beta_{wk_Y} -) \sim \text{Gam} \left( \xi_Y + y_{\cdot wk_Y}, \frac{1}{\eta_w + r_{k_Y} \theta_{\cdot k_Y}} \right),$
$(c_B -) \sim \text{Gam} \left( g_B + \gamma_B, \frac{1}{h_B + \sum_{k_B} \rho_{k_B}} \right), \quad (c_Y -) \sim \text{Gam} \left( g_Y + \gamma_Y, \frac{1}{h_Y + \sum_{k_Y} r_{k_Y}} \right),$
$(\zeta_d -) \sim \text{Gam} \left( \alpha_Y + K_Y a_Y, \frac{1}{\varepsilon_Y + \theta_d} \right), \quad (\epsilon -) \sim \text{Gam} \left( f_0 + \sum_{k=1}^{K_B} y_{...k}, \left( g_0 + \sum_{k=1}^{K_B} \rho_{k_B} \sum_{n=1}^N  \mathcal{Z}_n  \phi_{nk_B} \right)^{-1} \right),$
$(\zeta_w -) \sim \text{Gam} \left( a_0 + K_B \xi_B, \frac{1}{b_0 + \psi_w} \right), \quad (\eta_w -) \sim \text{Gam} \left( c_0 + K_Y \xi_Y, \frac{1}{d_0 + \beta_w} \right).$

Table 6.3: Gibbs sampling updates in J-GPPF

**Sampling of  $\gamma_B$**  : Using Lemma 2.1.4, one can show that  $x_{..k_B} \sim \text{Pois}(\rho_{k_B})$ . Integrating  $\rho_{k_B}$  and using Lemma 2.1.1, one can have  $x_{..k_B} \sim \text{NB}(\gamma_B, p_B)$ , where  $p_B = 1/(c_B + 1)$ . Similarly,  $y_{..k_B} \sim \text{Pois}(\rho_{k_B})$  and after integrating  $\rho_{k_B}$  and using Lemma 2.1.1, we have  $y_{..k_B} \sim \text{NB}(\gamma_B, p_B)$ . We now augment  $l_{k_B} \sim \text{CRT}(x_{..k_B} + y_{..k_B}, \gamma_B)$  and then following Lemma 2.2.2 sample:

$$(\gamma_B|-) \sim \text{Gam} \left( e_B + \sum_{k_B} l_{k_B}, (f_B - q_B)^{-1} \right), \quad (6.27)$$

where  $q_B = \sum_{k_B} \log(c_B / (c_B + \sum_n \phi_{nk_B} \phi_{k_B}^{-n})) / K_B$ .

**Sampling of  $\gamma_Y$**  : Using Lemma 2.1.4, one can show that  $y_{..(K_B+k_Y)} \sim \text{Pois}(r_{k_Y})$  and after integrating  $r_{k_Y}$  and using Lemma 2.1.1, we have  $y_{..(K_B+k_Y)} \sim \text{NB}(\gamma_Y, p_Y)$ , where  $p_Y = 1/(c_Y + 1)$ . We now augment  $m_{k_Y} \sim \text{CRT}(y_{..(K_B+k_Y)}, \gamma_Y)$  and then following Lemma 2.2.2 sample:

$$(\gamma_Y | -) \sim \text{Gam} \left( e_Y + \sum_{k_Y} m_{k_Y}, (f_Y - q_Y)^{-1} \right), \quad (6.28)$$

where  $q_Y = \sum_{k_Y} \log(c_Y / (c_Y + \theta_{k_Y})) / K_Y$ .

#### 6.4.2 Gibbs Sampling for J-GPPF with Missing Entries

Parameters whose update get affected in presence of missing entries are  $\rho_{k_B}$ ,  $\phi_{nk_B}$ ,  $\psi_{wk_B}$ ,  $r_{k_Y}$ ,  $\theta_{dk_Y}$ ,  $\beta_{wk_Y}$ . Sampling of these parameters follow from Lemma 2.1.1 and are given in Table 6.4 and 6.5. Here  $\mathcal{M}_B$  and  $\mathcal{M}_Y$  denote the set of missing entries in  $B$  and  $Y$  respectively.

$  \begin{aligned}  (\phi_{nk_B}   -) &\sim \text{Gam} \left( a_B + \sum_{\substack{m=1 \\ (n,m) \notin \mathcal{M}_B}}^{(n-1)} x_{nmk_B} + \sum_{\substack{m=(n+1) \\ (n,m) \notin \mathcal{M}_B}}^N x_{nmk_B} + \sum_{(d,w) \notin \mathcal{M}_Y} y_{dnwk_B}, \left( \sigma_n + \rho_{k_B} \left( \sum_{\substack{m=1 \\ (n,m) \notin \mathcal{M}_B}}^N \phi_{mk_B} + \epsilon \sum_{\substack{d,w \\ (d,w) \notin \mathcal{M}_Y}} Z_{nd} \psi_{wk_B} \right) \right)^{-1} \right), \\  (\psi_{wk_B}   -) &\sim \text{Gam} \left( \xi_B + \sum_{\substack{d \\ (d,w) \notin \mathcal{M}_Y}} y_{dnwk_B}, \left( \eta_w + \epsilon \rho_{k_B} \sum_{\substack{n,d \\ (d,w) \notin \mathcal{M}_Y}} Z_{nd} \phi_{nk_B} \right)^{-1} \right), \\  (\rho_{k_B}   -) &\sim \text{Gam} \left( \frac{\gamma_B}{K_B} + \sum_{\substack{n=1, n < m \\ (n,m) \notin \mathcal{M}_B}}^{(N-1)} x_{nmk_B} + \sum_{\substack{n,d,w \\ (d,w) \notin \mathcal{M}_Y}} y_{dnwk_B}, \left( c_B + \sum_{\substack{n=1, n < m \\ (n,m) \notin \mathcal{M}_B}}^{(N-1)} \phi_{nk_B} \phi_{mk_B} + \epsilon \sum_{\substack{n,d,w \\ (d,w) \notin \mathcal{M}_Y}} Z_{nd} \phi_{nk_B} \psi_{wk_B} \right)^{-1} \right).  \end{aligned}  $
---

Table 6.4: Sampling of  $\phi_{nk_B}$ ,  $\psi_{wk_B}$ ,  $\rho_{k_B}$  in J-GPPF with missing entries

#### 6.4.3 Special cases: Network Only GPPF (N-GPPF) and Corpus Only GPPF (C-GPPF)

A special case of J-GPPF appears when only the binary matrix  $B$  is modeled without the auxiliary matrix  $Y$ . The update equations of variables corresponding to N-GPPF can be obtained with  $Z = 0$ . Another special case of J-GPPF appears when only the count matrix  $Y$  is modeled without using the contribution from the

$$\begin{aligned}
(\theta_{dk_Y} | -) &\sim \text{Gam} \left( a_Y + \sum_{\substack{w \\ (d,w) \notin \mathcal{M}_Y}} y_{dwk_Y}, \left( \varsigma_d + r_{k_Y} \sum_{\substack{w \\ (d,w) \notin \mathcal{M}_Y}} \beta_{wk_Y} \right)^{-1} \right), \\
(\beta_{wk_Y} | -) &\sim \text{Gam} \left( \xi_Y + \sum_{\substack{d \\ (d,w) \notin \mathcal{M}_Y}} y_{dwk_Y}, \left( \eta_w + r_{k_Y} \sum_{\substack{d,n \\ (d,w) \notin \mathcal{M}_Y}} Z_{dn} \phi_{nk_Y} \right)^{-1} \right), \\
(r_{k_Y} | -) &\sim \text{Gam} \left( \frac{\gamma_Y}{K_Y} + \sum_{\substack{d,w \\ (d,w) \notin \mathcal{M}_Y}} y_{dwk_Y}, \left( c_Y + \sum_{\substack{d,w \\ (d,w) \notin \mathcal{M}_Y}} \theta_{dk_Y} \beta_{wk_Y} \right)^{-1} \right).
\end{aligned}$$

Table 6.5: Sampling of  $\theta_{dk_Y}$ ,  $\beta_{wk_Y}$ ,  $r_{k_Y}$  in J-GPPF with missing entries

network matrix  $\mathbf{B}$ .

#### 6.4.4 Computation Complexity

The Gibbs sampling updates of J-GPPF can be calculated in  $O(K_B S_B + (K_B + K_Y) S_Y + N K_B + D K_Y + V(K_B + K_Y))$  time, where  $S_B$  is the number of non-zero entries in  $\mathbf{B}$  and  $S_Y$  is the number of non-zero entries in  $\mathbf{Y}$ . It is obvious that for large matrices the computation is primarily of the order of  $K_B S_B + (K_B + K_Y) S_Y$ . Such complexity is a huge saving when compared to other methods like MMSB [Airoldi et al., 2008], that only models  $\mathbf{B}$  and incurs computation cost of  $O(N^2 K_B)$ ; and standard matrix factorization approaches [Salakhutdinov and Mnih, 2007] that work with the matrix  $\mathbf{Y}$  and incur  $O(DV K_Y)$  computation cost. In Fig. 6.1(a), we show the computation time for generating one million samples from Gamma, Dirichlet (of dimension 50), multinomial (of dimension 50) and truncated Poisson distributions using the samplers available from GNU

Scientific Library (GSL) on an Intel 2127U machine with 2 GB of RAM and 1.90 GHz of processor base frequency. To highlight the average complexity of sampling from Dirichlet and multinomial distributions, we further display another plot where the computation time is divided by 50 for these samplers only. One can see that to draw one million samples, our implementation of the sampler for truncated Poisson distribution takes the longest, though the difference from the Gamma sampler in GSL is not that significant.

## 6.5 Experimental Results

### 6.5.1 Experiments with Synthetic Data

We generate a synthetic network of size  $60 \times 60$  ( $\mathbf{B}$ ) and a count data matrix of size  $60 \times 45$  ( $\mathbf{Y}$ ). Each user in the network writes exactly one document and a user and the corresponding document are indexed by the same row-index in  $\mathbf{B}$  and  $\mathbf{Y}$  respectively. To evaluate the quality of reconstruction in presence of side-information and less of network structure, we hold-out 50% of links and equal number of non-links from  $\mathbf{B}$ . This is shown in Fig. 6.1(b) where the links are presented by brown, the non-links by green and the held-out data by deep blue. Clearly, the network consists of two groups.  $\mathbf{Y} \in \{0, 5\}^{60 \times 45}$ , shown in Fig 6.1(c), is also assumed to consist of the same structure as  $\mathbf{B}$  where the zeros are presented by deep blue and the non-zeros are represented by brown. The performance of N-GPPF is displayed in Fig. 6.2(a). Evidently, there is not much structure visible in the discovered partition of  $\mathbf{B}$  from N-GPPF and that is reflected in the poor value of AUC in Fig. 6.3(a). The parameter  $\epsilon$ , when fixed at a given value, plays an

important role in determining the quality of reconstruction for J-GPPF. As  $\epsilon \rightarrow 0$ , J-GPPF approaches the performance of N-GPPF on  $\mathbf{B}$  and we observe as poor a quality of reconstruction as in Fig. 6.2(a). When  $\epsilon$  is increased and set to 1.0, J-GPPF departs from N-GPPF and performs much better in terms of both structure recovery and prediction on held-out data as shown in Fig. 6.2(e) and Fig. 6.3(b). With  $\epsilon = 10.0$ , perfect reconstruction and prediction are recorded as shown in Fig. 6.2(i) and Fig. 6.3(c) respectively. In this synthetic example,  $\mathbf{Y}$  is purposefully designed to reinforce the structure of  $\mathbf{B}$  when most of its links and non-links are held-out. However, in real applications,  $\mathbf{Y}$  might not contain as much of information and the Gibbs sampler needs to find a suitable value of  $\epsilon$  that can carefully glean information from  $\mathbf{Y}$ .

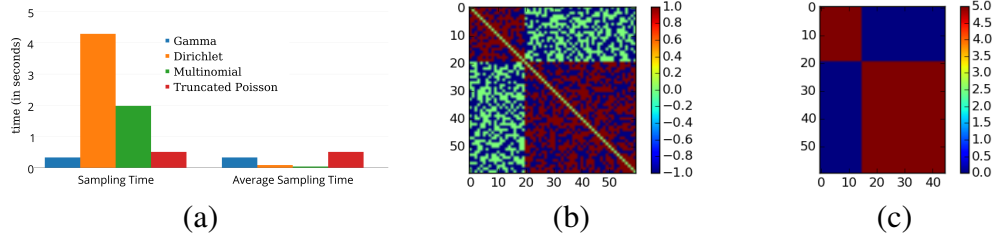


Figure 6.1: (a) Time to generate a million of samples, (b)  $\mathbf{B}$  with held-out data, (c)  $\mathbf{Y}$

There are few more interesting observations from the experiment with synthetic data that characterize the behavior of the model and match our intuitions. In our experiment with synthetic data  $K_{\mathbf{B}} = K_{\mathbf{Y}} = 20$  is used. Fig. 6.2(b) demonstrates the assignment of the users in the network communities and Fig. 6.2(d) illustrates the assignment of the documents to the combined space of network communities and the topics (with the network communities appearing before the topics

in the plot). For  $\epsilon = 0.001$ , we observe disjoint modeling of  $\mathbf{B}$  and  $\mathbf{Y}$ , with two latent factors modeling  $\mathbf{Y}$  and multiple latent factors modeling  $\mathbf{B}$  without any clear assignment. As we increase  $\epsilon$ , we start observing joint modeling of  $\mathbf{B}$  and  $\mathbf{Y}$ . For  $\epsilon = 1.0$ , as Fig. 6.2(h) reveals, two of the network latent factors and two of the factors for count data together model  $\mathbf{Y}$ , the contribution from the network factors being expectedly small. Fig. 6.2(f) shows how two of the exact same latent factors model  $\mathbf{B}$  as well. Fig. 6.2(j) and Fig. 6.2(l) show how two of the latent factors corresponding to  $\mathbf{B}$  dictate the modeling of both  $\mathbf{B}$  and  $\mathbf{Y}$  when  $\epsilon = 10.0$ . This implies that the discovery of latent groups in  $\mathbf{B}$  is dictated mostly by the information contained in  $\mathbf{Y}$ . In all these cases, however, we observe perfect reconstruction of  $\mathbf{Y}$  as shown in Fig. 6.2(c), Fig. 6.2(g) and Fig. 6.2(k).

## 6.5.2 Experiments with Real World Data

To evaluate the performance of J-GPPF, we consider N-GPPF, the infinite relational model (IRM) of [Kemp et al., 2006] and the Mixed Membership Stochastic Block Model (MMSB) [Airoldi et al., 2008] as the baseline algorithms.

### 6.5.2.1 NIPS Authorship Network

This dataset contains a list of all papers and authors from NIPS 1988 to 2003. We took the 234 authors who had published with the most other people and looked at their co-authorship information. After standard pre-processing and removing words that appear less than 50 times in the over-all corpus corresponding

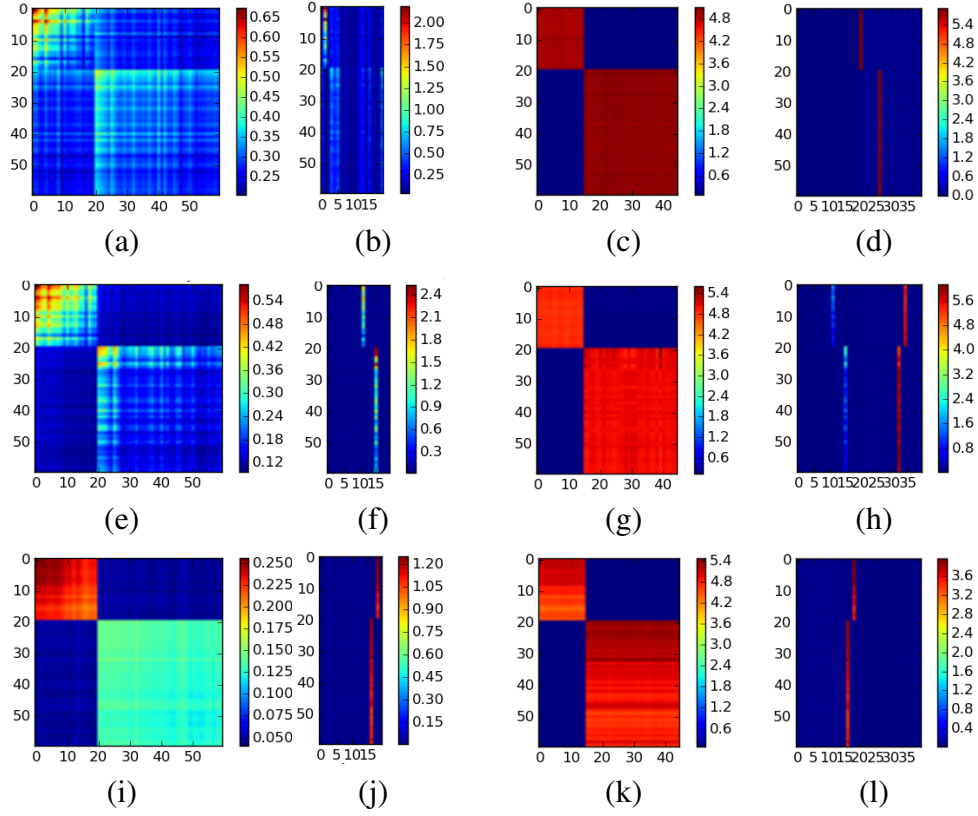


Figure 6.2: Performance of J-GPPF:  $\epsilon = 10^{-3}$  (top row),  $\epsilon = 1$  (middle row),  $\epsilon = 10$  (bottom row)

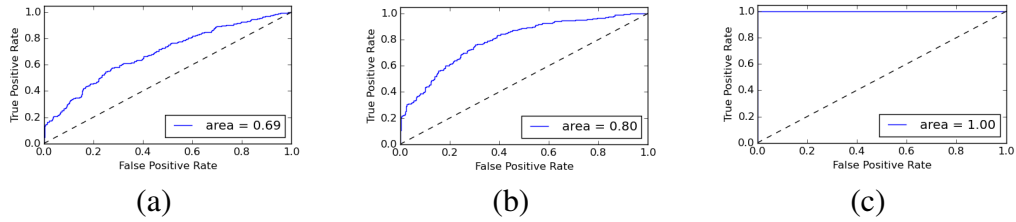


Figure 6.3: (a) AUC with  $\epsilon = 0.001$ , (b) AUC with  $\epsilon = 1.0$ , (c) AUC with  $\epsilon = 10.0$

to these users, the number of users in the graph who write at least one document, is 225 and the total number of unique words is 1354. The total number of documents is 1165.

#### **6.5.2.2 GoodReads Data**

Using the Goodreads API, we collect a base set of users with recent activity on the website. For each user in the base set, the user’s friends as well as friends of friends on the site are collected (two hops in the graph). This process is repeated over a 24-hour time period, with a new base set constructed each time (*i.e.* friends are not polled recursively). By running for a full day, multiple time zones are covered and the reviews are collected for all identified users, with a maximum of 200 reviews per user. Each review consists of a book ID and a rating from 0 to 5. Similar dataset has also been used in [Chaney et al., 2013]. After standard pre-processing and removing words that appear less than 10 times in the over-all corpus, the number of users in the graph is 84 and the total number of unique words is 189.

#### **6.5.2.3 Twitter Data**

The Twitter data that we use in the paper is a subset of the geo-tagged tweets collected by the authors in [Roller et al., 2012]. A subset of users located in the San Francisco city limits are extracted for our analysis. For each user in the San Francisco subset, we collect all of the accounts followed by that user with the Twitter API, and discard accounts that are not in San Francisco. This process creates an

undirected graph of users within the San Francisco subset. For side information, word counts are collected from the aggregated tweets for each user in the graph. After standard pre-processing and removing words that appear less than 25 times in the over-all corpus, the number of users in the graph is 670 and the total number of unique words is 538. All the tweets corresponding to one user are collapsed into a single document and so each user is associated with exactly one document in this dataset.

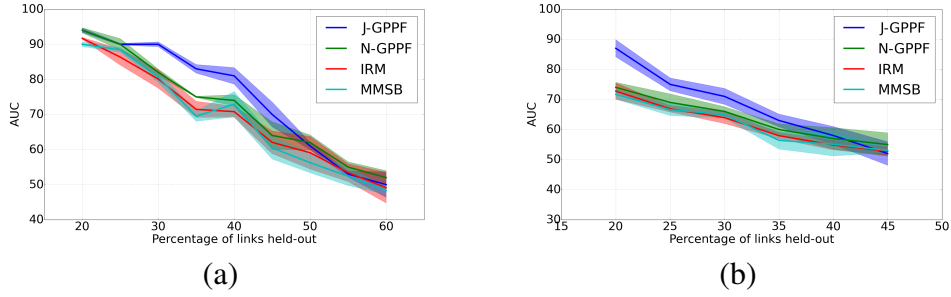


Figure 6.4: (a) NIPS Data, (b) GoodReads Data

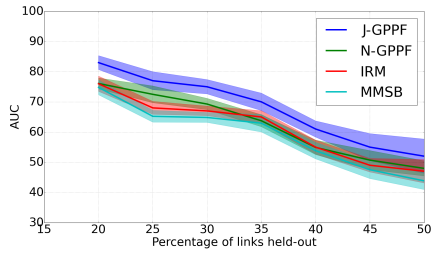


Figure 6.5: Twitter Data

#### 6.5.2.4 Experimental Setup and Results

In all the experiments, we initialize  $\epsilon$  to 2 and let the sampler decide what value works best for joint modeling. We use  $K_B = K_Y = 50$  and initialize all the hyper-parameters to 1. In the first set of experiments, for each dataset, we hold out data from **B** only and ran 20 different experiments and display the mean AUC and one standard error. In this setup, we consider N-GPPF, the infinite relational model (IRM) of [Kemp et al., 2006] and the Mixed Membership Stochastic Block Model (MMSB) [Airoldi et al., 2008] as the baseline algorithms. Fig. 6.4 and 6.5 demonstrate the performances of the models in predicting the held-out data. J-GPPF clearly has advantage over other network-only models when the network is sparse enough and the auxiliary information is sufficiently strong. However, all methods fail when the sparsity increases beyond a certain point. The performance of J-GPPF also drops below the performances of network-only models in highly sparse networks, as the sampler faces additional difficulty in extracting information from both **B** and **Y**.

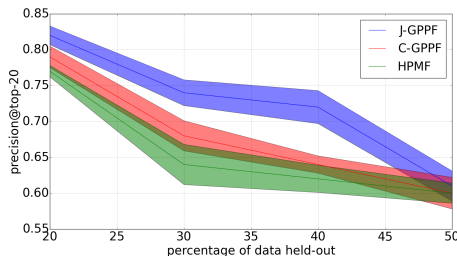


Figure 6.6: MAP NIPS

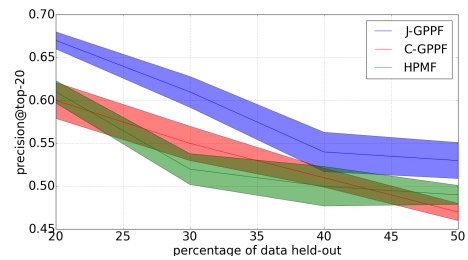


Figure 6.7: MAP GoodReads

In the second set of experiments, we hold out data from **Y** only and run 20 different experiments and display the precision@top-20 for J-GPPF. This evaluation

is structured along the lines of the work in [Gopalan et al., 2014a]. We calculate the intersection of the top 20 predicted set of words (arranged in the decreasing order of counts) and the top 20 words in a document and divide the number by 20 to get the precision for each document. We then calculate mean average precision (MAP) by taking the average of the precision over all the documents. C-GPPF and the hierarchical Poisson matrix factorization (HPMF) [Gopalan et al., 2013] are considered as the baselines, both of which model only  $\mathbf{Y}$ . Fig. 6.6 and 6.7 show that  $\mathbf{B}$  helps in boosting the predictive performance in J-GPPF over a wide range of fractions of the data that is held out from  $\mathbf{Y}$ .

## 6.6 Conclusion

We propose J-GPPF that jointly factorizes the network adjacency matrix and the associated side information that can be represented as a count matrix. The model has the advantage of representing true sparsity in adjacency matrix, in latent group membership, and in the side information. We derived an efficient MCMC inference method, and compared our approach to several popular network algorithms that model the network adjacency matrix. Experimental results confirm the efficiency of the proposed approach in utilizing side information to improve the performance of network models. In the next two chapters, we explore the implementations and applications of sequential knowledge transfer only.

## Chapter 7

### Nonparametric Dynamic Models

In the earlier chapters, we have seen how simultaneous knowledge transfer can be used to solve problems related to object recognition from images, text classification, and joint network and topic modeling for better network imputation. This chapter demonstrates how sequential knowledge transfer can be used effectively to model data that changes with time. In Section 7.1, we develop models specific to the analysis of count-valued and binary vectors that evolve with time. We invent quite a few novel tricks to solve a difficult inference problem associated with the proposed model. Interestingly, the same set of inference tricks can also be applied to other models, described in Section 7.2 and 7.3, for the analysis of network data and dyadic data respectively that change with time.

#### 7.1 Nonparametric Bayesian Dynamic Count and Binary Matrix Factorization

There has been growing interest in analyzing dynamic count and binary matrices, whose columns are data vectors that are sequentially collected over time. These data appear in many real world applications, such as text analysis, social network modeling, audio and language processing, and recommendation systems. The count data are discrete and nonnegative, have limited ranges, and often present

overdispersion; the binary data only have two possible values: 0 and 1; and both kinds of data commonly appear in big matrices that are extremely sparse. While the classical matrix factorization method using the Frobenius norm is effective for factorizing real matrices [Aharon et al., 2006; Candès et al., 2011; Gunasekar et al., 2013; Koren et al., 2009; Lawrence and Urtasun, 2009; Salakhutdinov and Mnih, 2008; Srebro et al., 2003], its inherent Gaussian assumption is often overly restrictive for modeling count and binary matrices. To take advantage of existing well-developed techniques for Gaussian data, one usually consider connecting a count observation to a latent Gaussian random variable using the lognormal-Poisson link, and connecting a binary observation using the probit or logit links. These generalized linear model [McCullagh and Nelder, 1989] based approaches, however, might involve heavy computation. For example, for an extremely sparse but huge-size count/binary matrix, linking each zero observation to a latent Gaussian random variable would impose a substantial computational and memory burden. In addition, there is often lack of intuitive interpretation of the inferred factorization in the latent Gaussian space.

Despite the disadvantages in both computation and interpretation, latent Gaussian based approaches are commonly used to analyze count and binary data. This is particularly true for dynamic modeling, since inference techniques for linear dynamic systems such as the Kalman filter are well developed, which can be readily applied once the dynamic count/binary data are transformed into the latent Gaussian space. For example, to analyze the temporal evolution of topics in a corpus, the dynamic topic model draws the topic proportion at each time stamp from

a logistic normal distribution, whose parameters are chained in a state space model that evolves with Gaussian noise [Blei and Lafferty, 2006]. Although the dynamic topic model is a discrete latent variable model, to model the topic proportion that explains the number of words assigned to a topic in a document, which is a count, it chooses to use the logistic normal link and imposes the temporal smoothness of model parameters in the latent Gaussian space.

Rather than modeling the dynamic evolving of count and binary data in the latent Gaussian space using a linear dynamic system, in this section, we consider a fundamentally different approach: we directly chain the positive Poisson rates of the count or binary data in a state space model that evolves with gamma noise. More specifically, we build a gamma Markov chain that sends  $\theta_{t-1}$ , a latent gamma random variable at time  $t - 1$ , as the shape parameter of the latent gamma random variable at time  $t$  as  $\theta_t | \theta_{t-1} \sim \text{Gamma}(\theta_{t-1}, 1/c)$ ; at each time point, we use  $\theta_t$  as the Poisson rate for a count as  $n_t \sim \text{Pois}(\theta_t)$ ; and the counts  $\{n_t\}_t$  are conditionally independent given  $\{\theta_t\}_t$ . If the observation is binary, then we assume the Bernoulli random variable is generated by thresholding a latent count as  $b_t = \mathbf{1}(n_t \geq 1)$ , which means  $b_t = 1$  if  $n_t \geq 1$  and  $b_t = 0$  if  $n_t = 0$ . We call this novel count-binary link function as the Poisson-Bernoulli link, under which the conditional posterior of the latent count follows a truncated Poisson distribution.

To apply the gamma Markov chain to dynamic count and binary matrix factorization, we extend it to a multivariate setting, which is integrated into a discrete latent variable model called Poisson factor analysis [Zhou et al., 2012]. Specifically, we factorize the observed dynamic count (binary) matrix under the Poisson

(Poisson-Bernoulli) likelihood, and chain the latent factor scores across time, where a gamma distributed factor score is linked via a Poisson distribution to a *latent* count that counts how many times the corresponding factor is used by the corresponding observation. To avoid tuning the latent dimension of factorization, we also employ a gamma process to automatically infer the number of factors, which can be potentially infinite as the number of observation grows. The key challenge for this unconventional Markov chain is to infer the gamma shape parameters, for which we discover a simple and effective solution.

This section makes the following contributions: 1) We construct a novel gamma Markov chain to model dynamic count and binary data. 2) We provide closed-form update equations to infer the parameters of the gamma Markov chain, using novel data augmentation and marginalizing techniques. 3) We integrate the gamma Markov chain into Poisson factor analysis to analyze dynamic count matrices. 4) We factorize a dynamic binary matrix under the proposed Poisson-Bernoulli likelihood, with extremely efficient computation for sparse observations. 5) We apply the developed techniques to real world data analysis, with state-of-the-art results.

### 7.1.1 Gamma Process Dynamic Poisson Factor Analysis

In this paper, we first consider a dynamic count matrix  $\mathbf{N} \in \mathbb{Z}^{V \times T}$ , whose  $T$  columns are  $V$ -dimensional count vectors sequentially observed. We consider a modified version of PFA as

$$\mathbf{N} \sim \text{Pois}(\Phi \Lambda \Theta),$$

where  $\Lambda = \text{diag}(\boldsymbol{\lambda})$  and  $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_\infty)$  is a vector representing the strengths of the countably infinite latent factors. Note that under the regular setting where different columns of  $\Theta$  are independently modeled, this parameterization is not a strict generalization of the beta-NB process PFA described in [Zhou and Carin, 2012; Zhou et al., 2012]. This is because if one follows the beta-NB process to let  $\theta_{tk} \sim \text{Gamma}(r_t, p_k/(1 - p_k))$ , and  $\lambda_k$  is assumed to be independent from  $\theta_{tk}$ , then  $\tilde{\theta}_{tk} := \lambda_k \theta_{tk} \sim \text{Gamma}(r_t, q_k/(1 - q_k))$ , where  $q_k = \frac{\lambda_k p_k}{1 + (\lambda_k - 1)p_k}$ . Thus  $\Lambda$  are redundant and can be absorbed into  $\Theta$  as  $\mathbf{N} \sim \text{Pois}(\Phi \tilde{\Theta})$ . In this paper, with the column index  $t$  corresponding to time, the modified representation would become necessary to impose temporal smoothness for consecutive columns, which are no longer assumed to be independent, as discussed below.

We consider a gamma process  $G \sim \text{GaP}(c, G_0)$ , a draw from which is expressed as  $G = \sum_{k=1}^{\infty} \lambda_k \delta_{\phi_k}$ , where  $\phi_k$  is an atom drawn from a  $V$ -dimensional base distribution as  $\phi_k \sim \text{Dir}(\eta, \dots, \eta)$  and  $\lambda_k = G(\phi_k)$  is the associated weight. We mark each atom  $\phi_k$  with a constant  $\theta_{(-1)k} = 0.01$ , and then generate a gamma Markov chain by letting:

$$\theta_{tk} | \theta_{(t-1)k} \sim \text{Gamma}(\theta_{(t-1)k}, 1/c_t), \quad t = 0, \dots, T.$$

We then integrate the weights of the gamma process  $\{\lambda_k\}$  and the infinite-dimensional gamma Markov chain into a gamma process dynamic Poisson factor analysis (dPFA)

model as:

$$\begin{aligned}
n_{vt} &= \sum_{k=1}^{\infty} n_{vtk}, \quad n_{vtk} \sim \text{Pois}(\lambda_k \phi_{vk} \theta_{tk}), \\
\phi_k &\sim \text{Dir}(\eta_1, \dots, \eta_V), \quad \theta_{tk} \sim \text{Gamma}(\theta_{(t-1)k}, 1/c_t), \\
G &\sim \text{GaP}(c, G_0), \quad c_t \sim \text{Gamma}(e_0, 1/f_0).
\end{aligned} \tag{7.1}$$

We further impose the gamma prior  $\text{Gamma}(e_0, 1/f_0)$  on both the concentration parameter  $c$  and the mass parameter  $\gamma_0 = G_0(\Omega)$ . Below we discuss how to infer the model parameters, in particular, how to solve the challenge of inferring each  $\theta_{tk}$ , which is the gamma shape parameter for  $\theta_{(t+1)k}$ .

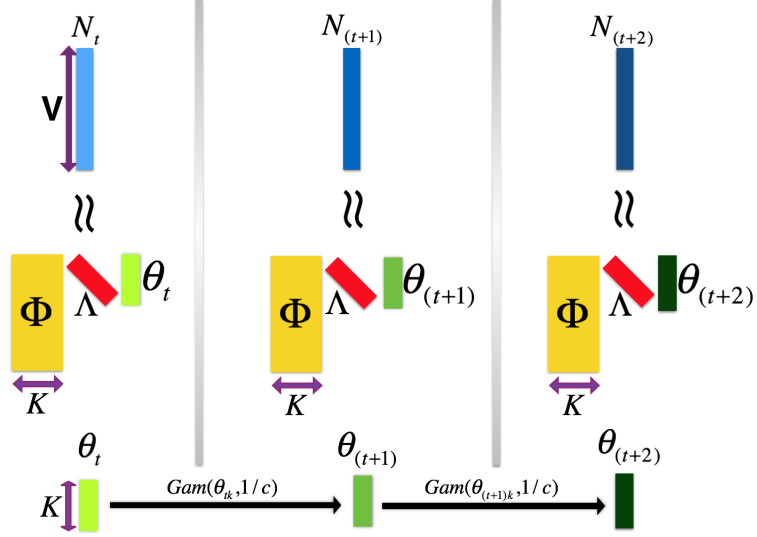


Figure 7.1: Illustration of GP-DPFA

#### 7.1.1.1 Inference via Gibbs Sampling

The proposed gamma process dPFA supports an countably infinite number of latent factors, but in practice it is impossible to instantiate all of them.

One common approach for exact inference for a nonparametric Bayesian model is to marginalize out the underlying stochastic process [Blackwell and MacQueen, 1973], and another common approach is to use slice sampling to adaptive truncate the number of atoms [Walker, 2007]. For simplicity, in this paper, we consider a finite approximation of the infinite model by truncating the number of factors to  $K$ , by letting

$$\lambda_k \sim \text{Gamma}(\gamma_0/K, 1/c), \quad (7.2)$$

which approaches the original infinite model as  $K \rightarrow \infty$ . Despite the significant challenge presented in inferring the gamma shape parameters, exploiting the data augmentation and marginalization techniques unique to the negative binomial distribution [Zhou and Carin, 2012, 2015], we derive closed-form Gibbs sampling update equations for all model parameters, as described below.

Exploiting the property that  $\sum_{v=1}^V \phi_{vk} = 1 \forall k$ , the likelihood of the latent counts, conditioned on  $(\Phi, \Theta, \lambda)$  can be expressed as

$$\begin{aligned} P(\{(n_{vjk})_{k=1}^K\} | \Phi, \Theta, \lambda) &= \prod_{j=1}^J \prod_{v=1}^V \prod_{k=1}^K \frac{(\lambda_k \phi_{vk} \theta_{jk})^{n_{vjk}}}{n_{vjk}!} e^{-\lambda_k \phi_{vk} \theta_{jk}} \\ &= \left( \prod_{j=1}^J \prod_{v=1}^V \prod_{k=1}^K \frac{1}{n_{vjk}!} \right) \prod_{k=1}^K \left\{ \left( \prod_{v=1}^V \phi_{vk}^{n_{v \cdot k}} \right) \left( \prod_{j=1}^J \theta_{jk}^{n_{\cdot jk}} \lambda_k^{n_{\cdot tk}} e^{-\lambda_k \theta_{jk}} \right) \right\}. \end{aligned} \quad (7.3)$$

**Sample  $n_{vjk}$ :** Using the relationship between the Poisson and multinomial distribution, as in Lemma 4.1 of [Zhou et al., 2012], given the observed counts and latent parameters, we have

$$(n_{vt1}, \dots, n_{vtK} | -) \sim \text{Mult} \left( n_{vt}, \frac{\lambda_1 \phi_{v1} \theta_{t1}}{\sum_k \lambda_k \phi_{vk} \theta_{tk}}, \dots, \frac{\lambda_K \phi_{vK} \theta_{tK}}{\sum_k \lambda_k \phi_{vk} \theta_{tk}} \right) \quad (7.4)$$

**Sample  $\phi_k$ :** Since in the likelihood we have  $(n_{1\cdot k}, \dots, n_{V\cdot k} | n_{\cdot\cdot k}, \phi_k) \sim \text{Mult}(n_{\cdot\cdot k}, \phi_k)$ , using the Dirichlet-multinomial conjugacy, the conditional posterior of  $\phi_k$  can be expressed as

$$(\phi_k | -) \sim \text{Dir}(\eta + \mathbf{n}_{1\cdot k}, \dots, \eta + \mathbf{n}_{V\cdot k}). \quad (7.5)$$

**Sample  $\lambda_k$ :** Since in the likelihood  $\mathbf{n}_{\cdot tk} \sim \text{Pois}(\lambda_k \theta_{tk})$ , using the gamma-Poisson conjugacy, the conditional posterior of  $\lambda_k$  can be expressed as

$$(\lambda_k | -) \sim \text{Gamma}\left(\mathbf{n}_{\cdot\cdot k} + \frac{\gamma_0}{K}, \frac{1}{c + \sum_t \theta_{tk}}\right). \quad (7.6)$$

**Sample  $\theta_{tk}$ :** Due to the Markovian construction, it is necessary to consider both backward and forward information for the inference of  $\theta_{tk}$ . Starting from the last time point  $t = T$ , one has  $n_{Tk} \sim \text{Pois}(\lambda_k \theta_{Tk})$ ,  $\theta_{Tk} \sim \text{Gamma}(\theta_{(T-1)k}, 1/c_T)$ . The marginalization of  $\theta_{Tk}$  leads to  $n_{Tk} \sim \text{NB}(\theta_{(T-1)k}, p_{Tk})$ , where  $p_{Tk} := \frac{\lambda_k}{c_T + \lambda_k}$  and  $p_{(T+1)k} := 0$ . The NB distribution can further be augmented with an auxiliary count variable as  $l_{Tk} \sim \text{CRT}(n_{Tk}, \theta_{(T-1)k})$ ,  $n_{Tk} \sim \text{NB}(\theta_{(T-1)k}, p_{Tk})$ . Following Lemma 2.2.2, the joint distribution of  $l_{Tk}$  and  $n_{Tk}$  is a Poisson-logarithmic distribution that can be represented as  $n_{Tk} \sim \sum_{t=1}^{l_{Tk}} \text{Log}(p_{Tk})$ ,  $l_{Tk} \sim \text{Pois}(-\theta_{(T-1)k} \ln(1 - p_{Tk}))$ . Thus  $l_{Tk}$  can be considered as the backward information from  $T$  to  $(T-1)$ . Given  $l_{(t+1)k}$ , the backward information from  $(t+1)$  to  $t$ , we then have

$$l_{(t+1)k} \sim \text{Pois}(-\theta_{tk} \ln(1 - p_{(t+1)k})), \quad n_{tk} \sim \text{Pois}(r_k \theta_{tk}).$$

The marginalization of  $\theta_{tk}$  leads to

$$(n_{tk} + l_{(t+1)k}) \sim \text{NB}(\theta_{(t-1)k}, p_{tk}), \quad p_{tk} := \frac{\lambda_k - \ln(1 - p_{(t+1)k})}{c_t + \lambda_k - \ln(1 - p_{(t+1)k})}.$$

Thus  $l_{tk}$ , the backward information from  $t$  to  $(t-1)$ , can be calculated as:

$$l_{tk} \sim \text{CRT}(n_{tk} + l_{(t+1)k}, \theta_{(t-1)k}). \quad (7.7)$$

With these information calculated backwards, for  $t = 0, \dots, T$ , one can sample  $\theta_{tk}$  *forwards* as:

$$(\theta_{tk}|-) \sim \text{Gamma}(\theta_{(t-1)k} + n_{tk} + l_{(t+1)k}, (1 - p_{tk})/c_t). \quad (7.8)$$

where  $n_{0k} := 0$  and  $\theta_{(-1)k} := 0.01$ .

**Sample**  $c_t$ ,  $c$  **and**  $\gamma_0$ . For  $t = 0, \dots, T$ , we sample  $c_t$  as:

$$(c_t|-) \sim \text{Gamma}\left(e_0 + \sum_k \theta_{(t-1)k}, \frac{1}{f_0 + \sum_k \theta_{tk}}\right). \quad (7.9)$$

We sample  $c$  as:

$$(c|-) \sim \text{Gamma}\left(e_0 + \gamma_0, \frac{1}{f_0 + \sum_k \lambda_k}\right). \quad (7.10)$$

Since  $n_{..k} \sim \text{NB}\left(\frac{\gamma_0}{K}, \frac{\sum_t \theta_{tk}}{c + \sum_t \theta_{tk}}\right)$ , we can sample  $\gamma_0$  using

$$(\ell_k|-) \sim \text{CRT}\left(n_{..k}, \frac{\gamma_0}{K}\right), \quad (7.11)$$

$$(\gamma_0|-) \sim \text{Gamma}\left(e_0 + \sum_k \ell_k, \frac{1}{f_0 - \sum_k \ln\left(1 - \frac{\sum_t \theta_{tk}}{c + \sum_t \theta_{tk}}\right)}\right). \quad (7.12)$$

### 7.1.1.2 Modeling Binary Observations

To model binary data, a novel data augmentation technique is introduced here. Rather than following the usual approach to link a binary observation to a

latent Gaussian random variable using the probit or logit links, a binary observation is linked to a latent count as

$$b = \mathbf{1}(n \geq 1), \quad n \sim \text{Pois}(\lambda),$$

which is named in this paper as the Poisson-Bernoulli (PoBe) link. We call the distribution of  $b$  given  $\lambda$  as the Poisson Bernoulli distribution, with PMF  $f_B(b|\lambda) = e^{-\lambda(1-b)}(1 - e^{-\lambda})^b$ ,  $b \in \{0, 1\}$ . The conditional posterior of the latent count  $n$  is simply  $(n|b, \lambda) \sim b \cdot \text{Pois}_+(\lambda)$ , where  $k \sim \text{Pois}_+(\lambda)$  is the truncated Poisson distribution with PMF  $f_K(k) = \frac{1}{1-e^{-\lambda}} \frac{\lambda^k e^{-\lambda}}{k!}$ ,  $k = 1, 2, \dots$ . Thus if  $b = 0$ , then  $n = 0$  almost surely (a.s.), and if  $b = 1$ , then  $n$  is drawn from a truncated Poisson distribution. To simulate the truncated Poisson random variable  $x \sim \text{Pois}_+(\lambda)$ , we use rejection sampling: if  $\lambda \geq 1$ , we draw  $x \sim \text{Pois}(\lambda)$  until  $x \geq 1$ ; if  $\lambda < 1$ , we draw  $y \sim \text{Pois}(\lambda)$  and  $u \sim \text{Unif}(0, 1)$ , and let  $x = y + 1$  if  $u < 1/(y + 1)$ . The acceptance rate is  $1 - e^{-\lambda}$  if  $\lambda \geq 1$  and  $\lambda^{-1}(1 - e^{-\lambda})$  if  $\lambda < 1$ . Thus the minimum acceptance rate is 63.2% (when  $\lambda = 1$ ).

With the PoBe link to connect an observed dynamic binary matrix to a dynamic latent count matrix, we are ready to apply the gamma process dPFA to dynamic binary matrix factorization. The only additional step is to add the sampling of the latent counts as

$$(n_{it}|b_{it}, \Phi, \Theta) \sim b_{it} \text{Pois}_+(\sum_k \phi_{kt} \theta_{kt}). \quad (7.13)$$

A clear advantage of the PoBe link over both the probit and logit links is that it is extremely efficient in handling sparse binary matrices, since if an element of the

binary matrix is zero, the corresponding latent count is zero a.s., for which there is no need to perform sampling.

### 7.1.2 Experimental Results

In this section, experimental results are reported on a variety of synthetic and real world datasets and GP-DPFA is compared with relevant baselines. The synthetic and coal-mine disaster datasets provide a test-bed of GPAR, a special case of GP-DPFA.

#### 7.1.2.1 Results with Synthetic Datasets

As in [Adams et al., 2009], three one-dimensional data sets are used with the following rate functions:

- A sum of an exponential and a Gaussian bump (SDS1):  $\theta(t) = 2\exp(-t/15) + \exp(-((t - 25)/10)^2)$  on the interval  $t = [0 : 1 : 50]$ .
- A sinusoid with increasing frequency (SDS2):  $\theta(t) = 5\sin(t^2) + 6$  on  $t = [0 : 0.2 : 5]$ .
- $\theta$  is the piecewise linear function on the interval  $t = [0 : 1 : 100]$  and is given by:  $\theta(t) = (2 + t/30)$  if  $0 \leq t \leq 30$ ,  $\theta(t) = (3 - (t - 30)/10)$  if  $31 \leq t \leq 50$ ,  $\theta(t) = (1 + 1.5 * (t - 50)/25)$  if  $51 \leq t \leq 75$  and  $\theta(t) = (2.5 + 0.5 * (t - 75)/25)$  if  $76 \leq t \leq 100$  (SDS3).

GPAR is compared with the sigmoidal Gaussian Cox process (SGCP) [Adams et al., 2009], log-Gaussian Cox process (LGCP) [Møller et al.], and the classical kernel

smoothing (KS) [Diggle, 1985]. These methods are considered as state-of-the-art in various scenarios involving modeling of count time series. Edge-corrected kernel smoothing is performed using a quartic kernel and a mean-square minimization technique is used for bandwidth selection. The squared-exponential kernel is used for both the SGCP and LGCP. Since the LGCP works with discretization, experiments are performed with 10, 25 and 100 bins. The rate functions provide ground truth and cumulative mean squared error (MSE) between the ground truth and the estimated rate are measured for all the models. Additionally, for each of the above series, the last five observations are withheld and MSE is measured between the true rate and the estimated rate over these withheld observations. The results are displayed in Table 2. “PMSE” stands for MSE in prediction for the last five years of data. The best results are presented in bold.

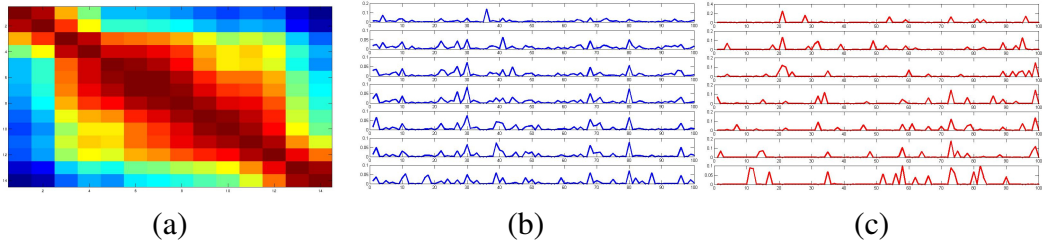


Figure 7.2: (a) correlation across topics over time, (b) latent factors dominant over time for GP-DPFA, (c) latent factors dominant over time for the baseline.

### 7.1.2.2 Results with Real World Datasets

**Coalmine Disaster Dataset:** The British coal mine disaster dataset [Adams et al., 2009] records the number of coalmine accidents arranged according to year from 1851 to 1962. To illustrate the robustness of the inference framework, the underlying rate is initialized to a large value 1000. Fig. 1(a) shows the estimated rate and the sampled value of the underlying rate after the 1<sup>st</sup> iteration. Fig. 1(b) shows the estimation of the underlying rate along with a “baseline” GP-DPFA model that does not use any temporal correlation. A box plot of the sampled rate is presented in Fig. 1(c) showing that the algorithm converges to a good estimate even with such a poor initialization. For these plots, 3000 iterations are used and the last 1000 samples are collected.

**State-of-the-Union Dataset (STU):** The STU dataset contains the transcripts of 225 US State of the Union addresses, from 1790 to 2014. Each transcript corresponding to each year is considered as one document. After removing stop words and terms that occur fewer than 7 times in one document or less than 20 times overall, there are 2375 unique words.

**Conference Abstract Dataset (Conf.):** The Conf. dataset consists of the abstracts of the papers appearing on DBLP for the second author of this paper from 2000 to 2013. For every year, a count vector of dimension  $V = 1771$  is maintained where the counts are the occurrences of the words appearing in all documents from

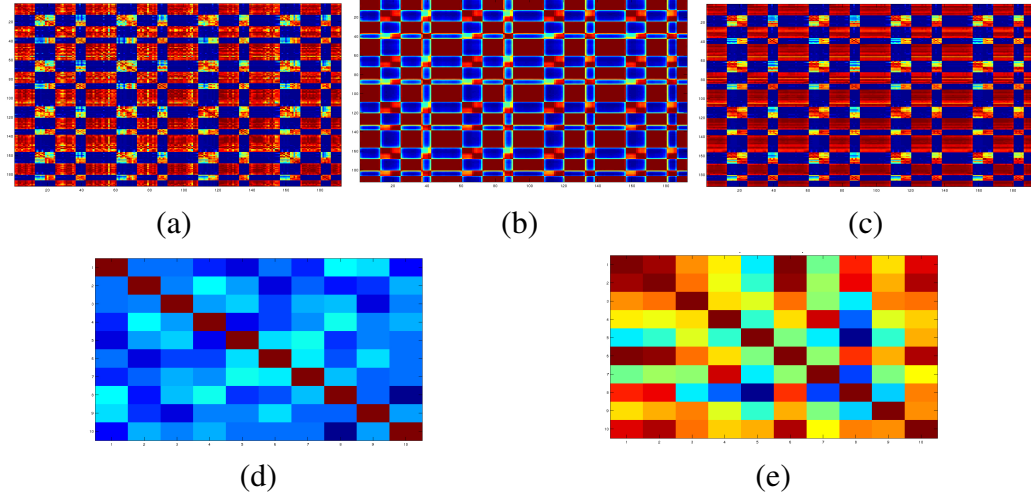


Figure 7.3: (a) correlation of the observed data across time, (b) correlation discovered in the latent space, (c) correlation between the observation and latent counts, (d) correlation between the ten most prominent latent factors for GP-DPFA, (e) correlation between the ten most prominent latent factors in the baseline model.

the given year, chosen after standard pre-processing like stemming and stop-words removal.

Table 3 displays the results from both STU and Conf. datasets. 20% of the words are held-out for each of the first 224 years in the STU data and 10% of the words are held-out for each of the first 13 years in the Conf. data, when training three different models: i) GP-DPFA, ii) DRFM [Han et al., 2014], and iii) a baseline model which is a simplified version of GP-DPFA that does not use temporal correlation for the latent rates. Additionally, all the data from the last year for both of these datasets are held-out. The underlying prediction problem is concerned with estimating the held-out words. For the prediction corresponding to each year, the words are ranked according to the estimated count and then two quantities are calculated: i)

precision@top- $M$  which is given by the fraction of the top- $M$  words, predicted by the model, that matches the true ranking of the words; and ii) recall@top- $M$  which is given by the fraction of words from the held-out set that appear in the top- $M$  ranking. In the experiments reported,  $M = 50$  is used. For the last year for which entire data is held-out, calculation of recall@top- $M$  is irrelevant. In Table 3, the column MP and MR signify mean precision and mean recall respectively over all the years that appear in the training set. The column PP signifies the predictive precision for the final year, for which the entire dataset is held out. Such measure is also adopted for the recommendation system in [Gopalan et al., 2014a] and is perhaps the only reasonable measure when the likelihoods between two different models like GP-DPFA and DRFM are not comparable. GP-DPFA almost always outperforms DRFM and both of these dynamic models convincingly beat the baseline model.

For the Conf. dataset, Fig. 4(a) shows the correlation discovered in the latent space over time, and Figs. 4 (b) and (c) show the normalized strengths of the latent factors (*i.e.*  $\lambda_k \theta_{tk} / \sum_k \lambda_k \theta_{tk}$ ) discovered by GP-DPFA and the baseline model, respectively. One can clearly see that the assignments to latent factors are strongly correlated with time for GP-DPFA but the baseline model tends to choose different latent factors for different years. In the experiments,  $K = 100$  is used and GP-DPFA infers that only a small subset of the 100 topics need to be active, implying an automatic model selection. The number of active latent topics is found to be around 14 on average. Examining some of the topics provides even more insight about the data. For example, the top words of a topic that has large weights

across all years include “network”, “graph-partition”, “algorithm”, “cluster” and “outlier”, whereas the top words of a topic that is dominant over a certain period of time include “Bregman”, “projection”, “clustering” and “ensemble”, revealing the author’s publication trend.

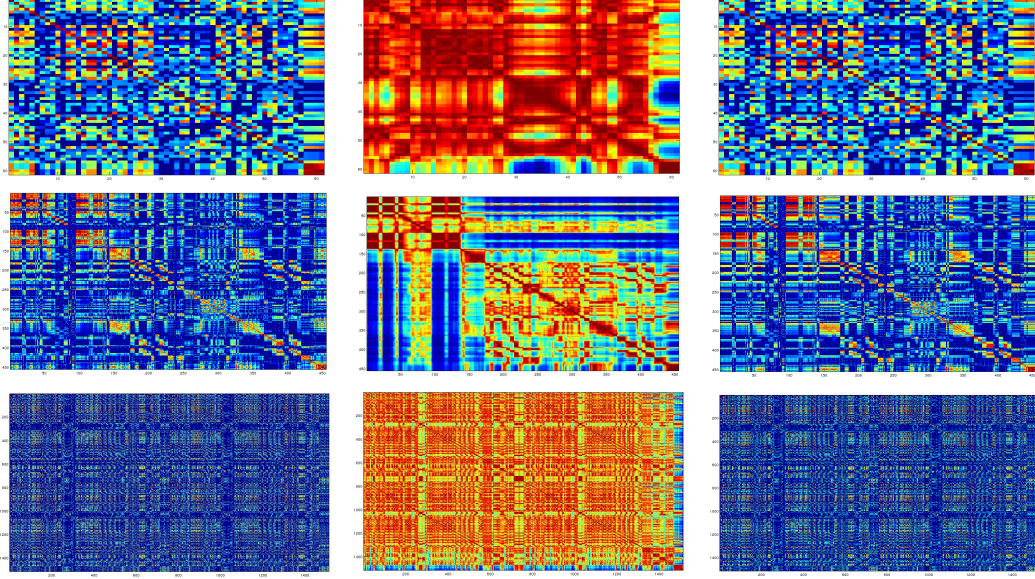


Figure 7.4: Top Row: Correlation plots for JSB chorales, Middle Row: Correlation plots for Piano.midi, Bottom Row: Correlation plots for Musedata. In each row, figures from left to right are plots that are analogous to Figs. 5 (a)-(c).

**Music Dataset:** Four different polyphonic music sequences of piano are used for experiments with GP-DPFA. Each of these datasets is a collection of binary strings indicating which of the keys are “on” at each time [Boulanger-Lewandowski et al., 2012; Poliner and Ellis, 2007]. “Nottingham” is a collection of 1200 folk tunes, “Piano.midi” is a classical piano MIDI archive, “MuseData” is an electronic library of orchestral and piano classical music, and “JSB chorales” refers to the

entire corpus of 382 four-part harmonized chorales by J. S. Bach. The polyphony (number of simultaneous notes) varies from 0 to 15 and the average polyphony is 3.9. We use an input of 88 binary units that span the whole range of piano from A0 to C8. In Fig. 5, results are displayed for one of the 1200 tunes from Nottingham data. Fig. 5(a) shows the correlation of the binary strings across time. Interestingly, a similar but more prominent correlation structure is discovered in the latent factor scores (*i.e.* across  $(\lambda_k \theta_{tk})_{k=1}^K$ 's), displayed in Fig. 5(b). Additionally, the correlations between the original data and the estimated latent counts are presented in Fig. 5(c). One can see that this correlation plot perfectly imitates the correlation between the original data, implying that the original data are faithfully reconstructed using GP-DPFA. Also, in Fig. 5(d) we display the correlation between the top ten  $\phi_k$ 's (ranked according to the magnitudes of the  $\lambda_k$ 's) discovered by GP-DPFA. We compare this plot with Fig. 5(e), which shows the correlation between the top ten  $\phi_k$ 's discovered by the non-dynamic baseline model. One can clearly see that GP-DPFA discovers comparatively less correlated latent factors.

The top, middle and bottom rows in Fig. 6 illustrate the correlation plots for one of the tunes in the JSB chorales dataset, the Piano.midi data and the MuseData, respectively. The left-most plot in each of the rows shows the correlation of the observed data. The plots in the middle illustrate the correlation discovered in the latent space and the plots in the last column shows the the correlation between the observed data and estimated latent counts. It is shown that even when the correlation structure is not clear in the original data, very clear correlation structure is discovered in the latent space, without sacrificing the data reconstruction quality.

## 7.2 Nonparametric Dynamic Network Modeling

Many complex social and biological interactions can be naturally represented as graphs. Often these graphs evolve over time. For example, an individual in a social network can get acquainted with a new person, an author can collaborate with a new author to write a research paper and proteins can change their interactions to form new compounds. Consequently, a variety of statistical and graph-theoretic approaches have been proposed for modeling both static and dynamic networks [Airoldi et al., 2008; Fu et al., 2009; Gopalan et al., 2012; Kemp et al., 2006; Kim and Leskovec, 2013; Sarkar and Moore, 2005; Snijders et al., 2010; Xu and Hero, 2014; Zhou, 2015].

Of particular interest in this work are scalable techniques that can identify groups or communities and track their evolution. Existing non-parametric Bayesian approaches for this task promise to solve the model selection problem of identifying an appropriate number of groups, but are computationally intensive, and often do not match the characteristics of real datasets. All such models assume that the data comes from a latent space that has *either* discrete sets of configurations [Foulds et al., 2011; Kim and Leskovec, 2013; Sarkar and Moore, 2005] *or* is modeled using Gaussian distribution [Fu et al., 2009; Ho et al., 2011; Xu and Hero, 2014]. Approaches that employ discrete latent states do not have closed-form inference updates, mostly due to the presence of probit or logit links. On the other hand, Gaussian assumption is often overly restrictive for modeling binary matrices. Since the inference techniques for linear dynamical systems are well-developed, one usually is tempted to connect a binary observation to a latent Gaussian random variable

using the probit or logit links. Such approaches, however, involve heavy computation and lack intuitive interpretation of the latent states.

This work attempts to address such inadequacies by introducing an efficient and effective model for binary matrices that evolve over time. Its contributions include:

- A novel non-parametric Gamma Process dynamic network model that predicts the number of latent network communities from the data itself.
- A technique for allowing the weights of these latent communities to vary smoothly over time using a Gamma-Markov chain, the inference of which is solved using an augmentation trick associated with the Negative Binomial distribution together with a forward-backward sampling algorithm, each step of which has closed-form updates.
- Empirical results indicating clear superiority of the proposed dynamic network model as compared to existing baselines for dynamic and static network modeling.

The rest of the section is organized as follows. Pertinent background and related works are outlined in Section 7.2.1. A detailed description of the Dynamic Gamma Process network model in Section 7.2.2. Empirical results for both synthetic and real-world data are reported in Section 7.2.3.

### **7.2.1 Related Work**

We mention select, most relevant approaches from a substantial literature on this topic. Among static latent variable based models, the Infinite Relational Model

(IRM [Kemp et al., 2006]) allows for multiple types of relations between entities in a network and an infinite number of clusters, but restricts these entities to belong to only one cluster. The Mixed Membership Stochastic Blockmodel (MMSB [Airoldi et al., 2008]) assumes that each node in the network can exhibit a mixture of communities. Though the MMSB has been applied successfully to discover complex network structure in a variety of applications, the computational complexity of the underlying inference mechanism is in the order of  $N^2$ , which limits its use to small networks. Computation complexity is also a problem with many other existing latent variable network models, such as the latent feature relational model [Miller et al., 2009] and its max margin version [Zhu, 2012], and the infinite latent attribute model [Palla et al., 2012]. Regardless, such models are adept at identifying high-level clusters and perform particularly well for link prediction in small, dense, static networks. The Assortative Mixed-Membership Stochastic Blockmodel (a-MMSB [Gopalan et al., 2012]) bypasses the quadratic complexity of the MMSB by making certain assumptions about the network structure that might not be true in general, such as assuming the probability of linking distinct communities is small, sub-sampling the network, and employing stochastic variational inference that uses *only* a noisy estimate of the gradients. The hierarchical Dirichlet process relational model [Kim et al., 2013] allows mixed membership with an unbounded number of latent communities; however, it is built on the a-MMSB whose assumptions could be restrictive.

There has been quite a bit of research with non-Bayesian [Hanneke et al., 2010; Snijders et al., 2010] as well as Bayesian approaches [Ho et al., 2011; Ishig-

uro et al., 2010; Sarkar and Moore, 2005; Xu and Hero, 2014] to study dynamic networks. The Bayesian approaches differ among themselves due to the assumptions in structures of the latent space they make. For example, Euclidean space models [Hoff et al., 2001; Sarkar and Moore, 2005] place nodes in a low dimensional Euclidean space and the network evolution is then modeled as a regression problem of future latent node location. On the other hand, certain models [Fu et al., 2009; Ho et al., 2011; Ishiguro et al., 2010] assume that the latent variables stochastically depend on the state at the previous time step. Some other models use multi-memberships [Foulds et al., 2011; Heaukulani and Ghahramani, 2013; Kim and Leskovec, 2013] wherein a node’s membership to one group does not limit its membership to other groups. Compared to these approaches, D-NGPPF models the latent factors using Gamma distribution and the shape parameter of the distribution of the latent factor at time  $t$  is modeled by the latent factor at time  $(t - 1)$ . The network entries are generated from a Truncated Poisson distribution whose rate is given by the underlying latent variables, some of which evolve over time and will be described in more details later.

### 7.2.2 Dynamic Gamma Process Poisson Factorization for Networks (D-NGPPF)

Consider a tensor  $\mathbb{B} \in \mathbb{Z}^{N \times N \times T}$ , whose  $T$  columns are sequentially observed  $N \times N$ -dimensional binary matrices, and are indexed by  $\{\mathbf{B}_t\}_{t=1}^T$ . Further, consider a gamma process  $G \sim \Gamma\text{P}(c, G_0)$ , a draw from which is expressed as  $G = \sum_{k=1}^{\infty} r_{0k} \delta_{\phi_k}$ , where  $\phi_k \in \Omega$  is an atom drawn from an  $N$ -dimensional base distribution  $\phi_k \sim \prod_{n=1}^N \text{Gamma}(e_0, 1/c_n)$  and  $r_{0k} = G(\phi_k)$  is the associated

weight. We mark each atom  $\phi_k$  with an  $r_{1k}$  and generate a gamma Markov chain by letting:

$$r_{tk}|r_{(t-1)k} \sim \text{Gam}(r_{(t-1)k}, 1/c), \quad t = \{1, \dots, T\}.$$

The  $(n, m)^{\text{th}}$  entry at time  $t$  is assumed to be generated as follows:

$$b_{tnm} = \mathbb{I}_{x_{tnm} \geq 1}, \quad x_{tnm} \sim \text{Pois} \left( \sum_k r_{tk} \phi_{nk} \phi_{mk} \right).$$

Similar to NGPPF, to complete the generative process, we put Gamma priors over  $c$  and  $c_n$  as:

$$c \sim \text{Gamma}(c_0, 1/d_0), \quad c_n \sim \text{Gamma}(f_0, 1/g_0). \quad (7.14)$$

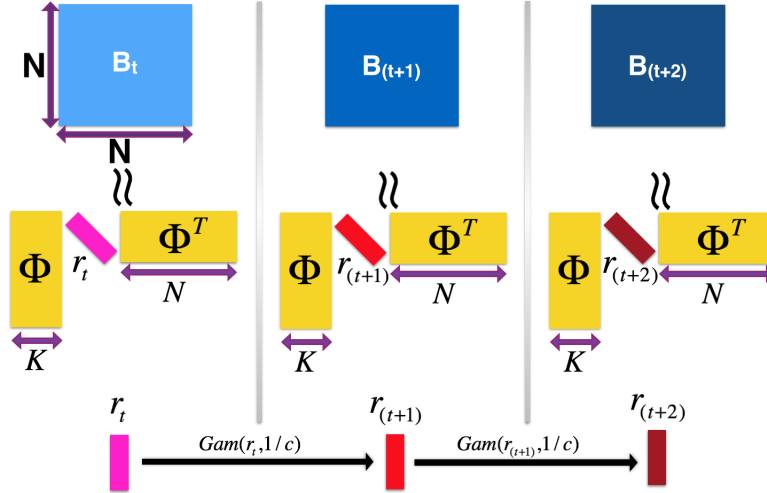


Figure 7.5: Illustration of Dynamic Network Model

In the formulation above of the dynamic network model, we assume that the weights of the latent factors evolve over time using a Gamma markov chain. At the  $t^{\text{th}}$  time instance, the proximity (or assignment) of the  $n^{\text{th}}$  entity of the network to the

$k^{\text{th}}$  latent factor is given by  $r_{tk}\phi_{nk}$  and hence the evolution of  $r_{tk}$  alone can capture the changes in characteristics of the  $n^{\text{th}}$  network entity. In many applications, one may also evolve  $\phi_{nk}$  over time, but we leave that as an interesting future work.

### 7.2.2.1 Gibbs Sampling for D-NGPPF

Similar to the implementation for N-GPPF, a finite approximation of the infinite model is considered by truncating the number of factors to  $K$  which approaches the original infinite model as  $K \rightarrow \infty$ .

**Sampling of  $x_{tnm}$  :**  $x_{tnm}$ 's are sampled only corresponding to the following entries:

$$(t, n, m) : t = \{1, \dots, T\}, n = \{1, \dots, (N-1)\}, m = \{(n+1), \dots, N\}.$$

For the above entries, the sampling goes as follows:

$$x_{tnm} \sim b_{tnm} \text{Pois}_+ \left( \sum_{k=1}^K r_{tk} \phi_{nk} \phi_{mk} \right). \quad (7.15)$$

Since, one can augment  $x_{tnm} \sim \text{Pois} \left( \sum_{k=1}^K r_{tk} \phi_{nk} \phi_{mk} \right)$  as  $x_{tnm} = \sum_{k=1}^K x_{tnmk}$ , where  $x_{tnmk} \sim \text{Pois}(r_{tk} \phi_{nk} \phi_{mk})$ , equivalently, one obtains the following according to Lemma 2.1.4:

$$(x_{tnmk})_{k=1}^K | \sim \text{mult} \left( (r_{tk} \phi_{nk} \phi_{mk})_{k=1}^K / \sum_{k=1}^K r_{tk} \phi_{nk} \phi_{mk}; x_{tnm} \right). \quad (7.16)$$

**Sampling of  $r_{tk}$  :** The data augmentation and marginalization techniques specific to the NB distribution [Acharya et al., 2015; Zhou and Carin, 2012] are utilized to sample  $r_{tk}$ . Despite the challenge present in inferring the gamma shape parameters,

closed-form Gibbs sampling update equations can be derived for all the  $r_{tk}$ 's. For  $t = T$ , one can sample:

$$r_{Tk} | \sim \text{Gam} \left( r_{(T-1)k} + x_{T..k}, 1/(c + s_k) \right), x_{T..k} = \sum_{n=1}^{(N-1)} \sum_{m=(n+1)}^N x_{Tnmk}, s_k = \sum_{n=1}^{(N-1)} \sum_{m=(n+1)}^N \phi_{nk} \phi_{mk}. \quad (7.17)$$

For  $t = (T - 1)$ , one needs to augment  $\ell_{Tk} \sim \text{CRT}(x_{T..k}, r_{(T-1)k})$ , after which, using Lemma 2.2.2 one obtains the following:

$$r_{(T-1)k} | \sim \text{Gam} \left( r_{(T-2)k} + x_{(T-1)..k} + \ell_{Tk}, 1/(c + s_k - \log(1 - p_{Tk})) \right), \quad (7.18)$$

$$x_{(T-1)..k} = \sum_{n=1}^{(N-1)} \sum_{m=(n+1)}^N x_{(T-1)nmk}, p_{Tk} = \frac{s_k}{(c + s_k)}.$$

For  $1 \leq t \leq (T-2)$ , the augmentation and sampling trick is very similar. One needs to augment  $\ell_{(t+1)k} \sim \text{CRT}(x_{(t+1)..k} + \ell_{(t+2)k}, r_{tk})$  and then sample  $r_{tk}$  according to Lemma 2.2.2:

$$r_{tk} | \sim \text{Gam} \left( r_{(t-1)k} + x_{t..k} + \ell_{(t+1)k}, 1/(c + s_k - \log(1 - p_{(t+1)k})) \right), \quad (7.19)$$

$$x_{t..k} = \sum_{n=1}^{(N-1)} \sum_{m=(n+1)}^N x_{tnmk}, p_{(t+1)k} = \frac{s_k - \log(1 - p_{(t+2)k})}{(c + s_k - \log(1 - p_{(t+2)k}))}.$$

For  $t = 0$ , augment  $\ell_{1k} \sim \text{CRT}(x_{1..k} + \ell_{2k}, r_{0k})$ . Then sample

$$r_{0k} | \sim \text{Gam}(\gamma_k + \ell_{1k}, 1/(c - \log(1 - p_{1k}))), \quad (7.20)$$

$$x_{1..k} = \sum_{n=1}^{(N-1)} \sum_{m=(n+1)}^N x_{1nmk}, p_{1k} = \frac{s_k - \log(1 - p_{2k})}{(c + s_k - \log(1 - p_{2k}))}.$$

**Sampling of  $\gamma_k$  :** Augment  $\ell_{0k} \sim \text{CRT}(\ell_{1k}, \gamma_k)$ . Then sample

$$\gamma_k | \sim \text{Gam} (a_0 + \ell_{0k}, 1/(b_0 - \log(1 - p_{0k}))), p_{0k} = \frac{\log(1 - p_{1k})}{(\log(1 - p_{1k}) - c)}.$$

**Sampling of  $\phi_{nk}$**  : Sampling of these parameters follow from Lemma 2.1.1 and are given as follows:

$$\phi_{nk} | \sim \text{Gam} \left( d_0 + \sum_{t=1}^T \left( \sum_{m=1}^{(n-1)} x_{tmnk} + \sum_{m=(n+1)}^N x_{tmnk} \right), 1 / \left( c_n + \sum_{t=1}^T \sum_{m=1, m \neq n}^N r_{tk} \phi_{mk} \right) \right). \quad (7.21)$$

**Sampling of  $c_n$  and  $c$**  : Sampling of these parameters follow from Lemma 2.1.2 and are given as follows:

$$c_n | \sim \text{Gam} \left( f_0 + K e_0, 1 / \left( g_0 + \sum_{k=1}^K \phi_{nk} \right) \right), \quad (7.22)$$

$$c | \sim \text{Gam} \left( \sum_{k=1}^K \left( \gamma_k + \sum_{t=0}^{(T-1)} r_{tk} \right) + c_0, 1 / \left( \sum_{k=1}^K \sum_{t=0}^T r_{tk} + d_0 \right) \right). \quad (7.23)$$

### 7.2.2.2 Gibbs Sampling for D-NGPPF with Missing Entries

Variables whose update get affected in presence of missing values are  $r_{tk}$ 's and  $\phi_{nk}$ 's. Rest of the update equations are same as in D-NGPPF without any missing value. Below, the updates are enlisted where  $\mathcal{M}_t$  denotes the set of missing entries in the network at the  $t^{\text{th}}$  time instance.

**Sampling of  $r_{tk}$**  : For  $t = T$ ,

$$r_{Tk} | \sim \text{Gam} \left( r_{(T-1)k} + x_{T..k}, 1 / (c + s_{Tk}) \right), \quad (7.24)$$

$$x_{T..k} = \sum_{\substack{n=1, m=(n+1) \\ (n,m) \notin \mathcal{M}_T}}^{(N-1), N} x_{Tnmk}, \quad s_{Tk} = \sum_{\substack{n=1, m=(n+1) \\ (n,m) \notin \mathcal{M}_T}}^{(N-1), N} \phi_{nk} \phi_{mk}.$$

For  $t = (T - 1)$ , augment  $\ell_{Tk} \sim \text{CRT}(x_{T..k}, r_{(T-1)k})$  and then sample

$$r_{(T-1)k} | \sim \text{Gam} \left( r_{(T-2)k} + x_{(T-1)..k} + \ell_{Tk}, 1 / (c + s_{(T-1)k} - \log(1 - p_{Tk})) \right), \quad (7.25)$$

$$x_{(T-1)k} = \sum_{\substack{n=1, m=(n+1) \\ (n,m) \notin \mathcal{M}_{(T-1)}}}^{(N-1), N} x_{(T-1)nmk}, s_{(T-1)k} = \sum_{\substack{n=1, m=(n+1) \\ (n,m) \notin \mathcal{M}_{(T-1)}}}^{(N-1), N} \phi_{nk} \phi_{mk}, p_{Tk} = \frac{s_{Tk}}{(c + s_{Tk})}.$$

For  $1 \leq t \leq (T - 2)$ , augment  $\ell_{(t+1)k} \sim \text{CRT}(x_{(t+1)k} + \ell_{(t+2)k}, r_{tk})$  and then sample  $r_{tk}$  as:

$$r_{tk} | \sim \text{Gam} \left( r_{(t-1)k} + x_{t..k} + \ell_{(t+1)k}, 1 / (c + s_{tk} - \log(1 - p_{(t+1)k})) \right), \quad (7.26)$$

$$x_{t..k} = \sum_{\substack{n=1, m=(n+1) \\ (n,m) \notin \mathcal{M}_t}}^{(N-1), N} x_{tnmk}, s_{tk} = \sum_{\substack{n=1, m=(n+1) \\ (n,m) \notin \mathcal{M}_t}}^{(N-1), N} \phi_{nk} \phi_{mk}, p_{(t+1)k} = \frac{s_{(t+1)k} - \log(1 - p_{(t+2)k})}{(c + s_{(t+1)k} - \log(1 - p_{(t+2)k}))}.$$

**Sampling of  $\phi_{nk}$  :**

$$\phi_{nk} | \sim \text{Gam} \left( d_0 + \sum_{t=1}^T \left( \sum_{\substack{m=1 \\ (m,n) \notin \mathcal{M}_t}}^{(n-1)} x_{tnmk} + \sum_{\substack{m=(n+1) \\ (n,m) \notin \mathcal{M}_t}}^N x_{tnmk} \right), 1 / \left( c_n + \sum_{t=1}^T \sum_{\substack{m=1, m \neq n \\ (n,m) \notin \mathcal{M}_t}}^N r_{tk} \phi_{mk} \right) \right). \quad (7.27)$$

## 7.2.3 Experiments

In this section, experimental results are reported for a synthetic data and three real world datasets. For all the experiments with synthetic and real world data, the Gibbs sampler is run with 2000 burn-in and 2000 collection iterations, and  $K = 50$  is maintained.

### 7.2.3.1 Synthetic Data

We generate a set of synthetic networks of size  $60 \times 60$  with three different groups that evolve over six different time stamps. These datasets are displayed in column (a) in both Fig. 7.6 and 7.7. In practice, this may represent a group of users in a social network whose friend circles change over time. The links in these graphs are presented by brown and the non-links are illustrated by deep blue. The

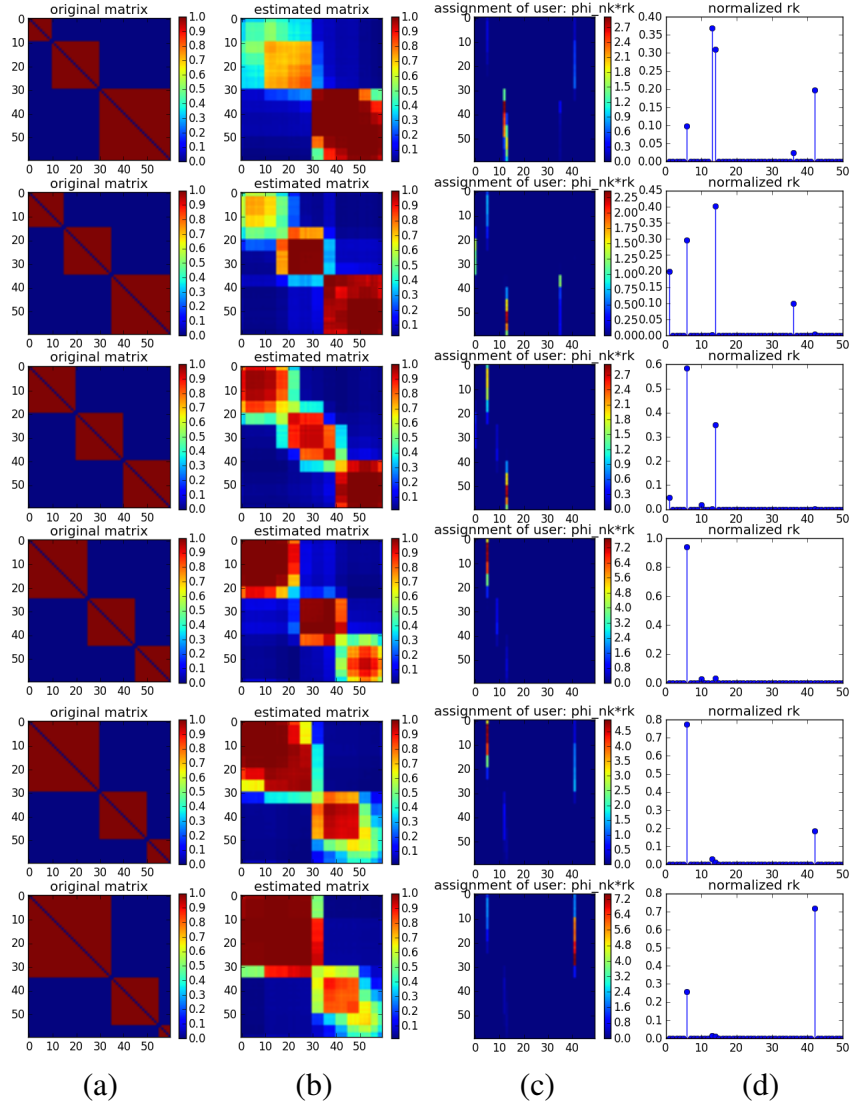


Figure 7.6: Results from D-NGPPF

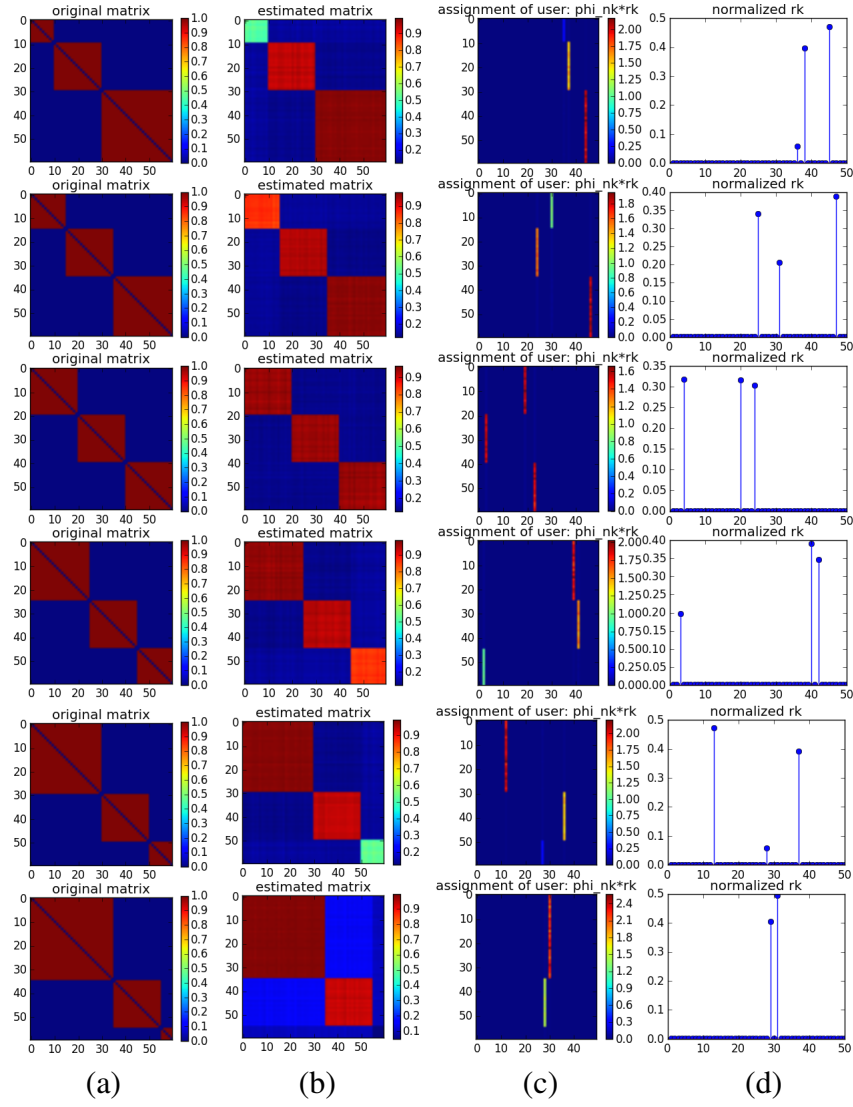


Figure 7.7: Results from N-GPPF

performance of D-NGPPF is displayed in columns (b), (c), and (d) of Fig. 7.6. Column (b) in Fig. 7.6 shows the groups discovered by D-NGPPF in the graph over different time-stamps. Note that the discovery of groups at any time instance is influenced by the groups present in other time instances. In column (c) of Fig. 7.6, the proximity of the users to the latent groups are displayed. The x-axis in each of these plots imply different latent groups and the y-axis represents the proximity of the  $n^{\text{th}}$  user to the  $k^{\text{th}}$  latent group at the  $t^{\text{th}}$  time instance, which is calculated as  $r_{tk}\phi_{nk}$ . In our experiments, 50 different latent groups are maintained ( $K = 50$ ), but the model assigns the users to only a few of the latent groups, a desired outcome. This observation is also reinforced by the plots in column (c) of Fig. 7.6. These plots denote the normalized weights of the different latent groups ( $r_{tk} / \sum_{k=1}^K r_{tk}$ ) at different time instances. In each time instance, only a few latent groups have positive weight. Expectedly, as displayed in columns (c) and (d) of Fig. 7.6, the latent factors that are dominant over different time instances vary smoothly with time. In Fig. 7.7, results are displayed for a baseline model that uses only N-GPPF for modeling the networks isolatedly at each different time slice. One can see that N-GPPF reconstructs the groups perfectly at each time instance as the groups are very clear-cut. However, different sets of latent groups dominate in modeling the networks at different time slices, as revealed in plots of columns (c) and (d) of Fig. 7.7. Unlike this toy example, most real world networks are sparse and groups are less distinct at any given time. The performance of a static network model is expected to be poorer in such settings, as it cannot link the solutions across time. This is explained more clearly alongside the results reported in the next subsection.

### 7.2.3.2 Real World Data

**NIPS Authorship Network Data:** The NIPS co-authorship network connects two people if they appear on the same publication in the NIPS conference in a given year. Network spans  $T = 17$  years (1987 to 2003). Following [Heaukulani and Ghahramani, 2013], only a subset of 110 authors, who are most connected over all the time periods, are considered. For evaluating the predictive performance, 25% of the links and equal number of non-links are held out from each of the 17 time instances. The rest of the data is used as training. DSBM [Xu and Hero, 2014], N-GPPF and MMSB [Airoldi et al., 2008] are considered as the baselines in the prediction problem. For both N-GPPF and MMSB, the networks for the different time instances are modeled isolatedly. We use the implementation from the authors of DBSM for the corresponding set of experiments. Since both DBSM and MMSB are parametric methods, we use  $K = 10$  for all the experiments which, as the literature reports, is found to produce best results for these set of models with these datasets. The objective is to infer the labels of the held out links and non-links. The quality of prediction is measured by AUC and the results are displayed in Table 7.1.

Dataset	D-NGPPF	DSBM	N-GPPF	MMSB
NIPS	<b>0.797</b> $\pm 0.016$	0.780 $\pm 0.010$	0.766 $\pm 0.012$	0.740 $\pm 0.009$
DBLP	<b>0.836</b> $\pm 0.013$	0.810 $\pm 0.013$	0.756 $\pm 0.020$	0.749 $\pm 0.014$
Infocom	<b>0.907</b> $\pm 0.008$	0.901 $\pm 0.006$	0.856 $\pm 0.011$	0.831 $\pm 0.006$

Table 7.1: AUC Results on Real World Data

**DBLP Data:** The DBLP co-authorship network is obtained from 21 Computer

Science conferences from 2000 to 2009 ( $T = 10$ ) [Tang et al., 2008]. Only top 209 people are considered in this datasets by taking 7-core of the aggregated network for the entire time. For each different time slice, 10% of the links and equal number of non-links are held out. The results are displayed in Table 7.1.

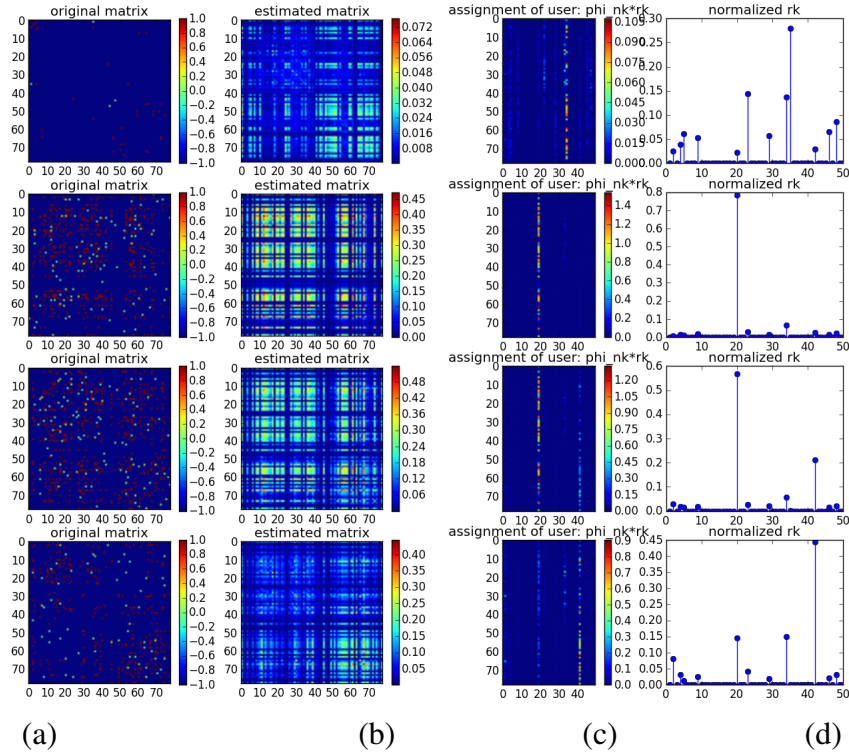


Figure 7.8: Infocom: Hour 5<sup>th</sup> to 8<sup>th</sup>

**Infocom Data:** The Infocom dataset represents the physical proximity interactions between 78 students at the 2006 Infocom conference, recorded by wireless detector remotes given to each attendee [J.Scott et al., 2009]. As in [Heaukulani and Ghahramani, 2013], the recordings are agglomerated into one hour-long time slices and only the reciprocated sightings are maintained. Also, the slices with less than

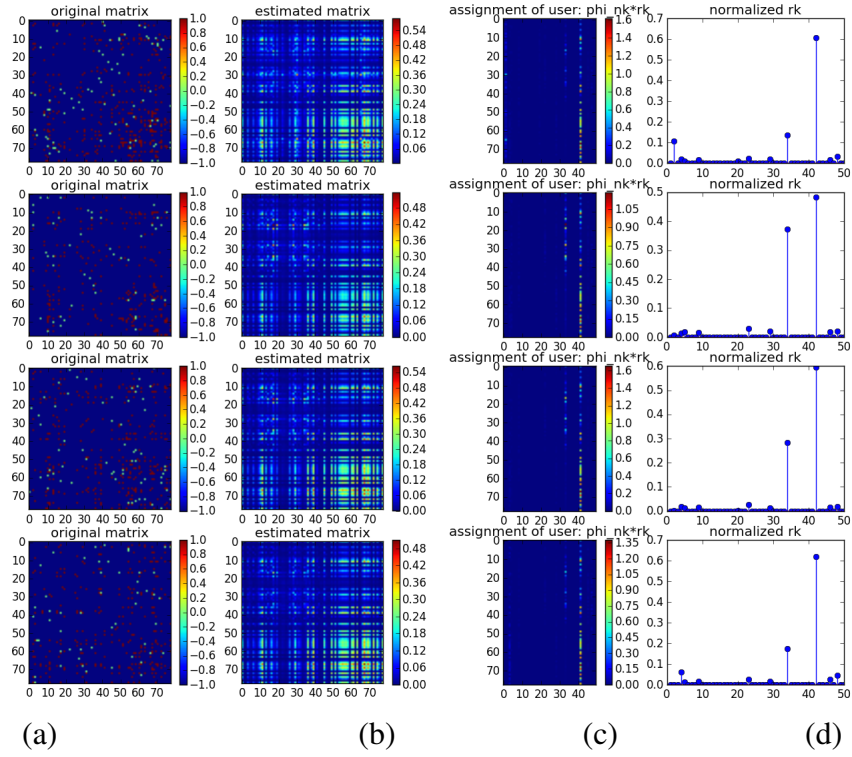


Figure 7.9: Infocom: Hour 9<sup>th</sup> to 12<sup>th</sup>

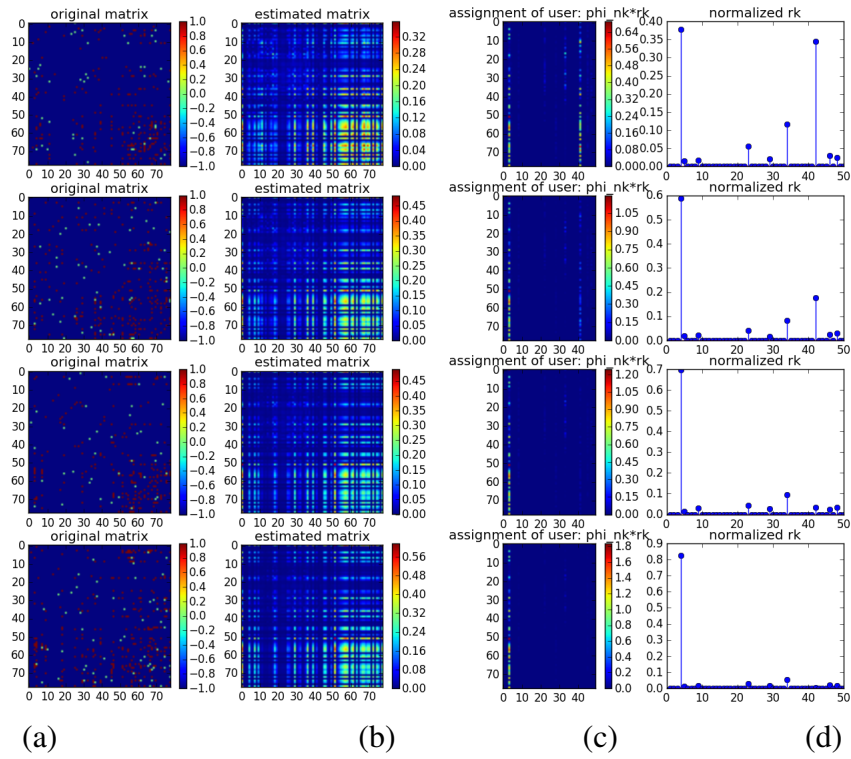


Figure 7.10: Infocom: Hour 13<sup>th</sup> to 16<sup>th</sup>

80 links (corresponding to late night and early morning hours), are removed, resulting in only 50 time slices. For each different time slice, 10% of the links and equal number of non-links are held out. The results are displayed in Table 7.1. One can see that D-NGPPF outperforms DSBM, a strong baseline for dynamic network modeling, and two other baselines for static network modeling.

To illustrate the effectiveness of D-NGPPF further in real world data, some findings are presented in Fig. 7.8 to Fig. 7.10 for the Infocom dataset. One can see the smooth transition of the dominant factors over time. Fig. 7.8, 7.9 and 7.10 present the results corresponding to the datasets at times  $T = 4$  to  $T = 8$ ,  $T = 9$  to  $T = 12$  and  $T = 13$  to  $T = 16$  respectively. Column (a) in each of these figures present the original network with some of the entities held out (indicated by green). Column (b) represents the cluster structures discovered by D-NGPPF, while column (c) and (d) signify the assignment of the users in the latent space and the weights of the latent factors respectively. Note that, for each time slice, very few links are available (indicated by deep brown) and hence the performance of N-GPPF for prediction of held-out links is poorer, as illustrated in Table 7.1.

### 7.3 Nonparametric Dynamic Count Matrix Factorization

Analysis of dyadic data, which represents relationship between two different sets of entities such as users and items, has been a prolific domain of research since the last decade, particularly due to their applications in recommendation systems [Christakopoulou and Banerjee, 2015; Deodhar and Ghosh, 2009; Koren et al., 2009], e-commerce [Raghavan et al., 2012], topic modeling [Ahmed and Xing,

2008; Blei et al., 2003; Du et al., 2015] and bio-informatics [Natarajan and Dhillon, 2014]. Successful as different techniques for such analysis are, a major limitation of them is that they are static models and ignore the temporal correlation and evolution of the relationships between entities, an attribute present in most real-world dyadic data. Among the handful of techniques that deal with the temporal correlation in recommendation systems, TimeSVD++ [Koren, 2009] and Bayesian probabilistic tensor factorization (BPTF) [Xiong et al., 2010] are worth mentioning. Such algorithms assume that the latent factors are distributed according to a normal distribution and an interaction of such latent factors generates the actual observation, which is clearly restrictive for count-valued dyadic data. Since the inference techniques for linear dynamical systems are well-developed, one usually is tempted to connect a count-valued observation to a latent Gaussian random variable, though such approaches incur heavy computation, fail to exploit the natural sparsity of the data and lack interpretation of the latent states. On the other hand, text mining researchers have developed numerous techniques for analyzing a corpus that evolves over time which is modeled as sequence of document-by-word count matrices. Some of these techniques are equipped with Kalman filtering based inference and a nonlinear transformation of the latent states to the discrete observations [Wang et al., 2008], while some others use temporal Dirichlet process and make arguably simplistic assumptions [Ahmed and Xing, 2008, 2010] to calculate an intractable posterior for MCMC sampling. A detailed discussion of and comparison with the existing works on dynamic topic model are beyond the scope of this paper and we only present results corresponding to dynamic collaborative filtering. To be

more specific, the contributions of this work include:

- A novel non-parametric Gamma Process dynamic count matrix factorization model that predicts the number of latent factors from the data itself.
- A technique for allowing the weights of these latent factors to vary smoothly over time using a Gamma-Markov chain, the inference of which is solved using an augmentation trick associated with the Negative Binomial distribution together with a forward-backward sampling algorithm, each step of which has closed-form updates.
- Empirical results indicating clear superiority of the proposed dynamic matrix factorization model as compared to existing baselines for dynamic and static count matrix factorization models.

The rest of the section is organized as follows. Pertinent background and related works are outlined in Section 7.3.1. A detailed description of the dynamic Gamma Process count matrix factor modeling is provided in Section 7.3.2. Empirical results for both synthetic and real-world data are reported in Section 7.3.3. Finally, the conclusion and future works are listed in Section 7.4.

### **7.3.1 Background and Related Work**

One of the notable contributions towards dynamic relational model is a similarity based approach [Ding and Li, 2005] where similarity score is calculated by reducing the importance of the older data. Sugiyama et al. [2004] proposes a personalized web search engine where they allow the profile of each user to evolve

over time. TimeSVD++ [Koren, 2009] assumes that the latent features consist of two parts, one that evolves over time and the other which does not and acts as bias. This model can effectively capture local changes of user preferences, though the performance depends on some of the regularization parameters, tuning of which is prohibitively expensive for large datasets. On the other hand, BPTF captures the global effect of time that are shared among all users and items and imposes prior over some of the regularization parameter for which the performance is relatively insensitive towards initialization of the corresponding hyper-parameters. However, BPTF incurs a computation cost  $O(DVK^2 + (D + V + T)K^3)$ , where  $K$  is the dimension of the latent space, which is expensive for large matrices. Such computation complexity is also a problem with other latent gaussian based approaches that minimize squared error (such as TimeSVD++) as they model both zeros and non-zeros, and the latent factors corresponding to both zeros and non-zeros need to be sampled.

### 7.3.2 Dynamic Gamma Process Poisson Factorization for Count Matrices (DCGPPF)

Consider a tensor  $\mathbb{Y} \in \mathbb{Z}^{D \times V \times T}$ , whose  $T$  columns are sequentially observed  $D \times V$ -dimensional count matrices, and are indexed by  $\{\mathbf{Y}_t\}_{t=1}^T$ . Further, consider a gamma process  $G \sim \text{GP}(c, G_0)$ , a draw from which is expressed as  $G = \sum_{k=1}^{\infty} r_{0k} \delta_{\theta_k}$ , where  $\theta_k \in \Omega$  is an atom drawn from a  $D$ -dimensional base distribution  $\theta_k \sim \prod_{d=1}^D \text{Gam}(g_0, 1/c_d)$  and  $r_{0k} = G(\theta_k)$  is the associated weight. We mark each atom  $\theta_k$  with an  $r_{1k}$  and generate a gamma Markov chain by letting:  $r_{tk}|r_{(t-1)k} \sim \text{Gam}(r_{(t-1)k}, 1/c)$ ,  $t = \{1, \dots, T\}$ . Additionally, each atom

$\theta_k$  is marked with an atom  $\beta_k$ , drawn from a  $V$ -dimensional base distribution as  $\beta_k \sim \prod_{w=1}^V \text{Gam}(h_0, 1/s_w)$ . The  $(d, w)^{\text{th}}$  entry at time  $t$  is assumed to be generated from a sum of latent counts as:  $y_{tdw} \sim \text{Pois}(\sum_k \lambda_{tdwk})$  where  $\lambda_{tdwk} = r_{tk}\theta_{dk}\beta_{wk}$ . One may consider  $\lambda_{tdwk}$  as the strength of the latent factor that dictates the relation between the  $d^{\text{th}}$  user and the  $w^{\text{th}}$  item at time  $t$ . Each latent factor contributes such a count and the total count aggregates the countably infinite latent factors. Each of these latent counts is composed of three parts. The parameter  $r_{tk}$  models the global popularity of the latent factor  $k$  at time  $t$ ,  $\theta_{dk}$  models the affinity of the  $d^{\text{th}}$  user to the  $k^{\text{th}}$  latent factor and  $\beta_{wk}$  models the popularity of the  $w^{\text{th}}$  word among the  $k^{\text{th}}$  latent factor. As described in Section 2.3, such modeling assumption is one instance of Poisson factor analysis. To complete the generative process, we put Gamma priors over  $c$ ,  $c_d$  and  $s_w$  as:

$$c \sim \text{Gam}(c_0, 1/d_0), c_d \sim \text{Gam}(e_0, 1/f_0), s_w \sim \text{Gam}(t_0, 1/u_0). \quad (7.28)$$

In the formulation above of the dynamic count factorization, we assume that the weights of the latent factors evolve over time using a Gamma markov chain. At the  $t^{\text{th}}$  time instance, the proximity (or assignment) of the  $d^{\text{th}}$  document to the  $k^{\text{th}}$  latent factor is given by  $r_{tk}\theta_{dk}$  and hence the evolution of  $r_{tk}$  alone can capture the changes in characteristics of the  $d^{\text{th}}$  document. Practical utility of such formulation is that a rating depends not only on the similarity between a given user and a given item, but also on how much these preferences match with the “global trend” prevalent at that point of time. For instance, if a user likes a horror movie but the overall trend of the month is that few people are watching them or talking about them, then

this user is probably not going to watch it neither. However, in many applications, one may also evolve  $\theta_{dk}$  over time, but we leave that as an interesting future work. Also, the factors  $\beta_k$  can adapt with time, for example, in applications like dynamic topic modeling [Zhai and Boyd-graber, 2013] where the vocabulary changes with time.

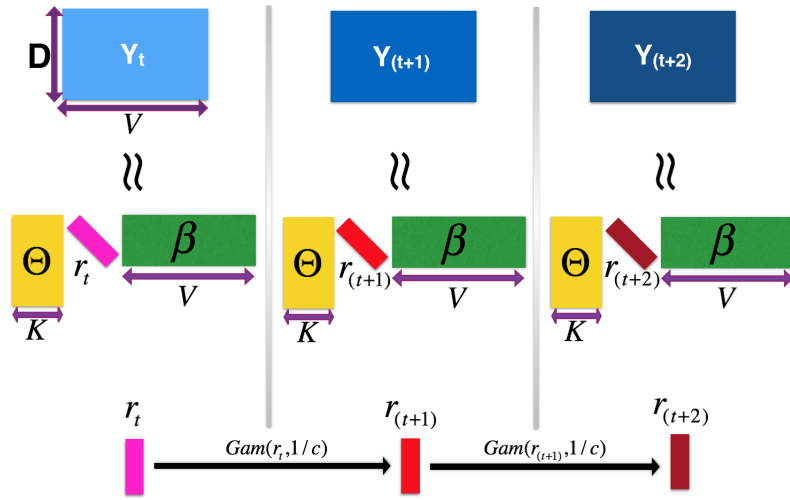


Figure 7.11: Illustration of D-CGPPF

### 7.3.2.1 Gibbs Sampling for Dynamic Poisson Matrix Factor Model

A finite approximation of the infinite model is considered by truncating the number of factors to  $K$  which approaches the original infinite model as  $K \rightarrow \infty$ . The sampling proceeds as follows:

**Sampling of  $x_{tdwk}$**  : This follows from the relation between Poisson and multinomial distribution, given in Lemma 2.1.4, and can be derived as:

$$(x_{tdwk})_{k=1}^K \sim \text{mult} \left( r_{tk} \theta_{dk} \beta_{wk} / \sum_{k=1}^K r_{tk} \theta_{dk} \beta_{wk}; y_{tdw} \right). \quad (7.29)$$

**Sampling of  $r_{tk}$**  : For  $t = T$ , sample  $r_{Tk}$  according to Lemma 2.1.1 as:

$$r_{Tk} | \sim \text{Gam} \left( r_{(T-1)k} + x_{T..k}, 1/(c + \theta_{.k}\beta_{.k}) \right), \quad (7.30)$$

For  $t = (T - 1)$ , first we integrate out  $r_{Tk}$  and according to Lemma 2.1.5, one obtains  $x_{T..k} \sim NB(r_{(T-1)k}, p_{Tk})$ , where  $p_{Tk} = \frac{\theta_{.k}\beta_{.k}}{(c + \theta_{.k}\beta_{.k})}$ . We then augment  $\ell_{Tk} \sim \text{CRT}(x_{T..k}, r_{(T-1)k})$  and according to Lemma 2.2.2 sample:

$$r_{(T-1)k} | \sim \text{Gam} \left( r_{(T-2)k} + x_{(T-1)..k} + \ell_{Tk}, 1/(c + \theta_{.k}\beta_{.k} - \log(1 - p_{Tk})) \right). \quad (7.31)$$

For  $t = (T - 2)$  to  $t = 1$ , following a repeated application of Lemma 2.1.5 and 2.2.2 augment  $\ell_{(t+1)k} \sim \text{CRT}(x_{(t+1)..k} + \ell_{(t+2)k}, r_{tk})$  and then sample

$$r_{tk} | \sim \text{Gam} \left( r_{(t-1)k} + x_{t..k} + \ell_{(t+1)k}, 1/(c + \theta_{.k}\beta_{.k} - \log(1 - p_{(t+1)k})) \right), \quad (7.32)$$

where  $p_{(t+1)k} = \frac{\theta_{.k}\beta_{.k} - \log(1 - p_{(t+2)k})}{(c + \theta_{.k}\beta_{.k} - \log(1 - p_{(t+2)k}))}$ . For  $t = 0$ , augment  $\ell_{1k} \sim \text{CRT}(x_{1..k} + \ell_{2k}, r_{0k})$  and according to Lemma 2.1.5 and 2.2.2 sample:

$$r_{0k} | \sim \text{Gam}(\gamma_k + \ell_{1k}, 1/(c - \log(1 - p_{1k}))), p_{1k} = \frac{\theta_{.k}\beta_{.k} - \log(1 - p_{2k})}{(c + \theta_{.k}\beta_{.k} - \log(1 - p_{2k}))}. \quad (7.33)$$

**Sampling of  $\gamma_k$**  : Augment  $\ell_{0k} \sim \text{CRT}(\ell_{1k}, \gamma_k)$  and according to Lemma 2.2.2, sample:

$$\gamma_k | \sim \text{Gam}(a_0 + \ell_{0k}, 1/(b_0 - \log(1 - p_{0k}))), p_{0k} = \frac{\log(1 - p_{1k})}{(\log(1 - p_{1k}) - c)}. \quad (7.34)$$

**Sampling of  $\theta_{dk}$  and  $\beta_{wk}$**  : Sampling of these variables can be derived according to Lemma 2.1.1 as:

$$\theta_{dk} | \sim \text{Gam}(g_0 + x_{.d.k}, 1/(c_d + r_{.k}\beta_{.k})), \quad (7.35)$$

$$\beta_{wk} | \sim \text{Gam}(h_0 + x_{..wk}, 1/(s_w + r_{.k}\theta_{.k})). \quad (7.36)$$

**Sampling of  $c_d$ ,  $s_w$  and  $c$  :** Sampling of these variables can be derived according to Lemma 2.1.2 and are given as:

$$c_d | \sim \text{Gam}(e_0 + Kg_0, 1/(f_0 + \theta_d)), \quad (7.37)$$

$$s_w | \sim \text{Gam}(t_0 + Kh_0, 1/(u_0 + \beta_w)), \quad (7.38)$$

$$c | \sim \text{Gam}\left(\sum_{k=1}^K \left(\gamma_k + \sum_{t=0}^{(T-1)} r_{tk}\right) + c_0, 1/\left(\sum_{k=1}^K \sum_{t=0}^T r_{tk} + d_0\right)\right). \quad (7.39)$$

A consequence of closed form updates for Gibbs sampling is that the computation per iteration for D-CGPPF is  $O((S + D + V + T)K)$  where  $S$  is the number of number of non-zero entries, which is a huge saving for sparse matrices compared to BPTF whose computation cost per iteration is  $O(DVK^2 + (D + V + T)K^3)$ . This follows from the underlying assumptions of Poisson distribution. When the observation is zero, the corresponding latent counts  $\{x_{tdwk}\}_{k=1}^K$  are zero with probability 1, and hence one needs to sample latent counts corresponding to non-zero entries only.

### 7.3.2.2 Gibbs Sampling for Dynamic Poisson Matrix Factor Model with Missing Entries

Variables whose update get affected in presence of missing values are  $r_{tk}$ 's and  $\theta_{dk}$ 's and  $\beta_{wk}$ 's. Rest of the update equations are same as in the dynamic Poisson matrix factor model without any missing value. Below, the updates are enlisted.  $\mathcal{M}_t$  denotes the set of missing entries in the matrix at the  $t^{\text{th}}$  time instance.

**Sampling of  $r_{tk}$  :** For  $t = T$ , sample

$$r_{Tk} | \sim \text{Gam}(r_{(T-1)k} + x_{T..k}, 1/(c + z_{Tk})), \quad (7.40)$$

where,

$$x_{T..k} = \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_T}}^{D,V} x_{Tdwk}, z_{Tk} = \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_T}}^{D,V} \theta_{dk} \beta_{wk}. \quad (7.41)$$

For  $t = (T - 1)$  augment  $\ell_{Tk} \sim \text{CRT}(x_{T..k}, r_{(T-1)k})$ . Then sample

$$r_{(T-1)k} | \sim \text{Gam} \left( r_{(T-2)k} + x_{(T-1)..k} + \ell_{Tk}, 1/(c + z_{(T-1)k} - \log(1 - p_{Tk})) \right), \quad (7.42)$$

where,

$$x_{(T-1)..k} = \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_{(T-1)}}}^{D,V} x_{(T-1)dwk}, z_{(T-1)k} = \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_{(T-1)}}}^{D,V} \theta_{dk} \beta_{wk}, p_{Tk} = \frac{z_{Tk}}{(c + z_{Tk})}. \quad (7.43)$$

For  $t = (T - 2)$  to  $t = 1$ , augment  $\ell_{(t+1)k} \sim \text{CRT}(x_{(t+1)..k} + \ell_{(t+2)k}, r_{tk})$ . Then sample  $r_{tk}$  as:

$$r_{tk} | \sim \text{Gam} \left( r_{(t-1)k} + x_{t..k} + \ell_{(t+1)k}, 1/(c + z_{tk} - \log(1 - p_{(t+1)k})) \right), \quad (7.44)$$

where,

$$x_{t..k} = \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_t}}^{D,V} x_{tdwk}, z_{tk} = \sum_{\substack{d=1, w=1 \\ (d,w) \notin \mathcal{M}_t}}^{D,V} \theta_{dk} \beta_{wk}, p_{(t+1)k} = \frac{s_{(t+1)k} - \log(1 - p_{(t+2)k})}{(c + s_{(t+1)k} - \log(1 - p_{(t+2)k}))}. \quad (7.45)$$

**Sampling of  $\theta_{dk}$  :**

$$\theta_{dk} | \sim \text{Gam} \left( d_0 + \sum_{t=1}^T \sum_{\substack{w=1 \\ (d,w) \notin \mathcal{M}_t}}^V x_{tdwk}, 1/(c_d + \sum_{t=1}^T \sum_{\substack{w=1 \\ (d,w) \notin \mathcal{M}_t}}^V r_{tk} \beta_{wk}) \right). \quad (7.46)$$

**Sampling of  $\beta_{wk}$  :**

$$\beta_{wk} | \sim \text{Gam} \left( h_0 + \sum_{t=1}^T \sum_{\substack{d=1 \\ (d,w) \notin \mathcal{M}_t}}^D x_{tdwk}, 1/(s_w + \sum_{t=1}^T \sum_{\substack{d=1 \\ (d,w) \notin \mathcal{M}_t}}^D r_{tk} \theta_{dk}) \right). \quad (7.47)$$

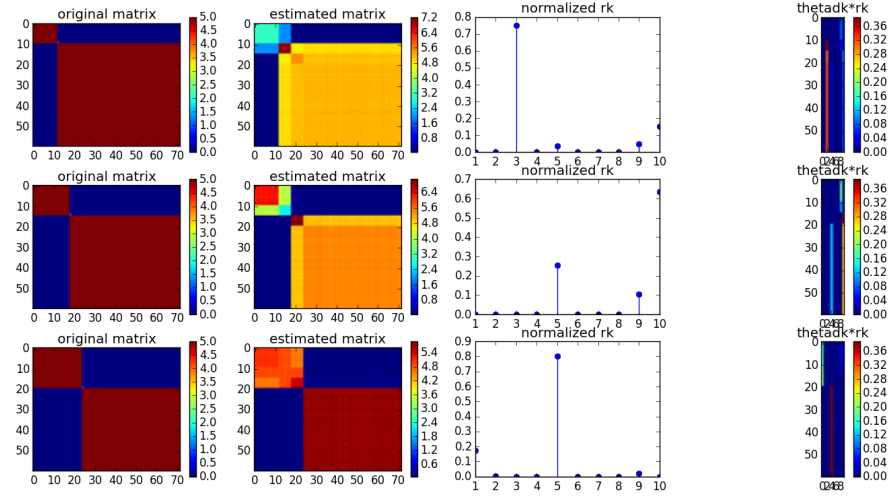


Figure 7.12: Results from D-CGPPF

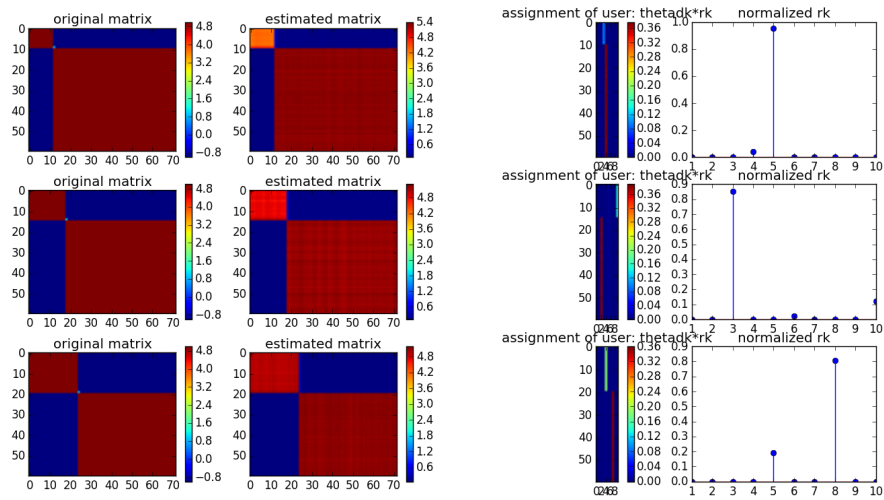


Figure 7.13: Results from C-GPPF

### 7.3.3 Experiments

#### 7.3.3.1 Synthetic Data

We generate a set of synthetic count matrices of size  $60 \times 72$  with two different groups that evolve over three different time stamps. These datasets are displayed in column (a) in both Fig. 7.12 and 7.13. In practice, this may represent a collection of users and movies where the preference of the users for certain movies change over time. The performance of D-CGPPF is displayed in columns (b), (c), and (d) of Fig. 7.12. Column (b) in Fig. 7.12 shows the groups discovered by D-CGPPF in the graph over different time-stamps. Note that the discovery of groups at any time instance is influenced by the groups present in other time instances. In column (c) of Fig. 7.12, the proximity of the users to the latent groups are displayed. The x-axis in each of these plots imply different latent groups and the y-axis represents the proximity of the  $n^{\text{th}}$  user to the  $k^{\text{th}}$  latent group at the  $t^{\text{th}}$  time instance, which is calculated as  $r_{tk}\theta_{dk}$ . In our experiments, 10 different latent groups are maintained ( $K = 10$ ), but the model assigns the users to only a few of the latent groups, a desired outcome. This observation is also reinforced by the plots in column (c) of Fig. 7.12. These plots denote the normalized weights of the different latent groups ( $r_{tk} / \sum_{k=1}^K r_{tk}$ ) at different time instances. In each time instance, only a few latent groups have positive weight. Expectedly, as displayed in columns (c) and (d) of Fig. 7.12, the latent factors that are dominant over different time instances vary smoothly with time. In Fig. 7.13, results are displayed for a baseline model that uses only N-GPPF for modeling the networks isolatedly at each different time slice. One can see that N-GPPF reconstructs the groups perfectly at each time instance as

the groups are very clear-cut. However, different sets of latent groups dominate in modeling the count matrices at different time slices, as revealed in plots of columns (c) and (d) of Fig. 7.13. Unlike this toy example, most real world dyadic datasets are sparse and groups are less distinct at any given time. The performance of a static count matrix factorization model is expected to be poorer in such settings, as it cannot link the solutions across time. This is explained more clearly alongside the results reported in the next subsection.

### 7.3.3.2 Real World Data

We now validate the performance of our model on three different movie rating datasets<sup>1,2</sup> popularly used in the recommender system literature.

- **MovieLens100k:** Movielens 100K is a movie rating dataset which contains 100k ratings provided by users, with 943 users and 1682 movies. Rating ranges from 0 to 5-star scale. 943 1682 and 8 different time slices.
- **MovieLens1M:** Movielens 1M is a movie rating dataset which contains 1 million ratings provided by users, with 6040 users and 3900 movies. Rating ranges from 0 to 5-star scale and over  $T = 35$  different time slices, each corresponding to different months.
- **Netflix:** Netflix also is a movie rating dataset which contains 100M ratings provided by users and is very skewed unlike the Movielens dataset, with 480189 users

---

<sup>1</sup><http://grouplens.org/datasets/movielens/>

<sup>2</sup> <http://www.netflixprize.com/>

and 17770 movies. Rating ranges from 0 to 5-star scale. Since this dataset is quite large, we sample a subset of this dataset and create a new training set consisting of around 3M ratings that belong to 93705 users, 3561 movies over 27 different time slices, each of which corresponds to ratings from a month.

To evaluate accuracy, we consider the recommendation problem as a ranking problem and use mean average precision (MAP) and Normalized Discounted Cumulative Gain (NDCG) as the metrics [Gopalan et al., 2013, 2014a]. For all the datasets, we measure the accuracy of prediction on a held out set consisting of 10% data instances randomly selected. Training is done on the remaining 90% of the data instances. During the training phase, the held out set is considered as missing data. We compare D-CGPPF with two strong baseline methods: BPTF [Xiong et al., 2010] and C-GPPF.

Dataset	D-CGPPF	BPTF	C-GPPF
Movielens100K	<b>0.597</b> $\pm$ 0.023	0.512 $\pm$ 0.010	0.238 $\pm$ 0.047
Movielens1M	<b>0.641</b> $\pm$ 0.010	0.632 $\pm$ 0.008	0.521 $\pm$ 0.019
Netflix	<b>0.490</b> $\pm$ 0.008	0.418 $\pm$ 0.002	0.251 $\pm$ 0.039

Table 7.2: MAP Results on Real World Data

Dataset	D-CGPPF	BPTF	C-GPPF
Movielens100K	<b>0.714</b> $\pm$ 0.016	0.703 $\pm$ 0.010	0.455 $\pm$ 0.012
Movielens1M	0.721 $\pm$ 0.013	<b>0.725</b> $\pm$ 0.013	0.585 $\pm$ 0.020
Netflix	<b>0.613</b> $\pm$ 0.007	0.592 $\pm$ 0.011	0.451 $\pm$ 0.018

Table 7.3: NDCG Results on Real World Data

## 7.4 Conclusion

This Chapter introduces the Dynamic Gamma Process Poisson Factorization framework for analyzing count and binary vectors and matrices that evolve over time. Efficient inference technique has been developed for modeling the temporal evolution of the latent components of the count matrix using a gamma Markov chain. Superior empirical performance on both synthetic and real world datasets makes the approach a promising candidate for modeling other count time-series data; for example, time-evolving tensors that appear quite frequently in analysis of electronic health records. In Section 9.1, we briefly describe how we can extend D-CGPPF for modeling count tensors that evolve over time. In the next Chapter, we will explore a different type of sequential knowledge transfer for a problem motivated from practical applications.

## Chapter 8

# Bayesian Combination of Classification and Clustering Ensembles

In several data mining applications, one builds an initial classification model that needs to be applied to unlabeled data acquired subsequently. Since the statistics of the underlying phenomena being modeled changes with time, these classifiers may also need to be occasionally rebuilt if performance degrades beyond an acceptable level. In such situations, it is desirable that the classifier functions well with as little labeling of new data as possible, since labeling can be expensive in terms of time and money, and a potentially error-prone process. Moreover, the classifier should be able to adapt to changing statistics to some extent, given the aforementioned constraints.

This chapter addresses the problem of combining multiple classifiers and clusterers in a fairly general setting, that includes the scenario sketched above. An ensemble of classifiers is first learnt on an initial labeled training dataset after which the training data can be discarded. Subsequently, when new unlabeled target data is encountered, a cluster ensemble is applied to it, thereby generating cluster labels for the target data. The heart of our approach is a Bayesian framework that combines both sources of information (class/cluster labels) to yield a consensus labeling of

the target data.

The setting described above is, in principle, different from transductive learning setups where both labeled and unlabeled data are available at the same time for model building [Silver and Bennett, 2008], as well as online methods [Blum, 1998]. Additional differences from existing approaches are described in the section on related works. For the moment we note that the underlying assumption is that similar new objects in the target set are more likely to share the same class label. Thus, the supplementary constraints provided by the cluster ensemble can be useful for improving the generalization capability of the resulting classifier system. Also, these supplementary constraints can be useful for designing learning methods that help determining differences between training and target distributions, making the overall system more robust against concept drift.

We also show that our approach can combine cluster and classifier ensembles in a privacy-preserving setting. This approach can be useful in a variety of applications. For example, the data sites can represent parties that are a group of banks, with their own sets of customers, who would like to have a better insight into the behavior of the entire customer population without compromising the privacy of their individual customers.

The remainder of the chapter is organized as follows. The next section addresses related work. The proposed Bayesian framework — named **BC<sup>3</sup>E**, from **B**ayesian **C**ombination of **C**lassifiers and **C**lusterer **E**nsembles — is described in Section 8.2. Issues with privacy preservation are discussed in Section 8.3 and the experimental results are reported in Section 8.4. Finally, Section 8.5 concludes the

chapter.

## 8.1 Related Work

The combination of multiple classifiers to generate an ensemble has been proven to be more useful compared to the use of individual classifiers [Oza and Tumer, 2008]. Analogously, several research efforts have shown that cluster ensembles can improve the quality of results as compared to a single clusterer — *e.g.*, see [Wang et al., 2011b] and references therein. Most of the motivations for combining ensembles of classifiers and clusterers are similar to those that hold for the standalone use of either classifier or cluster ensembles. Additionally, unsupervised models can provide supplementary constraints for classifying new data and thereby improve the generalization capability of the resulting classifier. These successes provide the motivation for designing effective ways of leveraging both classifier and cluster ensembles to solve challenging prediction problems.

Specific mechanisms for combining classification and clustering models however have been introduced only recently in the Bipartite Graph-based Consensus Maximization (**BGCM**) algorithm [Gao et al., 2009], the Locally Weighted Ensemble (**LWE**) algorithm [Gao et al., 2008] and, in the **C<sup>3</sup>E** algorithm [Acharya et al., 2011a]. Both **BGCM** and **C<sup>3</sup>E** have parameters that control the relative importance of classifiers and clusterers. In traditional semi-supervised settings, such parameters can be optimized via cross-validation. However, if the training and the target distributions are different, cross-validation is not possible. From this viewpoint, our approach (**BC<sup>3</sup>E**) can be seen as an extension of **C<sup>3</sup>E** [Acharya et al.,

2011a] that is capable of dealing with this issue in a more principled way. In addition, the algorithms in [Acharya et al., 2011a; Gao et al., 2008, 2009] do not deal with privacy issues, whereas our probabilistic framework can combine class labels with cluster labels under conditions where sharing of individual records across data sites is not permitted. It uses a soft probabilistic notion of privacy, based on a quantifiable information-theoretic formulation [Merugu and Ghosh, Nov, 2003]. Note that existing works on Bayesian classifier ensembles — *e.g.*, [Chipman et al., 2006; Edakunni and Vijayakumar, 2009; Ghahramani and Kim, 2003] — do not deal with privacy issues.

From the clustering side, the proposed model borrows ideas from the Bayesian Cluster Ensemble [Wang et al., 2011b]. In [Acharya et al., 2011b], we introduced some preliminary ideas that are further developed in our current chapter. In particular, the algorithm in [Acharya et al., 2011b] is not capable of automatically estimating the importance that classifiers and clusterers should have. This property is fundamental for applications where training and target distributions are different. In addition, the Bayesian model presented here is considerably different and requires more sophisticated inference and estimation procedures.

## 8.2 Probabilistic Model

We assume that a classifier ensemble has been (previously) induced from a training set. At this point and assuming a non-transductive setting, the training data can be discarded if so desired. Such a classifier ensemble is employed to generate a number of class labels (one from each classifier) for every object in the target

set. **BC<sup>3</sup>E** refines such classifier prediction with the help of a cluster ensemble. Each base clustering algorithm that is part of the ensemble partitions the target set, providing cluster labels for each of its objects. From this point of view, the cluster ensemble provides supplementary constraints for classifying those objects, with the rationale that similar objects — those that are likely to be clustered together across (most of) the partitions that form the cluster ensemble — are more likely to share the same class label.

Consider a target set  $\mathcal{X} = \{\mathbf{x}_n\}_{n=1}^N$  formed by  $N$  unlabeled objects. A classifier ensemble composed of  $r_1$  models has produced  $r_1$  class labels for every object  $\mathbf{x}_n \in \mathcal{X}$ . It is assumed that the target objects belong to  $k$  classes denoted by  $C = \{C_i\}_{i=1}^k$  and at least one object from each of these classes was observed in the training phase (*i.e.* we do not consider “novel” classes in the target set). Similarly, consider that a cluster ensemble comprised of  $r_2$  clustering algorithms has generated cluster labels for every object in the target set. The number of clusters need not be the same across different clustering algorithms. Also, it should be noted that the cluster labeled as  $l$  in a given data partition may not align with the cluster numbered  $l$  in another partition, and none of these clusters may correspond to class  $l$ . Given the class and cluster labels, the objective is to come up with refined class probability distributions  $\{(\hat{P}(C_i|\mathbf{x}_n))_{i=1}^k = \mathbf{y}_n\}_{n=1}^N$  of the target set objects. This framework is illustrated in Fig. 8.1.

The observed class and cluster labels are represented as  $\mathbf{W} = \{\{\mathbf{w}_{1nl}\}, \{\mathbf{w}_{2nm}\}\}$  where  $\mathbf{w}_{1nl}$  is the 1-of- $k$  representation of class label of the  $n^{\text{th}}$  object given by the  $l^{\text{th}}$  classifier, and  $\mathbf{w}_{2nm}$  is the 1-of- $k^{(m)}$  representation of cluster label assigned to

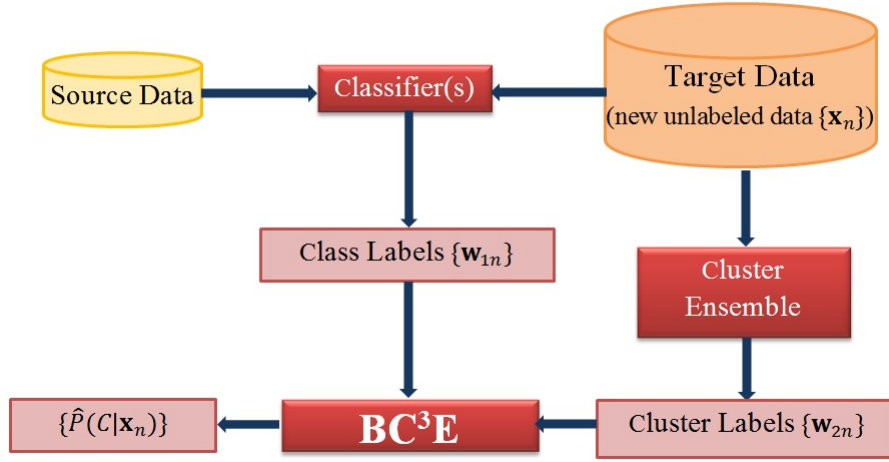


Figure 8.1: Combining Classifiers and Clusterers.

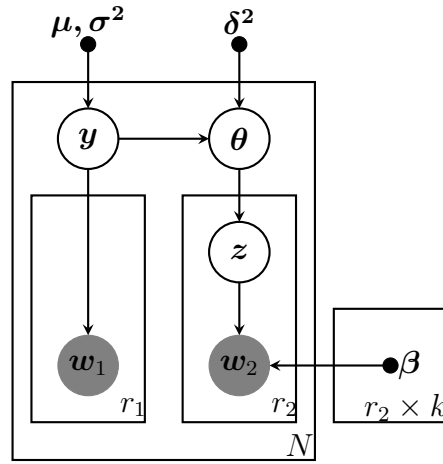


Figure 8.2: Graphical Model for **BC³E**

the  $n^{\text{th}}$  object by the  $m^{\text{th}}$  clusterer. A generative model is proposed to explain the observations  $\mathbf{W}$ , where each object  $\mathbf{x}_n$  has an underlying mixed-membership to the  $k$  different classes. Let  $f(\mathbf{y}_n)$  denote the latent mixed-membership vector for  $\mathbf{x}_n$ , where  $f(\mathbf{x}) = \frac{\exp(x_i)}{\sum_{i=1} \exp(x_i)}$  is the softmax function.  $\mathbf{y}_n$  is sampled from a normal distribution  $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ . Also, corresponding to the  $i^{\text{th}}$  class and  $m^{\text{th}}$  base clustering, we assume a multinomial distribution  $\beta_{mi}$  over the cluster labels of the  $m^{\text{th}}$  base clustering. Therefore,  $\beta_{mi}$  is of dimension  $k^{(m)}$  and  $\sum_{j=1}^{k^{(m)}} \beta_{mij} = 1$  if the  $m^{\text{th}}$  base clustering has  $k^{(m)}$  clusters. The data generative process, whose corresponding graphical model is shown in 8.2, can be summarized as follows.

For each  $\mathbf{x}_n \in \mathcal{X}$ :

1. Choose  $\mathbf{y}_n \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ , where  $\boldsymbol{\mu} \in \mathbb{R}^k$  is the mean and  $\boldsymbol{\Sigma} \in \mathbb{R}^{k \times k}$  is the covariance.
2. Choose  $\boldsymbol{\theta}_n \sim \mathcal{N}(\mathbf{y}_n, \delta^2 I_k)$ , where  $\delta^2 \geq 0$  is the scaling factor of the covariance of the normal distribution centered at  $\mathbf{y}_n$ , and  $I_k$  is the identity  $k \times k$  matrix.
3.  $\forall l \in \{1, 2, \dots, r_1\}$ , choose  $\mathbf{w}_{1nl} \sim f(\mathbf{y}_n)$ .
4.  $\forall m \in \{1, 2, \dots, r_2\}$ :
  - (a) Choose  $\mathbf{z}_{nm} \sim f(\boldsymbol{\theta}_n)$ , where  $\mathbf{z}_{nm}$  is a  $k$ -dimensional vector with 1-of- $k$  representation.
  - (b) Choose  $\mathbf{w}_{2nm} \sim \text{multinomial}(\boldsymbol{\beta}_{r\mathbf{z}_{nm}})$ .

The observed class labels  $\{\mathbf{w}_{1nl}\}$  are assumed to be sampled from the latent mixed-membership vector  $f(\mathbf{y}_n)$ . If the  $n^{\text{th}}$  object is sampled from the  $i^{\text{th}}$  class in the  $m^{\text{th}}$  base clustering (implying  $z_{nmi} = 1$ ), then its cluster label will be sampled from the multinomial distribution  $\beta_{mi}$ . This particular generative process is analogous to the one used by the Bayesian Cluster Ensemble in [Wang et al., 2011b]. The fact that  $\theta_n$  is sampled from  $\mathcal{N}(\mathbf{y}_n, \delta^2 I_k)$  needs further clarification. In practice, the observed class labels and cluster labels carry different intrinsic weights. If the observations from the classifiers are assigned too much weight compared to those from clustering, there is little hope for the clustering to enhance classification. Similarly, if the observations from the clustering are given too much of importance, the classification performance might deteriorate. Ideally, the unsupervised information is only expected to enhance the classification accuracy.

Aimed at building a “safe” model that can intelligently utilize or reject the unsupervised information,  $\theta_n$  is sampled from  $\mathcal{N}(\mathbf{y}_n, \delta^2 I_k)$  where the parameter  $\delta$  decides how much the observations from the clusterings can be trusted. If  $\delta^2$  is a large positive number,  $\mathbf{y}_n$  does not have to explain the posterior of  $\theta_n$ . From the generative model perspective, this means that the sampled value of  $\theta_n$  is not governed by  $\mathbf{y}_n$  anymore as the distribution has very large variance. On the other hand, if  $\delta^2$  is a small positive number,  $\mathbf{y}_n$  has to explain the posterior of  $\theta_n$  and hence the observations from the clustering. Therefore, the posteriors of  $\{\mathbf{y}_n\}$  are expected to get more accurate compared to the case if they only had to explain the classification results. A concrete quantitative argument for this intuitive statement will be presented later.

To address the log-likelihood function of  $\mathbf{BC}^3\mathbf{E}$ , let us denote the set of hidden variables by  $\mathbf{Z} = \{\{\mathbf{y}_n, \{\boldsymbol{\theta}_n\}, \{\mathbf{z}_{nm}\}\}\}$ . The model parameters can conveniently be represented by  $\zeta_0 = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \delta^2, \{\boldsymbol{\beta}_{mi}\}\}$ . The joint distribution of the hidden and observed variables can be written as:

$$p(\mathbf{X}, \mathbf{Z} | \zeta_0) = \prod_{n=1}^N p(\mathbf{y}_n | \boldsymbol{\mu}, \boldsymbol{\Sigma}) p(\boldsymbol{\theta}_n | \mathbf{y}_n, \delta^2 I_k) \prod_{l=1}^{r_1} p(w_{1nl} | f(\mathbf{y}_n)) \prod_{m=1}^{r_2} p(\mathbf{z}_{nm} | f(\boldsymbol{\theta}_n)) p(w_{2nm} | \boldsymbol{\beta}, \mathbf{z}_{nm}). \quad (8.1)$$

The inference and estimation is performed using Variational Expectation-Maximization (**VEM**) to avoid computational intractability due to the coupling between  $\boldsymbol{\theta}$  and  $\boldsymbol{\beta}$ .

## 8.2.1 Approximate Inference and Estimation:

### 8.2.1.1 Inference:

To obtain a tractable lower bound on the observed log-likelihood, we specify a fully factorized distribution to approximate the true posterior of the hidden variables:

$$q(\mathbf{Z} | \{\zeta_n\}_{n=1}^N) = \prod_{n=1}^N q(\mathbf{y}_n | \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) q(\boldsymbol{\theta}_n | \boldsymbol{\epsilon}_n, \boldsymbol{\Delta}_n) \prod_{m=1}^{r_2} q(\mathbf{z}_{nm} | \boldsymbol{\phi}_{nm}), \quad (8.2)$$

where  $\mathbf{y}_n \sim \mathcal{N}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$ ,  $\boldsymbol{\theta}_n \sim \mathcal{N}(\boldsymbol{\epsilon}_n, \boldsymbol{\Delta}_n)$ ,  $\mathbf{z}_{nm} \sim \text{Mult}(\boldsymbol{\phi}_{nm})$ .

$\zeta_n = \{\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n, \boldsymbol{\epsilon}_n, \boldsymbol{\Delta}_n, \{\boldsymbol{\phi}_{nm}\}\}$  is the set of variational parameters corresponding to the  $n^{\text{th}}$  object. Further,  $\boldsymbol{\mu}_n, \boldsymbol{\epsilon}_n \in \mathbb{R}^k$ ,  $\boldsymbol{\Sigma}_n, \boldsymbol{\Delta}_n \in \mathbb{R}^{k \times k} \forall n$  and  $\boldsymbol{\phi}_{nm} = (\phi_{nmi})_{i=1}^k \forall n, m$ ; where the components of the corresponding vectors are made explicit. To work with less parameters, all the covariance matrices are assumed to be diagonal. Therefore,  $\boldsymbol{\Sigma} = \text{diag}((\sigma_i)_{i=1}^k)$ ,  $\boldsymbol{\Sigma}_n = \text{diag}((\sigma_{ni})_{i=1}^k)$ , and  $\boldsymbol{\Delta}_n = \text{diag}((\delta_{ni})_{i=1}^k)$ . Using Jensen's inequality, a lower bound on the observed

log-likelihood can be derived as:

$$\log[p(\mathbf{X}|\zeta_0)] \geq \mathbf{E}_{q(\mathbf{Z})} [\log[p(\mathbf{X}, \mathbf{Z}|\zeta_0)]] + H(q(\mathbf{Z})) = \mathcal{L}(q(\mathbf{Z})) \quad (8.3)$$

where  $H(q(\mathbf{Z})) = -\mathbf{E}_{q(\mathbf{Z})}[\log[q(\mathbf{Z})]]$  is the entropy of the variational distribution  $q(\mathbf{Z})$ , and  $\mathbf{E}_{q(\mathbf{Z})}[\cdot]$  is the expectation w.r.t  $q(\mathbf{Z})$ .

Let  $\mathcal{Q}$  be the set of all distributions having a fully factorized form as given in (8.2). The optimal distribution that produces the tightest possible lower bound  $\mathcal{L}$  is given by:

$$q^* = \arg \min_{q \in \mathcal{Q}} \text{KL}(p(\mathbf{Z}|\mathbf{X}, \zeta_0) || q(\mathbf{Z})). \quad (8.4)$$

In equations (3), (5), (7), (9), (11), (12) and (13) in Table 8.1, the optimal values of the variational parameters that satisfy (8.4) are presented. Since the logistic normal distribution is not conjugate to multinomial, the update equations of all the parameters cannot be obtained in closed form. For the parameters that do not have a closed form solution for the update, we just present the part of the objective function that depends on the concerned parameter and some numeric optimization method has to be used for optimizing the lower bound. Since  $\phi_{nm}$  is a multinomial distribution, the updated values of the  $k$  components should be normalized to unity. Note that the optimal value of one of the variational parameters depends on the others and, therefore, an iterative optimization is adopted to minimize the lower bound till convergence is achieved.

4!

Update Equations	
$\phi_{nmi}^* \propto \exp\left(\epsilon_{ni} + \sum_{j=1}^{k^{(m)}} \beta_{mij} w_{2nmj}\right) \forall n, m, i. \quad (3)$	$\mu^* = \frac{1}{N} \sum_{n=1}^N \mu_n. \quad (4)$
$\kappa_n^* = \sum_{i=1}^k \exp(\mu_{ni} + \sigma_{ni}^2/2) \forall n. \quad (5)$	$\beta_{mij}^* \propto \sum_{n=1}^N \phi_{nmi} w_{2nmj} \forall j \in 1, 2, \dots, k^m. \quad (6)$
$\xi_n^* = \sum_{i=1}^k \exp(\epsilon_{ni} + \delta_{ni}^2/2) \forall n. \quad (7)$	$\delta^2 = \frac{1}{Nk} \sum_{n=1}^N \sum_{i=1}^k [(\epsilon_{ni} - \mu_{ni})^2 + \sigma_{ni}^2 + \delta_{ni}^2]. \quad (8)$
$\mathcal{L}_{[\delta_n^2]} = -\frac{1}{2} \sum_{i=1}^k \frac{\delta_{ni}^2}{\delta^2} - \frac{1}{2} \sum_{i=1}^k \log(\delta_{ni}^2) - \frac{r_2}{\xi_n} \sum_{i=1}^k \exp(\epsilon_{ni} + \delta_{ni}^2/2). \quad (9)$	$\mathcal{L}_{[\sigma^2]} = -\frac{N}{2} \sum_{i=1}^k \log(\sigma_i^2) - \frac{1}{2} \sum_{n=1}^N \sum_{i=1}^k \left[ \frac{\sigma_{ni}^2 + (\mu_{ni} - \mu_i)^2}{\sigma_i^2} \right]. \quad (10)$
$\mathcal{L}_{[\mu_n]} = -\frac{1}{2} \sum_{i=1}^k \sum_{n=1}^N \frac{(\mu_{ni} - \mu_i)^2}{\sigma_i^2} - \frac{1}{2\delta^2} \sum_{i=1}^k (\mu_{ni} - \epsilon_{ni})^2 + \sum_{l=1}^{r_1} \sum_{i=1}^k w_{lni} \mu_{ni} - \frac{r_1}{\xi_n} \sum_{i=1}^k \exp(\mu_{ni} + \sigma_{ni}^2/2). \quad (11)$	
$\mathcal{L}_{[\sigma_n^2]} = -\frac{1}{2} \sum_{i=1}^k \sum_{n=1}^N \frac{\sigma_{ni}^2}{\sigma_i^2} - \frac{1}{2} \sum_{i=1}^k \log(\sigma_{ni}^2) - \frac{1}{2} \sum_{i=1}^k \frac{\sigma_{ni}^2}{\delta^2} - \frac{r_1}{\kappa_n} \sum_{i=1}^k \exp(\mu_{ni} + \sigma_{ni}^2/2). \quad (12)$	
$\mathcal{L}_{[\epsilon_n]} = \sum_{m=1}^{r_2} \sum_{i=1}^k \phi_{nmi} \epsilon_{ni} - \frac{1}{\xi_n} \sum_{i=1}^k \exp(\epsilon_{ni} + \delta_{ni}^2/2) - \frac{1}{2} \sum_{i=1}^k \frac{(\epsilon_{ni} - \mu_{ni})^2}{\delta^2}. \quad (13)$	

Table 8.1: Equations for update of variational and model parameters in **BC<sup>3</sup>E**

Equations (5) and (7) present updates for two new parameters. These parameters come from  $\mathbb{E}_q(\log p(\mathbf{w}_{1nl}|f(\mathbf{y}_n)))$  and  $\mathbb{E}_q(\log p(\mathbf{z}_{nm}|f(\boldsymbol{\theta}_n)))$  respectively. Both of these integrations do not have analytic solution and hence a first order Taylor approximation is utilized as also done in [Blei and Lafferty, 2007]. A closer inspection of (11) reveals that  $\delta^2$  appears in the denominator of the term  $\sum_{i=1}^k (\mu_{ni} - \epsilon_{ni})^2 / \delta^2$  in the objective. Hence, larger values of  $\delta^2$  will nullify any effect from  $\epsilon_n$  which, in turn, is affected by the observations  $\{\mathbf{w}_{2nm}\}$  (as is obvious from (13)). On the other hand, if  $\delta^2$  is small enough,  $\epsilon_n$  can strongly impact the values of  $\boldsymbol{\mu}_n$ .

#### 8.2.1.2 Estimation:

For estimation, we maximize the optimized lower bound obtained from the variational inference w.r.t the free model parameters  $\zeta_0$  (by keeping the variational parameters fixed). The optimal values of the model parameters are presented in equations (4), (6) and (8). Since  $\beta_{mi}$  is a multinomial distribution, the updated values of  $k^{(m)}$  components should be normalized to unity. However, no closed form of update exists for  $\sigma^2$ , and a numeric optimization method has to be resorted to. The part of the objective function that depends on  $\sigma^2$  is provided in Eq. (10). Once the optimization in M-step is done, E-step starts and the iterative update is continued till convergence. The variational parameters  $\{\boldsymbol{\mu}_n\}_{n=1}^N$  are then investigated which serve as proxy for the refined posterior estimates of  $\{\mathbf{y}_n\}_{n=1}^N$ . The main steps of inference and estimation are concisely presented in Algorithm 2.

---

**Algorithm 2** Learning BC<sup>3</sup>E

---

**Input:**  $W$ .

**Output:**  $\theta^m, \{\mu_n\}_{n=1}^N$ .

Initialize  $\theta^m, \{\zeta_n\}_{n=1}^N$ .

Until Convergence

**E-Step**

Until Convergence

1. Update  $\kappa_n$  using Eq. (5)  $\forall n \in \{1, 2, \dots, N\}$ .
2. Update  $\xi_n$  using Eq. (7)  $\forall n \in \{1, 2, \dots, N\}$ .
3. Update  $\phi_{nmi}$  using Eq. (3)  $\forall n, m, i$ . Normalize  $\phi_{nm}$ .
4. Maximize (11) w.r.t.  $\mu_n \forall n$ .
5. Maximize (12) w.r.t.  $\sigma_n^2 \forall n$  s.t.  $\sigma_n^2 \geq 0$ .
6. Maximize (13) w.r.t.  $\epsilon_n \forall n$ .
7. Maximize (9) w.r.t.  $\delta_n^2 \forall n$  s.t.  $\delta_n^2 \geq 0$ .

**M-Step**

8. Update  $\mu$  using Eq. (4).
  9. Update  $\delta^2$  using Eq. (8).
  10. Update  $\beta_{mij}$  using Eq. (6)  $\forall m, i, j$ . Normalize  $\theta_{mi}$ .
  11. Maximize (10) w.r.t.  $\sigma^2$  s.t.  $\sigma^2 \geq 0$ .
- 

### 8.3 Privacy Preserving Learning

Most of the privacy-aware distributed data mining techniques developed so far have focused on classification or on association rules [Agrawal and Aggarwal, 2001; Evfimievski et al., 2002]. There has also been some work on distributed clustering for *vertically partitioned data* (different sites contain different attributes/features of a common set of records/objects) [Johnson and Kargupta, 1999], and on parallelizing clustering algorithms for *horizontally partitioned data* (i.e. the objects are distributed amongst the sites, which record the same set of features for each object) [Dhillon and Modha, 1999]. These techniques, however, do not specifically address privacy issues, other than through encryption [Vaidya and Clifton, 2003].

This is also true of earlier, data-parallel methods [Dhillon and Modha, 1999] that are susceptible to privacy breaches, and also need a central planner that dictates what algorithm runs on each site. Finally, recent works on distributed differential privacy focus on query processing rather than data mining [Chen et al., 2012].

In the sequel, we show that the inference and estimation in **BC<sup>3</sup>E** using **VEM** allows solving the cluster ensemble problem in a way that preserves privacy. Depending on how the objects with their cluster/class labels are distributed in different “data sites”, we can have three scenarios – i) Row Distributed Ensemble, ii) Column Distributed Ensemble, and iii) Arbitrarily Distributed Ensemble.

### 8.3.1 Row Distributed Ensemble:

In the row distributed ensemble learning framework, the test set  $\mathcal{X}$  is partitioned into  $D$  parts and different parts are assumed to be at different locations. The objects from partition  $d$  are denoted by  $\mathcal{X}_d$  so that  $\mathcal{X} = \cup_{d=1}^D \mathcal{X}_d$ . Now, a careful look at the E-step equations reveal that the update of variational parameters corresponding to each object in a given iteration is independent of those of other objects. Therefore, we can maintain a client-server based framework where the server only updates the model parameters (in the M-step) and the clients (there should be as many number of clients as there are distributed data sites) update the variational parameters.

For instance, consider a situation where a dataset is partitioned into two subsets  $\mathcal{X}_1$  and  $\mathcal{X}_2$  and these two subsets are located in two different data sites. Data site 1 has access to  $\mathcal{X}_1$  and a set of clustering and classification results pertaining

to objects belonging to  $\mathcal{X}_1$ . Similarly, data site 2 has access to  $\mathcal{X}_2$  and a set of clustering and classification results corresponding to  $\mathcal{X}_2$ . Further assume that a set of distributed classification (clustering) algorithms were used to generate the class (cluster) labels of the objects belonging to each set. Now, data site 1 can update the variational parameters  $\zeta_n, \forall \mathbf{x}_n \in \mathcal{X}_1$ . Similarly, data site 2 can update the variational parameters for all objects  $\mathbf{x}_n \in \mathcal{X}_2$ . Once the variational parameters are updated in the E-step, the server gathers information from two sites and updates the model parameters. Now, a closer inspection of the M-step update equations reveals that each of them contains a summation over the objects. Therefore, individual data sites can send only some collective information to the server without transgressing privacy. For example, consider the update equation for  $\beta_{mij}$ . Eq. (6) can be broken as follows:

$$\beta_{mij}^* \propto \sum_{\mathbf{x}_n \in \mathcal{X}_1} \phi_{nli} w_{2nli} + \sum_{\mathbf{x}_n \in \mathcal{X}_2} \phi_{nli} w_{2nli} \quad (8.14)$$

The first and second terms can be calculated in data sites 1 and 2 separately and sent to the server where the two terms can be added and  $\beta_{mij}$  can get updated  $\forall m, i, j$ . Similarly, the other M-step update equations (performed by the server in an analogous way) also do not reveal any information about class or cluster labels of objects belonging to different data sites.

### 8.3.2 Column Distributed Ensemble:

In the column distributed framework, different data sites share the same set of objects but only a subset of base clusterings or classification results are available to each data site. For example, consider that we have two data sites and four sets

of class and cluster labels and each data site has access to only two sets of classification or clustering results. Assume that data site 1 has access to the 1<sup>st</sup> and 2<sup>nd</sup> classification and clustering results and data site 2 has access to the rest of the results. As in the earlier case, a single server and two clients (corresponding to two different data sites) are maintained. Since each data site has access to all the objects, it is necessary to share the variational parameters corresponding to these objects. Therefore,  $\{\kappa_n, \xi_n, \boldsymbol{\mu}_n, \boldsymbol{\sigma}_n, \boldsymbol{\epsilon}_n, \boldsymbol{\delta}_n\}_{n=1}^N$  are all updated in the server (which is accessible from each client).

The site (and object) specific variational parameters  $\{\phi_{nmi}\}$ , however, cannot be shared and should be updated in individual sites. This means that the updates (5), (7), (11), (13), (9) and (12) should be performed in the server. On the other hand, the update for  $\{\phi_{nmi}\} \forall n, i$  and  $m \in \{1, 2\}$  (corresponding to the 1<sup>st</sup> and 2<sup>nd</sup> clustering or classification results) should be performed in data site 1. Similarly, the update for  $\{\phi_{nmi}\} \forall n, i$  and  $m \in \{3, 4\}$  has to be performed in data site 2. However, while updating  $\{\boldsymbol{\mu}_n\}$ , the calculation of the term  $\sum_{l=1}^{r_1} \sum_{i=1}^k w_{1nli} \mu_{ni}$  has to be performed without revealing the class labels  $\{\boldsymbol{w}_{1nl}\}$  to the server. To that end, it can be rewritten as:

$$\sum_{l=1}^{r_1} \sum_{i=1}^k w_{1nli} \mu_{ni} = \sum_{l=1}^2 \sum_{i=1}^k w_{1nli} \mu_{ni} + \sum_{l=3}^4 \sum_{i=1}^k w_{1nli} \mu_{ni}, \quad (8.15)$$

where the first term can be computed in data site 1 and the second term can be computed by data site 2 and then can be added in the server. It can be seen that  $\{\boldsymbol{w}_{1nl}\}$  can never be recovered by the server and hence privacy is ensured in the updates of the E-step. Except for  $\{\beta_{mij}\}$ , all other model parameters can be updated

in the server in the M-step. However, the parameters  $\{\beta_{mij}\}$  have to be updated separately inside the clients. Since  $\{\beta_{mij}\}$  do not appear in any update equation performed in the server, there is no need to send these parameters to the server either. Therefore, in essence, the clients update the parameters  $\{\phi_{nmi}\}$  and  $\{\beta_{mij}\}$  in E-step and M-step respectively, and the server updates the remaining parameters.

### 8.3.3 Arbitrarily Distributed Ensemble:

In an arbitrarily distributed ensemble, each data site has access to only a subset of the data points or a subset of the classification and clustering results. Fig. 8.3 shows a situation with arbitrarily distributed ensemble with six data sites.

We now refer to Fig. 8.4 and explain the privacy preserved EM update for this setting. As before, corresponding to each different data site, a client node is created. Clients that share a subset of the objects should have access to the variational parameters corresponding to common objects. To highlight the sharing of objects by clients, the test set  $\mathcal{X}$  is partitioned into four subsets —  $\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3$  and  $\mathcal{X}_4$  as shown in Fig. 8.3. Similarly, the columns are also partitioned into three subsets:  $G_1, G_2$ , and  $G_3$ .

Now, corresponding to each row partition, an “Auxiliary Server”(AS) node is created. Each AS updates the variational parameters corresponding to a set of shared objects. For example, in Fig. 8.4,  $AS_1$  updates the variational parameters corresponding to  $\mathcal{X}_1$  (using equations (7), (5), (11), (12), (13), and (9)). However, any variational parameter that is specific to both an object and a column is updated separately inside the corresponding client (and hence it is connected with  $C_1$

and  $C_2$ ). Therefore,  $\{\phi_{nmi} : n \in \mathcal{X}_1, m \in G_1\}$  are updated inside client 1 and  $\{\phi_{nmi} : n \in \mathcal{X}_1, m \in G_2 \cup G_3\}$  are updated inside client 2 (using Eq. (3)). Once all variational parameters are updated in the E-step, M-step starts. Corresponding to each column partition, an “Auxiliary Client” (AC) node is created. This node updates the model parameters  $\beta_{mij}$  (using Eq. (6)) which are specific to columns belonging to  $G_1$ . Since  $C_1, C_3$ , and  $C_5$  share the columns from the subset  $G_1$ ,  $AC_1$  is connected with these three nodes in Fig. 8.4. The remaining model parameters are, however, updated in a “Server” (using equations (4), (8), (10)).

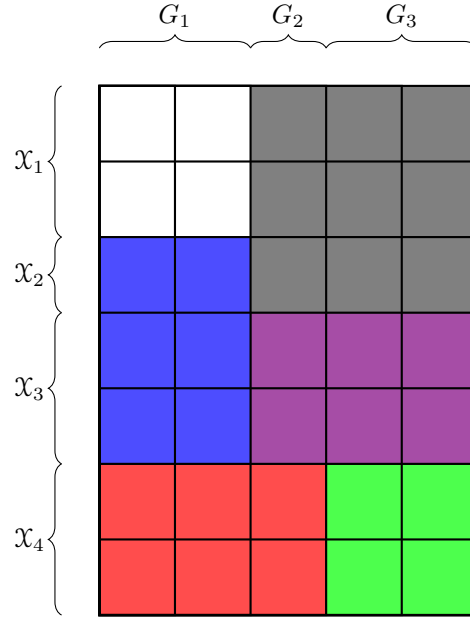


Figure 8.3: Arbitrarily Distributed Ensemble

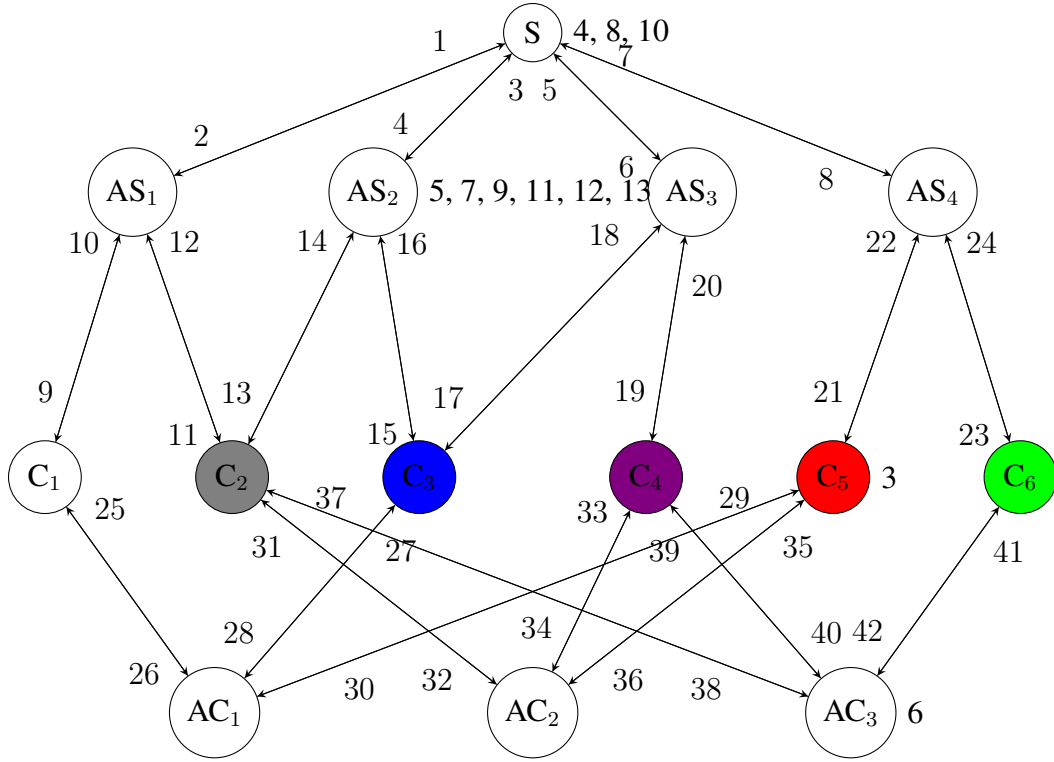


Figure 8.4: Parameter Update for Arbitrarily Distributed Ensemble

In Fig. 8.4, the bidirectional edges indicate that messages are sent to and from the connecting nodes. We have avoided separate arrows for each direction only to keep the figure uncluttered. The edges are also numbered near to their origin. For a comprehensive understanding of the privacy preservation, the messages transferred through each edge have also been enlisted in the supplementary material. The messages sent from the auxiliary servers to the main server are of the form given in Eq. (8.14) and are denoted as “partial sums”. Expectedly, messages sent out from a client node are “masked” in such a way that no other node can decode the cluster labels or class labels of points belonging to that client. This approach is

completely general and will work for any arbitrarily partitioned ensemble given that each partition contains at least two sets of classification results. Note that the ACs and ASs are only helpful in conceptual understanding of the parameter update and sharing. In practice, there is no real need for these extra storage devices/locations. Client nodes can themselves take the place of ASs and ACs and even the main server as long as the updates are performed in proper sequence<sup>1</sup>.

## 8.4 Experiments

In this section, two different sets of experiments are reported. The first set is for transfer learning with a text classification data from eBay Inc. The other set is for non-transductive semisupervised learning where some publicly available datasets are used to simulate the working environment of **BC<sup>3</sup>E**.

### 8.4.1 Transfer Learning:

To show the capability of **BC<sup>3</sup>E** in solving transfer learning problems, we use a large scale text classification dataset from eBay Inc. The training data consists of 83 million items sold over a three month period of time and the test set contains several millions of items sold a few days after the training period. More details about the dataset can be found in [Shen et al., 2012]. eBay organizes items into a six-level category structure where there are 39 top level nodes called *meta categories* and 20K+ bottom level nodes called *leaf categories*. The dataset is gen-

---

<sup>1</sup>Note that such framework allows running the updates of the same stage in parallel in different sites, thereby saving the computation time in large scale implementations.

erated when users provide the titles of items they intend to sell on eBay. Each title is limited to 50 characters, based on which the user gets recommendation of some *leaf categories* the item should belong to. Such categorization of the item helps a seller list an item in the correct branch of the product list, thereby allowing a buyer more easily search through a list of few million items sold via eBay every single day. A carefully designed  $k$ -Nearest Neighbor ( $k$ -NN) classifier (with the help of improved search engine algorithms) categorizes each of the items in less than 100 ms [Shen et al., 2012]. However, due to the large number of categories (20K), items belonging to similar types of categories often get misclassified.

To avoid such confusion, larger categories are formed by aggregating examples from categories which are relatively difficult to separate. Such aggregation is easy once the confusion matrix of the classification, obtained from a development dataset, is partitioned and strongly connected vertices (each vertex representing one of 20K *leaf categories*) are identified from the confusion graph, thereby forming a set of cliques which represent the large categories. Note that the large categories so discovered might not at all follow the internal hierarchy that is maintained. Next, clustering is performed with examples belonging to each of the large categories and the clustering results, along with the predictions from  $k$ -NN classification, are fed to **BC<sup>3</sup>E** (and also to its competitors *i.e.* **C<sup>3</sup>E**, **BGCM**, and **LWE**). The idea here is to first reduce the classification space and then use unsupervised information to refine the predictions from  $k$ -NN on a smaller number of categories. The number of *leaf categories* belonging to such large categories usually varies between 4-10.

However, the dataset is very dynamic and, typically over a span of three

months, 20% of new words are added to the existing vocabulary. One can retrain the existing  $k$ -NN classifier every three months, but the training process requires collecting new labeled data which is time consuming and expensive. One can additionally design classifiers to segregate examples belonging to each of the large categories. However, such approach might not improve much upon the performance of the initial  $k$ -NN classifier if the data changes so frequently. Therefore, we require a system that can adaptively predict newer examples without retraining the existing classifier or employing another set of classification algorithms. **BC<sup>3</sup>E** is useful in such settings. The parameter  $\delta$  can adjust the weights of prediction from classifiers and unsupervised information. As the results reported in Table 8.2 reveal, as long as the classification performance is not that poor, **BC<sup>3</sup>E** can improve on the performance of  $k$ -NN using the clustering ensemble.

The column “Group ID” denotes anonymized groups representing different large categories.  $|\mathcal{X}|$  shows the number of examples in the test data. The column “C<sup>3</sup>E-Ideal” shows the performance of C<sup>3</sup>E if the correct tuning parameter for C<sup>3</sup>E were known. For a transfer learning problem, estimating such tuning parameter requires some labeled data from the target set which is not available in our setting. If the tuning parameter is chosen from cross-validation on the training data, the final prediction on target set can get affected adversely if the underlying distribution changes (and in fact it does in our experiments). Therefore, we need to adopt a fail-safe approach where we can do at least as good as the  $k$ -NN prediction. The results reveal that **BC<sup>3</sup>E** significantly outperforms **BGCM** and **LWE**, and sometimes achieves as good a performance as C<sup>3</sup>E-Ideal (*i.e.* when correct tuning parameter

Group ID	$ \mathcal{X} $	$k$ -NN	BGCM	LWE	C <sup>3</sup> E-Ideal	BC <sup>3</sup> E
42	1299	64.90	73.78 ( $\pm 0.94$ )	76.86 ( $\pm 1.01$ )	83.99 ( $\pm 0.41$ )	83.68 ( $\pm 1.09$ )
84	611	63.67	69.23 ( $\pm 0.17$ )	75.24 ( $\pm 0.26$ )	81.18 ( $\pm 0.16$ )	76.27 ( $\pm 1.31$ )
86	2381	77.66	84.33 ( $\pm 2.74$ )	83.29 ( $\pm 1.02$ )	92.78 ( $\pm 0.35$ )	87.20 ( $\pm 0.91$ )
67	789	72.75	72.75 ( $\pm 0.07$ )	78.03 ( $\pm 0.72$ )	82.64 ( $\pm 0.82$ )	81.75 ( $\pm 1.37$ )
52	1076	76.95	77.01 ( $\pm 1.18$ )	77.49 ( $\pm 1.41$ )	88.38 ( $\pm 0.22$ )	85.04 ( $\pm 2.14$ )
99	827	84.04	85.12 ( $\pm 0.52$ )	86.90 ( $\pm 0.92$ )	91.54 ( $\pm 0.27$ )	91.17 ( $\pm 0.82$ )
48	3445	86.33	86.19 ( $\pm 0.25$ )	90.38 ( $\pm 1.03$ )	92.71 ( $\pm 0.31$ )	92.71 ( $\pm 1.16$ )
94	440	79.32	81.08 ( $\pm 0.73$ )	82.52 ( $\pm 0.83$ )	85.45 ( $\pm 0.09$ )	85.45 ( $\pm 0.79$ )
35	4907	82.41	82.10 ( $\pm 0.37$ )	85.08 ( $\pm 1.39$ )	88.16 ( $\pm 0.17$ )	88.22 ( $\pm 1.21$ )
45	1952	74.80	73.12 ( $\pm 0.81$ )	73.64 ( $\pm 1.68$ )	84.32 ( $\pm 0.23$ )	77.97 ( $\pm 0.47$ )

Table 8.2: Performance of **BC<sup>3</sup>E** on text classification data — Avg. Accuracies  $\pm$ (Standard Deviations).

of  $\mathbf{C^3E}$  is known). The performance of  $\mathbf{C^3E}$ -Ideal can essentially be considered as the best accuracy one could achieve from the given inputs (*i.e.* class and cluster labels) using other existing algorithms — **BGCM**, **LWE**,  $\mathbf{C^3E}$  — that work on the same design space. Though **BGCM** has a tuning parameter, its variation did not affect performance much and we just report results corresponding to unity value of this parameter.

#### 8.4.2 Semi-supervised Learning:

Six datasets are used in our experiments for semi-supervised learning: *Half-Moon* (a synthetic dataset with two half circles representing two classes), *Circles* (another synthetic dataset that has two-dimensional instances that form two concentric circles — one for each class), and four datasets from the *Library for Support Vector Machines* — *Pima Indians Diabetes*, *Heart*, *German Numer*, and *Wine*. In order to simulate semi-supervised settings where there is a very limited amount of labeled instances, small percentages (see the values reported in Table 8.3) of the instances are randomly selected for training, whereas the remaining instances are used for testing (target set). We perform 20 trials for every dataset. For running experiments with **BGCM**, and  $\mathbf{C^3E}$ , the parameters reported in [Gao et al., 2009] and [Acharya et al., 2014a] are used respectively. The parameters of  $\mathbf{BC^3E}$  are initialized randomly and approximately 10 EM iterations are enough to get the results reported in Table 8.3. The classifier ensemble consists of decision tree (C4.5), linear discriminant, and generalized logistic regression. Cluster ensembles are generated by means of multiple runs of *k*-means [Acharya et al., 2014a]. **LWE** [Gao et al.,

2008] is better suited for transfer learning applications and hence has been left out from comparison. The column “Best” in Table 8.3 refers to the performance of the best classifier in the ensemble. Note that **BC<sup>3</sup>E** has superior performance for the most difficult problems, where one has an incentive to use a more complex mechanism. Most importantly, **BC<sup>3</sup>E** has the privacy preserving property not present in any of its counterparts.

## 8.5 Conclusion

The **BC<sup>3</sup>E** model proposed in this chapter has been shown to be useful for difficult non-transductive semisupervised and transfer learning problems. A good trade-off between accuracy and privacy has also been established empirically – a property absent in any of **BC<sup>3</sup>E**’s competitors. With minor modification, **BC<sup>3</sup>E** can also handle soft outputs from classification and clustering ensembles which can further improve the results.

Dataset (% of tr. data)	$ \mathcal{X} $	Ensemble	Best	BGCM	C <sup>3</sup> E	BC <sup>3</sup> E
Half-moon(2%)	784	92.53( $\pm 1.83$ )	93.02( $\pm 0.82$ )	92.16( $\pm 1.47$ )	<b>99.64</b> ( $\pm 0.08$ )	98.23( $\pm 2.03$ )
Circles(2%)	1568	60.03( $\pm 8.44$ )	95.74( $\pm 5.15$ )	78.67( $\pm 0.54$ )	<b>99.61</b> ( $\pm 0.83$ )	97.91( $\pm 0.74$ )
Pima(2%)	745	68.16( $\pm 5.05$ )	69.93( $\pm 3.68$ )	69.21( $\pm 4.83$ )	70.31( $\pm 4.44$ )	<b>72.83</b> ( $\pm 0.49$ )
Heart(7%)	251	77.77( $\pm 2.55$ )	79.22( $\pm 2.20$ )	82.78( $\pm 4.82$ )	<b>82.85</b> ( $\pm 5.25$ )	82.53( $\pm 1.14$ )
G. Numer(10%)	900	70.96( $\pm 1.00$ )	70.19( $\pm 1.52$ )	73.70( $\pm 1.06$ )	74.44( $\pm 3.44$ )	<b>74.61</b> ( $\pm 1.62$ )
Wine(10%)	900	79.87( $\pm 5.68$ )	80.37( $\pm 5.47$ )	75.37( $\pm 13.66$ )	<b>83.62</b> ( $\pm 6.27$ )	82.20( $\pm 1.07$ )

Table 8.3: Comparison of **BC<sup>3</sup>E** with **C<sup>3</sup>E** and **BGCM** — Avg. Accuracies  $\pm$ (Standard Deviations).

## Chapter 9

### Future Work

Some of the specific future works are already listed in the corresponding chapters. Below, we present two other works which are potential extensions of the nonparametric Gamma-Poisson factorization framework.

#### 9.1 Dynamic Count Tensor Factorization

Automatic analysis of electronic health records (EHRs) has ushered in the era of data-driven approaches for improved clinical research, prognosis, and patient management. Unfortunately, EHR data do not always reliably represent some medical concepts that the clinical researchers are familiar with. Some recent studies have focused on EHR-derived phenotyping [Ho et al., 2014a,b], which aim at mapping the EHR data to specific medical concepts without human supervision. In particular, these works represent the interaction among the patients, diagnosis and medications in the form of a count tensor and discover the phenotypes using a low rank decomposition of this count tensors where the factors are sufficiently sparse for human experts to interpret. Often, such tensors evolve over time, representing changes in patients' medical condition. The generative process of a potential model that can track the changes in patients' medical condition in a latent space is provided below:  $y_{td_1d_2d_3} \sim \text{Pois}(\sum_k \theta_{td_1k} \phi_{d_2k} \beta_{d_3k})$ , where  $\theta_{td_1k} \sim \text{Gam}(\theta_{(t-1)d_1k}, 1/c)$  and

$y_{td_1 d_2 d_3}$  is the interaction of the  $d_1^{\text{th}}$  patient,  $d_2^{\text{th}}$  medication and  $d_3^{\text{th}}$  diagnosis at time  $t$ . We assume  $\phi_{d_2 k} \sim \text{Gam}(a_{d_2}, 1/b_k)$  and  $\beta_{d_3 k} \sim \text{Gam}(e_{d_3}, 1/f_k)$ . Here,  $\theta_{td_1}$  represents the phenotype of the  $d_1^{\text{th}}$  patient at time  $t$ , while  $\phi_{d_2}$  represents the phenotype of the  $d_2^{\text{th}}$  medication and  $\beta_{d_3}$  represents the phenotype of the  $d_3^{\text{th}}$  diagnosis. One can additionally impose Gamma priors on the scale and shape parameters of these Gamma distributions. Inference in such model is straightforward based on Lemma 2.1.1, 2.1.2, and 2.2.2.

## 9.2 Distributed Count Matrix Factorization

Another important implication of Lemma 2.2.2 is that one can share some of the latent factors across multiple groups in a hierarchical fashion and can still perform closed form Gibbs sampling. This is useful when there is a need for hierarchical modeling in an application of distributed matrix factorization. Consider multiple sites denoted by  $S_1, \dots, S_M$  each of which contains a set of documents, conveniently represented as a set of count matrices  $\{\mathbf{Y}_m\}_{m=1}^M$ . For distributed Poisson matrix factorization, one can maintain a global set of  $r_k$ 's, which represent the overall strength of the  $k^{\text{th}}$  topic, and  $\beta_{wk}$ 's which are the global set of topics. Corresponding to the site  $S_m$ , one can generate  $r_{mk} \sim \text{Gam}(r_k, 1/c_m)$  and  $\beta_{mwk} \sim \text{Gam}(\beta_{wk}, 1/e_m)$ . The  $(d, w)^{\text{th}}$  entry in the site  $S_m$  can be generated as:  $y_{mdw} \sim \text{Pois}(\sum_k r_{mk} \theta_{dk} \beta_{mwk})$ , where  $\theta_{dk} \sim \text{Gam}(a_d, 1/b_m)$ . We can put Gamma priors over  $r_k$  and  $\beta_{wk}$  as:  $r_k \sim \text{Gam}(\gamma_0/K, 1/c)$  and  $\beta_{wk} \sim \text{Gam}(f_w, 1/g_k)$ . One can also impose Gamma priors on the parameters of these Gamma distributions.

## Bibliography

- A. Acharya, E. R. Hruschka, J. Ghosh, and S. Acharyya. C<sup>3</sup>E: A Framework for Combining Ensembles of Classifiers and Clusterers. In *Proc. of 10th Int. Workshop on MCS*, 2011a.
- A. Acharya, E.R. Hruschka, and J. Ghosh. A privacy-aware bayesian approach for combining classifier and cluster ensembles. In *SocialCom/PASSAT*, pages 1169–1172, 2011b.
- A. Acharya, R.J. Mooney, and J. Ghosh. Active multitask learning using doubly supervised latent dirichle allocation. In *NIPS 13 Workshop on Topic Models*, 2013a.
- A. Acharya, A. Rawal, R. J. Mooney, and E. R. Hruschka. Using both supervised and latent shared topics for multitask learning. In *Proc. of ECML PKDD, Part II, LNAI 8189*, pages 369–384, 2013b.
- A. Acharya, E.R. Hruschka, J. Ghosh, and S. Acharyya. An optimization framework for combining ensembles of classifiers and clusterers with applications to nontransductive semisupervised learning and transfer learning. *ACM Trans. Knowl. Discov. Data*, 9:1:1–1:35, August 2014a.
- A. Acharya, R. J. Mooney, and J. Ghosh. Active multitask learning using both latent and supervised shared topics. In *Proc. of SDM*, Philadelphia, Pennsylvania, April 2014b.
- A. Acharya, J. Ghosh, and M. Zhou. Nonparametric Bayesian Factor Analysis for Dynamic Count Matrices. In *Proc. of AISTATS*, pages 1–9. 2015.
- R. P. Adams, I. Murray, and D. J. C. MacKay. Tractable nonparametric bayesian inference in poisson processes with gaussian process intensities. In *Proc. of ICML*, pages 9–16, 2009.

- D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Symposium on Principles of Database Systems*, 2001.
- M. Aharon, M. Elad, and A. M. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Trans. Signal Process.*, 2006.
- A. Ahmed and E. Xing. Dynamic non-parametric mixture models and the recurrent chinese restaurant process : with applications to evolutionary clustering. *Proc. of SDM*, 2008.
- A. Ahmed and E. Xing. Timeline: A dynamic hierarchical dirichlet process model for recovering birth/death and evolution of topics in text stream. *Proc. of UAI*, 2010.
- E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *JMLR*, 9:1981–2014, jun 2008.
- R. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6:1817–1853, 2005.
- R.K. Ando. Applying alternating structure optimization to word sense disambiguation. In *Proc. of Computational Natural Language Learning*, 2006.
- C. Antoniak. Mixtures of Dirichlet processes with applications to bayesian non-parametric problems. *The Annals of Statistics*, (2):1152–1174, 1974.
- A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. In *Proc. of NIPS*, 2007.
- S. Arora, R. Ge, and A. Moitra. Learning topic models – going beyond svd. In *Proc. of FOCS*, pages 1–10, 2012.
- A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh. On smoothing and inference for topic models. In *Proc. of UAI*, pages 27–34, 2009.
- B. Bakker and T. Heskes. Task clustering and gating for Bayesian multitask learning. *JMLR*, 4, 2003.

- R. Balasubramanyan and W. W. Cohen. Block-LDA: Jointly modeling entity-annotated text and entity-entity links. In *Proc. of SDM*, pages 450–461, 2011.
- B. Ball, B. Karrer, and M.E.J. Newman. Efficient and principled method for detecting communities in networks. *Phys. Rev. E*, 84, Sep 2011.
- S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *Learning Theory and Kernel Machines*, volume 2777, pages 567–580. 2003.
- S. Bickel, J. Bogojeska, T. Lengauer, and T. Scheffer. Multi-task learning for HIV therapy screening. In *Proc. of ICML*, pages 56–63, 2008.
- I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.
- D. Blackwell and J. MacQueen. Ferguson distributions via Pólya urn schemes. *The Annals of Statistics*, 1973.
- D. M. Blei and J. D. Lafferty. Dynamic topic models. In *Proc. of ICML*, 2006.
- D. M. Blei and J. D. Mcauliffe. Supervised topic models. In *Proc. of NIPS*, 2007.
- D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *JMLR*, 3: 993–1022, 2003.
- D.M. Blei and J.D. Lafferty. A correlated topic model of science. *Annals of Applied Statistics*, 1(1):17–35, 2007.
- A. Blum. On-line algorithms in machine learning. In Fiat and Woeginger, editors, *Online Algorithms: The State of the Art*. LNCS Vol.1442, Springer, 1998.
- K. D. Bollacker and J. Ghosh. Knowledge transfer mechanisms for characterizing image datasets. In *Soft Computing and Image Processing*. Physica-Verlag, Heidelberg, 2000.
- A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *JMLR*, 6:1579–1619, December 2005.
- A. Bordes, L. Bottou, P. Gallinari, and J. Weston. Solving multiclass support vector machines with larank. In *Proc. of ICML*, pages 89–96, 2007.

- N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proc. of ICML*, 2012.
- L.D. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *Proc. of ICCV*, pages 1365–1372, 2009.
- T. Broderick, L. Mackey, J. Paisley, and M. I. Jordan. Combinatorial clustering and the beta negative binomial process. *arXiv:1111.1802v5*, 2013.
- W. L. Buntine. Variational extensions to EM and Multinomial PCA. In *Proc. of ECML*, pages 23–34, 2002.
- S. Rendle C. Freudenthaler, L. Schmidt-Thieme. Bayesian factorization machines. In *Proc. of NIPS Workshop on Sparse Representation and Low-rank Approximation*, 2011.
- E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis. *Journal of the ACM*, 2011.
- J. Canny. Gap: a factor model for discrete data. In *Proc. of SIGIR*, 2004.
- R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, July 1997.
- A. T. Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Intell. Neuroscience*, 2009:4:1–4:17, January 2009.
- N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear classification. *JMLR*, 7:1205–1230, 2006.
- Y. S. Chan and H. T. Ng. Domain adaptation with active learning for word sense disambiguation. In *Proc. of ACL*, pages 49–56, 2007.
- A.J.B. Chaney, P. Gopalan, and D.M. Blei. Poisson trust factorization for incorporating social networks into personalized item recommendation. In *NIPS Workshop: What Difference Does Personalization Make?*, 2013.
- E. Chang, S. Tong, K. Goh, and C. Chang. Support Vector Machine Concept-Dependent Active Learning For Image Retrieval. *IEEE Transactions on Multimedia*, 2005.

- J. Chang and D. Blei. Relational topic models for document networks. In *Proc. of AISTATS*, 2009.
- J. Chang, J. B. Graber, and D. M. Blei. Connections between the lines: augmenting social networks with text. In *Proc. of KDD*, pages 169–178, 2009.
- R. Chen, A. Reznichenko, P. Francis, and J. Gehrke. Towards statistical queries over distributed private user data. In *Proc. of the 9th USENIX conference on Networked Systems Design and Implementation*, 2012.
- H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian ensemble learning. In *Proc. of NIPS*, pages 265–272, 2006.
- K. Christakopoulou and A. Banerjee. Collaborative ranking with a push at the top. In *Proc. of WWW*, pages 205–215, 2015.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.
- M. Deodhar and J. Ghosh. Mining for most certain predictions from dyadic data. In *Proc. of KDD*, 2009.
- I. S. Dhillon and D. S. Modha. A data-clustering algorithm on distributed memory multiprocessors. In *Proc. Large-scale Parallel Knowledge Discovery and Data Mining Systems Workshop, ACM SIGKnowledge Discovery and Data Mining*, August 1999.
- P. Diggle. A kernel method for smoothing point process data. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 34:138–147, 1985.
- Y. Ding and X. Li. Time weight collaborative filtering. In *Proc. of CIKM*, pages 485–492, 2005.
- J. Donahue and K. Grauman. Annotator rationales for visual recognition. In *Proc. of ICCV*, pages 1395–1402, 2011.
- G. Druck, B. Settles, and A. McCallum. Active learning by labeling features. In *Proc. of EMNLP*, pages 81–90, 2009.

- N. Du, M. Farajtabar, A. Ahmed, A.J. Smola, and L. Song. Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In *Proc. of KDD (to appear)*, 2015.
- N. U. Edakunni and S. Vijayakumar. Efficient online classification using an ensemble of bayesian linear logistic regressors. In *8th Int. Workshop on MCS*, pages 102–111, 2009.
- J. Eisenstein, A. Ahmed, and E. P. Xing. Sparse additive generative models of text. In *Proc. of ICML*, pages 1041–1048, 2011.
- A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proc. of KDD*, 2002.
- T. Evgeniou, C.A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *JMLR*, 6:615–637, 2005.
- T. Evgeniou, M. Pontil, and O. Toubia. A convex optimization approach to modeling consumer heterogeneity in conjoint estimation. *Marketing Science*, 26(6): 805–818, November 2007.
- R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9:1871–1874, June 2008.
- A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *Proc. of CVPR*, pages 1778–1785, 2009.
- T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *Ann. Statist.*, 1973.
- J. R. Foulds, C. Dubois, A. U. Asuncion, C. T. Butts, and P. Smyth. A dynamic relational infinite feature model for longitudinal social networks. In *Proc. of AISTATS*, volume 15, pages 287–295, 2011.
- W. Fu, L. Song, and E. P. Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proc. of ICML*, pages 329–336, 2009.
- C. Gabriella, R.D. Christopher, F. Lixin, W. Jutta, and B. Cédric. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

- J. Gao, W. Fan, J. Jiang, and J. Han. Knowledge transfer via multiple model local structure mapping. In *Proc. of KDD*, pages 283–291, 2008.
- J. Gao, F. Liang, W. Fan, Y. Sun, and J. Han. Graph-based consensus maximization among multiple supervised and unsupervised models. In *Proc. of NIPS*, pages 1–9, 2009.
- Z. Ghahramani and H. Kim. Bayesian classifier combination. Technical report, 2003. URL <http://learning.eng.cam.ac.uk/zoubin/papers/GhaKim03.pdf>.
- P. Gopalan, D. M. Mimno, S. Gerrish, M. J. Freedman, and D. M. Blei. Scalable inference of overlapping communities. In *Proc. of NIPS*, pages 2258–2266, 2012.
- P. Gopalan, J.M. Hofman, and D.M. Blei. Scalable recommendation with poisson factorization. *CoRR*, abs/1311.1704, 2013. URL <http://arxiv.org/abs/1311.1704>.
- P. Gopalan, F. Ruiz, R. Ranganath, and D. Blei. Bayesian nonparametric poisson factorization for recommendation systems. In *Proc. of AISTATS*, 2014a.
- P.K. Gopalan, L. Charlin, and D. Blei. Content-based recommendations with poisson factorization. In *Proc. of NIPS*, pages 3176–3184. 2014b.
- S. Gunasekar, A. Acharya, N. Gaur, and J. Ghosh. Noisy matrix completion using alternating minimization. In *Proc. of ECML PKDD, Part II, LNAI 8189*, pages 194–209, 2013.
- S. Han, L. Du, E. Salazar, and L. Carin. Dynamic rank factor model for text streams. In *Proc. of NIPS*, 2014.
- S. Hanneke, W. Fu, and E.P. Xing. Discrete temporal models of social networks. *Electronic Journal of Statistics*, 4:585–605, 2010.
- A. Harpale and Y. Yang. Active learning for multi-task adaptive filtering. In *Proc. of ICML*, pages 431–438. Omnipress, 2010.
- C. Heaukulani and Z. Ghahramani. Dynamic probabilistic models for latent feature propagation in social networks. In *Proc. of ICML*, pages 275–283, 2013.

- G. E. Hinton and S. Osindero. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:2006, 2006.
- N. L. Hjort. Nonparametric Bayes estimators based on beta processes in models for life history data. *Ann. Statist.*, 1990.
- J. C. Ho, J. Ghosh, S. R. Steinhubl, W. F. Stewart, J. C. Denny, B. A. Malin, and J. Sun. Limestone: High-throughput candidate phenotype generation via tensor factorization. *Journal of Biomedical Informatics*, 52:199–211, 2014a.
- J.C. Ho, J. Ghosh, and J. Sun. Marble: High-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. In *Proc. of KDD*, pages 115–124, 2014b.
- Q. Ho, L. Song, and E.P. Xing. Evolving cluster mixed-membership blockmodel for time-varying networks. In *Proc. of AISTATS*. 2011.
- P.D. Hoff, A.E. Raftery, and M.S. Handcock. Latent space approaches to social network analysis. *Journal Of The American Statistical Association*, 97:1090–1098, 2001.
- M. D. Hoffman, D. M. Blei, and P. R. Cook. Bayesian nonparametric matrix factorization for recorded music. In *Proc. of ICML*, pages 439–446, 2010.
- P. A. D. F. R. Højén-Sørensen, O. Winther, and L. K. Hansen. Mean-field approaches to independent component analysis. *Neural Comput.*, 14:889–918, 2002.
- S.J. Hwang and K. Grauman. Reading between the lines: Object localization using implicit cues from image tags. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(6): 1145–1158, June 2012.
- K. Ishiguro, T. Iwata, N. Ueda, and J. B. Tenenbaum. Dynamic infinite relational model for time-varying relational data analysis. In *Proc. of NIPS*, pages 919–927. 2010.
- L. Jacob, F. Bach, and J. Vert. Clustered multi-task learning: A convex formulation. *CoRR*, abs/0809.2085, 2008.

- P. Jain and A. Kapoor. Active learning for large multi-class problems. In *Proc. of CVPR*, pages 762–769, 2009.
- R. Jenatton, J. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. *JMLR*, 12:2777–2824, nov 2011.
- E. Johnson and H. Kargupta. Collective, hierarchical clustering from distributed, heterogeneous data. In *Large-Scale Parallel Knowledge Discovery and Data Mining Systems*, volume 1759 of *LNCScience*, pages 221–244, 1999.
- N. L. Johnson, A. W. Kemp, and S. Kotz. *Univariate Discrete Distributions*. John Wiley & Sons, 2005.
- M. I. Jordan. Hierarchical models, nested models, and completely random measures. In *Frontiers of Statistical Decision Making and Bayesian Analysis: In Honor of James O. Berger*, pages 207–217. Springer, 2010.
- A.J. Joshi, F. Porikli, and N. Papanikolopoulos. Multi-class active learning for image classification. In *Proc. of CVPR*, pages 2372–2379, 2009.
- J.Scott, R.Gass, J.Crowcroft, P.Hui, C.Diot, and A.Chaintreau. CRAW-DAD data set dartmouth/campus (v. 2009-05-29). Downloaded from <http://crawdad.org/dartmouth/campus/>, May 2009.
- G. Jun and J. Ghosh. An efficient active learning algorithm with knowledge transfer for hyperspectral remote sensing data. In *Proc. of International Geosci. and Sens. Symposium*, volume 1, pages I–52–I–55, 2008.
- C. Kemp, J.B. Tenenbaum, T.L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proc. of AAAI*, pages 381–388, 2006.
- D. Il Kim, P. Gopalan, D. M. Blei, and E. B. Sudderth. Efficient online inference for bayesian nonparametric relational models. In *Proc. of NIPS*, pages 962–970, 2013.
- M. Kim and J. Leskovec. Nonparametric multi-group membership model for dynamic networks. In *Proc. of NIPS*, pages 1385–1393. 2013.

- S. Kim and E. P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proc. of ICML*, pages 543–550, 2010.
- Y. Kim and S. Choi. Scalable variational bayesian matrix factorization with side information. In *Proc. of AISTATS*, pages 493–502, 2014.
- J. F. C. Kingman. Completely random measures. *Pacific Journal of Mathematics*, 21(1):59–78, 1967.
- J.F.C. Kingman. *Poisson Processes*. Oxford University Press, 1993.
- Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proc. of KDD*, pages 426–434, 2008.
- Y. Koren. Collaborative filtering with temporal dynamics. In *Proc. of KDD*, pages 447–456, 2009.
- Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 2009.
- A. Kovashka, S. Vijayanarasimhan, and K. Grauman. Actively selecting annotations among objects and attributes. In *Proc. of ICCV*, pages 1403–1410, 2011.
- N. Kumar, A.C. Berg, P.N. Belhumeur, and S.K. Nayar. Describable visual attributes for face verification and image search. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 33, pages 1962–1977, October 2011.
- C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by betweenclass attribute transfer. In *Proc. of CVPR*, pages 951–958, 2009.
- N. D. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *Proc. of ICML*, pages 601–608, 2009.
- D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Proc. of NIPS*, 2001.
- J. Leskovec and J.M. Julian. Learning to discover social circles in ego networks. In *Proc. of NIPS*, pages 539–547. 2012.

- Y.J. Lim and Y.W. Teh. Variational bayesian approach to movie rating prediction. In *Proc. of KDDCup*, 2007.
- Y. Low, D. Agarwal, and A. J. Smola. Multiple domain user personalization. In *Proc. of KDD*, pages 123–131, 2011.
- D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.
- H. Ma, H. Yang, M. R. Lyu, and I. King. Sorec: Social recommendation using probabilistic matrix factorization. In *Proc. of CIKM*, pages 931–940, 2008.
- A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and role discovery in social networks with experiments on enron and academic email. *J. Artif. Int. Res.*, 30(1):249–272, October 2007.
- P. McCullagh and J. A. Nelder. *Generalized linear models*. Chapman & Hall, 2nd edition, 1989.
- A. Menon and C. Elkan. Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases*, volume 6912 of *Lecture Notes in Computer Science*, pages 437–452. Springer Berlin / Heidelberg, 2011.
- S. Merugu and J. Ghosh. Privacy perserving distributed clustering using generative models. In *Proc. of ICDM*, pages 211–218, Nov, 2003.
- K. T. Miller, T. L. Griffiths, and M. I. Jordan. Nonparametric latent feature models for link prediction. In *Proc. of NIPS*, pages 1276–1284, 2009.
- J. W. Miskin. Ensemble learning for independent component analysis. Technical report, in *Advances in Independent Component Analysis*, 2000.
- J. Møller, A. R. Syversveen, and R. P. Waagepetersen. Log gaussian cox processes. *Scandinavian Journal of Statistics*, 25:451–482.
- R.M. Nallapati, A. Ahmed, E.P. Xing, and W.W. Cohen. Joint latent topic models for text and citations. In *Proc. of KDD*, pages 542–550, 2008.
- N. Natarajan and I.S. Dhillon. Inductive matrix completion for predicting gene-disease associations. *Bioinformatics*, 30(12):60–68, 2014.

- R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of computational and graphical statistics*, 2000.
- R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants, 1999.
- N. C. Oza and K. Tumer. Classifier ensembles: Select real-world applications. *Inf. Fusion*, 9:4–20, January 2008.
- P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2): 111–126, 1994.
- K. Palla, Z. Ghahramani, and D. A. Knowles. An infinite latent attribute model for network data. In *Proc. of ICML*, pages 1607–1614, 2012.
- S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345–1359, 2010.
- A. Passos, P. Rai, J. Wainer, and H. Daumé III. Flexible modeling of latent task structures in multitask learning. In *Proc. of ICML*, pages 1103–1110, 2012.
- J. Pitman. *Combinatorial stochastic processes*, volume 1875. Springer-Verlag, 2006.
- M.D. Plumbley and E. Oja. A “nonnegative pca” algorithm for independent component analysis. *IEEE Transactions on Neural Networks*, 15(1):66–76, 2004.
- G. E. Poliner and D. P. W. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP J. Adv. Sig. Proc.*, 2007.
- I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proc. of KDD*, pages 569–577, 2008.
- G.J. Qi, Xian-Sheng H., Yong R., Jinhui T., and Hong-Jiang Z. Two-dimensional active learning for image classification. In *Proc. of CVPR*, pages 1–8, 2008.

- A. Quattoni, S. Wang, L. P. Morency, M. Collins, and T. Darrell. Hidden-state conditional random fields. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.
- H. Raghavan, O. Madani, and R. Jones. Interactive feature selection. In *Proc. of IJCAI*, pages 841–846, 2005.
- S. Raghavan, S. Gunasekar, and J. Ghosh. Review quality aware collaborative filtering. In *Proc. of RecSys*, pages 123–130, 2012.
- P. Rai, A. Saha, H. Daumé, III, and S. Venkatasubramanian. Domain adaptation meets active learning. In *Proc. of NAACL HLT Workshop on Active Learning for Natural Language Processing*, pages 27–32, 2010.
- D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. In *Proc. of EMNLP*, pages 248–256, 2009.
- S. Rendle. Factorization machines. In *Proc. of ICDM*, 2010.
- S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proc. of WSDM*, pages 81–90, 2010.
- S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proc. of WWW*, pages 811–820, 2010.
- S. Robertson and I. Soboroff. The TREC 2002 filtering track report. In *Text Retrieval Conference*, 2002.
- S. Roller, M. Speriosu, S. Rallapalli, B. Wing, and J. Baldridge. Supervised text-based geolocation using language models on an adaptive grid. In *Proc. of EMNLP-CoNLL*, pages 1500–1510, 2012.
- M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *Proc. of UAI*, pages 487–494, 2004.
- N. Roy and A. K. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proc. of ICML*, pages 441–448, 2001.

- T. N. Rubin, A. Chambers, P. Smyth, and M. Steyvers. Statistical topic models for multi-label document classification. *CoRR*, abs/1107.2462, 2011.
- B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation, 2008.
- A. Saha, P. Rai, H. Daum III, and S. Venkatasubramanian. Online learning of multiple tasks and their relationships. *JMLR - Proceedings Track*, 15:643–651, 2011.
- R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Proc. of NIPS*, 2007.
- R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proc. of ICML*, pages 880–887, 2008.
- P. Sarkar and A.W. Moore. Dynamic social network analysis using latent space models. *SIGKDD Explor. Newsl.*, 7:31–40, dec 2005.
- J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4: 639–650, 1994.
- B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.
- D. Shen, J. Ruvini, and B. Sarwar. Large-scale item categorization for e-commerce. In *Proc. of CIKM*, 2012.
- X. Shi, W. Fan, and J. Ren. Actively transfer domain knowledge. In *Proc. of ECML PKDD - Part II*, pages 342–357, 2008.
- B. Siddiquie, R. S. Feris, and L. S. Davis. Image ranking and retrieval based on multi-attribute queries. In *Proc. of CVPR*, pages 801–808, 2011.
- J. Silva and L. Carin. Active learning for online bayesian matrix factorization. In *Proc. of KDD*, pages 325–333, 2012.
- D. L. Silver and K. P. Bennett. Guest editor’s introduction: special issue on inductive transfer learning. *Machine Learning*, 73:215–220, December 2008.

- T.A.B. Snijders, G.G. Bunt, and C.E.G. Steglich. Introduction to stochastic actor-based models for network dynamics. *Social Networks*, 32(1):44–60, 2010.
- N. D. Socci, D. D. Lee, and H. S. Seung. The rectified gaussian distribution. In *Proc. of NIPS*, pages 350–356, 1998.
- M. Spain and P. Perona. Some objects are more equal than others: Measuring and predicting importance. In *Proc. of ECCV*, pages 523–536, 2008.
- N. Srebro, T. Jaakkola, et al. Weighted low-rank approximations. In *Proc. of ICML*, 2003.
- K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *Proc. of WWW*, pages 675–684, 2004.
- J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Zhong Su. Arnetminer: Extraction and mining of academic social networks. In *Proc. of KDD*, pages 990–998, 2008.
- Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101:1566–1581, December 2006.
- Y. W. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for HDP. In *Proc of NIPS*, 2007.
- R. Thibaux and M. Jordan. Hierarchical beta processes and the indian buffet process. In *Proc. of AISTATS*, 2007.
- M. K. Titsias. The infinite gamma-Poisson feature model. In *Proc. of NIPS*, 2008.
- A. Torralba, K.P. Murphy, and W.T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(5): 854–869, May 2007.
- J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proc. of KDD*, pages 206–215, 2003.

- S. Vijayanarasimhan, P. Jain, and K. Grauman. Hashing hyperplane queries to near points with applications to large-scale active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(2):276–288, 2014.
- S. G. Walker. Sampling the Dirichlet mixture model with slices. *Communications in Statistics Simulation and Computation*, 2007.
- C. Wang, D. Blei, and D. Heckerman. Continuous Time Dynamic Topic Models. In *Proc. of UAI*, 2008.
- C. Wang, B. Thiesson, C. Meek, and D. Blei. Markov topic models. In *Proc. of AISTATS*, 2009.
- C. Wang, J. W. Paisley, and D. M. Blei. Online variational inference for the hierarchical Dirichlet process. *JMLR - Proceedings Track*, 15:752–760, 2011a.
- H. Wang, H. Shan, and A. Banerjee. Bayesian cluster ensembles. *Statistical Analysis and Data Mining*, 1:1–17, January 2011b.
- X. Wang, N. Mohanty, and A. McCallum. Group and topic discovery from relations and their attributes. In *Proc. of NIPS*, pages 1449–1456, 2006.
- Y. Wang and L. Carin. Levy measure decompositions for the beta and gamma processes. In *Proc. of ICML*, 2012.
- K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proc. of ICML*, pages 1113–1120, 2009.
- Z. Wen and C. Lin. Towards finding valuable topics. In *Proc. of SDM*, pages 720–731, 2010.
- R. L. Wolpert, M. A. Clyde, and C. Tu. Stochastic expansions using continuous dictionaries: Lévy Adaptive Regression Kernels. *Annals of Statistics*, 2011.
- L. Xiong, X. Chen, T. Huang, J. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proc. of SDM*, 2010.

- K.S. Xu and A.O. Hero. Dynamic stochastic blockmodels for time-evolving social networks. *J. Sel. Topics Signal Processing*, 8(4):552–562, 2014.
- Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with Dirichlet process priors. *JMLR*, 8:35–63, 2007.
- J. Yang and J. Leskovec. Overlapping community detection at scale: A nonnegative matrix factorization approach. In *Proc. of WSDM*, pages 587–596, 2013.
- J. Yang, J. J. McAuley, and J. Leskovec. Community detection in networks with node attributes. In *Proc. of ICDM*, pages 1151–1156, 2013.
- T. Yoshida. Toward finding hidden communities based on user profile. *J. Intell. Inf. Syst.*, 40(2):189–209, April 2013.
- C. J. Yu and T. Joachims. Learning structural SVMs with latent variables. In *Proceedings of International Conference on Machine Learning*, pages 1169–1176, 2009.
- O. F. Zaidan, J. Eisner, and C. Piatko. Machine learning with annotator rationales to reduce annotation cost. In *Proc. of the NIPS Workshop on Cost Sensitive Learning*, 2008.
- K. Zhai and J.L. Boyd-graber. Online latent dirichlet allocation with infinite vocabulary. In *Proc. of ICML*, pages 561–569, 2013.
- J. Zhang, Z. Ghahramani, and Y. Yang. Flexible latent variable models for multi-task learning. *Machine Learning*, 73(3):221–242, December 2008.
- J. Zhou, J. Chen, and J. Ye. Clustered Multitask Learning Via Alternating Structure Optimization. In *Proc. of NIPS*, 2011.
- M. Zhou. Infinite edge partition models for overlapping community detection and link prediction. In *Proc. of AISTATS*, pages 1135–1143. 2015.
- M. Zhou and L. Carin. Augment-and-conquer negative binomial processes. In *Proc. of NIPS*, 2012.
- M. Zhou and L. Carin. Negative binomial process count and mixture modeling. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2015.

- M. Zhou, L. Hannah, D. Dunson, and L. Carin. Beta-negative binomial process and Poisson factor analysis. In *Proc. of AISTATS*, pages 1462–1471, 2012.
- Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan. Large-scale parallel collaborative filtering for the netflix prize. In *Proc. 4th Intl Conf. Algorithmic Aspects in Information and Management, LNCS 5034*, 2008.
- J. Zhu. Max-margin nonparametric latent feature models for link prediction. In *Proc. of ICML*, 2012.
- J. Zhu, A. Ahmed, and E. P. Xing. MedLDA: maximum margin supervised topic models for regression and classification. In *Proc. of ICML*, pages 1257–1264, 2009.
- J. Zhu, N. Chen, H. Perkins, and B. Zhang. Gibbs max-margin topic models with fast sampling algorithms. In *Proc. of ICML*, pages 124–132, 2013.

## Vita

Ayan Acharya received his Bachelor of Engineering degree from the Electronics and Telecommunication Engineering Department, Jadavpur University, Kolkata in 2009 and Masters degree from the Department of Electrical and Computer Engineering at the University of Texas at Austin in May 2012. He worked as interns in eBay Research Lab, Qualcomm Inc., Yahoo! Labs, and CognitiveScale Inc. His specialization is in data mining and statistical machine learning.

Permanent address: 915 East 41st Street, apt 206  
Austin, Texas 78751

This dissertation was typeset with L<sup>A</sup>T<sub>E</sub>X<sup>†</sup> by the author.

---

<sup>†</sup>L<sup>A</sup>T<sub>E</sub>X is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's T<sub>E</sub>X Program.