A Mixture Model for Heterogeneous Data with Application to Public Healthcare Data Analysis

Master's thesis
Applied Mathematics

Johannes Sirola

Advisor
Dr. Arto Klami

Examiners
Dr. Arto Klami
Prof. Jukka Corander

Department of Mathematics and Statistics
University of Helsinki
2016

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta/Osasto — Fakultet/Sektion — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Mathematics and Statistics

Tekijä — Författare — Author

Johannes Sirola

Työn nimi — Arbetets titel — Title

A Mixture Model for Heterogeneous Data with Application to Public Healthcare Data Analysis

Oppiaine — Läroämne — Subject		
Applied Mathematics		
Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	04-10-2016	5+82+17

Tiivistelmä — Referat — Abstract

In this thesis we present an algorithm for doing mixture modeling for heterogeneous data collections. Our model supports using both Gaussian- and Bernoulli distributions, creating possibilities for analysis of many kinds of different data. A major focus is spent to developing scalable inference for the proposed model, so that the algorithm can be used to analyze even a large amount of data relatively fast.

In the beginning of the thesis we review some required concepts from probability theory and then proceed to present the basic theory of an approximate inference framework called variational inference. We then move on to present the mixture modeling framework with examples of the Gaussian- and Bernoulli mixture models. These models are then combined to a joint model which we call GBMM for *Gaussian and Bernoulli Mixture Model*. We develop scalable and efficient variational inference for the proposed model using state-of-the-art results in Bayesian inference. More specifically, we use a novel data augmentation scheme for the Bernoulli part of the model coupled with overall algorithmic improvements such as incremental variational inference and multicore implementation. The efficiency of the proposed algorithm over standard variational inference is highlighted in a simple toy data experiment. Additionally, we demonstrate a scalable initialization for the main inference algorithm using a state-of-the-art random projection algorithm coupled with k-means++ clustering. The quality of the initialization is studied in an experiment with two separate datasets. As an extension to the GBMM model, we also develop inference for categorical features. This proves to be rather difficult and our presentation covers only the derivation of the required inference algorithm without a concrete implementation.

We apply the developed mixture model to analyze a dataset consisting of electronic patient records collected in a major Finnish hospital. We cluster the patients based on their usage of the hospital's services over 28-day time intervals over 7 years to find patterns that help in understanding the data better. This is done by running the GBMM algorithm on a big feature matrix with 269 columns and more than 1.7 million rows. We show that the proposed model is able to extract useful insights from the complex data, and that the results can be used as a guideline and/or preprocessing step for possible further, more detailed analysis that is left for future work.

Avainsanat — Nyckelord — Keywords

Clustering, Mixture Modeling, Incremental Variational Inference, Machine Learning, Bayesian Statistics

Säilytyspaikka — Förvaringsställe — Where deposited

Kumpula Campus Library

Muita tietoja — Övriga uppgifter — Additional information

Acknowledgements

This thesis was written while I was interning at the Multi-Source Probabilistic Inference research group at Helsinki Institute for Information Technology HIIT. I would like to thank prof. Jukka Corander for encouraging me to apply for a summer internship at HIIT and pointing out an interesting topic for my Bachelor's thesis. This was my first touch to machine learning and pointed the way for my further studies.

The topic of this thesis evolved around collaboration with researchers at the University Hospital of Turku. I would like to thank prof. Tarja Laitinen for allowing the use of their data, and Mikko Stepanov for help with preprocessing and other technical details considering the data.

Finally, I want to thank my advisor Dr. Arto Klami for giving me the opportunity to work with interesting machine learning projects over the two years I spent at HIIT. This time has been invaluable for my professional growth and I feel the experience and tools I acquired during this time will carry a long way into the future. I am also grateful for all the guidance and trust I received – the road here was not always easy, but it was definitely worth it.

Contents

List of abbreviations

Abbreviation	Explanation
GMM	Gaussian mixture model
BMM	Bernoulli mixture model
GBMM	Gaussian- and Bernoulli mixture model
ELBO	evidence lower bound
r.v.	a random variable / random vector
pmf	probability mass function
pdf	probability density function
a.s.	almost surely, i.e. with probability 1
KL divergence	Kullback-Leibler divergence
VI	variational inference
SVI	stochastic variational inference
IVI	incremental variational inference

List of symbols

Symbol	Explanation
x / $\mathbf{x}$ / $\mathbf{X}$	a scalar/vector/matrix
$\mathbf{x}^T$	transpose of $\mathbf{x}$
$\langle \cdot \rangle$, $\mathbb{E}[\cdot]$	expectation
$\phi(\cdot)$ / $\Phi(\cdot)$	standard normal density/distribution function
$\lvert \cdot \rvert$	determinant
$\lVert \cdot \rVert_F$	Frobenius norm
$p(\cdot)$	a generic probability density function
$D_{KL}(p\lvert\lvert q)$	Kullback-Leibler divergence between distributions p and q
$D_B(p, q)$	Bhattacharyya distance between distributions p and q

Chapter 1

Introduction

1.1 Background

Every day, an enormous amount of data is collected in the world by different sensors, social media updates, online transactions etc. The related huge datasets are often called examples of *big data*. Many companies and institutions, as well as ordinary people, could benefit from analyzing this data, but unfortunately the volume and complexity of the data often make the task too hard for humans. Following the increase in computing power in the recent years, *machine learning* has become the major tool for processing huge volumes of this data automatically. In fact, we use these systems every day for example to find movie recommendations, recognize faces, fraud detection and bunch of other things. Recent news about a Go-playing AI[1] beating top human players, self driving cars making their way into public roads, and others are making machine learning and 'data science' known topics to the ordinary people as well.

In this thesis we focus on a subfield of machine learning called *unsupervised machine learning*, which is a general term for methods that try to find some structure in unlabeled data. The idea for the subject of this thesis came from a possibility to work with a digitalized healthcare application. We were provided a data set consisting of electronic patient records collected in a major Finnish hospital over a time period of 7 years. The goal of our

[1]Go is a traditional Chinese board game, often described as one of the most complex games in the world. Creating an AI that can beat the top players in the world is thought to be a major feat.

research is to understand the complex data better by means of statistical analysis and visualizations of the findings.

We approach the problem by creating a machine learning model that can automatically find relevant patterns in the data. More specifically, we specify a Bayesian mixture model that finds similar groups of examples within heterogeneous data collections. Solving this type of a problem is generally called *clustering*. The data was preprocessed so that our samples depict one month-long time periods of the patients based on the treatment they received during that month. We then cluster the (patient, time interval)- tuples and use the results to create different visualizations that describe the data. We also look for insights that would provide motivation for further analysis that can be left for future work.

From a technical point of view, we pay special attention to the scalability and efficiency of our solution. The approach we take on inference is based on so called variational inference, which has lately gotten a lot of attention due to its ability to scale up to big data.

1.2 Structure of the thesis

We begin by going through necessary background from probability theory and Bayesian statistics in Chapter 2. Chapter 3 introduces the basic principles of the standard variational inference and also discusses some of the more recent developments that are used to make the inference faster and more scalable. Together these two chapters review the core methodology that is used to develop inference for the models presented in the later chapters.

Chapter 4 is used to specify two common mixture models, namely the Gaussian- and Bernoulli mixture models. The notation established here will be used later, most importantly in Chapter 5 that contains the main technical contributions of the thesis. First, the mixture models for heterogeneous data is specified by combining the Gaussian- and Bernoulli mixture models from Chapter 4 to a joint mixture model. The rest of the chapter is used to explain the inference related to the model.

In Chapter 6 we show a multinomial mixture model could be specified using a probit data augmentation. We also briefly discuss recent results which would offer a better, more practical way of implementing the model.

Chapter 7 is used to discuss the application to the public healthcare dataset. We describe how the modeling was done and provide visualizations

of the results.

Finally, the conclusions are presented in Chapter 8, while the technical details concerning the inference algorithms in Chapters 5 and 6 are shown in Appendices A and B.

1.3 Contributions of the thesis

The main contributions of the thesis are as follows:

1. We specify a hierarchical, fully Bayesian mixture model for clustering heterogeneous data. A state-of-the-art data augmentation scheme is employed to make efficient variational inference possible.

2. We present scalable inference for the model using some of the very latest results in Bayesian inference and machine learning. The core parts of the inference process are:

   - Scalable initialization for the main inference algorithm using a random projection k-means(++) algorithm. We empirically demonstrate the performance of the algorithm in two separate experiments.
   - Scalable and fast incremental variational inference for the presented mixture model.
   - A simple parallel implementation of the proposed incremental variational inference algorithm that runs on multiple CPU cores for faster computation speed with large datasets.

   The proposed multicore implementation of the incremental variational inference algorithm is shown to offer substantially faster computation speeds over the default variational inference in a toy data experiment.

3. Application of our model to digitalized healthcare. We cluster a dataset that contains over 1.7 million data points that are represented by 269-dimensional feature vectors. We present visualizations that characterize the found clusters and how they relate to for example to death probabilities of the patients and the number of days they will be treated in the hospital in the future.

Chapter 2

Preliminaries

2.1 Review of probability

In this section we review probability theory in sufficient detail for the purposes of this thesis. A more rigorous treatment of the presented results can be found in textbooks such as [40] or [46].

2.1.1 Random variables and random vectors

A triplet $(\Omega, \mathcal{F}, P)$ is called a *probability space* if

(i) Ω is a nonempty set.

(ii) $\mathcal{F} \subset \Omega$ is a σ-algebra.

(iii) P is a *probability measure* on $(\Omega, \mathcal{F})$, i.e. P is a measure on $(\Omega, \mathcal{F})$ and $P(\Omega) = 1$.

The set Ω is called *sample space* and the sets $A \in \mathcal{F}$ *events*.

A measurable function

$$X : \Omega \to \mathbb{R}$$

is called a (real) *random variable*. In a similar fashion, a measurable function

$$X : \Omega \to \mathbb{R}^n$$

is called a *random vector*. We consider the vectors be represented as column vectors by default. A shorthand notation r.v. is used to denote both random variables and random vectors.

We define the *indicator random variable* of an event $A \subset \mathbb{R}^n$ by setting

$$\mathbb{1}_A(\omega) = \begin{cases} 1, & \text{if } \omega \in A, \\ 0, & \text{otherwise.} \end{cases}$$

We use the well established notation from probability theory and do not explicitly write down the argument ω when operating with random variables.

2.1.2 Distributions

The *distribution* of a random vector X is a probability measure P_X defined by

$$P_X(A) = P(X^{-1}(A)) \quad \text{for } A \in \mathbb{R}^n.$$

If $\mu = P_X$ we write $X \sim \mu$ for X 'follows the distribution μ'.

The *(cumulative) distribution function* of a random vector X is defined by

$$F_X(x) = P_X(X \leq x), \quad x \in \mathbb{R}^n.$$

The distribution function completely determines the distribution of X.

If the sample space Ω is at most countably infinite, we say that the distribution of a random vector X defined in the corresponding probability space is *discrete*. We then define the *probability mass function* (called often simply pmf) of X by

$$f_X(x) = P(X_1 = x_1, ..., X_n = x_n).$$

We note that from the properties of the probability measure P it follows that $0 \leq f_X \leq 1$ and $\sum_x f_X(x) = 1$.

The distribution of a random vector X is called *continuous* if for any set $A \in \mathbb{R}^n$ we have

$$P(X \in A) = \int_A f_X(x)\,\mathrm{d}x.$$

In such case we call f_X the *probability density function* or simply the *density* of X (abbreviated pdf). The probability density function is not unique as it can be freely altered in a set that has measure zero. When calling two probability densities equal we thus understand them to be equal *almost surely*, often abbreviated a.s. in the literature, meaning they differ only in a set that has zero measure.

For the rest of the chapter we will formulate the results for continuous random variables. The same results apply also to the discrete case and they are obtained by simply replacing the integrals with sums.

Let $X = (X_1, X_2)^T$ be a random vector with a continuous distribution. The distribution of X_i alone ($i = 1, 2$) is called a marginal distribution of X_i. The marginal distributions can be found by integrating the joint density over the remaining variable, e.g.

$$f_{X_1}(x_1) = \int f_{X_1,X_2}(x_1, x_2)\,\mathrm{d}x_2,$$

which is also called *marginalization*.

An important formula in Bayesian inference is the formula for *conditional density*: given $X_2 = x_2$ the conditional density of X_1 is given by

$$f_{X_1|X_2}(x_1|x_2) = \begin{cases} \frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)}, & \text{if} f_{X_2}(x_2) > 0, \\ 0, & \text{otherwise.} \end{cases}$$

Note that the denominator is found by marginalizing the joint density f_{X_1,X_2}. From the above formula we also get the *multiplication rule*

$$f_{X_1,X_2}(x_1, x_2) = f_{X_1|X_2}(x_1|x_2) f_{X_2}(x_2).$$

We note that the above formulas for marginal and conditional distributions work similarly also in the multidimensional setting and for joint distributions where some variables are discrete and some are continuous. In such cases we simply integrate over the continuous variables and sum over the discrete variables.

2.1.3 Independence and conditional independence

We say that random vectors $X_1, ..., X_n$ are *independent* if their joint distribution function factorizes as

$$F_{X_1,...,X_n}(x_1, ..., x_n) = \prod_{i=1}^{n} F_{X_i}(x_i),$$

for all values of $x_1, ..., x_n$. If the random vectors in addition have the same distribution μ, we often use a shorthand notation and say that the X_i are

independent and identically distributed (i.i.d.) and follow the distribution μ. If this is the case, then also the joint pdf and the joint pmf factorize similarly the product of the marginal pdf's and pmf's. If for all values of $x_1, ..., x_n$ and y we have

$$f_{X_1,...,X_n|Y}(x_1, ..., x_n|y) = \prod_{i=1}^{n} f_{X_i|Y}(x_i|y),$$

then $X_1, ..., X_n$ are called *conditionally independent* given Y.

One remark concerning the notation: For the most parts of the thesis we will use a shorthand notation, usually $p(\cdot)$, to denote any probability distribution without explicitly showing the relating r.v. in the subscript. This notation is well established in statistics literature and helps to avoid unnecessary cluttering of notation, especially when working with complex statistical models with several variables.

2.1.4 Expectations

Let X be a continuous random variable or a random vector and g a measurable function. The *expected value* (also *expectation* or *mean*) of the transformation $g(X)$ is given by

$$\mathbb{E}[g(X)] = \int g(x) f_X(x) \, \mathrm{d}x,$$

whenever the result is finite. The expectation is either a scalar or a vector, depending on g and the dimensionality of X. If $X \in \mathbb{R}$ and $g(x) = x^k$, we call the result the kth *moment* of X. Note that the expectations are by default calculated with respect to the measure $\mu = P_X$. If we want to calculate the expectation with respect to another distribution, say ν, we write $\mathbb{E}_\nu[\cdot]$. This notation is used throughout the thesis, as we will be constantly calculating expectations of the same distribution under different probability measures.

The *variance* of a random variable X is defined as

$$\mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2].$$

Using the definition of variance and the linearity of expectation it is easy to check that the important formula

$$\mathrm{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

holds. If $X \in \mathbb{R}^n$ is a random vector it's *covariance* is defined as the $n \times n$ matrix

$$\mathrm{Cov}(X) = \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^{\mathrm{T}}].$$

For the diagonal elements it is true that

$$[\mathrm{Cov(X)}]_{ii} = \mathrm{Var}(X_i), \quad i = 1, ..., n.$$

Finally, we define the *conditional expectation* of X given another random variable Y. We begin by defining the conditional expectation of X given $Y = y$ by

$$\mathbb{E}[X|Y = y] = \int x f_{X|Y}(x|y)\,\mathrm{d}x.$$

For a fixed y we thus have $\mathbb{E}[X|Y = y] = g(y)$ for some function g. We then extend the formula by calling the conditional expectation of X given Y the random variable $\mathbb{E}[X|Y]$ for which

$$\mathbb{E}[X|Y] = g(Y), \quad \text{where } g(y) = \mathbb{E}[X|Y = y].$$

A useful property of the conditional expectation is the *tower property*, which tells us that we can calculate the expectation of X iteratively by first conditioning on another r.v. Y as

$$\mathbb{E}[X] = \mathbb{E}_Y[\mathbb{E}_{X|Y}[X|Y]].$$

2.1.5 Strong law of large numbers and Monte Carlo integration

Let X and $X_1, X_2, ..., X_n$ be i.i.d. random variables with finite expectations. Additionally denote $\mathbb{E}[X] = \mu$ and $S_n = \sum_{i=1}^{n}(X_i - \mu)$. The *strong law of large numbers* states that

$$P\Big[\lim_{n\to\infty}\Big|\frac{S_n}{n}\Big| = 0\Big] = 1,$$

which means that the sample average $\sum_{i=1}^{n} X_i/n$ converges to the mean μ almost surely as n tends to infinity. This can be used to approximate certain expectations by sampling.

Let g be a function such that the expectation $\mathbb{E}[g(X)]$ is finite. If we know how to sample from the distribution of X, the *Monte Carlo integral* of $g(X)$ can be computed as

$$\mathbb{E}[g(X)] = \int g(x) f_X(x)\,\mathrm{d}x \approx \frac{1}{n}\sum_{i=1}^{n} g(x_i),$$

where x_i are independent samples drawn from the distribution of X. The strong law of large numbers guarantees that the estimate on the right hand side converges to the true value of the expectation.

2.2 About Bayesian inference

In this section we present some basic principles behind Bayesian inference. More details about the topics touched here can be found for example in [26].

Suppose we have observed some data $\mathbf{x}$ and specified a model (probability distribution) that we think is generating the data. Denoting the model parameters by $\boldsymbol{\theta}$, the distribution of the observations, $p(\mathbf{x}|\boldsymbol{\theta})$, is called a *sampling distribution*.

In Bayesian inference we are often interested in predicting the value of future data $\tilde{\mathbf{x}}$ given the already observed data $\mathbf{x}$. This is done by finding the *posterior predictive distribution* $p(\tilde{\mathbf{x}}|\mathbf{x})$ of the new data by integrating $\boldsymbol{\theta}$ out in the joint distribution $p(\tilde{\mathbf{x}}, \boldsymbol{\theta}|\mathbf{x})$:

$$p(\tilde{\mathbf{x}}|\mathbf{x}) = \int p(\tilde{\mathbf{x}}|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathbf{x})\,\mathrm{d}\boldsymbol{\theta}.$$

Here $p(\boldsymbol{\theta}|\mathbf{x})$ is the *posterior distribution* of the parameters conditioned on the data. Often times also the posterior distribution itself is of particular interest. This is very much the case in this thesis, as we focus on creating a model for the current data without the need to find the posterior predictive distribution at all. Regardless of the end goal, posterior inference is essential in the process.

The posterior distribution is calculated using *Bayes' theorem* as the conditional density

$$p(\boldsymbol{\theta}|\mathbf{x}) = \frac{p(\mathbf{x}|\boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{x})}. \tag{2.1}$$

In this context we have already observed the data and the quantity $p(\mathbf{x}|\boldsymbol{\theta})$ is called the *likelihood* of the model. In order to use the formula we also

have to specify a *prior distribution* $p(\boldsymbol{\theta})$ for the parameters. This is a part of the model selection and is usually handled by choosing a standard prior distribution whose parameters are then set to values that represent our prior thoughts about the true parameter values. Another common option is to assign an *uninformative* prior such as the uniform distribution for the parameters. In this case the interpretation is that we have no particular idea of the true parameter values before we are acquire any data. Another observation we notice in formula (2.1) is that the normalizing constant $p(\mathbf{x})$, called *marginal likelihood*, is possibly a high dimensional integral and often very difficult to evaluate analytically. Fortunately, the posterior can also be calculated only up to a constant as

$$p(\boldsymbol{\theta}|\mathbf{x}) \propto p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}), \tag{2.2}$$

after which it may be possible to normalize the expression by recognizing the functional form of the distribution in question.

2.2.1 The exponential family and conjugate priors

The distributions belonging to the *exponential family* of distributions are of the form

$$p(\mathbf{x}|\boldsymbol{\theta}) = h(\mathbf{x})g(\boldsymbol{\theta}) \exp\{\boldsymbol{\theta}^T \mathbf{t}(\mathbf{x})\}, \tag{2.3}$$

where h, g and $\mathbf{t}$ are some functions. This definition is also called the *canonical form* of exponential family distributions. The parameters $\boldsymbol{\theta}$ are here called the *natural parameters* of the distribution and $\mathbf{t}(\mathbf{x})$ is the *sufficient statistic*. The word 'sufficient' highlights the fact that the distribution depends on the data $\mathbf{x}$ only through the function $\mathbf{t}$ (the term $h(\mathbf{x})$ is just a normalizing constant). From the definition we easily see that the sufficient statistics here are additive: If $\mathbf{x}_1, ..., \mathbf{x}_n$ are i.i.d. random variables from an exponential family distribution, then the joint distribution $p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{x}_n|\boldsymbol{\theta})$ is also in the exponential family and the corresponding sufficient statistic is given by $\sum_{n=1}^{N} \mathbf{t}(\mathbf{x}_n)$.

A worthy note is that the exponential family is in the heart of an important class of statistical models called *generalized linear models* (GLM) [53], and various other models. Many commonly used distributions, for example gamma, Gaussian and Dirichlet distributions belong to the exponential fam-

ily[1]. This serves as a good motivation for its applications, as the generic form in (2.3) can be used to derive many useful results for a wide range of distributions at the same time.

A prior distribution $p(\boldsymbol{\theta})$ is called a *conjugate prior* of a sampling distribution $p(\mathbf{x}|\boldsymbol{\theta})$ if the posterior $p(\boldsymbol{\theta}|\mathbf{x})$ is of the same functional form as the prior. If the prior is chosen as conjugate, then recognizing the posterior by looking at the unnormalized density (2.2) is straightforward. Although in general the existence of a conjugate prior is not guaranteed, for exponential family distributions it is always available [8].

2.2.2 Posterior inference

For almost all but the most simple models exact calculation of the posterior is not possible. In this case finding the solution has to be approached by other means. An important class of simulation methods called Markov chain Monte Carlo (MCMC) algorithms can be used to draw correlated samples from a Markov chain [58] that has the posterior distribution as its equilibrium distribution. Popular examples of such algorithms include the Gibbs sampling- [27] and the Metropolis-Hastings algorithms [31, 54]. These methods are guaranteed to asymptotically sample from the true posterior, but there are questions one has to address: How many samples are needed to represent the posterior in sufficient accuracy? Has the algorithm converged to the equilibrium distribution reasonably well? How to tune the parameters so that the sampling is efficient and the parameter space is explored thoroughly? Also, in general the computational cost of the sampling algorithms is often significant even after tuning the parameters for optimal performance.

The other alternative approach is to approximate the posterior by some simpler distribution. A classical example of this is the Laplace approximation [67], where the approximating distribution is Gaussian. The upside of the approximations is the simple interpretation of the model parameters and usually faster computation time compared to the simulation methods. The downside is that the result is at best still an approximation to the true posterior and the fit might not be appropriate for certain models or use cases. This makes it important to understand the limitations of the approximations when choosing which method to use for posterior inference.

[1]Most of the commonly used exponential family distributions are not traditionally presented in the canonical form in (2.3).

In this thesis we will focus on method called variational inference [6, 10], which is an example of a rather general way to approximate the true posterior by some simpler distribution. An introduction to the method is given in Chapter 3, after which the derived theory will be used for inference in the model proposed in Chapter 5.

2.2.3 Data augmentation

Sometimes, when it is hard to work with a complicated joint distribution $p(\mathbf{x}, \boldsymbol{\theta}) = p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$, the problem may be simplified by *augmenting* the model with so called *latent variables* or *auxiliary variables* $\mathbf{z}$. If the marginal joint density of the augmented model is the same as the joint density in the original model, that is

$$\int p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}) \, \mathrm{d}\mathbf{z} = p(\mathbf{x}, \boldsymbol{\theta}),$$

we can simply use the augmented joint density instead and forget about the latent variables when interpreting the results. This is useful if the augmented posterior $p(\mathbf{z}, \boldsymbol{\theta}|\mathbf{x})$ can be handled easier than the original posterior $p(\boldsymbol{\theta}|\mathbf{x})$. In this thesis we will use this kind of a construct to augment a mixture of Bernoulli distributions with suitable latent variables, which makes the otherwise complicated inference easier.

2.2.4 Plate diagram

Hierarchical Bayesian probability models are often visualized as *directed acyclic graphs* (or DAGs) where the nodes depict the parameters and data, and the edges point out the dependencies between the nodes [14]. A *plate diagram* is a specific style of representing these graphs that we adopt in this thesis.

An example plate diagram of a simple Bayesian model is shown in Figure 2.1. The model could be for example a generative model for N coin tosses where θ denotes the probability of landing heads. The unobserved variables (here the probability parameter θ) are depicted by white nodes and the observed data (the outcomes of the coin tosses) by the shaded nodes. The variables inside the plate are replicated as many times as indicated by the symbol in the corner of the plate. In this thesis we leave the hyperparameters out of the plate diagram to make the figures clearer. Finally,

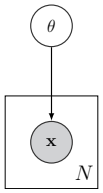

Figure 2.1: A plate diagram of a simple graphical model. White nodes are used to denote the variables in the model while the colored nodes are used for observed data. The variables inside the plate are replicated by the number of times indicated here by the symbol N. The edge from θ to $\mathbf{x}$ is used to specify the dependencies in the model. In the example the joint distribution of the observed data and the model parameters is given by $p(\mathbf{x}, \theta) = p(\theta) \prod_{n=1}^{N} p(x_n|\theta)$.

the edges between the nodes depict the dependency structure in the model. For example, here the joint distribution that specifies the model would be $p(\mathbf{x}, \theta) = p(\theta) \prod_{n=1}^{N} p(x_n|\theta)$. We emphasize the fact that the plate diagram is mainly used for clarifying the model structure and one still needs to specify the distributions of all the variables to have a valid statistical model.

2.3 Some probability distributions

This section is used for reviewing the probability distributions and some of their key properties that will be used in the thesis. Along with the probability density- and mass functions, we are also interested in certain expected values, as they will be required when deriving variational inference for models that use these distributions.

2.3.1 Common distributions

We begin by describing some well known distributions that are extensively used in statistical modeling.

Normal distribution

A normal (or Gaussian) distribution is arguably the most important and well studied distribution used in statistics, machine learning and various

other fields. It arises in countless applications and many natural phenomena which is one of the reasons of its popularity. It is also a fairly well behaving from a computational point of view, which makes it even more appealing for modeling tasks.

A random vector $\mathbf{x} : \Omega \to \mathbb{R}^n$ is said to have a (multivariate) normal distribution with mean vector $\boldsymbol{\mu} \in \mathbb{R}^n$ and positive definite covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$, denoted by $\mathbf{x} \sim \mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, if it has a probability density function

$$p(\mathbf{x}) = (2\pi)^{-\frac{n}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})}, \quad \mathbf{x} = (x_1, ..., x_n)^T.$$

Often in Bayesian statistics, and also later in this thesis, the distribution is parametrized instead by using a *precision matrix* $\boldsymbol{\Lambda}$, which is the inverse of the covariance matrix of $\mathbf{x}$:

$$\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}.$$

Using this parametrization usually leads to slightly more compact notation when doing standard Bayesian calculations.

The mean and the covariance matrix are conveniently given, as the naming proposes, as

$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu}$$

and

$$\mathrm{Cov}(\mathbf{x}) = \boldsymbol{\Sigma} = \boldsymbol{\Lambda}^{-1}.$$

The family of normal distributions is closed under affine transformations: If $\mathbf{x} \in \mathbb{R}^D$, $\mathbf{x} \sim \mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\mathbf{b} \in \mathbb{R}^D$ and $\mathbf{A} \in \mathbb{R}^{N \times D}$ we have

$$\mathbf{b} + \mathbf{A}\mathbf{x} \sim \mathrm{N}(\mathbf{b} + \mathbf{A}\mathbf{x}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T).$$

A special case of the multivariate normal distribution is acquired when there is only one component. In this case we say that X has a (univariate) normal distribution with mean μ and variance σ^2, denoted by $X \sim \mathrm{N}(\mu, \sigma^2)$. Using the same alternative parametrization as in the multivariate case, we denote $\sigma^2 = \tau^{-1}$ and call τ the precision of the distribution. The variance of a univariate Gaussian is simply the second parameter of the distribution:

$$\mathrm{Var}(X) = \sigma^2 = \tau^{-1}.$$

The density function of a univariate Gaussian is usually denoted by ϕ and the corresponding distribution function is given by

$$\Phi(a) = \int_{-\infty}^{a} \phi(x)\,\mathrm{d}x,$$

which cannot be expressed in terms of elementary functions. However, all reasonable statistical software packages provide efficient implementations to evaluate it.

It is a well known result that the marginal distributions of a multivariate normal distribution are univariate normal distributions. In the special case of a multivariate normal distribution with diagonal covariance, the density function also factorizes to the product of the marginal distributions. From a Bayesian point of view, the normal distribution is a conjugate prior for the mean of another normal distribution.

Truncated normal distribution

A Gaussian random vector $\mathbf{x} \sim \mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is truncated to set $\mathcal{C} \subset \mathbb{R}^n$ if the probability density function of $\mathbf{x}$ is given by

$$p(\mathbf{x}) = P(\mathcal{C})^{-1}\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})\mathbb{1}_{\{\mathbf{x}\in\mathcal{C}\}}.$$

The difference to the non-truncated distribution is that the truncated density is normalized by multiplying with the constant $P(\mathcal{C})^{-1}$ and its support is restricted to the set $\mathcal{C}$. We note that other distributions can be truncated in a similar way.

Gamma distribution

A random variable $X : \Omega \to (0, \infty)$ is said to have a Gamma distribution with shape $a > 0$ and rate $b > 0$, denoted by $X \sim \mathrm{Gamma}(a, b)$, if it has a probability density function

$$p(x) = \frac{b^a}{\Gamma(a)} x^{a-1} e^{-bx},$$

where Γ is the Euler gamma function defined by

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x}\,\mathrm{d}x, \quad t > 0.$$

The expected value of a Gamma distributed random variable is given by

$$\mathbb{E}[X] = \frac{a}{b}.$$

Gamma distribution is the conjugate prior for the precision of a univariate normal distribution.

Normal-Gamma distribution

The normal-gamma distribution is used as a prior for a univariate normal distribution with unknown mean and precision. A random vector (μ, λ) has a normal-gamma distribution if it has the joint density

$$p(\mu, \lambda | \mu_0, \beta_0, a_0, b_0) = \mathrm{N}(\mu | \mu_0, (\beta_0 \lambda)^{-1}) \, \mathrm{Gamma}(\lambda | a_0, b_0),$$

where $\mathrm{N}(\cdot|\cdot,\cdot)$ and $\mathrm{Gamma}(\cdot|\cdot,\cdot)$ denote the probability density functions of univariate normal- and gamma distributions respectively. As seen from the definition, the distribution of λ is a gamma distribution and its moments are thus easy to calculate. As for μ, the mean is simply μ_0 and the second moment can be found out by applying the tower property of conditional expectations to the mixed random variable $\mu|\lambda$.

Bernoulli distribution

A Bernoulli random variable can be though of as the outcome of a single experiment with two outcomes. Formally, a random variable $X : \Omega \to \{0, 1\}$ follows a Bernoulli distribution with success probability p, denoted by $X \sim \mathrm{Bernoulli}(p)$, if it has a probability mass function

$$p(x) = p^x (1 - p)^{1-x}.$$

The expected value of X is simply p. The parameter p is often parametrized by taking the *logit-transform* of p:

$$\psi = \mathrm{logit}(p) = \log \frac{p}{1 - p}.$$

The resulting parameter ψ is called *log-odds*. The logit transformation is also the canonical link function for Bernoulli likelihood in the theory of generalized linear models, which partly explains its popularity as the parametrization. There the resulting model is called *logistic regression* [17, 69]. The inverse logit-transformation that defines the model likelihood in ψ is given by the *logistic* (also called *sigmoid*) function $\sigma : \mathbb{R} \to [0, 1]$:

$$p = \sigma(\psi) = (1 + e^{-\psi})^{-1}.$$

Unfortunately the likelihood $p(\psi)$ is not of any form that could be easily combined with a prior distribution to yield an analytically tractable posterior.

In fact, also approximative Bayesian inference requires some effort. In section 2.3.2 we describe a recent data augmentation strategy that deals with this problem.

It should also noted that the log-odds parametrization is not the only one used: Another common parametrization arises from using a *probit* link, specified by

$$\psi = \Phi^{-1}(p),$$

where Φ is the distribution function of a standard normal distribution. This link function results in a generalized linear model known as *probit regression* [12], where the idea itself dates back to the 1930's.

The difference between the logit- and the probit link functions is that the former has slightly flatter tails. Generally, the logit link might be preferred because of the intuitive interpretation of modeling log-odds. In this thesis we choose to use the log-odds parametrization also because of the new data augmentation scheme that makes handling the model with logit parametrization easier than it would be with the probit link.

Multinomial distribution

A random vector $\mathbf{x} : \Omega \to \{(x_1, ..., x_K)^T \in \{0, ..., n\}^K : \sum_{k=1}^K x_k = n\}$ is said to have multinomial distribution with a number of trials $n > 0$ and event probabilities $p_1, ..., p_K$, where $\sum_{k=1}^K p_k = 1$, if it has a probability mass function

$$p(\mathbf{x}) = \frac{\Gamma(\sum_{k=1}^K x_k + 1)}{\prod_{k=1}^K \Gamma(x_k + 1)} \prod_{k=1}^K p_k^{x_k}.$$

We denote this by $\mathbf{x} \sim \text{Mult}(n, \mathbf{p})$. The expected value of each component of a multinomial random vector is given as

$$\mathbb{E}[x_k] = np_k.$$

A multinomial distribution is the distribution for the number of observations in each of the K different categories after n independent trials where the kth category is chosen with probability p_k. A multinomial distribution with one trial is often called a *categorical distribution*.

Dirichlet distribution

A random vector $\mathbf{x} : \Omega \to \{(x_1, ..., x_K)^T \in (0, 1)^K : \sum_{k=1}^K x_k = 1\}$ has a K-dimensional Dirichlet distribution with concentration parameter $\boldsymbol{\alpha} =$

$(\alpha_1, ..., \alpha_K)$, denoted by $\mathbf{x} \sim \text{Dir}(\boldsymbol{\alpha})$, if it has a probability density function

$$p(\mathbf{x}) = \frac{1}{\text{B}(\boldsymbol{\alpha})} \prod_{k=1}^{K} x_k^{\alpha_k - 1}, \quad \mathbf{x} = (x_1, ..., x_n),$$

where the normalizing constant $\text{B}(\boldsymbol{\alpha})$ is given by

$$\text{B}(\boldsymbol{\alpha}) = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k)}.$$

The expected value of each component x_k of $\mathbf{x}$ is given by

$$\mathbb{E}[X_k] = \frac{\alpha_k}{\sum_k \alpha_k}.$$

Additionally, the expected value of the logarithm of each x_k is given by

$$\mathbb{E}[\ln x_k] = \psi(\alpha_k) - \psi(\sum_k \alpha_k),$$

where ψ is the digamma function defined by

$$\psi(x) = \frac{d}{dx} \ln \Gamma(x).$$

Dirichlet distribution is the conjugate prior for a multinomial distribution. We denote a K-dimensional Dirichlet(α,...,α) distribution with SymDir(K,α).

2.3.2 Pólya-Gamma distribution

The Pólya-Gamma distribution was recently introduced by Polson et al. [61] to provide a new data augmentation strategy for Bayesian inference in models with binomial likelihoods. In this thesis we use the augmentation to make variational inference in a Bernoulli mixture model analytically tractable. Below we present the selected properties of the Pólya-Gamma distribution that will be needed in this thesis.

Definition and properties

A random variable $X : \Omega \to (0, \infty)$ has a Pólya-Gamma distribution with parameters $b > 0$ and $c \in \mathbb{R}$ if it has a density function

$$p(x) = \cosh^b(c/2) \frac{2^{b-1}}{\Gamma(b)} \sum_{n=0}^{\infty} (-1)^n \frac{\Gamma(n+2)}{\Gamma(n+1)} \frac{(2n+b)}{\sqrt{2\pi x^3}} e^{-\frac{(2n+b)^2}{8x} - \frac{c^2}{2}x}.$$

Alternatively the distribution can be characterized by the relation

$$X \overset{d}{=} \frac{1}{2\pi^2} \sum_{k=1}^{\infty} \frac{g_k}{(k-1/2)^2 + c^2/(4\pi^2)},$$

where g_k are i.i.d. Gamma$(b, 1)$-distributed random variables and $\overset{d}{=}$ denotes equality in distribution. Despite the fact that the density function looks rather intimidating, the Pólya-Gamma distribution has some attractive properties.

The general Pólya-Gamma distribution arises from exponential tilting of a PG$(b, 0)$-density:

$$\text{PG}(\omega|b, c) = \frac{\exp(-\frac{c^2}{2}\omega)p(\omega|b, 0)}{\mathbb{E}_\omega[\exp(-\frac{c^2}{2}\omega)]}, \tag{2.4}$$

where the expectation in the denominator is taken with respect to a PG$(1, 0)$-distribution and given as

$$\mathbb{E}_\omega[\exp(-\frac{c^2}{2}\omega)] = \cosh^{-b}(c/2).$$

This follows from the Laplace transform of a PG$(1, 0)$ distributed random variable (see Polson et al. for details).

Conveniently, the expectation of a general Pólya-Gamma distribution can be computed analytically. Let $\omega \sim \text{PG}(b, c)$, where $c \neq 0$. Then the expectation of ω is given by

$$\mathbb{E}[\omega] = \frac{b}{2c} \tanh(c/2).$$

The main result in Polson et al. states that binomial likelihoods parametrized by log-odds can be written as a mixture of Gaussians with respect to a Pólya-Gamma distribution:

$$\frac{(e^\psi)^a}{(1+e^\psi)^b} = 2^{-b} e^{\kappa\psi} \int_0^\infty e^{-\omega\psi^2/2} p(\omega)\, \mathrm{d}\omega, \tag{2.5}$$

where $a \in \mathbb{R}$, $\kappa = a - b/2$ and $\omega \sim \text{PG}(b, 0)$. The result can be applied for example to the Bernoulli, binomial and negative binomial likelihoods. In this thesis we will work with the Bernoulli likelihood parametrized with the log-odds ψ, which can be written as

$$\text{Bernoulli}(x|\psi) = \frac{(e^\psi)^x}{1+e^\psi}.$$

We see that this corresponds to the left hand side in (2.5) with $a = x$, $b = 1$ and $\kappa = x - 1/2$.

In practice the result is used as in 2.2.3 by explicitly introducing Pólya-Gamma random variables ω into the model and noting that the unnormalized likelihood with the augmented variables is given by

$$p(x, \omega|\psi) = p(x|\omega, \psi)p(\omega) \propto 2^{-b} e^{\kappa\psi - \omega\psi^2/2} p(\omega).$$

The likelihood is quadratic in ψ and thus placing a Gaussian prior on ψ results in the conditional $p(\psi|\omega, x)$ being a Gaussian as well. On the other hand, the conditional $p(\omega|\psi, x)$ is seen to be in the Pólya-Gamma family since it is acquired by exponential tilting of the $\mathrm{PG}(b, 0)$ prior as in (2.4):

$$p(\omega|\psi, x) = \frac{\exp(-\omega\psi^2/2)p(\omega|b, 0)}{\mathbb{E}_\omega[\exp(-\omega\psi^2/2)]} = \mathrm{PG}(\omega|b, \psi).$$

Knowing these conditional distributions is essential in developing many inference algorithms such as Gibbs sampling and variational inference.

Related work

Since the original paper by Polson et al., the Pólya-Gamma augmentation strategy has found its way into several applications. Zhou et al. [71] proposed a state-of-the-art Bayesian univariate negative binomial regression model with Gibbs sampling and variational inference. Building on top of this work, Klami et al. [45] developed a novel multivariate regression model for count valued data and used it to predict public transport passenger counts in a smart cities application. In 2015, Linderman et al. [50] showed how to use the augmentation with multinomial data and presented applications in correlated topic models and Gaussian processes with multinomial observations among others. Gan et al. [25] augment deep sigmoid belief networks with Pólya-Gamma latent variables to derive efficient inference, which is crucial for deep, multi-layered graphical models. The augmentation has also been used to improve the inference in logistic topic models [16, 72].

Chapter 3

Variational Inference

Assume we have some independently observed data samples from our model, denoted by $\mathbf{x}$, and a set of model parameters and latent variables denoted by $\boldsymbol{\theta}$. *Variational inference* (VI) [6, 10] tries to approximate the true posterior $p(\boldsymbol{\theta}|\mathbf{x})$ with some tractable distribution $q(\boldsymbol{\theta})$, called *variational distribution*, so that the distributions would be as close to each other as possible. In this chapter we present the basic theory of variational inference and also go through the most widely used special case of it called *mean-field approximation*. We also discuss some recent developments which can be used to scale the variational inference further to big data applications. The theory shown here will be used to derive the inference algorithms for the models that are presented in Chapters 4, 5 and 6.

3.1 General variational inference framework

In the following we use the integral symbol to denote either integration or summation for the continuous and discrete parts of the integrand respectively. One common way to find the approximating distribution is to minimize the Kullback-Leibler (KL) divergence of the true posterior p from the variational distribution q. The KL divergence in this case is given by

$$D_{KL}(q||p) = -\int q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta}|\mathbf{x})}{q(\boldsymbol{\theta})} \, \mathrm{d}\boldsymbol{\theta}. \tag{3.1}$$

Without a proof we note that the KL divergence is not a metric since it is not symmetric and does not satisfy the triangle inequality. Also it is always nonnegative and zero only if $p = q$ almost surely.

Rewriting (3.1) and rearranging the terms gives

$$\ln p(\mathbf{x}) = \mathcal{L}(q) + D_{KL}(q||p), \tag{3.2}$$

where the term

$$\mathcal{L}(q) = \int q(\boldsymbol{\theta}) \ln \frac{p(\mathbf{x}, \boldsymbol{\theta})}{q(\boldsymbol{\theta})} \, \mathrm{d}\boldsymbol{\theta} \tag{3.3}$$

is called *variational free energy* or *evidence lower bound* (ELBO).

As the KL divergence is always nonnegative we see that $\mathcal{L}(q)$ gives a lower bound on the log-evidence $\ln p(\mathbf{x})$. Because the log-evidence is a constant, the KL divergence can be minimized by equivalently maximizing $\mathcal{L}(q)$, which shows that this is essentially an optimization problem. As noted, it is trivial that the KL divergence is minimized when $p(\boldsymbol{\theta}|\mathbf{x}) = q(\boldsymbol{\theta})$. However, if the true posterior is intractable, we have to put some constraints on $q(\cdot)$ so that it becomes tractable and at the same provides a good approximation to the true posterior.

On the direction of the KL divergence

Before we go on, it is worth noting that here the forward KL divergence $D_{KL}(q||p)$ is used instead of the reversed one given by $D_{KL}(p||q)$. The reason for this is that the former leads to integrating over $q(\cdot)$, which can be made easy by choosing a simple enough variational approximation $q(\cdot)$, whereas the latter would include the much harder task of integrating over p. The difference between using the forward and the backward KL divergence lies in the fact that they tend to under- and overestimate the posterior variance respectively. Some analysis on why this is indeed the case is provided in [8]. We stress the fact that the underestimation of the posterior variance by the forward KL divergence $D_{KL}(q||p)$ is a drawback of the method, and one should be aware of this theoretical property when using variational inference. One algorithm acquired by using the reverse KL divergence is called *Expectation Propagation* (EP) [55].

3.2 Mean-field variational-Bayes

One way of restricting the form of q is to assume that the variational distribution factors into I independent blocks, that is

$$q(\boldsymbol{\theta}) = \prod_{i=1}^{I} q_i(\boldsymbol{\theta}_i). \tag{3.4}$$

This is the mean-field assumption. It should be noted that the assumption makes no claims about the functional form of the different $q_i(\boldsymbol{\theta}_i)$.

3.2.1 Gradient based optimization

Practically all machine learning problems are solved by optimizing some objective function to find the best parameter values with respect to the chosen objective. Often the objective function is chosen so that it is *convex* (or *concave*), which makes it possible to efficiently use gradient based optimization algorithms to solve the problem. Even if this is not the case, by using gradient based methods one is often guaranteed to find at least a local optimum of the objective. The field that studies this kind of optimization problems is called *convex optimization*.

The convex optimization toolbox can also be employed to solving the variational inference problem. The general idea is to use the ELBO in (3.3) as the objective function that is to be maximized. This is actually not a convex optimization problem but, as mentioned, gradient based methods are still useful. Using the mean-field assumption (3.4) leads to computing the gradient of the ELBO with respect to each of the variational parameters $\boldsymbol{\theta}_i$. In practice one then ends up with a coordinate ascent algorithm for solving the optimal parameters $\boldsymbol{\theta}_i$, which is also what we get in the next section with another derivation. The gradient based approach is explained in more detail for example in [6].

3.2.2 Derivation of the mean-field variational updates

The idea is still to maximize $\mathcal{L}(q)$ with respect to each of the q_i in turn but, by using a clever trick, we can actually avoid taking the gradient of the ELBO altogether. The derivation presented here will follow closely to that shown in [8]. To ease notation, let q_i stand for $q_i(\boldsymbol{\theta}_i)$. Now substituting (3.4) into (3.3) and separating the jth variational density q_j gives

$$
\begin{aligned}
\mathcal{L}(q) &= \int \prod_i q_i \ln p(\mathbf{x}, \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta} - \int \Big[\prod_i q_i \Big] \sum_i \ln q_i \, \mathrm{d}\boldsymbol{\theta} \\
&= \int q_j \Big[\int \prod_{i \neq j} q_i \ln p(\mathbf{x}, \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}_i \Big] \, \mathrm{d}\boldsymbol{\theta}_j - \sum_i \int \Big[\prod_i q_i \Big] \ln q_i \, \mathrm{d}\boldsymbol{\theta} \\
&= \int q_j \Big[\int \prod_{i \neq j} q_i \ln p(\mathbf{x}, \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}_i \Big] \, \mathrm{d}\boldsymbol{\theta}_j - \sum_i \int q_i \ln q_i \, \mathrm{d}\boldsymbol{\theta}_i \\
&= \int q_j \Big[\int \prod_{i \neq j} q_i \ln p(\mathbf{x}, \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}_i \Big] \, \mathrm{d}\boldsymbol{\theta}_j - \int q_j \ln q_j \, \mathrm{d}\boldsymbol{\theta}_j + c_{-j}, \qquad (3.5)
\end{aligned}
$$

where c_{-j} denotes a term that does not depend on q_j. Now define a new distribution $\tilde{p}(\mathbf{x}, \boldsymbol{\theta}_j)$ through

$$
\ln \tilde{p}(\mathbf{x}, \boldsymbol{\theta}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{x}, \boldsymbol{\theta})] + \text{const.}
$$

Here $\mathbb{E}_{i \neq j}[\cdot]$ denotes an expectation with respect to all other variational distributions except q_j, that is

$$
\mathbb{E}_{i \neq j}[\ln p(\mathbf{x}, \boldsymbol{\theta})] = \int \ln p(\mathbf{x}, \boldsymbol{\theta}) \prod_{i \neq j} q_i \, \mathrm{d}\boldsymbol{\theta}_i.
$$

Thus we can write (3.5) as

$$
\int q_j \ln \tilde{p}(\mathbf{x}, \boldsymbol{\theta}_j) \, \mathrm{d}\boldsymbol{\theta}_j - \int q_j \ln q_j \, \mathrm{d}\boldsymbol{\theta}_j + c_{-j}. \qquad (3.6)
$$

Now suppose the variables $q_{i \neq j}$ are fixed and (3.6) has to be maximized over all possible forms of distributions q_j. We recognize that (3.6) is the negative KL divergence between q_j and $\tilde{p}(\mathbf{x}, \boldsymbol{\theta}_j)$ plus the constant that does not depend on q_j. Thus to maximize (3.6) with respect to q_j the KL divergence must be minimized. The minimum is of course reached when $q_j = \tilde{p}(\mathbf{x}, \boldsymbol{\theta}_j)$. This means that the optimal solution for the jth variational distribution can be found with

$$
\ln q_j^*(\boldsymbol{\theta}_j | \mathbf{x}) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{x}, \boldsymbol{\theta})] + \text{const.} \qquad (3.7)
$$

Exponentiating and taking the normalizing constant into account we see that the optimal variational density of the jth parameter block is given by

$$
q_j^*(\boldsymbol{\theta}_j | \mathbf{x}) = \frac{\exp(\mathbb{E}_{i \neq j}[\ln p(\mathbf{x}, \boldsymbol{\theta})])}{\int \exp(\mathbb{E}_{i \neq j}[\ln p(\mathbf{x}, \boldsymbol{\theta})]) \, \mathrm{d}\boldsymbol{\theta}_j}.
$$

We note that in practice it is easier to work with the form in (3.7) as the normalizing constant can often be worked out easily if the functional form of the distribution is recognized.

The solution in (3.7) gives a set of coupled equations where the parameters of each distribution q_i depend on the parameters of the other variational distributions. This suggests an iterative algorithm where the parameters of each q_i are first initialized and then updated in turn according to the equations until convergence. Fortunately, it has been proved that convergence here is guaranteed, but only to a local optimum [43]. Moreover, the convergence is also monotonic, which aids possible debugging in the implementation phase. The lower bound can be conveniently calculated by rewriting the expression in (3.3) as:

$$\mathcal{L}(q) = \mathbb{E}_q\Big[\ln \frac{p(\mathbf{x}, \boldsymbol{\theta})}{q(\boldsymbol{\theta})}\Big] = \mathbb{E}_q[\ln p(\mathbf{x}, \boldsymbol{\theta})] - \sum_i \mathbb{E}_{q_i}[\ln q_i(\boldsymbol{\theta}_i)].$$

3.3 Stochastic and incremental variational inference

The standard variational inference procedure requires a full pass through all available data before the updates to the parameters can be made. For small datasets this is not a problem, but for larger datasets or cases where the inference has to be faster, a few modifications have been proposed. To understand these, we distinguish the *local- and global variational parameters* of the variational approximation from each other. A local variational parameter ϕ_n is a parameter that is directly associated with the corresponding data point $\mathbf{x}_n$. In practice these are often unobserved latent variables. On the other hand, the global variational parameters $\boldsymbol{\theta}$ are parameters which are shared between several data points. The ideas which the following algorithms are based on were first presented in the case of the conceptually quite similar *Expectation Maximization* (EM) algorithm [20] by Neal and Hinton in 1998 [57].

Stochastic variational inference (SVI) [32] works by considering only a *mini-batch*, denoted by B, of uniformly sampled data points of the original data during each iteration. The local variational parameters $\hat{\boldsymbol{\phi}}_B$ that correspond to the data points in the mini-batch are updated, after which an intermediate estimate $\hat{\boldsymbol{\theta}}_B$ of the global variational parameters based on B is calculated. This intermediate estimate is then combined with the current

Algorithm 1: Stochastic variational inference (SVI)

input : Data $\mathbf{X}$, initial values for global variational parameters $\boldsymbol{\theta}^{(0)}$.
output: Variational parameters.

Choose a step size function μ.
while *ELBO not converged* **do**
    Sample mini-batch B uniformly from $\mathbf{X}$.
    Compute local variational parameters $\hat{\boldsymbol{\phi}}_B$ for data points in B.
    Compute intermediate global variational parameters $\hat{\boldsymbol{\theta}}_B$ based on the mini-batch specific local variational parameters $\hat{\boldsymbol{\phi}}_B$.
    Update the estimate for the global variational parameters by setting $\boldsymbol{\theta}^{(t)} \leftarrow (1 - \mu(t))\boldsymbol{\theta}^{(t-1)} + \mu(t)\hat{\boldsymbol{\theta}}_B$.
end

estimate of the global parameters $\boldsymbol{\theta}^{(t-1)}$ to yield an updated estimate $\boldsymbol{\theta}^{(t)}$ of the global parameters by weighting each term accordingly with a step-size $\mu(t)$ that depends on the iteration t:

$$\boldsymbol{\theta}^{(t)} \leftarrow (1 - \mu(t))\boldsymbol{\theta}^{(t-1)} + \mu(t)\hat{\boldsymbol{\theta}}_B.$$

Pseudocode for this procedure is shown in Algorithm 1. Using SVI requires the specification of the batch size and the step-size, which makes it implementation slightly more difficult compared to the default VI algorithm.

Incremental variational inference (IVI) has lately been used as another way to scale up variational inference [3, 37]. It is based on exploiting the additivity property of sufficient statistics in exponential family distributions. The data is first divided into J batches $\{B_j\}_{j=1}^{J}$ and each full iteration of the algorithm consists of processing the batches in the following way. First we subtract the batch specific sufficient statistics $\mathbf{t}^j(\mathbf{X})$ from the full dataset statistics:

$$\mathbf{t}(\mathbf{X}) \leftarrow \mathbf{t}(\mathbf{X}) - \mathbf{t}^j(\mathbf{X}).$$

We then update the batch specific local variational parameters $\boldsymbol{\phi}_{B_j}$, calculate the new batch subset statistics $\mathbf{t}^j(\mathbf{X})$ and add these back to the full dataset statistics:

$$\mathbf{t}(\mathbf{X}) \leftarrow \mathbf{t}(\mathbf{X}) + \mathbf{t}^j(\mathbf{X}),$$

After this we update the global variational parameters $\boldsymbol{\theta}$ based on the full dataset statistics and move on to the next batch B_{j+1}, which is processed in

Algorithm 2: Incremental variational inference (IVI)

input : Data $\mathbf{X}$, initial values for local variational parameters ϕ.
output: Variational parameters.

Divide sample indices $1, ..., N$ into batches B_j, $j = 1, ..., J$.
Calculate initial expected batch specific sufficient statistics $\mathbf{t}^j(\mathbf{X})$, expected full dataset sufficient statistics $\mathbf{t}(\mathbf{X})$ and the global variational parameters $\boldsymbol{\theta}$.
while *ELBO not converged* **do**
    for $j = 1$ *to* J **do**
        $\mathbf{t}(\mathbf{X}) \leftarrow \mathbf{t}(\mathbf{X}) - \mathbf{t}^j(\mathbf{X})$.
        Compute local variational parameters $\boldsymbol{\phi}_{B_j}$ corresponding to data points in B_j and update the expected batch sufficient statistics $\mathbf{t}^j(\mathbf{X})$.
        $\mathbf{t}(\mathbf{X}) \leftarrow \mathbf{t}(\mathbf{X}) + \mathbf{t}^j(\mathbf{X})$.
        Update global variational parameters $\boldsymbol{\theta}$ based on the full dataset sufficient statistics $\mathbf{t}(\mathbf{X})$.
    end
end

the same way. During each full sweep over the data the global parameters are thus updated J times, once after processing each batch. Again, the variational parameters are updated until convergence of the ELBO. Pseudocode for the algorithm is shown in Algorithm 2.

The upside of the IVI as compared to the SVI is that there is no need to specify any additional parameters for the optimization. Also, the optimization process is completely deterministic which makes the algorithm in principle more stable and the lower bound is guaranteed to increase monotonically unlike in SVI. Moreover, the implementation of the procedure is very easy and only comes at the cost of slightly increased space complexity of the algorithm as we need to store the subset statistics for each batch. For these reasons, we adopt the IVI in our implementation for the model proposed in Chapter 5. In the same chapter we will also empirically compare the performance of IVI against the default VI in the case of a Gaussian mixture model.

Chapter 4

Gaussian and Bernoulli Mixture Models

Mixture models refer to models that are of the form

$$p(\mathbf{x}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k p(\mathbf{x}|\boldsymbol{\theta}_k), \tag{4.1}$$

where $0 \leq \pi_k \leq 1$ and $\sum_k \pi_k = 1$. The observations are acquired by *mixing* the *base distributions* $p(\mathbf{x}|\boldsymbol{\theta}_k)$ with the weights π_k. This provides a natural clustering framework: We think of the weight π_k as probability of picking a mixture component, here called *cluster*, indexed by k. The observation is then generated from the base distribution $p(\mathbf{x}|\boldsymbol{\theta}_k)$ corresponding to the chosen cluster. Our task is then, given the observations, to infer the cluster probabilities π_k and the parameters of the base distributions $p(\mathbf{x}|\boldsymbol{\theta}_k)$. The most common way to infer the parameters of a general mixture model is the EM-algorithm mentioned earlier. For practical reasons we write the generative model as an augmented model where the chosen cluster of each data point $\mathbf{x}$ is represented by a corresponding K-dimensional latent vector $\mathbf{z}$ for which the kth component is 1 if the chosen cluster was k and 0 otherwise. Using this formulation we can equivalently write (4.1) as

$$p(\mathbf{x}|\boldsymbol{\theta}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\boldsymbol{\theta}, \mathbf{z})$$
$$= \sum_{\mathbf{z}} p(\mathbf{z}) \prod_{k=1}^{K} p(\mathbf{x}|\boldsymbol{\theta}_k)^{z_k}. \tag{4.2}$$

This corresponds to the data augmentation in (2.2.3), but here also the distribution of the latent variables is of interest; It will characterize the cluster probabilities of the individual data points used for training the model.

Next we will present generative formulations for two common mixture models: the Gaussian- and Bernoulli mixture models. This is aimed to be an introductory chapter where the main purpose is to establish the notation, and also to introduce the models individually. After this it is straightforward to approach the main goal of the thesis: To merge the two models and develop efficient variational inference for the joint mixture, which will be done in Chapter 5.

4.1 Gaussian mixture model

A *Gaussian mixture model* (GMM) is acquired from (4.1) when the base distributions are Gaussian. The GMM is the most common of all mixture models used in various applications. It dates back a long time [19, 60] and has been researched extensively. Newer applications include for example speech recognition [62], image segmentation [29], anomaly detection [48] and deep GMMs [59].

In this thesis we present a special case of the GMM where the covariances of each of the mixture components are diagonal, thus assuming the data dimensions to be independent. This is done for computational reasons: Instead of the full covariance matrices we only need to estimate the diagonal elements. This also simplifies the calculations that are required to derive the inference algorithm.

4.1.1 The generative model

In addition to the base Gaussians, we also need to specify the priors for the Gaussian parameters and the mixture weights. Below we use the superscript

(G) for the observations $\mathbf{x}^{(G)}$ to highlight the fact that they are assumed to be Gaussian. We will be using corresponding superscripts later in the thesis for Bernoulli- and multinomial distributions as well.

We now specify a D_G-dimensional Bayesian Gaussian mixture model with diagonal covariances and K mixture components by setting

$$\begin{aligned}
\mathbf{x}_n^{(G)} &\sim \mathrm{N}(\mu_{\mathbf{z}_n}, \Lambda_{z_n}^{-1}), \\
\boldsymbol{\mu}_k &\sim \mathrm{N}(\boldsymbol{\mu}_0, (\beta_0 \boldsymbol{\Lambda}_k)^{-1}), \\
\lambda_{kd} &\sim \mathrm{Gamma}(a_0, b_0), \\
\mathbf{z}_n &\sim \mathrm{Mult}(1, \boldsymbol{\pi}), \\
\boldsymbol{\pi} &\sim \mathrm{SymDir}(K, \alpha_0),
\end{aligned} \tag{4.3}$$

for $n = 1, 2, ..., N$, $k = 1, 2, ..., K$ and $d = 1, 2, ..., D_G$. Here we denoted $\boldsymbol{\Lambda}_k = \mathrm{diag}(\lambda_{k1}, ..., \lambda_{kD_G})$. The scalars a_0, b_0, α_0 and β_0 are positive hyperparameters and $\boldsymbol{\mu}_0$ is a real vector used as the hyperparameter for the cluster means. It should be noted that the distributions of both $\mathbf{x}_n^{(G)}$ and $\boldsymbol{\mu}_k$ factorize into a product of Gaussians, as the components are in both cases independent. A Normal-Gamma prior is put on each (μ_{kd}, λ_{kd}) to exploit conjugacy. The one-out-of-K coded unobserved latent vectors $\mathbf{z}_n$ indicate which cluster the nth data point belongs to. A conjugate Dirichlet prior is chosen for the multinomial parameter $\boldsymbol{\pi}$. The plate diagram illustrating the model is shown in Figure 4.1.

Let us denote $\mathbf{X}^{(G)} = ((\mathbf{x}_1^{(G)})^T, ..., (\mathbf{x}_N^{(G)})^T)^T$ and $\mathbf{Z} = (\mathbf{z}_1^T, ..., \mathbf{z}_N^T)^T$ for the full data matrix and the matrix of the one-out-of-K coded latent variables of cluster memberships respectively. We also let $\boldsymbol{\Lambda}$ and $\boldsymbol{\mu}$ to stand for the collection of the parameters $\boldsymbol{\Lambda}_k$ and $\boldsymbol{\mu}_k$ $(k = 1, 2, ..., K)$ respectively.

The joint distribution of the full model factorizes to the conditionals as

$$p(\mathbf{X}^{(G)}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = p(\mathbf{X}^{(G)} | \mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(\mathbf{Z} | \boldsymbol{\pi}) p(\boldsymbol{\pi}) p(\boldsymbol{\mu} | \boldsymbol{\Lambda}) p(\boldsymbol{\Lambda}),$$

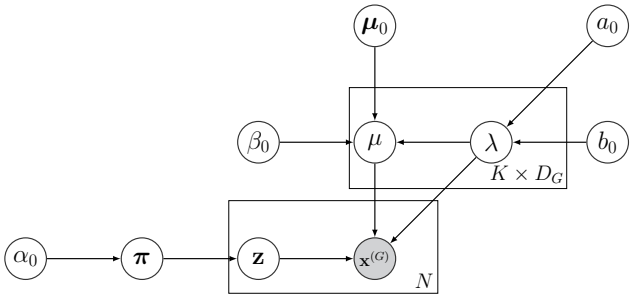

Figure 4.1: The plate diagram for a Bayesian Gaussian mixture model with diagonal covariance. For each of the K Gaussian clusters the means are drawn from a multivariate Gaussian prior and the precisions for individual components from a Gamma prior. A Dirichlet prior is put on the mixture weights $\boldsymbol{\pi}$. An observation is sampled by first drawing a cluster c from the multinomial specified by $\boldsymbol{\pi}$ and finally sampling $\mathbf{x}^{(G)}$ from the Gaussian specified by c.

where the factors are given by

$$
\begin{aligned}
p(\mathbf{X}^{(G)}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \prod_{n=1}^{N} \prod_{k=1}^{K} \prod_{d=1}^{D_G} \mathrm{N}(x_{nd}^{(G)}|\mu_{kd}, \lambda_{kd}^{-1})^{z_{nk}}, \\
p(\mathbf{Z}|\boldsymbol{\pi}) &= \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}}, \\
p(\boldsymbol{\pi}) &= \frac{\Gamma(\alpha_0 K)}{\Gamma(\alpha_0)^K} \prod_{k=1}^{K} \pi_k^{\alpha_0 - 1}, \\
p(\boldsymbol{\mu}|\boldsymbol{\Lambda}) &= \prod_{k=1}^{K} \prod_{d=1}^{D_G} \mathrm{N}(\mu_{kd}|\mu_{0d}, (\beta_0 \lambda_{kd})^{-1}), \\
p(\boldsymbol{\Lambda}) &= \prod_{k=1}^{K} \prod_{d=1}^{D_G} \mathrm{Gamma}(\lambda_{kd}|a_0, b_0).
\end{aligned}
$$

The distributions in (4.3) specify the generative Gaussian mixture model. Variational inference for this model will be a special case of the inference algorithm we present in Section 5.2.

4.2 Bernoulli mixture model

Bernoulli mixture model (BMM) is another example of a specific mixture model that has made its way into real world applications. It has been proposed for example as model for clustering binary images [44] and various other tasks in pattern recognition, such as feature selection, classification or dimensionality reduction [64].

4.2.1 The generative model

In this thesis we consider a Bernoulli mixture model where the output dimensions are assumed independent. The generative model for a D_B-dimensional Bernoulli mixture with K components is specified by

$$\begin{aligned} x_{nd}^{(B)} &\sim \text{Bernoulli}(p_{\mathbf{z}_n d}), \\ p_{kd} &= \sigma(\psi_{nk}) = (1 + e^{-\psi_{nk}})^{-1}, \\ \psi_{kd} &\sim \text{N}(\nu_{kd}, 1), \\ \nu_{kd} &\sim \text{N}(\nu_0, \gamma_0^{-1}), \\ \mathbf{z}_n &\sim \text{Mult}(1, \boldsymbol{\pi}), \\ \boldsymbol{\pi} &\sim \text{SymDir}(K, \alpha_0), \end{aligned} \tag{4.4}$$

for $n = 1, 2, ..., N$, $k = 1, 2, ..., K$ and $d = 1, 2, ..., D_B$. The distributions associated to the clusters, $\boldsymbol{\pi}$ and latent vectors $\mathbf{z}_n$ are the same as in the Gaussian mixture model. In fact this part could be the same for any mixture model. The Bernoulli distributions are parametrized by log-odds, with a Gaussian prior for the corresponding parameters ψ.

The formulas in (4.4) describe the generative process but, as mentioned earlier in Chapter 2, posterior inference poses problems because of the Bernoulli likelihood. To overcome this, we use the data augmentation described in Section 2.3. More specifically, we introduce a set of Pólya-Gamma latent variables

$$\omega_{nd} \sim \text{PG}(1, 0)$$

to the model, which makes our variational approach to posterior inference analytically tractable as the conditionals for ψ and ω will both be known. The plate diagram corresponding to the augmented model is shown in Figure 4.2.

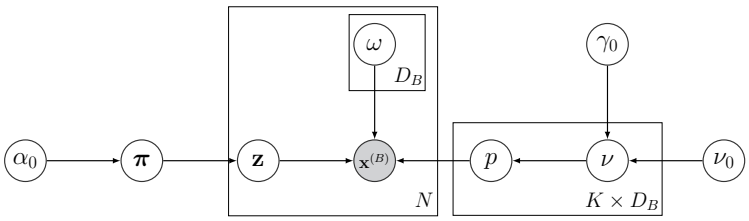

Figure 4.2: The plate diagram of a Bernoulli mixture model augmented with Pólya-Gamma latent variables. The cluster specific Bernoulli success probabilities p_{kd} depend on Gaussian variables ψ_{kd} through the logistic transformation $p_{kd} = \sigma(\psi_{kd}) = (1+e^{-\psi_{nk}})^{-1}$. The data augmentation scheme introduces Pólya-Gamma random variables ω for each dimension of every data point to enable analytically tractable variational inference.

Denoting $\boldsymbol{\psi}$, $\boldsymbol{\nu}$ and $\boldsymbol{\omega}$ for the collection of parameters ψ_{kd}, ν_{kd} and ω_{nd} respectively, the joint distribution of the parameters in the augmented model factorizes as

$$p(\mathbf{X}^{(B)}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\psi}, \boldsymbol{\nu}, \boldsymbol{\omega}) = p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\psi}, \boldsymbol{\omega})p(\mathbf{Z}|\boldsymbol{\pi})p(\boldsymbol{\pi})p(\boldsymbol{\psi}|\boldsymbol{\nu})p(\boldsymbol{\nu})p(\boldsymbol{\omega}).$$

The distributions $p(\mathbf{Z}|\boldsymbol{\pi})$ and $p(\boldsymbol{\pi})$ are exactly the same as in the GMM case in (4.3), and the other terms are given by

$$\begin{aligned}
p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\psi}, \boldsymbol{\omega}) &= \prod_{n=1}^{N}\prod_{k=1}^{K}\prod_{d=1}^{D_B}\left[2^{-1}e^{\kappa_{nd}\psi_{kd}-\omega_{nd}\psi_{kd}^2/2}\right]^{z_{nk}}, \\
p(\boldsymbol{\psi}|\boldsymbol{\nu}) &= \prod_{k=1}^{K}\prod_{d=1}^{D_B}\mathrm{N}(\psi_{kd}|\nu_{kd}, 1), \\
p(\boldsymbol{\nu}) &= \prod_{k=1}^{K}\prod_{d=1}^{D_B}\mathrm{N}(\nu_{kd}|\nu_0, \gamma_0^{-1}) \\
p(\boldsymbol{\omega}) &= \prod_{n=1}^{N}\prod_{d=1}^{D_B}\mathrm{PG}(\omega_{nd}|1, 0).
\end{aligned}$$

Here we used the identity in (2.5) for writing the augmented Bernoulli

likelihood $p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\psi}, \boldsymbol{\omega})$ as

$$\begin{aligned}\prod_{n=1}^{N}\prod_{k=1}^{K}\prod_{d=1}^{D_B}\Big[\text{Bernoulli}(\sigma(\psi_{kd}))\Big]^{z_{nk}} &= \prod_{n=1}^{N}\prod_{k=1}^{K}\prod_{d=1}^{D_B}\Big[\frac{(e^{\psi_{kd}})^{x_{nd}^{(B)}}}{1+e^{\psi_{kd}}}\Big]^{z_{nk}} \\ &= \prod_{n=1}^{N}\prod_{k=1}^{K}\prod_{d=1}^{D_B}\Big[2^{-1}e^{\kappa_{nd}\psi_{kd}-\omega_{nd}\psi_{kd}^2/2}\Big]^{z_{nk}},\end{aligned}$$

where $\kappa_{nd} = x_{nd}^{(B)} - 1/2$.

We have now specified a mixture model also for binary outputs. Now that we have the basics down, it is time to proceed to combine the Gaussian- and Bernoulli mixture models into a single mixture model.

Chapter 5

Mixture Model for Heterogeneous Data

A lot of work has been put into designing scalable clustering models. For example, Hore et al. [33] proposed a clustering model that scales to massive data by running k-means on disjoint subsets of the data, after which the found cluster ensemble is combined to form a global set of centroids. While easily scalable, this approach is clearly heuristic and does not offer the same intuitive explanation of the data as a probabilistic model would. Feldman et al. [24] propose scalable inference for Gaussian mixtures using coresets (a small weighted subset of the data) [1], but they focus solely on maximum likelihood estimation [15] and the EM-algorithm. Verbeek et al. [68] and Thiesson [66] discuss fast, large scale mixture modeling, but again both use variations of the EM-algorithm instead of fully Bayesian inference.

Variational Bayesian methods have been used for mixture models as well; Subedi et al. [65], employ the standard VI framework to do a normal-inverse Gaussian mixture model. Most of the scalable VI solutions seem to rely on stochastic variational inference [38, 70] or sequential variational approximation [49]. Sudderth et al. [37] use incremental variational inference, but without the faster parallel version we choose employ later in the chapter. Archembeau et al. [3] propose a truly scalable, distributed version of IVI for latent Dirichlet allocation [11], but implementing their algorithm requires some extra work because of issues like synchronization etc. We will discuss this approach briefly in Section 5.3.1.

In this chapter we present the main contribution of the thesis: A practical and efficient fully Bayesian mixture model that fits modeling scenarios where

the data is heterogeneous. Using recent advantages in machine learning and Bayesian statistics, we provide a state-of-the-art inference algorithm for our model that is both easy to implement and efficiently scales up to big data.

Consider a situation where we want to cluster data that has both numeric and binary dimensions. A common way is to interpret the numeric dimensions as Gaussian and the binary dimensions clearly correspond to Bernoulli data. This kind of a data is represented in the form $[\mathbf{X}^{(G)}\, \mathbf{X}^{(B)}]$, where $[\cdot, \cdot]$ denotes row-wise concatenation of the Gaussian and Bernoulli data matrices. The data is clearly heterogeneous and the theoretically correct solution for modeling it is to specify a generative probabilistic model that takes into account the different likelihoods of the data dimensions. If we assume that all of the output dimensions are independent, we achieve this simply by combining the Gaussian- and Bernoulli mixture models already presented in Chapter 4 into one single joint mixture model.

5.1 Specifying the joint mixture

Tying two mixture models together through the shared latent variables $\mathbf{z}$ is in principle easy. We use the notation established in Chapter 4 and simply write the joint density that combines the Gaussian- and Bernoulli mixtures as

$$\begin{aligned} p(\mathbf{X}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\psi}, \boldsymbol{\nu}, \boldsymbol{\omega}) = p(\mathbf{X}^{(G)}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\psi}, \boldsymbol{\omega}) \\ \times p(\mathbf{Z}|\boldsymbol{\pi}) p(\boldsymbol{\pi}) p(\boldsymbol{\mu}|\boldsymbol{\Lambda}) p(\boldsymbol{\Lambda}) p(\boldsymbol{\psi}|\boldsymbol{\nu}) p(\boldsymbol{\nu}) p(\boldsymbol{\omega}), \end{aligned}$$

where the factors are given as in (4.3) and (4.4). The plate diagram corresponding to this joint density is shown in Figure 5.1. We call the joint mixture model simply GBMM for *Gaussian- and Bernoulli Mixture Model*.

By looking at the plate diagram or the joint density we indeed see that the outputs of the Gaussian and Bernoulli parts are conditionally independent given the latent vectors $\mathbf{z}$. This means that only in the inference of $\mathbf{Z}$ do we have to think of the two mixtures explicitly. We also note that once the inference has been derived for the GBMM model, it can just as easily be used to fit mixtures that have only Gaussian or Bernoulli components.

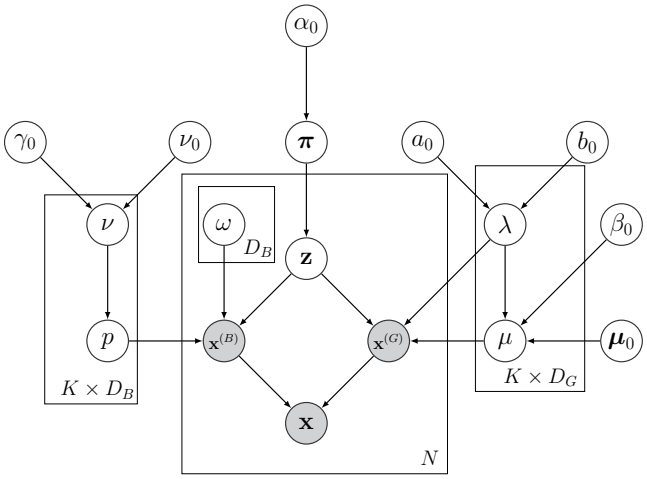

Figure 5.1: Plate diagram of the joint mixture model supporting both Gaussian- and Bernoulli likelihoods. The models are tied together by the latent variables $\mathbf{z}$ indicating the cluster memberships. The outputs from each model are then concatenated to yield the final output $\mathbf{x}$.

5.2 Variational inference for the GBMM model

We use mean-field variational inference to approximate the posterior with

$$p(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\psi}, \boldsymbol{\nu}, \boldsymbol{\omega}|\mathbf{X}) \approx q(\mathbf{Z})q(\boldsymbol{\pi})q(\boldsymbol{\mu}, \boldsymbol{\Lambda})q(\boldsymbol{\psi})q(\boldsymbol{\nu})q(\boldsymbol{\omega}).$$

Because we have assumed the dimensions of the data independent in the generative model, also the variational distributions for the Gaussian and Bernoulli parameters will factorize similarly.

As described in Section 3, we want to find the variational distributions $q_i(\cdot)$ that minimize the KL divergence of the true posterior p from $q(\cdot) = \prod_i q_i(\cdot)$. The calculations are done by applying the formula (3.7) that was derived in Chapter 2 and are shown in detail in Appendix A. The optimal

variational distributions under the mean-field assumption are found to be

$$
\begin{aligned}
q(\mathbf{Z}) &= \prod_{n=1}^{N} \mathrm{Mult}(\mathbf{z}_n|1, \hat{\mathbf{r}}_n), \\
q(\boldsymbol{\pi}) &= \mathrm{Dir}(\boldsymbol{\pi}|\hat{\boldsymbol{\alpha}}), \\
q(\boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \prod_{k=1}^{K}\prod_{d=1}^{D_G} \mathrm{N}(\mu_{kd}|\hat{m}_{kd}, (\hat{\beta}_k\lambda_{kd})^{-1}) \,\mathrm{Gamma}(\lambda_{kd}|\hat{a}_k, \hat{b}_{kd}), \\
q(\boldsymbol{\psi}) &= \prod_{k=1}^{K}\prod_{d=1}^{D_B} \mathrm{N}(\psi_{kd}|\hat{\nu}_{kd}, \hat{\tau}_{kd}^{-1}), \\
q(\boldsymbol{\nu}) &= \prod_{k=1}^{K}\prod_{d=1}^{D_B} \mathrm{N}(\nu_{kd}|\hat{n}_{kd}, \hat{\gamma}^{-1}), \\
q(\boldsymbol{\omega}) &= \prod_{n=1}^{N}\prod_{d=1}^{D_B} \mathrm{PG}(\omega_{nd}|1, \hat{e}_{nd}),
\end{aligned} \tag{5.1}
$$

The related local variational parameters are given by

$$
\begin{aligned}
&\hat{\mathbf{r}}_n = (\hat{r}_{n1}, ..., \hat{r}_{nK})^T, \quad \hat{r}_{nk} = \frac{s_{nk}}{\sum_{i=1}^{K} s_{ni}}, \\
&s_{nk} \propto \exp\Big\{\frac{1}{2}\Big(\sum_{d=1}^{D_G}(\psi(\hat{a}_k) - \ln\hat{b}_{kd}) - \sum_{d=1}^{D_G}(\frac{\hat{a}_k}{\hat{b}_{kd}}(x_{nd}^{(G)} - \hat{m}_{kd})^2 + \frac{1}{\hat{\beta}_k})\Big) \\
&\qquad + \sum_{d=1}^{D_B}\Big[\kappa_{nd}\hat{\nu}_{kd} - \frac{1}{4\hat{e}_{nd}}\tanh(\hat{e}_{nd}/2)(\hat{\tau}_{kd}^{-1} + \hat{\nu}_{kd}^2)\Big] + \psi(\hat{\alpha}_k) - \psi(\sum_{k=1}^{K}\hat{\alpha}_k)\Big\}, \\
&\hat{e}_{nd} = \sqrt{\sum_{k=1}^{K}\hat{r}_{nk}(\hat{\tau}_{kd}^{-1} + \hat{\nu}_{kd}^2)},
\end{aligned} \tag{5.2}
$$

and the global variational parameters by

$$
\begin{aligned}
\hat{\boldsymbol{\alpha}} &= (\hat{\alpha}_1, ..., \hat{\alpha}_K)^T, \quad \hat{\alpha}_k = \alpha_0 + \hat{R}_k, \\
\hat{R}_k &= \sum_{n=1}^{N} \hat{r}_{nk}, \\
\hat{m}_{kd} &= \frac{\beta_0 \mu_{0d} + \sum_{n=1}^{N} \hat{r}_{nk} x_{nd}^{(G)}}{\hat{R}_k + \beta_0}, \\
\hat{\beta}_k &= \hat{R}_k + \beta_0, \\
\hat{a}_k &= \frac{\hat{R}_k}{2} + a_0, \\
\hat{b}_{kd} &= b_0 + \frac{\hat{R}_k \beta_0}{2(\hat{R}_k + \beta_0)} \Big(\mu_{0d} - \frac{\sum_{n=1}^{N} \hat{r}_{nk} x_{nd}^{(G)}}{\hat{R}_k} \Big)^2 + \frac{1}{2} \sum_{n=1}^{N} \hat{r}_{nk} (x_{nd}^{(G)})^2 \\
&\quad - \frac{(\sum_{n=1}^{N} \hat{r}_{nk} x_{nd}^{(G)})^2}{2\hat{R}_k}, \\
\hat{\nu}_{kd} &= \frac{\sum_{n=1}^{N} \hat{r}_{nk} \kappa_{nd} + \hat{n}_{kd}}{\sum_{n=1}^{N} \Big[\hat{r}_{nk} \frac{1}{2\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2) \Big] + 1}, \\
\hat{\tau}_{kd} &= \sum_{n=1}^{N} \Big[\hat{r}_{nk} \frac{1}{2\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2) \Big] + 1, \\
\hat{n}_{kd} &= \frac{\hat{\nu}_{kd} + \gamma_0 \nu_0}{1 + \gamma_0}, \\
\hat{\gamma} &= 1 + \gamma_0,
\end{aligned}
\tag{5.3}
$$

where $\kappa_{nd} = x_{nd}^{(B)} - 1/2$. The local parameters $\hat{r}_{nk}$ are usually called *responsibilities*, that is, they express the probability for the nth data point to belong to the kth cluster. The hyperbolic tangent functions, that are somewhat untypical in this kind of calculations, show up in the formulas as the expected value of the Pólya-Gamma random variables with respect to the variational distribution:

$$
\mathbb{E}_q[\omega_{nd}] = \frac{1}{2\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2).
$$

By looking at the equations in (5.3), it is easy to see that the expected full dataset sufficient statistics required for updating the global variational parameters are given by

$$
\begin{aligned}
&\hat{\mathbf{t}}_1 \in \mathbb{R}^{K\times D_G}, \qquad [\hat{\mathbf{t}}_1]_{kd} = \sum_n \hat{r}_{nk} x_{nd}^{(G)}, \\
&\hat{\mathbf{t}}_2 \in \mathbb{R}^{K\times D_G}, \qquad [\hat{\mathbf{t}}_2]_{kd} = \sum_n \hat{r}_{nk} (x_{nd}^{(G)})^2, \\
&\hat{\mathbf{t}}_3 \in \mathbb{R}^{K\times D_B}, \qquad [\hat{\mathbf{t}}_3]_{kd} = \sum_n \hat{r}_{nk} \kappa_{nd}, \\
&\hat{\mathbf{R}} \in \mathbb{R}^{K}, \qquad \hat{R}_k = \sum_n \hat{r}_{nk}, \\
&\hat{\mathbf{E}} \in \mathbb{R}^{K\times D_B}, \qquad [\hat{\mathbf{E}}]_{kd} = \sum_n \hat{r}_{nk} \frac{1}{2\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2).
\end{aligned} \tag{5.4}
$$

We use the term 'expected sufficient statistics' to refer to weighting the mixture specific sufficient statistics by weights acquired by taking the expectation of the unobserved latent vectors $\mathbf{z}$.

For efficient computation of the lower bound we additionally compute the entropy term

$$\sum_{n,k} \hat{r}_{nk} \ln \hat{r}_{nk},$$

and the bound for $\boldsymbol{\omega}$ given by

$$\sum_{n,d} -\ln(\cosh(\hat{e}_{nd}/2)) + \frac{\hat{e}_{nd}}{4} \tanh(\hat{e}_{nd}/2)$$

while updating the responsibilities $\hat{\mathbf{r}}$.

The standard variational inference algorithm for the GBMM model is illustrated in Algorithm 3. Convergence of the algorithm is assessed by monitoring the change in the evidence lower bound given by

$$
\begin{aligned}
&\mathbb{E}_q[\ln p(\mathbf{X}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \Lambda, \boldsymbol{\psi}, \boldsymbol{\nu}, \boldsymbol{\omega})] - \mathbb{E}_q[\ln q(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \Lambda, \boldsymbol{\psi}, \boldsymbol{\nu}, \boldsymbol{\omega})] \\
=&\mathbb{E}_q[\ln p(\mathbf{Z}|\boldsymbol{\pi})] - \mathbb{E}_q[\ln q(\mathbf{Z})] + \mathbb{E}_q[\ln p(\boldsymbol{\pi})] - \mathbb{E}_q[\ln q(\boldsymbol{\pi})] \\
&+ \mathbb{E}_q[\ln p(\boldsymbol{\mu}, \boldsymbol{\Lambda})] - \mathbb{E}_q[\ln q(\boldsymbol{\mu}, \boldsymbol{\Lambda})] + \mathbb{E}_q[\ln p(\boldsymbol{\psi}|\boldsymbol{\nu})] - \mathbb{E}_q[\ln q(\boldsymbol{\psi})] \\
&+ \mathbb{E}_q[\ln p(\boldsymbol{\nu})] - \mathbb{E}_q[\ln q(\boldsymbol{\nu})] + \mathbb{E}_q[\ln p(\boldsymbol{\omega})] - \mathbb{E}_q[\ln q(\boldsymbol{\omega})] \\
&+ \mathbb{E}_q[\ln p(\mathbf{X}^{(G)}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \mathbb{E}_q[\ln p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\omega}, \boldsymbol{\psi})],
\end{aligned} \tag{5.5}
$$

where the expressions for the individual terms are given in Appendix A.

Algorithm 3: Variational Inference for Gaussian- and Bernoulli Mixture Model (VI-GBMM)

input : Data $\mathbf{X}$, initial values for $\hat{\mathbf{r}}$, $\hat{\boldsymbol{\nu}}$ and $\hat{\boldsymbol{\tau}}$.
output: Global variational parameters and responsibilities $\hat{\mathbf{r}}$.
Initialize expected sufficient statistics $\hat{\mathbf{t}}_1, \hat{\mathbf{t}}_2, \hat{\mathbf{t}}_3, \hat{\mathbf{R}}, \hat{\mathbf{E}}$.
while *ELBO not converged* **do**
    Update the global parameters using current expected sufficient statistics.
    Update the responsibilities $\hat{\mathbf{r}}$ using current estimates for the global parameters and calculate new expected sufficient statistics $\hat{\mathbf{t}}_1, \hat{\mathbf{t}}_2, \hat{\mathbf{t}}_3, \hat{\mathbf{R}}$ and $\hat{\mathbf{E}}$.
end

On the number of mixture components

A common issue with optimizing using maximum likelihood based methods and the EM-algorithm is that the clusters can shrink so that they contain only one of the data points. In this case the variance inside the cluster becomes zero and this usually causes numerical issues in the objective function to be optimized. Also, increasing the number of mixture components always makes the fit to the data better, but at the cost of making the model more complex and prone to *overfitting*. This is why choosing the correct number of mixture components is especially important for these methods.

Bayesian methods and variational inference on the other hand deal with these problems to some extent. An empty cluster will just become equal to the specified prior. The Bayesian treatment of a mixture model also comes with an automatic method of balancing between the model complexity and fitting the data, because the full distributions of the parameters are used. More details on this can be found in the variational inference section in the book by Bishop [8]. We also mention that the modeling can be done so that the number of mixture components is automatically inferred from the data. One example of a model that does this is called *Dirichlet process mixture model*, for which variational methods can still be used for inference [9].

5.3 Scaling up computation

We now show how to use the incremental variational inference that was introduced in Chapter 3 in the case of the GBMM model. Additionally, we parallelize the calculation of the local variational parameters and subset statistics for faster computation speed for large datasets.

5.3.1 Parallel incremental variational inference

All of the expected sufficient statistics in (5.4) are clearly additive, which makes incremental variational inference straightforward. We split the data points into J batches $B_1, ..., B_J$ and use the notation in (5.4) with superscript j to denote for the corresponding expected batch sufficient statistics. Because of the additivity property of the sufficient statistics, it is also trivial to implement the algorithm partly in parallel: For each batch B_j in turn, the updates for the responsibilities $\hat{r}$ and the computation of the batch sufficient statistics are easily distributed over multiple processor cores on a modern multicore CPU. This implementation of the IVI algorithm is depicted in Algorithm 4.

General discussion

Our method cannot process the whole data in parallel, since the full dataset sufficient statistics need to be updated after processing each batch. In the case of really big data, machine learning algorithms are usually deployed on a computing cluster consisting of several machines and in this case allowing parallel processing of the whole data would be beneficial. One such approach was introduced in recent work by Archembeau et al. [3], where they present a framework for *distributed asynchronous incremental variational inference*. The authors demonstrate the algorithm for latent Dirichlet allocation topic model and report a significant boost in performance compared to the standard IVI. The idea behind the algorithm is roughly the following:

1. The J data batches are distributed to J different worker machines along with the current estimates for the global variational parameters.

2. Each of the J workers compute the local variational parameters and batch sufficient statistics for their respective batch using the local copy

of the global variational parameters and, when ready, send the results to a master computing node.

3. Once the master node receives a new set of sufficient statistics from some worker, update the global variational parameters and send the worker the updated global variational parameters.

As the authors point out, the algorithm design becomes more complicated as the algorithm has to be robust to delays and inaccurate updates. Nevertheless, distributed algorithms like this are becoming increasingly important and will only attract more attention in the future. Currently for example the Apache Spark[1] framework provides a good interface for deploying distributed, large scale machine learning algorithms.

We would like to stress that our implementation of the GBMM algorithm is not distributed, but merely uses multiple CPU cores on a single machine. Our implementation can be thought to be somewhere in between of the framework proposed by Archembeau et al. and the standard IVI. To recap the upsides of our choice of implementation, our algorithm is:

1. Faster than the standard IVI algorithm since it computes the responsibilities and sufficient statistics inside each batch in parallel.

2. Trivial to implement in a multicore, one machine setup.

5.3.2 Initializing the cluster assignments

A bad initialization of the model can easily lead to slow convergence or reaching a (bad) local optima. Initializing the global variational parameters is often done by subjective choice, for example with the aid of calculating some simple initial statistics from the data. For initializing the hard cluster assignments required by our implementation of the GBMM algorithm we present the following alternatives based on the popular k-means clustering algorithm [51]:

1. The basic version of k-means, which works by finding clusters that minimize the squared distances of the data points to their chosen cluster centers. The algorithm starts with random data points chosen as

[1] `http://spark.apache.org/`

Algorithm 4: Parallel Incremental Variational Inference for Gaussian- and Bernoulli Mixture Model (P-IVI-GBMM)

input : Data $\mathbf{X}$, initial values for $\hat{\mathbf{r}}$, $\hat{\boldsymbol{\nu}}$ and $\hat{\boldsymbol{\tau}}$, number of batches J, number of CPU's C.

output: Global variational parameters and responsibilities $\hat{\mathbf{r}}$.

Divide sample indices $1, ..., N$ into batches B_j, $j = 1, ..., J$ and further into CPU specific batches $B_{j,c}$, $c = 1, ..., C$.
Calculate initial expected batch- and full dataset sufficient statistics.
while *ELBO not converged* **do**
    for $j = 1$ *to* J **do**
        Update global parameters using current full dataset expected sufficient statistics.
        Subtract current batch statistics from the full dataset statistics: $\hat{\mathbf{t}}_1 \leftarrow \hat{\mathbf{t}}_1 - \hat{\mathbf{t}}_1^j, \quad \hat{\mathbf{t}}_2 \leftarrow \hat{\mathbf{t}}_2 - \hat{\mathbf{t}}_2^j, \quad \hat{\mathbf{t}}_3 \leftarrow \hat{\mathbf{t}}_3 - \hat{\mathbf{t}}_3^j,$ $\hat{\mathbf{R}} \leftarrow \hat{\mathbf{R}} - \hat{\mathbf{R}}^j, \quad \hat{\mathbf{E}} \leftarrow \hat{\mathbf{E}} - \hat{\mathbf{E}}^j.$
        for $c = 1$ *to* C *in parallel* **do**
            Update the responsibilities $\hat{\mathbf{r}}_n$ for $n \in B_{jc}$ using current estimates for the global parameters and calculate expected sufficient statistics $\hat{\mathbf{t}}_1^{jc}, \hat{\mathbf{t}}_2^{jc}, \hat{\mathbf{t}}_3^{jc}, \hat{\mathbf{R}}^{jc}$ and $\hat{\mathbf{E}}^{jc}$ for batch $B_{j,c}$.
        end
        Sum up the calculated batch statistics from different CPU's: $\hat{\mathbf{t}}_1^j \leftarrow \sum_c \hat{\mathbf{t}}_1^{jc}, \quad \hat{\mathbf{t}}_2^j \leftarrow \sum_c \hat{\mathbf{t}}_2^{jc}, \quad \hat{\mathbf{t}}_3^j \leftarrow \sum_c \hat{\mathbf{t}}_3^{jc},$ $\hat{\mathbf{R}}^j \leftarrow \sum_c \hat{\mathbf{t}}_3^{jc}, \quad \hat{\mathbf{E}}_j \leftarrow \sum_c \hat{\mathbf{E}}^{jc}.$
        Add the current batch statistics back to the full dataset statistics: $\hat{\mathbf{t}}_1 \leftarrow \hat{\mathbf{t}}_1 + \hat{\mathbf{t}}_1^j, \quad \hat{\mathbf{t}}_2 \leftarrow \hat{\mathbf{t}}_1 + \hat{\mathbf{t}}_2^j, \quad \hat{\mathbf{t}}_3 \leftarrow \hat{\mathbf{t}}_3 + \hat{\mathbf{t}}_3^j,$ $\hat{\mathbf{R}} \leftarrow \hat{\mathbf{R}} + \hat{\mathbf{R}}^j, \quad \hat{\mathbf{E}} \leftarrow \hat{\mathbf{E}} + \hat{\mathbf{E}}^j.$
    end
end

the initial cluster centers and, if chosen badly, this can result in poor clustering.

2. The k-means++ algorithm [4]. This is a more robust alternative to the basic k-means. The only difference is that the initial cluster centers are chosen so that centers far away from each other are preferred with high probability.

3. The method by Boutsidis et al. [13]: The k-means or k-means++ algorithm, applied to a new dataset $\mathbf{X}'$ acquired by reducing the dimensionality of the original dataset $\mathbf{X}$ by a suitable *random projection*. We refer to this method as RP-k-means(++). While suboptimal, this method is suitable also for larger data matrices, and thus suits our goal of building a scalable clustering model. We will elaborate on the details below.

Random projection based initialization with k-means(++)

The random projection-based dimensionality reduction aims to preserve the Euclidean distances that the k-means uses for clustering. The dimensionality reduction is done by creating a new, lower dimensional data matrix of appropriately scaled values. In more detail, the k-means objective can be written as

$$L(\mathbf{X}, \mathbf{A}) = \|\mathbf{X} - \mathbf{A}\mathbf{A}^T\mathbf{X}\|_F^2, \tag{5.6}$$

where $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the data matrix, $\|\cdot\|_F$ denotes the Frobenius norm defined by $\|\mathbf{X}\|_F = \sqrt{\sum_{ij} x_{ij}^2}$ and $\mathbf{A} \in \mathbb{R}^{n \times k}$ is a cluster indicator matrix with the property $\mathbf{A}_{nk} = 1/\sqrt{s_k}$ if and only if $\mathbf{x}_n$ belongs to cluster k. The number of points that belong to cluster k is denoted by s_k. The random projection method works by forming a new data matrix $\mathbf{X}' \in \mathbb{R}^{n \times d'}$ by computing $\mathbf{X}' = \mathbf{X}\mathbf{R}$, where $\mathbf{R} \in \mathbb{R}^{d \times d'}$ is a random matrix computed by setting each element to $\pm 1/\sqrt{d'}$ uniformly at random. After this we apply k-means or k-means++ to the new matrix $\mathbf{X}'$ and as a result get a cluster indicator matrix $\mathbf{A}'$, which is also a solution to the original clustering problem. The random projection based clustering can be compared to k-means clustering on the original data by evaluating the objective in (5.6) for both methods.

We conducted two simple experiments on separate datasets to empirically validate that the RP-k-means++ clustering produces results comparable to the baseline k-means++ algorithm. The datasets are described below:

1. The first data set is a synthetic one, where we generated data from 10 different 100-dimensional Gaussian clusters with spherical covariance matrices. We set the standard deviations to 2, that is $\Sigma_k = \sqrt{2}\mathbf{I}$ for all k. The mean vector of cluster k was chosen by uniform random sampling with replacement from the set $\{1, ..., 100\}$ for all k.

2. The other dataset we used is the popular MNIST dataset of handwritten digits[2]. It contains 60,000 examples, each given as a 784-dimensional vector of grayscale pixel values corresponding to 28×28 pixel images of numerical digits 0-9. We used the first 5,000 examples in the data and removed columns (pixels) that were identically zero, which resulted in a $5{,}000 \times 663$ data matrix to be used for clustering. The natural number of clusters here is 10, as there are 10 different digits in the data.

For the experiments we used the more robust k-means++ version with $k = 10$ clusters. We used 5 restarts for each of the k-means++ runs and the solution corresponding to the best value of the objective function was chosen as the representative result. The quality of the solutions was measured by the normalized objective function $L(\mathbf{X}, \mathbf{A})/\|\mathbf{X}\|_F^2$, which we plot against the number of dimensions after the dimensionality reduction. We also plot the baseline solution of k-means++ applied to the original data as a constant horizontal line. For each number of dimensions kept, we ran the RP-k-means++ clustering 10 times and recorded the average value of the normalized objective function, along with the minimum and maximum values for comparison.

The results are shown in Figures 5.2 and 5.3. We see that if the dimensionality of the random projection is at least a fourth of the original data dimension, the found clusterings seem to be comparable to those by the k-means++ algorithm on the original data. The advantage is that the random projection allows the k-means algorithm to run with improved computational and memory cost, which is beneficial if the dimensionality of the original data matrix $\mathbf{X}$ is high. We note though, that the RP-k-means++ method is stochastic in nature and, as seen in the plot, the quality of the solutions do vary from run to run.

5.3.3 Performance evaluation using toy data

The proposed GBMM algorithm was implemented using the R-programming language[3]. Table 5.3.3 lists the packages, all of which are available from

[2]Dataset publicly available from `http://yann.lecun.com/exdb/mnist/`. Accessed on 03–08–2016.

[3]`https://www.r-project.org/`

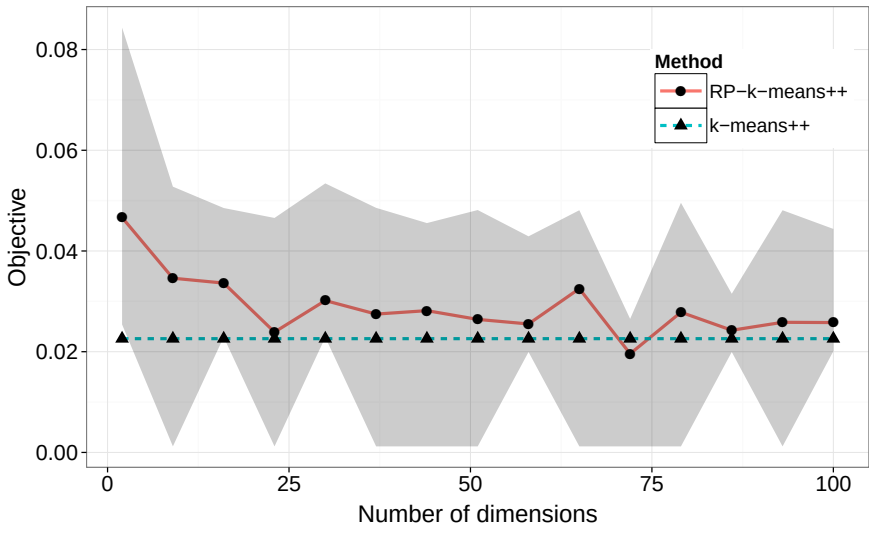

Figure 5.2: Benchmark on synthetic data. The plot shows the normalized k-means objective function plotted against the data dimension after random projection. The red line shows the average value of the normalized objective across 10 RP-k-means++ runs while the shadowing illustrates the range on the quality of the solution. The blue horizontal line shows the objective for a baseline solution by a k-means++ run on the original full data. The RP-k-means++ solution yields results comparable to the baseline k-means++ solution, and already using a fourth of the original dimensions seems to result in good performance on average. There is some variation in the quality of the solutions because of the stochastic nature of the algorithm.

CRAN[4], that were used as a part of the implementation. All tests were run on a Linux machine with 8 Intel Xeon E5649 processor cores and 16 gigabytes of RAM.

Experiment setup and objectives

The aim here is to empirically show the advantages of incremental variational inference over the standard variational inference. We created the data by generating 20 million data points from a 2-dimensional Gaussian mixture with 10 components all having diagonal covariance. The generated data thus

[4] `https://cran.r-project.org/`

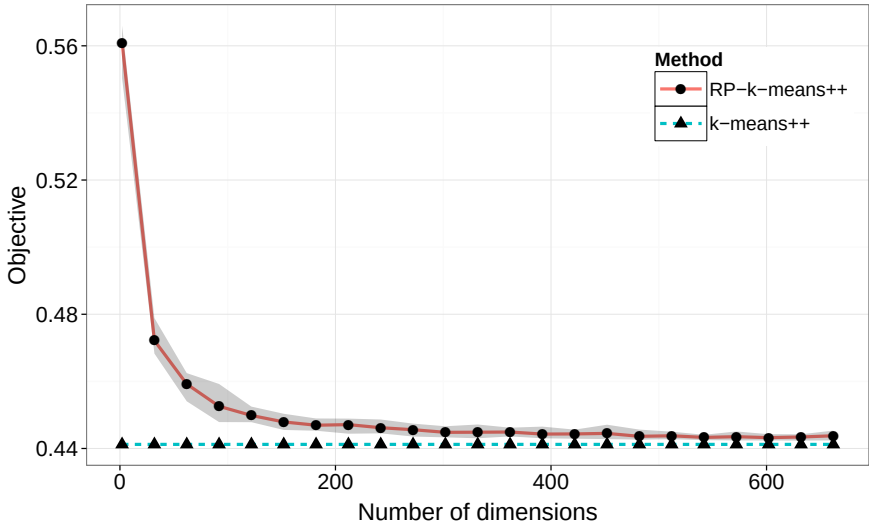

Figure 5.3: Benchmark on the MNIST-data. The visuals in the plot are to be interpreted similarly as in Figure 5.2. Here the RP-k-means++ solution clearly achieves performance comparable to the baseline k-means++ algorithm when around 300 dimensions are used in the random projection matrix $\mathbf{X}'$. A good solution is achieved already with a smaller number of dimensions. Importantly, the quality of the RP-k-means++ solutions seems to be rather stable across the different runs.

Package	Description / Used for
Rcpp	Interface for implementing the computationally heavy parts in C++. These include updating the responsibilities and expected sufficient statistics, calculating the lower bound, and initialization.
parallel	Parallel computing.
Matrix	Sparse matrix support.

Table 5.1: A list of R-packages that were used in implementing the GBMM-algorithm.

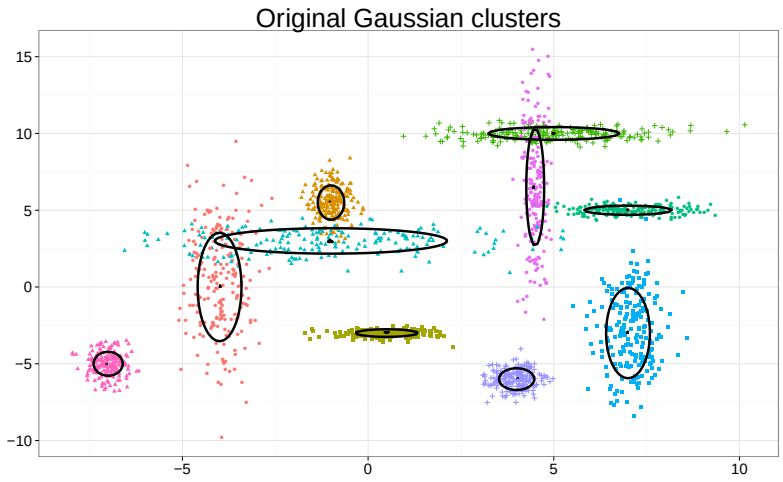

Figure 5.4: The first 2000 data points drawn from a 2-dimensional Gaussian mixture with 10 components. The different mixture components are highlighted by the colors, the contour lines and the point shapes.

matches our specification for the GMM in (4.3) and the inference task should therefore be easy. The first 2000 data points are visualized in Figure 5.4.

The initialization for the responsibilities $\hat{r}$ was done by hard cluster assignments using the k-means implementation in the default R package 'stats'. The algorithm was considered converged if the change in the lower bound between two full iterations was less than $\epsilon = 1$.

Comparing the batch sizes in IVI

We compare our incremental variational inference implementation of the GMM with different number of batches J, run on all 8 CPU cores. This experiment has two main purposes. Firstly, we want to know how much using a larger number of batches improves the convergence speed of the algorithm, as measured by the number of iterations required until convergence. On the other hand we are interested in finding the number of batches for which the computation time is the fastest. This is not clear a priori, as using a lower number of batches results in faster parallel processing because the communication overhead is smaller. Figure 5.5 shows the convergence of the lower bound as function of iteration and Figure 5.6 the computation time for different batch sizes. Note that batch size $J = 1$ corresponds to the standard

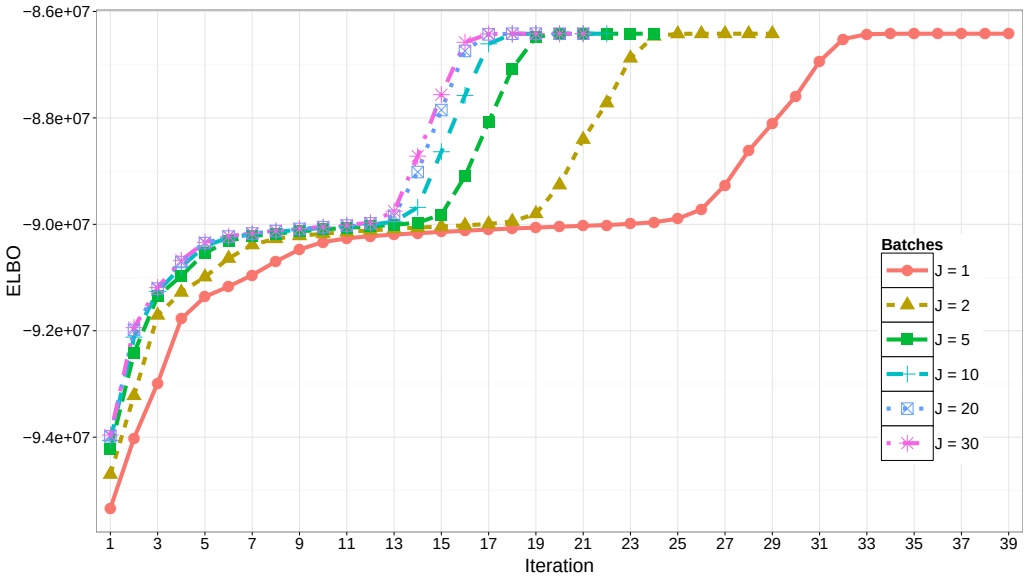

Figure 5.5: Convergence of the evidence lower bound as a function of the algorithm iteration for different number of batches used. Using more batches result in faster convergence of the lower bound, but after $J > 5$ batches are used, the total computation time starts to suffer because of the additional communication overhead, as seen in Figure 5.6.

variational inference.

From the lower bound we clearly see that convergence is the faster the more batches are used. However, after 20 batches the improvement after adding more batches is negligible. Importantly, the incremental variant with sufficiently many batches ($J \geq 5$) seems to be able to exit a local optimum faster, as seen in Figure 5.5 where the ELBO increases slowly for several iterations in the middle of the optimization process. Increasing the batch size too much increases the computation time as shown in Figure 5.6. This happens because the different CPU's are working with increasingly smaller amounts of data, which produces communication overhead as the algorithm ends up using too much time fetching the results as compared to the actual computation time. Based on these observations we conclude that the best batch size here is $J = 5$, which leads to the fastest overall computation time and it also converges quickly in terms of the required iterations. Notably the incremental algorithm proves to be superior to the standard variational

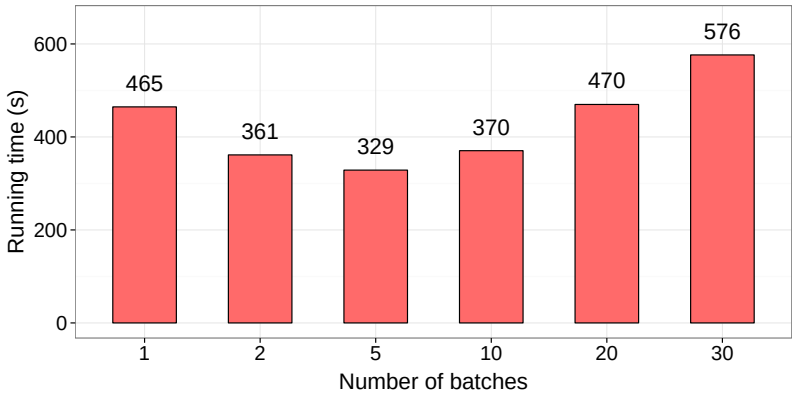

Figure 5.6: Running time of the GMM algorithm for different number of batches. The fastest solution is acquired with a batch size $J = 5$. When too many batches are used, the communication overhead that comes from handling the subsets is bigger than the convergence speedup acquired from updating the global parameters more often.

inference corresponding to the case $J = 1$.

As for the actual quality of the clustering, the cluster allocations of the different data points after the k-means initialization and the final iteration of the incremental GMM with batch size of $J = 5$ are shown in Figure 5.7. We see that while k-means produces a decent initialization, it is unable to find the correct cluster structure present in the data. Our GMM implementation does considerably better and finds the correct model parameters and maximum a posteriori estimates for almost all of the data points.

Parallel computation vs. a single CPU core

We ran the same learning task with $J = 1$ and $J = 5$ batches using 1 and 8 CPU cores for both alternatives to measure the effect of using multiple cores to the total computation time. The results are shown in Figure 5.8. Using 5 batches is faster with both CPU settings due to the smaller number of iterations required for convergence. As expected, parallel computation provides a major speedup compared to the 1 CPU implementation. Parallel computing with 5 batches is almost 7 times faster than the 1 core 1 batch setup. The advantage of using incremental variational inference also shows clearly: The 5 batch setup is 39 and 27 percent faster than when using the

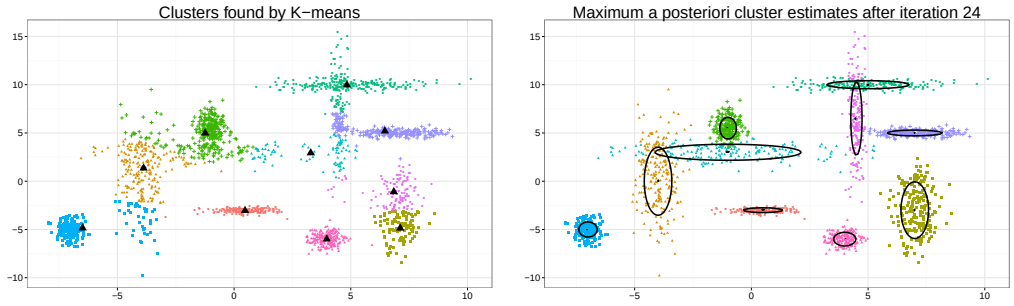

Figure 5.7: Left: Initialization of the cluster assignments by the k-means algorithm. The black triangles depict the cluster centers and the coloring and point labels the cluster memberships (note that the coloring is different to that in Figure 5.4). K-means is not able to find the elongated data clouds corresponding to the different mixture components correctly. Right: Maximum a posteriori estimates of cluster memberships after final iteration of the incremental GMM algorithm with $J = 5$ batches. The GMM accurately finds the correct cluster structure and Gaussian parameters as supposed to, since the data generating process matches the model.

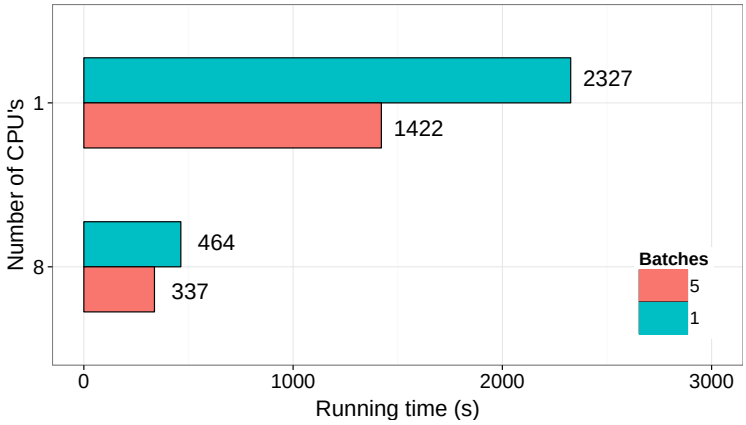

Figure 5.8: Total computation times of the incremental GMM algorithm using different number of batches and CPU cores. Using incremental variational inference speeds up computation with both CPU settings, but most of the performance gain comes from the parallel implementation.

1 batch version with 1 and 8 CPU cores respectively.

Conclusion

We observed that the incremental variational inference with a suitably chosen batch size outperforms the baseline standard variational inference by a clear margin independent of the CPU setup. This is because the convergence of the lower bound is faster due to the more frequent updates of the global parameters. We also find that our multicore implementation of the algorithm provides a major performance boost, given that the data size is big enough so that communication overhead does not dominate over the computation time. As the implementation of the (multicore) incremental variational inference in these kind of models is easy, we find no reason why it should not be the default implementation.

5.4 Roundup of key methodology

We conclude the Chapter with a recap of the methods we used in the modeling. Our main contribution is the development of an efficient and scalable mixture model for heterogeneous data. The key properties of the model and our inference algorithm are:

1. The model supports efficient inference for both Bernoulli and Gaussian data.

2. Our initialization using random projection k-means++ works for arbitrary high dimensional data.

3. We provided derivations for a state-of-the-art incremental variational inference algorithm.

4. Additionally, we showed how the algorithm can be easily implemented on a modern multicore system for increased performance.

Together, these points make up for an algorithm that can be used in many practical clustering applications. We emphasize the fact that the presented inference represents state of the art in terms of efficiency. Also, we are not aware of previous work that would use parallelized IVI in the way we do.

Chapter 6

Extending to Multinomial Data

In this chapter we extend the GBMM model to support categorical data. This means that instead of simple binary values the responses are chosen from C categories, where $C \geq 2$. The two most common models for this task are the multinomial logit- [2] and probit [39, 41] models, which are extensions of the respective models in the binary case. Both of the models are used in various applications, especially in the fields of econometrics and social sciences. Examples include modeling exchange rate policy classification [22] and accident-injury severities [52], demand forecasting [18], analyzing panel data [42] and voting research [21]. As most of the work with these models is done in methodology-wise rather conservative domains, and also simply because the models are hard, the practical use of the logit- and probit models has so far been mostly concentrated on standard statistical analysis instead big data- and machine learning applications. In the univariate case the models generally seem to give similar results, but it has been argued that in the multivariate setup this might depend on the data [30].

Recently, the Pólya-Gamma augmentation strategy was extended also to the multinomial case [50], but we were not aware of this work by the time we considered how to extend to the multinomial case. As a consequence, we choose to use the multinomial probit model, which is more approachable than the standard multinomial logit model due to the model specification through normal distributions. At the end of the chapter we will present the basic ideas of the Pólya-Gamma augmented strategy for the multinomial case, as we currently see this as the best way to do the multinomial mixture model.

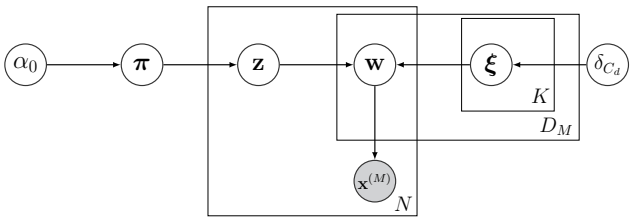

Figure 6.1: Plate diagram of a probit multinomial mixture model. The probit augmentation is done by introducing a Gaussian latent vectors $\mathbf{w}$ for each data point and data dimension. The prior of each $\mathbf{w}$ depends on the chosen cluster indicated by $\mathbf{z}_n$ and hence the data is conditional on $\mathbf{z}$ only through $\mathbf{w}$. The class label chosen by each $x_{nd}^{(M)}$ is given by $\arg\max_c\{w_{ndc}|c = 1, ..., C_d\}$.

6.1 Multinomial probit model

We consider again the general mixture model framework with the same notation as in the previous chapters. Let the number of the dimensions be D_M and denote the value of the dth dimension and the nth sample by $x_{nd}^{(M)}$. We augment the model with a set of Gaussian latent vectors

$$\mathbf{w}_{nd} \sim \mathrm{N}(\boldsymbol{\xi}_{\mathbf{z}_n d}, \mathbf{I}).$$

As usual, we want conjugacy and specify a Gaussian prior on the mean vectors $\boldsymbol{\xi}$:

$$\boldsymbol{\xi}_{\mathbf{z}_n d} \sim \mathrm{N}(0, \delta_{C_d}^{-1}\mathbf{I}).$$

Here δ_{C_d} is a precision hyperparameter that depends on the number of categories in the multinomial corresponding to the dth dimension, denoted by C_d. We get the actual data by setting

$$x_{nd}^{(M)} = \arg\max_c w_{ndc}.$$

In other words, we set $x_{nd}^{(M)}$ to be the index for which the corresponding dimension of the latent vector $\mathbf{w}_{nd}$ is the largest. The plate diagram corresponding to the probit multinomial mixture model is shown in Figure 6.1.

It should be noted that the model is not identifiable, as shifting the mean of the latent Gaussians $\mathbf{w}_{nd}$ does not change the likelihood. This would be important if the latent vectors had some natural interpretation in a specific application. Some ways to address this issue are described in [39]. In our

mixture model we are mostly interested in good cluster allocations and as such the meaning of the latent variables is not especially interesting, which is why do not consider this to be a problem.

Let $\mathbf{W}$ and $\boldsymbol{\Xi}$ stand for the collection of parameter vectors $\mathbf{w}_{nd}$ and $\boldsymbol{\xi}_{nd}$ respectively. The full likelihood of the model can then be written as

$$p(\mathbf{X}^{(M)}, \mathbf{W}, \boldsymbol{\Xi}, \mathbf{Z}, \boldsymbol{\pi}) = p(\mathbf{X}^{(M)}|\mathbf{W})p(\mathbf{W}|\mathbf{Z}, \boldsymbol{\Xi})p(\boldsymbol{\Xi})p(\mathbf{Z}|\boldsymbol{\pi})p(\boldsymbol{\pi}),$$

where the factors $p(\mathbf{Z}|\boldsymbol{\pi})$ and $p(\boldsymbol{\pi})$ are again exactly the same as in the previous chapters. Denote $\mathcal{C}_{nd} = \{\mathbf{w} \in \mathbb{R}^{C_d} | w_i < w_c \quad \forall i \neq c, \quad x_{nd} = c\}$ for the C_d-dimensional cone that defines the allowed values for $\mathbf{w}_{nd}$. We then have the other factors as

$$\begin{aligned}
p(\mathbf{X}^{(M)}|\mathbf{W}) &= \prod_{n=1}^{N} \prod_{d=1}^{D_M} \mathbb{1}_{\{\mathbf{w}_{nd} \in \mathcal{C}_{nd}\}}, \\
p(\mathbf{W}|\mathbf{Z}, \boldsymbol{\Xi}) &= \prod_{n=1}^{N} \prod_{k=1}^{K} \prod_{d=1}^{D_M} \mathrm{N}(\mathbf{w}_{nd}|\boldsymbol{\xi}_{kd}, \mathbf{I})^{z_{nk}}, \\
p(\boldsymbol{\Xi}) &= \prod_{k=1}^{K} \prod_{d=1}^{D_M} \mathrm{N}(\boldsymbol{\xi}_{kd}|0, \delta_{C_d}^{-1}\mathbf{I}).
\end{aligned}$$

The factor $p(\mathbf{X}^{(M)}|\mathbf{W})$ merely ensures that the latent vectors $\mathbf{w}$ are consistent with the observations.

6.2 Variational inference

Using the mean-field assumption, the posterior is approximated with

$$p(\mathbf{Z}, \boldsymbol{\pi}, \mathbf{W}, \boldsymbol{\Xi}|\mathbf{X}^{(M)}) \approx q(\mathbf{Z})q(\boldsymbol{\pi})q(\boldsymbol{\Xi})q(\mathbf{W}).$$

The variational distribution $q(\boldsymbol{\pi})$ will be exactly as in (5.1). Also $q(\mathbf{Z})$ has the same form as in (5.1), but one additional term resulting from the multinomial part must be added to the proportions s_{nk} in (5.2). Rather tedious derivations, shown in detail in Appendix B, reveal that

$$s_{nk} \propto \exp\Big\{\text{terms in (5.2)} - \frac{1}{2}\sum_{d=1}^{D_M}\Big[\sum_{c=1}^{C_d}(\hat{w^2}_{ndc} - 2\hat{w}_{ndc}\hat{o}_{kdc} + (\hat{o}^2_{kdc} + \hat{\delta}_k^{-1}))\Big]\Big\}. \tag{6.1}$$

and that the best variational distributions for the other terms are given by

$$
\begin{aligned}
q(\boldsymbol{\Xi}) &= \prod_{k=1}^{K}\prod_{d=1}^{D_M} \mathrm{N}(\hat{\boldsymbol{\xi}}_{kd}|\hat{\mathbf{o}}_{kd}, \hat{\delta}_k^{-1}\mathbf{I}), \\
q(\mathbf{W}) &= \prod_{n=1}^{N}\prod_{d=1}^{D_M} \mathrm{N}_{x_{nd}^{(M)}}(\mathbf{w}_{nd}|\hat{\boldsymbol{\xi}}_{nd}, \mathbf{I}),
\end{aligned} \tag{6.2}
$$

where $\mathrm{N}_{x_{nd}^{(M)}}(\cdot|\cdot,\cdot)$ denotes a truncated normal distribution such that the dimension c for which $x_{nd}^{(M)} = c$ is the largest. The parameters of the distributions are given by

$$
\begin{aligned}
\hat{\mathbf{o}}_{kd} &= \frac{\sum_{n=1}^{N} \hat{r}_{nk}\langle \mathbf{w}_{nd}\rangle}{\hat{R}_k + \delta_{C_d}} = \frac{\sum_{n=1}^{N} \hat{r}_{nk}\hat{\mathbf{w}}_{nd}}{\hat{R}_k + \delta_{C_d}}, \\
\hat{\delta}_k &= \hat{R}_k + \delta_{C_d}, \\
\hat{\boldsymbol{\xi}}_{nd} &= \sum_{k=1}^{K} \hat{r}_{nk}\hat{\mathbf{o}}_{kd},
\end{aligned} \tag{6.3}
$$

from which we see that the additional expected sufficient statistic required for updating the global variational parameters is given by $\sum_{n=1}^{N} \hat{r}_{nk}\hat{\mathbf{w}}_{nd}$.

In equations (6.1) and (6.3) we now have a set of expectations given by

$$
\begin{aligned}
\hat{\mathbf{w}}_{nd} &= (\hat{w}_{nd1}, ..., \hat{w}_{ndC_d})^T = (\mathbb{E}_q[w_{nd1}], ..., \mathbb{E}_q[w_{ndC_d}])^T, \\
\hat{w^2}_{ndc} &= \mathbb{E}_q[w_{ndc}^2].
\end{aligned} \tag{6.4}
$$

These are the first and second moments of the different components of the latent vectors $\mathbf{w}_{nd}$. As it turns out, these unfortunately do not have closed form expressions, which means numerical approximations are needed. The required formulas along with the computational challenges associated with the model poses will be discussed in the next section.

6.2.1 Computational challenges

The challenge in evaluating the expectations in (6.4) arises from the fact that this requires evaluation of C_d-dimensional integrals, which are not trivial

because the variational distribution of each $\mathbf{w}_{nd}$ is truncated. Assume now that the observation $x_{nd}^{(M)}$ is equal to c, that is the cth dimension of $q(\mathbf{w}_{nd})$ is the largest. After some manipulation it can be shown that the expectations can be computed with the formulas

$$
\begin{aligned}
\hat{w}_{ndi} &= \hat{\xi}_{ndi} - \mathcal{Z}_{nd}^{-1}\mathbb{E}_u[\phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi}) \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})], \quad \text{for} \quad i \neq c, \\
\hat{w}_{ndc} &= \hat{\xi}_{ndc} + \sum_{l \neq c} (\hat{\xi}_{ndl} - \hat{w}_{ndl}), \\
\hat{w^2}_{ndi} &= \hat{\xi}_{ndi}^2 + 1 - (\hat{\xi}_{ndc} + \hat{\xi}_{ndi})(\hat{\xi}_{ndi} - \hat{w}_{ndi}) \\
&\quad - \mathcal{Z}_{nd}^{-1}\mathbb{E}_u[u\phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi}) \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})], \quad \text{for} \quad i \neq c, \\
\hat{w^2}_{ndc} &= \hat{\xi}_{ndc}^2 + \mathcal{Z}_{nd}^{-1}\mathbb{E}_u[u^2 \prod_{j \neq c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})] + 2\hat{\xi}_{ndc} \sum_{l \neq c} (\hat{\xi}_{ndl} - \hat{w}_{ndl}), \\
\mathcal{Z}_{nd} &= \mathbb{E}_u[\prod_{i \neq c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi})],
\end{aligned} \tag{6.5}
$$

where u denotes a standard normal distributed random variable. These formulas look somewhat daunting, but all of them can be calculated approximately using fairly simple Monte-Carlo integrals.

Even though approximating a single expectation above is in principle easy, we face a much bigger problem in the fact that we would need to evaluate the above expectations at every iteration of the algorithm for all N data samples and D_M dimensions, both of which could be large depending on the application. In principle we could only draw one sufficiently big set of values from the standard normal distribution and use the same values for all computations in (6.5). This would reduce the number of required simulations tremendously. We also note that parallelizing the computations corresponding to different data samples is easy. Still, a good amount of computation would be needed just for this step. Unless we have enough memory available on our machine, we would also want to calculate the expectations only when required, that is when we are updating the responsibilities $\hat{\mathbf{r}}$ and calculating the sufficient statistics.

Our conclusion is that as such the presented way to incorporate the multinomial extension to the GBMM model would be inefficient in practice. We thus decided not to implement support to categorical data into our model.

6.2.2 Multinomial logit via Pólya-Gamma augmentation

As mentioned earlier, recent work by Linderman et al. [50] show how to extend the Pólya-Gamma augmentation to multinomial likelihood. We learned about this work only after working through our multinomial probit solution. We think that the Pólya-Gamma method represents the current state of the art way to handle also the multinomial logit. This is why we briefly go through with the main ideas to show how one should proceed with the modeling when a more efficient algorithm than our multinomial probit is required.

The problem with multinomial likelihood is that it is not immediately clear how the probability mass function

$$p(\mathbf{x}) = \frac{\Gamma(\sum_{k=1}^{K} x_k + 1)}{\prod_{k=1}^{K} \Gamma(x_k + 1)} \prod_{k=1}^{K} p_k^{x_k}$$

could be written in the form required by the Pólya-Gamma identity (2.5). Linderman et al. use a 'stick-breaking' representation of the multinomial likelihood, writing it recursively in terms of binomial distributions as

$$\text{Mult}(\mathbf{x}|n, \mathbf{p}) = \prod_{k=1}^{K-1} \text{Bin}(x_k|n_k, \tilde{p}_k), \tag{6.6}$$

where

$$n_k = n - \sum_{j<k} x_j, \qquad \tilde{p}_k = \frac{p_k}{1 - \sum_{j<k} p_j}, \quad k = 2, ..., K, \tag{6.7}$$

$n_1 = n = \sum_k x_k$ and $\tilde{p}_1 = p_1$. Here the binomial pmf is given by

$$\text{Bin}(x|n, p) = \binom{n}{x} p^x (1-p)^{1-x}$$

and each $\tilde{p}_k$ represents the fraction of the remaining probability mass corresponding to the kth category. The next step is to set $\tilde{p}_k = \sigma(\psi_k)$, after which the likelihood can be written in terms of the vector $\boldsymbol{\psi}$ as

$$\text{Mult}(\mathbf{x}|n, \boldsymbol{\psi}) = \prod_{k=1}^{K-1} \binom{n_k}{x_k} \frac{(e^{\psi_k})^{x_k}}{(1 + e^{\psi})^{n_k}}. \tag{6.8}$$

We note that in our mixture model the multinomial is in fact be categorical distribution, in which case the binomial coefficients $\binom{n_k}{x_k}$ in the above identity would simply cancel out. From equation (6.8) we immediately recognize that when choosing $a_k = x_k$ and $b_k = n_k$ for each $k = 1, ..., K-1$, we can use the identity (2.5) to write the Pólya-Gamma augmented likelihood as

$$p(\mathbf{x}, \boldsymbol{\omega}|\boldsymbol{\psi}) = p(\mathbf{x}|\boldsymbol{\omega}, \boldsymbol{\psi})p(\boldsymbol{\omega}) = \prod_{k=1}^{K-1} \binom{n_k}{x_k} 2^{-b_k} e^{\kappa_k \psi_k - \omega_k \psi_k^2/2} p(\omega_k), \tag{6.9}$$

where $\kappa_k = x_k - n_k/2$ and $p(\omega_k)$ is the density of a $\text{PG}(n_k, 0)$ random variable. From this factorized form we see that specifying a Gaussian prior on each of the ψ_k results in the conditional $p(\boldsymbol{\psi}|\boldsymbol{\omega}, \mathbf{x})$ being a diagonal Gaussian. Moreover, the conditional for $\boldsymbol{\omega}$ is given as the product of Pólya-Gamma distributions: $p(\boldsymbol{\omega}|\boldsymbol{\psi}, \mathbf{x}) = \prod_{k=1}^{K-1} \text{PG}(\omega_k|n_k, \psi_k)$.

It should be noted that the stick-breaking formulation of the multinomial likelihood is asymmetric, which is not the case with the standard multinomial logit- and probit models. In their paper Linderman et al. discuss this, and show that this does not hurt the representational capacity of the model.

The formulation of the augmented multinomial likelihood in (6.9) should result in a straightforward derivation of variational inference for a multinomial mixture model. This could then be combined with a Gaussian mixture model to yield a Gaussian- and multinomial mixture model with the Bernoulli mixture model as its special case. We emphasize again that to our knowledge this is currently the best way to specify a multinomial mixture model for which efficient inference can be developed.

Chapter 7

Application to Public Healthcare Dataset

Some of the most hyped big data applications can be brought under the term *digitalized healthcare*. This means applying data science and machine learning methods for example to help offering patients better treatment, make more accurate diagnoses and to optimize the use of expensive healthcare resources. The socio-economic importance of the possible applications is undoubtedly huge.

In this chapter we use our developed GBMM clustering model to an electronic patient record dataset. We aim to describe the timeline of each patient as a series of clusters found by our algorithm, based on various types of medical data collected from the patients. The found clusters and cluster sequences are analyzed in an exploratory data analysis fashion. Our goal was simply to get a better understanding of the data, and see what kind of patterns we are able to uncover. The motivation for this kind of research is clear, as understanding the treatment periods of the patients better would help in optimizing the resource allocation of the hospital and thus boosting cost efficiency. In this work the focus was explicitly on developing the analysis tools, and as such the results are not analyzed from a medical perspective. The main contribution of our work is that we provide the first tool for understanding the data better. We also demonstrate how this kind of analysis can be used to summarize and draw insights from the data, which we hope will spark interest for further research.

Related work

Some work on analyzing electronic patient record datasets has been presented before. Najjar et al. [56] use a novel variation of the k-prototypes algorithm [36] to cluster medical records of elderly patients that have been diagnosed with heart failure diseases. Their approach is scalable to big data, but not probabilistic. Later, the same authors proposed a joint mixture of Gaussians and multinomials coupled with a mixture of hidden Markov models (HMM) [5] to model the same data. The clustering part using a joint mixture of Gaussians and multinomials corresponds to our ideas, but their inference is done using the EM-algorithm instead of fully Bayesian inference. They also do not discuss anything related to the scalability of their solution. Other examples of related work include text data mining of hospital patient records [7, 47] and pattern analysis in clinical processes [23, 34, 35].

7.1 The dataset

We were provided access to an electronic patient record dataset from a major Finnish hospital. The data consists of time-stamped patient records including diagnoses, results of laboratory experiments and basic information such as age and gender, collected over approximately ten years of time. The information in the dataset is diverse, ranging from simple numeric values to text fields explaining the diagnoses. The dataset provides a good overall snapshot of patients who use hospital services in Finland, as we have data corresponding to all age groups, diseases etc. without any filtering. We would like to point out that detailed data of this nature, collected over such a long period of time and composed to one source is likely to be quite unique, even worldwide. In this study, we restricted our analysis to the approximately 110,000 patients who had been recorded to visit a hospital ward at least twice in the data.

7.1.1 Feature extraction

We chose to model a 7 year period of time for each patient, starting from the moment they first visit any of the hospital wards. We split the 7 year time period in 92 time intervals, each consisting of 28 consecutive days. Our idea is to extract a set of *features* from the data that would somehow characterize the kind of treatment that the patient received in the hospital during the

corresponding 28 day time interval. A feature here is simply another name for each dimension in the data and the resulting data matrix that is fed to the machine learning algorithm is called a *feature matrix*. After extracting the features we cluster the (time interval, patient)-tuples with the idea that the prototype clusters would represent typical one month periods in the hospital.

As we know, our GBMM model supports Gaussian and Bernoulli features. Keeping this in mind, we created a feature matrix for the (patient, time window) tuples that had the following features:

- Features 1-12 are continuous variables thresholded to range $[0, 1]$ that describe the number of days a patient spent at 12 different hospital wards during the 28 day period in question. The transformation is illustrated in Figure 7.1. We use this kind of transformation because we want to clearly separate the endpoints 0 and 28 from the other values, and also more or less equate the values in between.

- Feature 13 is a binary feature for gender.

- Features 14-149 are binary features that indicate whether a certain diagnosis was made during the time interval. We chose to include the most common diagnoses as such, while the rarer ones were labeled as the corresponding root in the diagnosis tree specified by the ICD-10[1] diagnostic tool.

- Features 150-269 are binary features that indicate whether a certain set of laboratory experiments (consisting of many individual experiments) was done during the time interval.

As we have 92 time intervals and around 110,000 patients, the feature matrix is now approximately of size 10,120,000 $\times$ 269, but very sparse since majority of the patients are healthy most of the time. This motivates us to remove all rows that correspond to having no record of activity in the hospital during the time window, essentially meaning the patient was either healthy and at home, or dead during the corresponding time window. This procedure reduces the size of the feature matrix, which will still be sparse, to approximately 1,700,000 $\times$ 269, while giving practically the same results as

[1]The abbreviation ICD stands for 'International Statistical Classification of Diseases and Related Health Problems' and it is used as the standard tool to classify diseases and other health conditions worldwide.

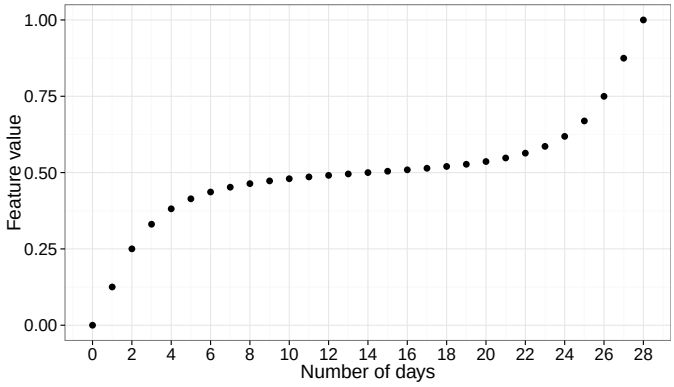

Figure 7.1: An illustration of the mapping used to transform the number of days spent at each hospital ward to the interval $[0, 1]$.

in the case with the zero rows included. Importantly, this step also reduces the computation time significantly.

7.2 Running the algorithm

For initialization we used the random projections k-means algorithm with $d' = 80$ dimensions and 3 random restarts for the ordinary k-means[2]. We then ran the GBMM model for our specified maximum number of 400 iterations. The lower bound continued to increase until the end, as shown in Figure 7.2. This is largely because of the size of the dataset: The effect that the change in individual cluster probabilities have on the ELBO add up. This makes choosing a good convergence threshold harder, as we want to have reasonable computation time while ensuring sufficient convergence. As most of the increase happened during the first 200 iterations and the cluster allocations remained relatively stable after this, we conclude that the algorithm converged well enough for our purposes.

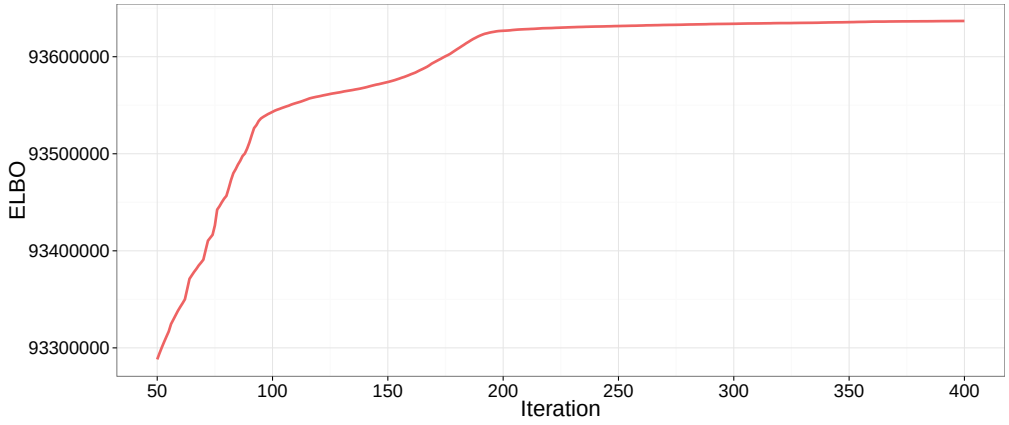

Figure 7.2: The convergence of the lower bound during iterations 50-400 while running the GBMM algorithm for the patient data. The steep increase in ELBO that corresponds to the first 50 iterations is not included in the plot so that the later changes are more visible.

7.3 Results

7.3.1 Visualizing the clusters

After performing maximum a posteriori cluster assignments on the results we have 42 clusters containing more than 50 samples. A sample cluster discovered by the algorithm is shown in Figures 7.3 (features corresponding to the wards) and 7.4 (binary features). The example cluster represents patients who spent most of their time in the neurology ward, but could also visit the internal medicine, skin diseases, or surgery wards. We also get information about the most common diagnoses and laboratory experiments done for patients in the cluster. In Figure 7.4 we have included the reference probability of a laboratory experiment being made when a patient enters the hospital along with the probabilities corresponding to the found cluster. We see that there are some experiments that are clearly more common in the cluster than in the data in general.

[2]Our data matrix is stored as a sparse matrix in R and the k-means++ implementation we used ran very slowly while using this data format. Consequently, we chose to use the ordinary k-means algorithm for the initialization.

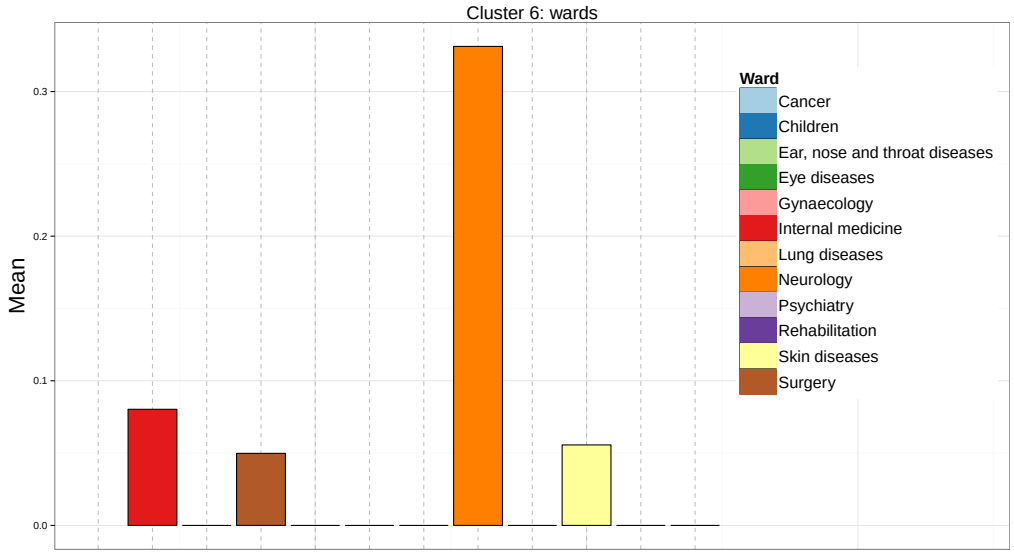

Figure 7.3: An example cluster found by the GBMM algorithm. The plot shows the hospital ward profile in the cluster, characterized by the mean vector of the estimated Gaussian distribution. For example, on average approximately 3 days are spent in the neurology ward, and 1 day at the internal medicine ward. The other means correspond to less than 1 day spent in the ward in question.

7.3.2 Analyzing the cluster sequences

We created a sequence of 92 clusters for each patient, representing the 7 year observation period in terms of the clusters. The removed rows were treated as either healthy- or dead cluster, depending on the status of each patient on the corresponding time window, and we thus have 52 clusters in total.

Some statistics based on the cluster allocations

We calculated how many days on average will the patients in each cluster spend in the hospital during the next 12 months, given the current cluster allocation. This is illustrated in Figure 7.5 along with 25% and 75% quantiles. Based on this, it seems that the psychiatric ward tends to be the clearest indicator of high usage of hospital services. Other significant wards include internal medicine, surgery and cancer. The quantiles show that the variances

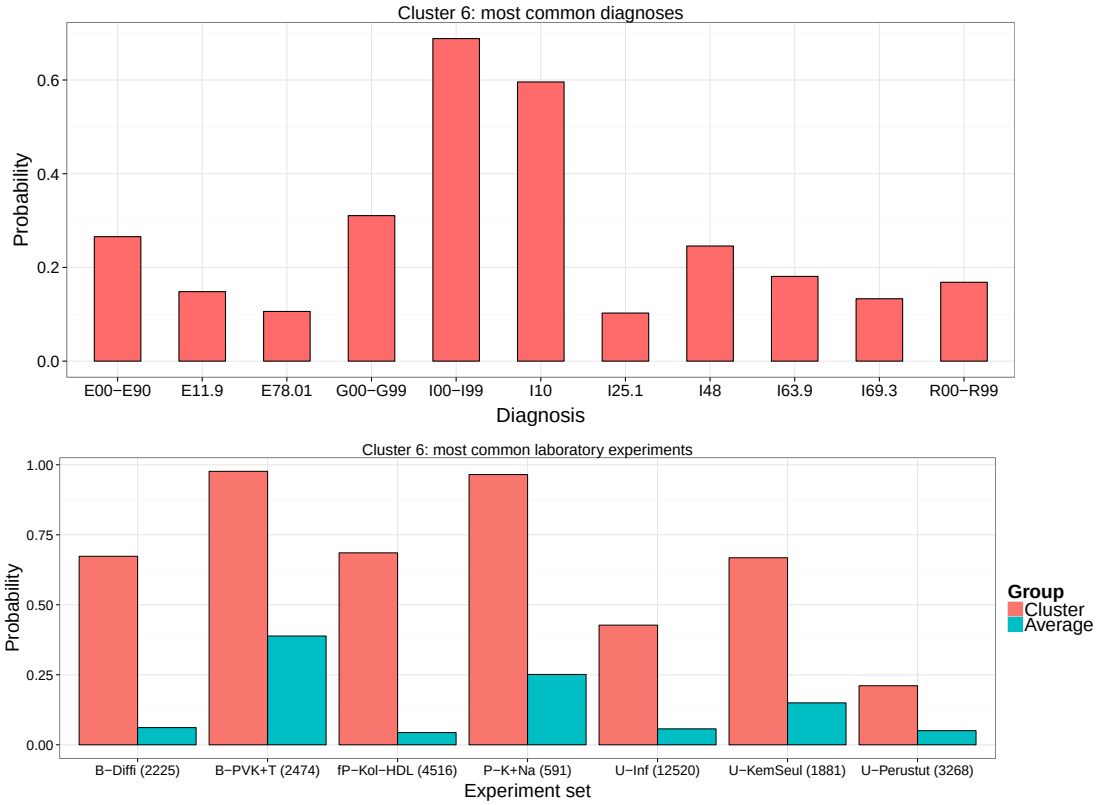

Figure 7.4: An example cluster found by the GBMM algorithm. The plot shows the most common diagnoses and laboratory experiments in the cluster. The probabilities of laboratory experiments in the example cluster are compared to the corresponding probabilities calculated for all hospital visits in the data. Note that the abbreviations of the laboratory experiments are taken from the data directly, and are thus in Finnish.

of the predicted days are rather high and the bottom quantile is 0 in most clusters.

Another point of interest is how the clusters relate to the death probability of the patients. This is an interesting metric since treatment of dying patients can be extremely expensive, and in some cases it might not even prolong the patients' life. We approach this question with a simple visualization: For each cluster we can calculate the portion of the patients who have died in the next m months ($m = 1, ..., 24$) following the time step when the cluster observation

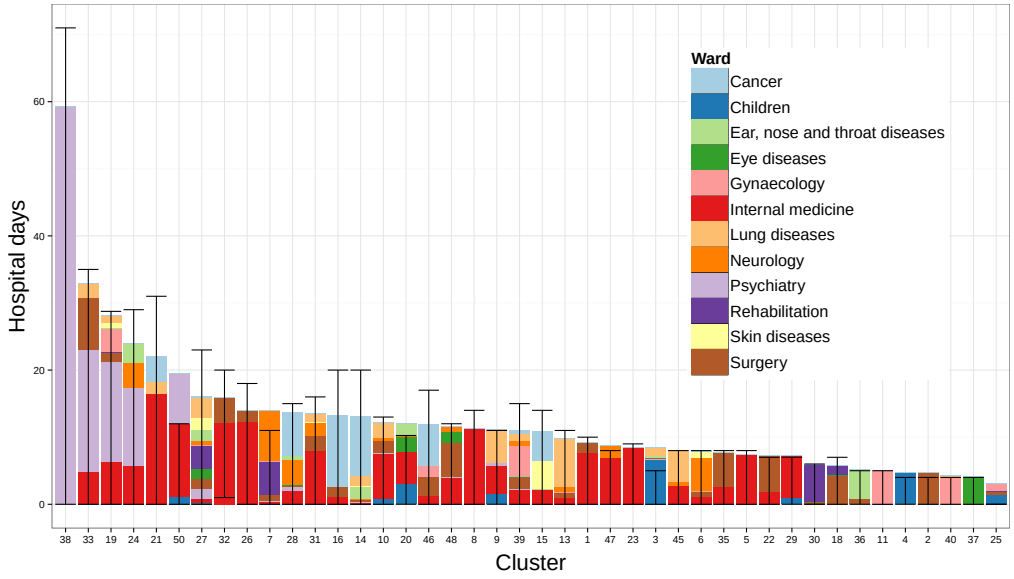

Figure 7.5: Illustration of the number of days a patient will spend in the hospital during the next 12 months given the current cluster. The coloring of the bars shows the relative time spent at the corresponding hospital department during a 28 day time window. The error bars represent 25% and 75% quantiles.

was made. In Figure 7.6 we have plotted the cumulative probability of death for the 5 clusters that have the highest death probability 12 months after observing the corresponding cluster allocation. For reference, also the average death probability across all clusters was plotted (except the 'dead' cluster of course). Out of the 5 clusters visualized, as many as 4 correspond to clusters that clearly describe the cancer ward (combined with other wards). This is intuitive, since cancer is probably one of the most common diagnosis that can lead to death rather quickly.

Finding interesting cluster transitions

We also looked if knowing more than one past cluster states will help in predicting the cluster the patient will be in the next time window. This was done by comparing conditional distributions of the type

$$p(C_{t+1}|\mathcal{C}_{t,s}),$$

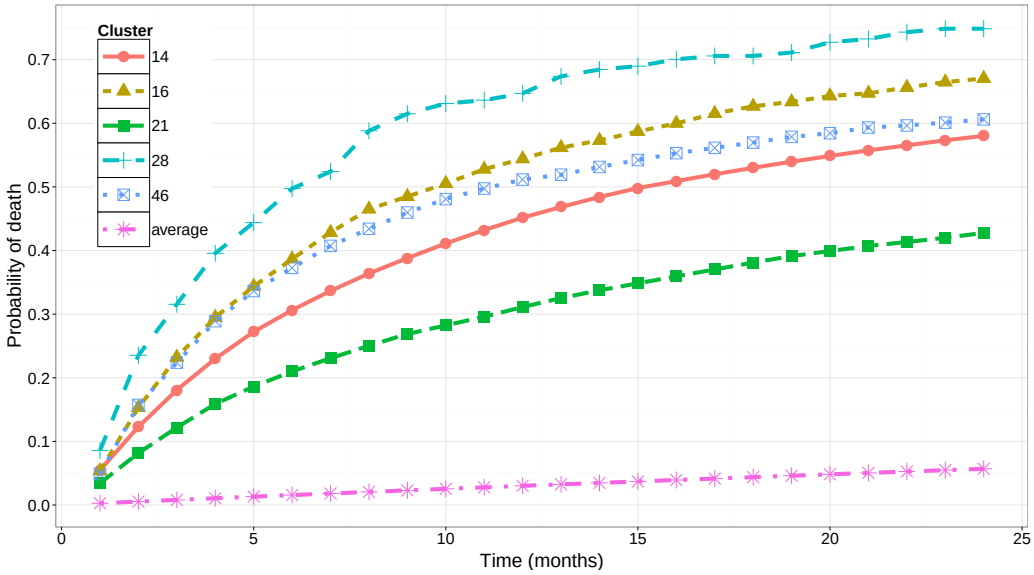

Figure 7.6: Cumulative death probability as a function of time when a patient is allocated to a given cluster at time step $t = 0$. The plot shows the top 5 clusters with the highest death probability 12 months after the observed cluster allocation. The bottom line shows the average death probability across all clusters (excluding the dead cluster). The 4 clusters corresponding to the highest death probability are all clearly identified as clusters describing the cancer ward. The cluster corresponding to the 5th highest death probability describes the best the type of patients who spend on average 10 days in the internal medicine ward. The cancer ward is present in this cluster as well, although not as clearly as in the top 4 clusters that indicate a high probability of death.

where C_{t+1} is the target cluster we want to predict and $\mathcal{C}_{t,s}$ corresponds to a cluster history of length s that was observed right before time step $t + 1$. These distributions were estimated by considering a data generating model with SymDir(52,1) prior for the observed transition counts (corresponding to pseudo-counts of 1) and a multinomial likelihood. We note that this corresponds to Bayesian modeling of the transitions as an order s Markov chain. This choice of prior and likelihood results in a Dirichlet posterior for the transition probabilities.

Using this approach, we compare the 1st order conditional distributions

against the corresponding 2nd order transition distributions by calculating the *Bhattacharyya distance* between these. The Bhattacharyya distance is a way to measure the distance between two probability distributions p and q, and it is defined by

$$D_B(p,q) = -\ln \int \sqrt{p(x)q(x)}\,\mathrm{d}x.$$

It is easy to see that $0 \leq D_B \leq \infty$ and that $D_B(p,q) = 0$ if $p = q$ a.s. Calculating the Bhattacharyya distance for two Dirichlet distributions $\mathrm{Dir}(\boldsymbol{\alpha})$ and $\mathrm{Dir}(\boldsymbol{\beta})$ is easy [63], and the result is given by

$$\begin{aligned} D_B(\mathrm{Dir}(\boldsymbol{\alpha}), \mathrm{Dir}(\boldsymbol{\beta})) = \ln\Gamma(\sum_k \frac{\alpha_k+\beta_k}{2}) + \frac{1}{2}\sum_k \Big[\ln\Gamma(\alpha_k) + \ln\Gamma(\beta_k)\Big] \\ - \sum_k \ln\Gamma(\frac{\alpha_k+\beta_k}{2}) - \frac{1}{2}\Big[\ln\Gamma(\sum_k \alpha_k) + \ln\Gamma(\sum_k \beta_k)\Big]. \end{aligned}$$

We note that using Dirichlet distributions for the transition probabilities also take into account the observed counts in the likelihood, which the naive method of simply normalizing the observed counts does not do.

Using the Bhattacharyya distance, we can identify the conditional distributions that seem to differ the most from each other. The purpose of this analysis is to aid, and somewhat automate, the process of finding interesting cluster transitions which could then be analyzed more carefully.

Going back to the example cluster, we can for instance calculate the Bhattacharyya distance between the first- and second order transition distributions that have the example cluster (labeled by 6) as the last known state: $D_B(p(C_{t+1}|C_t = 6), p(C_{t+1}|C_t = 6, C_{t-1} = c_{t-1}))$. Table 7.3.2 shows the 5 largest distances corresponding to different second order transition distributions which we could estimate from at least $N = 30$ observations. Essentially this tells us that knowing the second order history $(C_t = 6, C_{t-1} = 15)$ changes the first order transition distribution $p(C_{t+1}|C_t = 6)$ the most. We can use this for example to prioritize our analysis to the cluster histories that drastically seem to change the expected next cluster. In Figure 7.7 we have visualized the differences between the above first- and second order transition distributions. To make the plot cleaner, we have grouped the clusters that have low transition probability to one single cluster labeled 'other'. We see that there are some significant differences between the two transition distributions. For example the probability of the next state being 15 is about 15%

c_{t-1}	Distance
15	82.42
23	80.52
35	75.74
10	73.01
2	72.56

Table 7.1: Bhattacharyya distance between the first- and second order transition distributions $p(C_{t+1}|C_t = 6)$ and $p(C_{t+1}|C_t = 6, C_{t-1} = c_{t-1})$ for clusters c_{t-1} that correspond to the 5 largest distances. The distances were only calculated for histories for which more than $N = 30$ second order transitions were observed.

higher if we know the patient was in cluster 15 also at time step $t-1$ before observing cluster 6. We also note that the probability of getting healthy drops over 20% if we know the second order history as compared to the first oder history. This is a common and intuitive observation across the calculations we made; The second order histories usually describe a longer sick period than the first order history, in which case also the next cluster tends to be a sick cluster with higher probability.

7.3.3 Discussion

We presented a few ways to visualize the found clusters, and also did some simple analysis on how the clusters can be used to predict future behavior of the patients. Additionally, we presented an automated way of filtering interesting cluster transitions for further analysis based on the Bhattacharyya distance. Overall, the clustering reveals the hospital wards that usually are associated with long stay in the hospital along with the related diagnoses and laboratory experiments. We are also able to calculate simple statistics such as the death probability of the patients in the next 24 months based on the current cluster allocations. As of now, our model yields results that allows us to summarize the original data in various ways and draw ideas for further, more detailed analysis. To effectively interpret the results from a medical point of view, a medical professional would be needed to check the findings.

As one future idea we mention straight regression from the covariates to

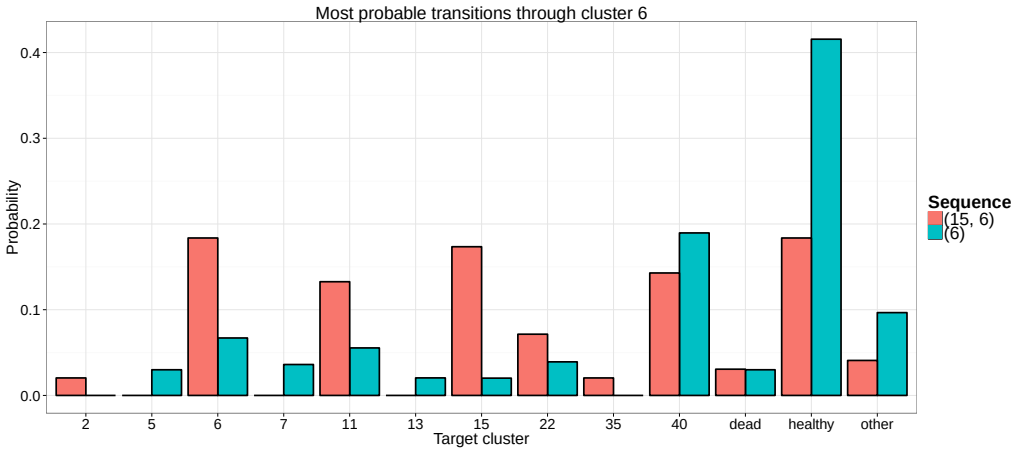

Figure 7.7: Comparison between the 1st and 2nd order transition distributions $p(C_{t+1}|C_t = 6)$ and $p(C_{t+1}|C_t = 6, C_{t-1} = 15)$. All target clusters with probability more than 0.02 are visualized as their own clusters, and other targets are merged to the cluster 'other'.

the upcoming hospital days, which might provide more detailed information about the features of interest. Also different classification algorithms could be used to predict what type of a patient the data represents, given past measurements and other information. Regarding these possibilities, we note that it would also be possible to use the found cluster allocations for the different time windows as features in further analysis.

Chapter 8

Conclusion

In this thesis we presented a Bayesian mixture model for analyzing heterogeneous data. We specified the model so that it can model both Gaussian- and Bernoulli data, which should suit modeling a lot of different datasets. A multinomial probit data augmentation was proposed as an extension that allows modeling also multinomial data, but was found to be inefficient for practical use on large data. We also briefly discussed a recent result [50] that provides a better alternative to extend our model to multinomial data, but that we were not aware of by the time when we derived the multinomial probit mixture model.

The main contribution of the thesis is the detailed coverage on the specification of the proposed mixture model as well as the full derivations for efficient and scalable incremental variational inference for parameter estimation. The incremental algorithm was found to be superior to the standard variational inference procedure in a toy data experiment. We showed that the convergence to the optimum is usually notably faster in case of the incremental algorithm, while the only downside is the requirement to store the subset sufficient statistics, which is not a problem in practice. Based on these results we conclude that, while being so easy to implement, the incremental algorithm should be used whenever possible. Additionally, we showed how to easily parallelize parts of the algorithm on multicore setups to boost the performance even further. Moreover, we also presented a scalable initialization to our mixture model using a random projections k-means(++) clustering algorithm. Even though the models in this thesis are not the easiest ones, we think this thesis should act as a good reference point for anyone wanting to learn about variational inference through examples. At the same time

we also presented some state-of-the-art practical ways to enhance the standard variational inference, such as the incremental variational inference and parallel computing.

As a practical application, we used the proposed model for exploratory data analysis of real electronic patient record data. The model found clusters that correspond to typical one month usage profiles of hospital services among the patients in the data. On top of the clustering results we calculated simple statistics to be used for further analysis, such as predicting the need of treatment in the future. Our findings mostly resemble the intuition; For example the patients having psychiatric issues tend to require treatment for extended periods of time. As of now, the main advantage from using the model is that the complex data can be compressed to a set of cluster allocations and known probability distributions. These in turn can be used to extract insights, such as interesting cluster transitions, that could lead to discovering ideas for further and more detailed analysis of the data.

Bibliography

[1] Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Geometric approximation via coresets. In *Combinatorial and Computational Geometry, MSRI*, pages 1–30. University Press, 2005.

[2] A. Agresti. *Categorical Data Analysis*. Wiley Series in Probability and Statistics. Wiley, 2003.

[3] Cedric Archambeau and Beyza Ermis. Incremental Variational Inference for Latent Dirichlet Allocation. `http://arxiv.org/pdf/1507.05016v2.pdf`, 2015. Accessed: 15-05-2016.

[4] David Arthur and Sergei Vassilvitskii. K-means++: The Advantages of Careful Seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[5] Leonard E. Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *Ann. Math. Statist.*, 37(6):1554–1563, 12 1966.

[6] Matthew Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, University of London, 2003.

[7] Casey Bennett and Thomas Doub. Data mining and electronic health records: Selecting optimal clinical treatments in practice. *CoRR*, abs/1112.1668, 2011.

[8] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[9] David M. Blei and Michael I. Jordan. Variational inference for dirichlet process mixtures. *Bayesian Anal.*, 1(1):121–143, 03 2006.

[10] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. `https://arxiv.org/pdf/1601.00670v2.pdf`, 2016. Accessed: 06-06-2016.

[11] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[12] C. I. Bliss. The Method of Probits. *Science*, 79:38–39, 1934.

[13] C. Boutsidis, A. Zouzias, M. W. Mahoney, and P. Drineas. Randomized Dimensionality Reduction for k-Means Clustering. *IEEE Transactions on Information Theory*, 61(2):1045–1062, Feb 2015.

[14] Wray L. Buntine. Operations for learning with graphical models. *CoRR*, abs/cs/9412102, 1994.

[15] L. Le Cam. Maximum likelihood: An introduction. *International Statistical Review / Revue Internationale de Statistique*, 58(2):153–171, 1990.

[16] Jianfei Chen, Jun Zhu, Zi Wang, Xun Zheng, and Bo Zhang. Scalable inference for logistic-normal topic models. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2445–2453. Curran Associates, Inc., 2013.

[17] D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, 20(2):215–242, 1958.

[18] C. Daganzo. *Multinomial Probit: The Theory and Its Application to Demand Forecasting*. Economic Theory, Econometrics, and Mathematical Economics Series. Academic Press, 1979.

[19] N. E. Day. Estimating the components of a mixture of normal distributions. *Biometrika*, 56(3):463–474, 1969.

[20] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[21] Jay K. Dow and James W. Endersby. Multinomial probit and multinomial logit: a comparison of choice models for voting research. *Electoral Studies*, 23(1):107 – 122, 2004.

[22] Justin M. Dubas, Byung-Joo Lee, and Nelson C. Mark. A multinomial logit approach to exchange rate policy classification with an application to growth. *Journal of International Money and Finance*, 29(7):1438–1462, November 2010.

[23] H. Elghazel, V. Deslandres, K. Kallel, and A. Dussauchoy. Clinical pathway analysis using graph-based approach and markov models. In *Digital Information Management, 2007. ICDIM '07. 2nd International Conference on*, volume 1, pages 279–284, Oct 2007.

[24] Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2142–2150. Curran Associates, Inc., 2011.

[25] Zhe Gan, Ricardo Henao, David E. Carlson, and Lawrence Carin. Learning deep sigmoid belief networks with data augmentation. In *AISTATS*, 2015.

[26] A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, 2003.

[27] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, Nov 1984.

[28] M. Girolami and S. Rogers. Variational Bayesian multinomial probit regression with Gaussian process priors. *Neural Computation*, 18(8):1790–1817, 2006.

[29] Nicola Greggio, Alexandre Bernardino, Cecilia Laschi, Paolo Dario, and José Santos-Victor. Fast estimation of gaussian mixture models for image segmentation. *Machine Vision and Applications*, 23(4):773–789, 2012.

[30] Eugene D Hahn and Refik Soyer. Probit and logit models: Differences in the multivariate realm. *The Journal of the Royal Statistical Society, Series B*, pages 1–12, 2005.

[31] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[32] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic Variational Inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

[33] P. Hore and L. O. Hall. Scalable clustering: a distributed approach. In *Fuzzy Systems, 2004. Proceedings. 2004 IEEE International Conference on*, volume 1, pages 143–148 vol.1, July 2004.

[34] Zhengxing Huang, Wei Dong, Lei Ji, Chenxi Gan, Xudong Lu, and Huilong Duan. Discovery of clinical pathway patterns from event logs using probabilistic topic models. *J. of Biomedical Informatics*, 47:39–57, February 2014.

[35] Zhengxing Huang, Xudong Lu, and Huilong Duan. Latent treatment pattern discovery for clinical processes. *Journal of Medical Systems*, 37(2):1–10, 2013.

[36] Zhexue Huang. Clustering large data sets with mixed numeric and categorical values. In *In The First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 21–34, 1997.

[37] Michael Hughes and Erik Sudderth. Memoized online variational inference for dirichlet process mixture models. In *Advances in Neural Information Processing Systems 26*, pages 1133–1141. 2013.

[38] Viet Huynh, Dinh Phung, Svetha Venkatesh, Long Nguyen, Matthew Hoffman, and Hung Bui. Scalable nonparametric bayesian multilevel clustering. In *Proceedings of the 32th Conference in Uncertainty in Artificial Intelligence*, UAI 2016, 2016.

[39] Simon Jackman. Estimation and Inference via Bayesian Simulation: An Introduction to Markov Chain Monte Carlo. *American Journal of Political Science*, 44(2):375–404, 2000.

[40] Jean Jacod and Philip Protter. *Probability Essentials*. Springer, 2004.

[41] Siddhartha Chib James H. Albert. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

[42] Lester Johnson and David Hensher. Application of multinomial probit to a two-period panel data set. *Transportation Research Part A: General*, 16(5):457 – 464, 1982.

[43] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37(2):183–233, 1999.

[44] A. Juan and E. Vidal. Bernoulli mixture models for binary images. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 367–370 Vol.3, Aug 2004.

[45] Arto Klami, Abhishek Tripathi, Johannes Sirola, Lauri Väre, and Frederic Roulland. Latent feature regression for multivariate count data. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 462–470. JMLR, 2015.

[46] Achim Klenke. *Probability Theory : A Comprehensive Course, Second Edition*. Universitext. Springer, 2014.

[47] M. Kushima, K. Araki, M. Suzuki, T. Yamazaki, and N. Sonehara. Research on text data mining of hospital patient records within electronic medical records. In *Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on*, pages 1500–1505, Dec 2014.

[48] Lishuai Li, M Gariel, RJ Hansman, and R Palacios. Anomaly detection in onboard-recorded flight data using cluster analysis. In *Digital Avionics Systems Conference (DASC), 2011 IEEE/AIAA 30th*, pages 4A4–1. IEEE, 2011.

[49] Dahua Lin. Online learning of nonparametric mixture models via sequential variational approximation. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances*

in Neural Information Processing Systems 26, pages 395–403. Curran Associates, Inc., 2013.

[50] Scott Linderman, Matthew Johnson, and Ryan P Adams. Dependent multinomial models made easy: Stick-breaking with the polya-gamma augmentation. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3438–3446. Curran Associates, Inc., 2015.

[51] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[52] Nataliya V. Malyshkina and Fred L. Mannering. Markov switching multinomial logit model: An application to accident-injury severities. *Accident Analysis & Prevention*, 41(4):829 – 838, 2009.

[53] P. McCullagh and J.A. Nelder. *Generalized Linear Models, Second Edition*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1989.

[54] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[55] Thomas P. Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, UAI '01, pages 362–369, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[56] A. Najjar, C. Gagné, and D. Reinharz. A novel mixed values k-prototypes algorithm with application to health care databases mining, Dec 2014.

[57] Radford M. Neal and Geoffrey E. Hinton. *Learning in Graphical Models*, chapter A View of the EM Algorithm that Justifies Incremental, Sparse, and other Variants, pages 355–368. Springer Netherlands, Dordrecht, 1998.

[58] J.R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998.

[59] Aaron Oord and Benjamin Schrauwen. Factoring variations in natural images with deep gaussian mixture models. In Z. Ghahramani, M. Welling, C. Cortes, N.d. Lawrence, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3518–3526. Curran Associates, Inc., 2014.

[60] Karl Pearson. Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 185:71–110, 1894.

[61] Nicholas G. Polson, James G. Scott, and Jesse Windle. Bayesian Inference for Logistic Models Using Pólya–Gamma Latent Variables. *Journal of the American Statistical Association*, 108(504):1339–1349, 2013.

[62] D. Povey, L. Burget, M. Agarwal, P. Akyazi, K. Feng, A. Ghoshal, O. Glembek, N. K. Goel, M. Karafiát, A. Rastrow, R. C. Rose, P. Schwarz, and S. Thomas. Subspace gaussian mixture models for speech recognition. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4330–4333, March 2010.

[63] T. W. Rauber, A. Conci, T. Braun, and K. Berns. Bhattacharyya probabilistic distance of the Dirichlet density and its application to Split-and-Merge image segmentation. In *2008 15th International Conference on Systems, Signals and Image Processing*, pages 145–148, June 2008.

[64] Mehreen Saeed, Kashif Javed, and Haroon Atique Babri. Machine learning using bernoulli mixture models: Clustering, rule extraction and dimensionality reduction. *Neurocomput.*, 119:366–374, November 2013.

[65] Sanjeena Subedi and Paul D. McNicholas. Variational bayes approximations for clustering via mixtures of normal inverse gaussian distributions. *Advances in Data Analysis and Classification*, 8(2):167–193, 2014.

[66] Bo Thiesson. Fast large-scale mixture modeling with component-specific data partitions. In *NIPS-2010: Advances in Neural Information Processing Systems 23*. MIT Press, December 2010.

[67] Luke Tierney and Joseph B. Kadane. Accurate Approximations for Posterior Moments and Marginal Densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986.

[68] Jakob Verbeek, Nikos Vlassis, and Jan Nunnink. A variational EM algorithm for large-scale mixture modeling. In S. Vassiliades, L.M.J. Florack, J.W.J. Heijnsdijk, and A. van der Steen, editors, *9th Annual Conference of the Advanced School for Computing and Imaging (ASCI '03)*, pages 136–143, Heijen, Netherlands, June 2003.

[69] Strother H. Walker and David B. Duncan. Estimation of the probability of an event as a function of several independent variables. *Biometrika*, 54(1-2):167–179, 1967.

[70] Chong Wang, John William Paisley, and David M. Blei. Online variational inference for the hierarchical dirichlet process. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 752–760, 2011.

[71] Mingyuan Zhou, Lingbo Li, Lawrence Carin, and David B. Dunson. Lognormal and gamma mixed negative binomial regression. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1343–1350, New York, NY, USA, 2012. ACM.

[72] Jun Zhu, Xun Zheng, and Bo Zhang. Improved bayesian logistic supervised topic models with data augmentation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, (ACL) 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 187–195, 2013.

Appendix A

Variational Updates for the GBMM-model

Variational distributions

We denote $\langle \cdot \rangle$ for expectation taken w.r.t. all variational distributions except the one we are calculating, and C for a generic constant. The variational distributions for the Gaussian and Bernoulli mixture model introduced in Chapter 5 are found by applying the formula (3.7) presented in Chapter 3.

Calculation of $q(\mathbf{Z})$

$$\ln q(\mathbf{Z}) = \langle \ln p(\mathbf{X}^{(G)}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})\rangle + \langle \ln p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\omega}, \boldsymbol{\psi})\rangle + \langle \ln p(\mathbf{Z}|\boldsymbol{\pi})\rangle + C.$$

For the Gaussian part we have

$$\begin{aligned}\langle \ln p(\mathbf{X}^{(G)}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})\rangle = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\Big[-\frac{1}{2}(D_G \ln(2\pi) - \langle \ln|\boldsymbol{\Lambda}_k|\rangle \\ + \langle (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k)^T \boldsymbol{\Lambda}_k (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k)\rangle)\Big]. \qquad \text{(A.1)}\end{aligned}$$

Now for the Bernoulli part ($\kappa_{nd} = x_{nd}^{(B)} - 1/2$ and $b = 1$ in PG-identity):

$$\begin{aligned}\langle \ln p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\omega}, \boldsymbol{\psi})\rangle &= \langle \sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{d=1}^{D_B} z_{nk} \ln 2^{-1} e^{\kappa_{nd}\psi_{kd} - \omega_{nd}\psi_{kd}^2/2}\rangle \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K}\Big[z_{nk}\sum_{d=1}^{D_B}\Big[-\ln 2 + \kappa_{nd}\langle\psi_{kd}\rangle - \frac{1}{2}\langle\omega_{nd}\rangle\langle\psi_{kd}^2\rangle\Big]\Big].\end{aligned} \tag{A.2}$$

For the cluster assignment part we have

$$\langle \ln p(\mathbf{Z}|\boldsymbol{\pi})\rangle = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\langle \ln \pi_k\rangle. \tag{A.3}$$

Now (A.1) - (A.3) put together gives

$$\begin{aligned}\ln q(\mathbf{Z}) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\Big[&-\frac{1}{2}(D_G \ln(2\pi) - \langle \ln|\boldsymbol{\Lambda}_k|\rangle + \langle (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k)^T \boldsymbol{\Lambda}_k (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k)\rangle) \\ &+ \sum_{d=1}^{D_B}\Big[-\ln 2 + \kappa_{nd}\langle\psi_{kd}\rangle - \frac{1}{2}\langle\omega_{nd}\rangle\langle\psi_{kd}^2\rangle\Big] + \langle \ln \pi_k\rangle\Big].\end{aligned}$$

We now define

$$\begin{aligned}\ln s_{nk} := &-\frac{1}{2}(D_G \ln(2\pi) - \langle \ln|\boldsymbol{\Lambda}_k|\rangle + \langle (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k)^T \boldsymbol{\Lambda}_k (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k)\rangle) \\ &+ \sum_{d=1}^{D_B}\Big[-\ln 2 + \kappa_{nd}\langle\psi_{kd}\rangle - \frac{1}{2}\langle\omega_{nd}\rangle\langle\psi_{kd}^2\rangle\Big] + \langle \ln \pi_k\rangle.\end{aligned}$$

When all of the variational distributions are known, the required expectations

can be calculated as

$$\begin{aligned}
&\langle \ln |\mathbf{\Lambda}_k| \rangle = \sum_{d=1}^{D_G} \langle \ln \lambda_{kd} \rangle = \sum_{d=1}^{D_G} (\psi(\hat{a}_k) - \ln \hat{b}_{kd}), \\
&\langle \ln \pi_k \rangle = \psi(\hat{\alpha}_k) - \psi(\sum_{k=1}^{K} \hat{\alpha}_k), \\
&\langle (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k)^T \mathbf{\Lambda}_k (\mathbf{x}_n^{(G)} - \boldsymbol{\mu}_k) \rangle = \sum_{d=1}^{D_G} (\frac{\hat{a}_k}{\hat{b}_{kd}} (x_{nd}^{(G)} - \hat{m}_{kd})^2 + \frac{1}{\hat{\beta}_k}), \\
&\langle \psi_{kd} \rangle = \hat{\nu}_{kd}, \\
&\langle \psi_{kd}^2 \rangle = \hat{\tau}_{kd}^{-1} + \hat{\nu}_{kd}^2, \\
&\langle \omega_{nd} \rangle = \frac{1}{2\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2).
\end{aligned}$$

Thus

$$\begin{aligned}
s_{nk} \propto \exp \Big\{ \frac{1}{2} \Big(&\sum_{d=1}^{D_G} (\psi(\hat{a}_k) - \ln \hat{b}_{kd}) - \sum_{d=1}^{D_G} (\frac{\hat{a}_k}{\hat{b}_{kd}} (x_{nd}^{(G)} - \hat{m}_{kd})^2 + \frac{1}{\hat{\beta}_k}) \Big) \\
&+ \sum_{d=1}^{D_B} \Big[\kappa_{nd} \hat{\nu}_{kd} - \frac{1}{4\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2) (\hat{\tau}_{kd}^{-1} + \hat{\nu}_{kd}^2) \Big] + \psi(\hat{\alpha}_k) - \psi(\sum_{k=1}^{K} \hat{\alpha}_k) \Big\}.
\end{aligned} \tag{A.4}$$

After exponentiating and normalizing we see that the variational distribution is a product of multinomials

$$q(\mathbf{Z}) = \prod_n \text{Mult}(1, \hat{\mathbf{r}}_n)$$

where

$$\hat{r}_{nk} = \frac{s_{nk}}{\sum_i s_{ni}}.$$

Calculation of $q(\boldsymbol{\pi})$

$$\begin{aligned}
\ln q(\boldsymbol{\pi}) &= \langle \ln p(\mathbf{Z}|\boldsymbol{\pi}) \rangle + \ln p(\boldsymbol{\pi}) + C \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} \langle z_{nk} \rangle \ln \pi_k + \sum_{n=1}^{N} (\alpha_0 - 1) \ln \pi_k + C \\
&= \sum_{k=1}^{K} (\hat{R}_k + \alpha_0 - 1) \ln \pi_k + C,
\end{aligned}$$

where $\hat{R}_k = \sum_{n=1}^{N} \hat{r}_{nk}$. Exponentiating this gives

$$q(\boldsymbol{\pi}) \propto \prod_k \pi_k^{\alpha_0 + \hat{R}_k - 1},$$

which corresponds to a $\text{Dir}(\hat{\boldsymbol{\alpha}})$-distribution, where $\hat{\alpha}_k = \alpha_0 + \hat{R}_k$.

Calculation of $q(\boldsymbol{\mu}, \boldsymbol{\Lambda})$

First we note that by the independence assumptions made in the model specification in (4.3) the variational distribution clearly factorizes across the clusters and data dimensions as

$$q(\boldsymbol{\mu}, \boldsymbol{\Lambda}) = \prod_{k=1}^{K} \prod_{d=1}^{D_G} q(\mu_{kd}, \lambda_{kd}).$$

We thus continue by calculating the variational distribution for each of these factors.

$$\begin{aligned}
\ln q(\mu_{kd}, \lambda_{kd}) &= \langle \ln p(\mathbf{X}^{(G)}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \rangle + \ln p(\boldsymbol{\mu}, \boldsymbol{\Lambda}) + C \\
&= \sum_{n=1}^{N} \langle z_{nk} \rangle \Big(\frac{1}{2} \ln \lambda_{kd} - \frac{\lambda_{kd}}{2} (x_{nd}^{(G)} - \mu_{kd})^2 \Big) \\
&\quad + \frac{1}{2} \ln \lambda_{kd} - \frac{\beta_0 \lambda_{kd}}{2} (\mu_{kd} - \mu_{0d})^2 + (a_0 - 1) \ln \lambda_{kd} - b_0 \lambda_{kd} + C \\
&= (\frac{\hat{R}_k}{2} + a_0 - 1) \ln \lambda_{kd} - b_0 \lambda_{kd} + \frac{1}{2} \ln \lambda_{kd} \\
&\quad - \frac{\lambda_{kd}}{2} \Big[\sum_{n=1}^{N} \hat{r}_{nk} (x_{nd}^{(G)} - \mu_{kd})^2 + \beta_0 (\mu_{kd} - \mu_{0d})^2 \Big] + C
\end{aligned}$$

After completing the square for μ_{kd}, rearranging the terms and exponentiating we have

$$
\begin{aligned}
q(\mu_{kd}, \lambda_{kd}|\mathbf{X}) \propto \lambda_{kd}^{1/2} \exp\Big\{ & -\frac{\lambda_{kd}(\hat{R}_k + \beta_0)}{2}\Big(\mu_{kd} - \frac{\beta_0\mu_{0d} + \sum_{n=1}^N \hat{r}_{nk} x_{nd}^{(G)}}{\hat{R}_k + \beta_0}\Big)^2\Big\} \\
& \times \lambda_{kd}^{\hat{R}_k/2 + a_0 - 1} \exp\Big\{ -\Big[b_0 + \frac{\hat{R}_k\beta_0}{2(\hat{R}_k + \beta_0)}\Big(\mu_{0d} - \frac{\sum_{n=1}^N \hat{r}_{nk} x_{nd}^{(G)}}{\hat{R}_k}\Big)^2 \\
& + \frac{1}{2}\sum_{n=1}^N \hat{r}_{nk}(x_{nd}^{(G)})^2 - \frac{(\sum_{n=1}^N \hat{r}_{nk} x_{nd}^{(G)})^2}{2\hat{R}_k}\Big]\lambda_{kd}\Big\}.
\end{aligned}
$$

From this we recognize the kernel of a Normal-Gamma

$$
\mathrm{N}(\mu_{kd}|\hat{m}_{kd}, (\hat{\beta}_k\lambda_{kd})^{-1})\,\mathrm{Gamma}(\lambda_{kd}|\hat{a}_k, \hat{b}_{kd}),
$$

where

$$
\begin{aligned}
\hat{m}_{kd} &= \frac{\beta_0\mu_{0d} + \sum_{n=1}^N \hat{r}_{nk} x_{nd}^{(G)}}{\hat{R}_k + \beta_0}, \\
\hat{\beta}_k &= \hat{R}_k + \beta_0, \\
\hat{a}_k &= \frac{\hat{R}_k}{2} + a_0, \\
\hat{b}_{kd} &= b_0 + \frac{\hat{R}_k\beta_0}{2(\hat{R}_k + \beta_0)}\Big(\mu_{0d} - \frac{\sum_{n=1}^N \hat{r}_{nk} x_{nd}^{(G)}}{\hat{R}_k}\Big)^2 + \frac{1}{2}\sum_{n=1}^N \hat{r}_{nk}(x_{nd}^{(G)})^2 - \frac{(\sum_{n=1}^N \hat{r}_{nk} x_{nd}^{(G)})^2}{2\hat{R}_k}.
\end{aligned}
$$

Calculation of $q(\boldsymbol{\psi})$

Also here the variational distribution clearly factorizes as

$$
q(\boldsymbol{\psi}) = \prod_{k=1}^K \prod_{d=1}^{D_B} q(\psi_{kd}).
$$

The factors are calculated by the standard procedure:

$$
\begin{aligned}
\ln q(\psi_{kd}) &= \langle \ln p(\mathbf{X}^{(B)}|\mathbf{Z},\boldsymbol{\omega},\boldsymbol{\psi})\rangle + \langle \ln p(\boldsymbol{\psi}|\boldsymbol{\nu},\boldsymbol{\tau})\rangle + C \\
&= \langle \sum_{n=1}^{N} z_{nk} \ln(2^{-1} e^{\kappa_{nd}\psi_{kd} - \omega_{nd}\psi_{kd}^2/2})\rangle + \langle -\frac{1}{2}(\psi_{kd} - \nu_{kd})^2 \rangle + C \\
&= \psi_{kd} \sum_{n=1}^{N} \hat{r}_{nk}\kappa_{nd} - \frac{\psi_{kd}^2}{2} \sum_{n=1}^{N} \hat{r}_{nk}\langle \omega_{nd}\rangle - \frac{\psi_{kd}^2}{2} + \psi_{kd}\langle \nu_{kd}\rangle + C \\
&= -\frac{\sum_{n=1}^{N} \hat{r}_{nk}\langle \omega_{nd}\rangle + 1}{2} \Big(\psi_{kd} - \frac{\sum_{n=1}^{N} \hat{r}_{nk}\kappa_{nd} + \langle \nu_{kd}\rangle}{\sum_{n=1}^{N} \hat{r}_{nk}\langle \omega_{nd}\rangle + 1}\Big)^2 + C.
\end{aligned}
$$

This is a logarithm of a Gaussian kernel. Thus the variational distribution is $\mathrm{N}(\hat{\nu}_{kd}, \hat{\tau}_{kd}^{-1})$ where

$$
\begin{aligned}
\hat{\nu}_{kd} &= \frac{\sum_{n=1}^{N} \hat{r}_{nk}\kappa_{nd} + \langle \nu_{kd}\rangle}{\sum_{n=1}^{N} \hat{r}_{nk}\langle \omega_{nd}\rangle + 1} = \frac{\sum_{n=1}^{N} \hat{r}_{nk}\kappa_{nd} + \hat{n}_{kd}}{\sum_{n=1}^{N} \Big[\hat{r}_{nk} \frac{1}{2\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2)\Big] + 1}, \\
\hat{\tau}_{kd} &= \sum_{n=1}^{N} \hat{r}_{nk}\langle \omega_{nd}\rangle + 1 = \sum_{n=1}^{N} \Big[\hat{r}_{nk} \frac{1}{2\hat{e}_{nd}} \tanh(\hat{e}_{nd}/2)\Big] + 1.
\end{aligned}
$$

Calculation of $q(\boldsymbol{\omega})$

This variational distribution factorizes as

$$
q(\boldsymbol{\omega}) = \prod_{n=1}^{N} \prod_{d=1}^{D_B} q(\omega_{nd}).
$$

Again, we calculate the factors separately.

$$
\begin{aligned}
\ln q(\omega_{nd}) &= \langle \ln p(\mathbf{X}^{(B)}|\mathbf{Z},\boldsymbol{\omega},\boldsymbol{\psi})\rangle + \langle \ln p(\boldsymbol{\omega})\rangle + C \\
&= \langle \sum_{k} z_{nk} \ln(2^{-1} e^{\kappa_{nd}\psi_{kd} - \omega_{nd}\psi_{kd}^2/2})\rangle + \ln p(\omega_{nd}) + C \\
&= -\frac{\sum_{k} \hat{r}_{nk}\langle \psi_{kd}^2\rangle}{2}\omega_{nd} + \ln p(\omega_{nd}) + C.
\end{aligned}
$$

This is recognized as exponential tilting of a $\mathrm{PG}(1,0)$ density, yielding

$$q(\omega_{nd}) = \mathrm{PG}(1, \hat{e}_{nd}),$$

where

$$\hat{e}_{nd} = \sqrt{\sum_k \hat{r}_{nk} \langle \psi_{kd}^2 \rangle} = \sqrt{\sum_k \hat{r}_{nk} (\hat{\tau}_{kd}^{-1} + \hat{\nu}_{kd}^2)}.$$

The expected value is given as

$$\mathbb{E}_q[\omega_{kd}] = \frac{1}{2\hat{e}_{kd}} \tanh(\hat{e}_{kd}/2).$$

Calculation of $q(\boldsymbol{\nu})$

This variational distribution factorizes as

$$\prod_{k=1}^{K} \prod_{d=1}^{D_B} q(\nu_{kd})$$

and the factors are acquired by calculating

$$\begin{aligned}
\ln q(\nu_{kd}) &= \langle \ln p(\psi_{kd} | \nu_{kd}) \rangle + \ln p(\nu_{kd}) + C \\
&= -\frac{1}{2} \Big(\langle (\psi_{kd} - \nu_{kd})^2 \rangle + \gamma_0 (\nu_{kd} - \nu_0)^2 \Big) + C \\
&= -\frac{1 + \gamma_0}{2} (\nu_{kd} - \frac{\langle \psi_{kd} \rangle + \gamma_0 \nu_0}{1 + \gamma_0})^2 + C.
\end{aligned}$$

From this we recognize the log kernel of a normal $\mathrm{N}(\nu_{kd} | \hat{n}_{kd}, \hat{\gamma}^{-1})$ - distribution, where

$$\begin{aligned}
\hat{n}_{kd} &= \frac{\langle \psi_{kd} \rangle + \gamma_0 \nu_0}{1 + \gamma_0} = \frac{\hat{\nu}_{kd} + \gamma_0 \nu_0}{1 + \gamma_0}, \\
\hat{\gamma} &= 1 + \gamma_0.
\end{aligned}$$

Lower bound

The individual terms in the evidence lower bound are given by

$$\mathbb{E}_q[\ln p(\mathbf{Z}|\boldsymbol{\pi})] - \mathbb{E}_q[\ln q(\mathbf{Z})] = \sum_{k=1}^{K}[\hat{R}_k(\psi(\hat{\alpha}_k) - \psi(\sum_{k=1}^{K}\hat{\alpha}_k)) - \sum_{n=1}^{N}\hat{r}_{nk}\ln\hat{r}_{nk}],$$

$$\begin{aligned}\mathbb{E}_q[\ln p(\boldsymbol{\pi})] - \mathbb{E}_q[\ln q(\boldsymbol{\pi})] = \ln\Gamma(\alpha_0 K) - K\ln\Gamma(\alpha_0) - \ln\Gamma(\sum_{k=1}^{K}\hat{\alpha}_k) \\ + \sum_{k=1}^{K}\ln\Gamma(\hat{\alpha}_k) + \sum_{k=1}^{K}(\alpha_0 - \hat{\alpha}_k)(\psi(\hat{\alpha}_k) - \psi(\sum_{k=1}^{K}\hat{\alpha}_k)),\end{aligned}$$

$$\begin{aligned}\mathbb{E}_q[\ln p(\boldsymbol{\mu},\boldsymbol{\Lambda})] - \mathbb{E}_q[\ln q(\boldsymbol{\mu},\boldsymbol{\Lambda})] = \sum_{k=1}^{K}\sum_{d=1}^{D_G}\Big[\frac{1}{2}\ln\frac{\beta_0}{\hat{\beta}_k} - \frac{\beta_0}{2}\Big[\frac{1}{\hat{\beta}_k} + \frac{\hat{a}_k}{\hat{b}_{kd}}(\hat{m}_{kd} - \mu_{0d})^2\Big] \\ + \frac{1}{2} + a_0\ln b_0 - \hat{a}_k\ln\hat{b}_{kd} + \ln\frac{\Gamma(\hat{a}_k)}{\Gamma(a_0)} \\ + (a_0 - \hat{a}_k)(\psi(\hat{a}_k) - \ln\hat{b}_{kd}) - \hat{a}_k(\frac{b_0}{\hat{b}_{kd}} - 1)\Big],\end{aligned}$$

$$\mathbb{E}_q[\ln p(\boldsymbol{\psi}|\boldsymbol{\nu})] - \mathbb{E}_q[\ln q(\boldsymbol{\psi})] = -\frac{1}{2}\sum_{k=1}^{K}\sum_{d=1}^{D_B}(\hat{\nu}_{kd} - \hat{n}_{kd})^2 + \frac{1}{\hat{\gamma}} + \ln\hat{\tau}_{kd},$$

and

$$\mathbb{E}_q[\ln p(\boldsymbol{\nu})] - \mathbb{E}_q[\ln q(\boldsymbol{\nu})] = \frac{1}{2}\sum_{k=1}^{K}\sum_{d=1}^{D_B}\Big[\ln\frac{\gamma_0}{\hat{\gamma}} - \gamma_0(\hat{n}_{kd} - v_0)^2 - \frac{\gamma_0}{\hat{\gamma}} + 1\Big].$$

The Pólya-Gamma-part requires a little bit of manipulation. Let $\mathbb{E}_\omega$

denote expectation with respect to a PG(1,0)-distribution. Then

$$
\begin{aligned}
&\mathbb{E}_q[\ln p(\boldsymbol{\omega})] - \mathbb{E}_q[\ln q(\boldsymbol{\omega})] \\
&= \mathbb{E}_q\Big[\sum_{n=1}^{D}\sum_{d=1}^{D_B} \ln \frac{\mathrm{PG}(\omega_{nd}|1,0)}{\mathrm{PG}(\omega_{nd}|1,\hat{e}_{nd})}\Big] \\
&= \mathbb{E}_q\Big[\sum_{n=1}^{N}\sum_{d=1}^{D_B} \ln \Big[\mathrm{PG}(\omega_{nd}|1,0) \times \frac{\mathbb{E}_\omega[\exp(-\frac{\hat{e}_{nd}^2}{2}\omega_{nd})]}{\exp(-\frac{\hat{e}_{nd}^2}{2}\omega_{nd})\mathrm{PG}(\omega_{nd}|1,0)}\Big]\Big] \\
&= \mathbb{E}_q\Big[\sum_{n=1}^{N}\sum_{d=1}^{D_B} \ln \frac{\cosh^{-1}(\hat{e}_{nd}/2)}{\exp(-\frac{\hat{e}_{nd}^2}{2}\omega_{nd})}\Big] \\
&= \sum_{n=1}^{N}\sum_{d=1}^{D_B} \Big[-\ln(\cosh(\hat{e}_{nd}/2)) + \frac{\hat{e}_{nd}}{4}\tanh(\hat{e}_{nd}/2)\Big].
\end{aligned}
$$

Finally, the parts involving the data likelihoods are given by

$$
\begin{aligned}
\mathbb{E}_q[\ln p(\mathbf{X}^{(G)}|\mathbf{Z},\boldsymbol{\mu},\boldsymbol{\Lambda})] = \frac{1}{2}\sum_{k=1}^{K}\sum_{d=1}^{D_G} \Big[\hat{R}_k(-\ln(2\pi) + \psi(\hat{a}_k) - \ln\hat{b}_{kd} - \frac{1}{\hat{\beta}_k}) \\
- \frac{\hat{a}_k}{\hat{b}_{kd}}(\sum_{n=1}^{N}\hat{r}_{nk}(x_{nd}^{(G)})^2 - 2\hat{m}_{kd}\sum_{n=1}^{N}\hat{r}_{nk}x_{nd}^{(G)} + \hat{R}_k\hat{m}_{kd}^2)\Big],
\end{aligned}
$$

and

$$
\begin{aligned}
&\mathbb{E}_q[\ln p(\mathbf{X}^{(B)}|\mathbf{Z},\boldsymbol{\omega},\boldsymbol{\psi})] \\
&= \sum_{k=1}^{K}\sum_{d=1}^{D_M}\Big[-\hat{R}_k\ln 2 + \hat{\nu}_{kd}\sum_{n=1}^{N}\hat{r}_{nk}\kappa_{nd} - \frac{1}{2}(\hat{\nu}_{kd}^2 + \hat{\tau}_{kd}^{-1})\sum_{n=1}^{N}\hat{r}_{nk}\frac{1}{2\hat{e}_{nd}}\tanh(\hat{e}_{nd}/2)\Big].
\end{aligned}
$$

Appendix B

Variational Updates for Multinomial Extension of the GBMM-model

Calculation of $q(\mathbf{Z})$

The calculation is otherwise similar to what is presented in the GBMM-model, but the extra term $\langle \ln p(\mathbf{W}|\mathbf{Z}, \Xi)\rangle$ is needed when calculating the variational distribution $q(\mathbf{Z})$:

$$\begin{aligned}\ln q(\mathbf{Z}) = &\langle \ln p(\mathbf{X}^{(G)}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})\rangle + \langle \ln p(\mathbf{X}^{(B)}|\mathbf{Z}, \boldsymbol{\omega}, \boldsymbol{\psi})\rangle \\ &+ \langle \ln p(\mathbf{W}|\mathbf{Z}, \Xi)\rangle + \langle \ln p(\mathbf{Z}|\boldsymbol{\pi})\rangle + C.\end{aligned}$$

After simplifying we have

$$\langle \ln p(\mathbf{W}|\mathbf{Z}, \Xi)\rangle = \sum_{n=1}^{N}\sum_{k=1}^{K}\Big[-\frac{z_{nk}}{2}\sum_{d=1}^{D_M}\Big[C_d \ln(2\pi) + \langle (\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})^T(\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})\rangle\Big]\Big],$$

where the expectation is given as

$$\langle (\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})^T(\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})\rangle = \sum_{c=1}^{C_d}(\hat{w^2}_{ndc} - 2\hat{w}_{ndc}\hat{o}_{kdc} + (\hat{o}^2_{kdc} + \hat{\delta}^{-1}_k)).$$

Adding this new term to equation (A.4) in Appendix A gives the unnormalized responsibilities as

$$s_{nk} \propto \exp\Big\{\frac{1}{2}\Big(\sum_{d=1}^{D_G}(\psi(\hat{a}_k) - \ln\hat{b}_{kd}) - \sum_{d=1}^{D_G}(\frac{\hat{a}_k}{\hat{b}_{kd}}(x_{nd}^{(G)} - \hat{m}_{kd})^2 + \frac{1}{\hat{\beta}_k})\Big)$$
$$+ \sum_{d=1}^{D_B}\Big[\kappa_{nd}\hat{\nu}_{kd} - \frac{1}{4\hat{e}_{nd}}\tanh(\hat{e}_{nd}/2)(\hat{\tau}_{kd}^{-1} + \hat{\nu}_{kd}^2)\Big]$$
$$- \frac{1}{2}\sum_{d=1}^{D_M}\Big[\sum_{c=1}^{C_d}(\hat{w^2}_{ndc} - 2\hat{w}_{ndc}\hat{o}_{kdc} + (\hat{o}_{kdc}^2 + \hat{\delta}_k^{-1}))\Big] + \psi(\hat{\alpha}_k) - \psi(\sum_{k=1}^{K}\hat{\alpha}_k)\Big\}.$$

After this, the calculations for $q(\mathbf{Z})$ are exactly the same as in Appendix A.

Calculation of $q(\Xi)$

Again we see that the variational distribution factorizes over the clusters and data dimensions as

$$q(\Xi) = \prod_{k=1}^{K}\prod_{d=1}^{D_B} q(\boldsymbol{\xi}_{kd}).$$

We have

$$\begin{aligned}
\ln q(\boldsymbol{\xi}_{kd}) &= \langle \ln p(\mathbf{W}|\mathbf{Z},\Xi)\rangle + \ln p(\Xi) + C \\
&= \sum_{n=1}^{N}\hat{r}_{nk}\Big[-\frac{1}{2}\langle(\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})^T(\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})\rangle\Big] - \frac{\delta_{C_d}}{2}\boldsymbol{\xi}_{kd}^T\boldsymbol{\xi}_{kd} + C \\
&= -\frac{1}{2}\Big[\boldsymbol{\xi}_{kd}^T\boldsymbol{\xi}_{kd}(\hat{R}_k + \delta_{C_d}) - 2\boldsymbol{\xi}_{kd}^T\sum_{n=1}^{N}\hat{r}_{nk}\langle\mathbf{w}_{nd}\rangle\Big] + C \\
&= -\frac{\hat{R}_k + \delta_{C_d}}{2}\Big[\boldsymbol{\xi}_{kd}^T\boldsymbol{\xi}_{kd} - 2\frac{\boldsymbol{\xi}_{kd}^T\sum_{n=1}^{N}\hat{r}_{nk}\langle\mathbf{w}_{nd}\rangle}{\hat{R}_k + \delta_{C_d}}\Big] + C.
\end{aligned}$$

From this we recognize the functional form of a multivariate Gaussian

$$\mathrm{N}(\boldsymbol{\xi}_{kd}|\hat{\mathbf{o}}_{kd}, \hat{\delta}_k^{-1}\mathbf{I}),$$

with parameters given by

$$\hat{\mathbf{o}}_{kd} = \frac{\sum_{n=1}^{N} \hat{r}_{nk} \langle \mathbf{w}_{nd} \rangle}{\hat{R}_k + \delta_{C_d}} = \frac{\sum_{n=1}^{N} \hat{r}_{nk} \hat{\mathbf{w}}_{nd}}{\hat{R}_k + \delta_{C_d}},$$
$$\hat{\delta}_k = \hat{R}_k + \delta_{C_d}.$$

Calculation of $q(\mathbf{W})$

The variational distribution factorizes as

$$q(\mathbf{W}) = \prod_{n=1}^{N} \prod_{d=1}^{D_B} q(\mathbf{w}_{nd}).$$

The derivations presented here follow the ideas and derivations given in [28]. Without loss of generality assume $x_{nd}^{(M)} = c$. We begin by calculating the variational distributions $q(\mathbf{w}_{nd})$.

$$\begin{aligned}
\ln q(\mathbf{w}_{nd}) &= \ln p(\mathbf{X}^{(M)}|\mathbf{W}) + \langle \ln p(\mathbf{W}|\boldsymbol{\xi}, \mathbf{Z}) \rangle + C \\
&= \ln p(x_{nd}^{(M)}|\mathbf{w}_{nd}) - \frac{1}{2} \sum_{k=1}^{K} \hat{r}_{nk} \langle (\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})^T (\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd}) \rangle + C \\
&= \ln p(x_{nd}^{(M)}|\mathbf{w}_{nd}) - \frac{1}{2} (\mathbf{w}_{nd}^T \mathbf{w}_{nd} \underbrace{\sum_{k=1}^{K} \hat{r}_{nk}}_{=1} - 2\mathbf{w}_{nd}^T \sum_{k=1}^{K} \hat{r}_{nk} \langle \boldsymbol{\xi}_{kd} \rangle) + C \\
&= \ln p(x_{nd}^{(M)}|\mathbf{w}_{nd}) - \frac{1}{2} (\mathbf{w}_{nd}^T \mathbf{w}_{nd} - 2\mathbf{w}_{nd}^T \sum_{k=1}^{K} \hat{r}_{nk} \langle \boldsymbol{\xi}_{kd} \rangle) + C.
\end{aligned}$$

Thus

$$q(\mathbf{w}_{nd}) \propto \mathbb{1}_{\{\mathbf{w}_{nd} \in \mathcal{C}_{nd}\}} \exp\Big\{ -\frac{1}{2} (\mathbf{w}_{nd}^T \mathbf{w}_{nd} - 2\mathbf{w}_{nd}^T \sum_{k=1}^{K} \hat{r}_{nk} \langle \boldsymbol{\xi}_{kd} \rangle) \Big\},$$

where $\mathcal{C}_{nd} = \{\mathbf{w} \in \mathbb{R}^{C_d} | w_i < w_c \quad \forall i \neq c, \quad x_{nd} = c\}$. This is recognized as the kernel of a truncated C_d-dimensional multivariate normal distribution

$$\mathrm{N}_{x_{nd}^{(M)}}(\mathbf{w}_{nd}|\hat{\boldsymbol{\xi}}_{nd}, \mathbf{I}),$$

where $\hat{\boldsymbol{\xi}}_{nd} = \sum_{k=1}^{K} \hat{r}_{nk} \langle \boldsymbol{\xi}_{kd} \rangle = \sum_{k=1}^{K} \hat{r}_{nk} \hat{\mathbf{o}}_{kd}$ and the truncation is such that for a data point for which $x_{nd}^{(M)} = c$ the c^{th} dimension is the largest. We use $\mathrm{N}_{x_{nd}^{(M)}}(\cdot|\cdot,\cdot)$ to denote a truncated normal distribution where the largest dimension is specified by $x_{nd}^{(M)}$.

We proceed to calculate the expectations that are required for the variational updates. We denote $\hat{\mathbf{w}}_{nd} := \mathbb{E}_q[\mathbf{w}_{nd}] = (\hat{w}_{nd1}, ..., \hat{w}_{ndC_d})^T$ for the first moments and $\hat{\mathbf{w}^2}_{nd} := \mathbb{E}_q[\mathbf{w}_{nd}^2] = (\hat{w^2}_{nd1}, ..., \hat{w^2}_{ndC_d})^T$ for the second moments of the truncated distribution.

The variational distribution $q(\mathbf{w}_{nd})$ can be written as

$$q(\mathbf{w}_{nd}) = \mathbb{1}_{\{\mathbf{w}_{nd} \in \mathcal{C}_{nd}\}} \mathcal{Z}_{nd}^{-1} \prod_{i=1}^{C_d} \mathrm{N}(w_{ndi} | \hat{\xi}_{ndi}, 1), \tag{B.1}$$

where $\mathcal{Z}_{nd} = P(\mathbf{w}_{nd} \in \mathcal{C}_{nd})$ and the normal distributions for each w_{ndi} are untruncated. Let u and u_i be generic standard normal distributed random variables. Using the affine transformation property of normal distribution and the equation (B.1) we have

$$\begin{aligned}
\mathcal{Z}_{nd} &= P(\mathbf{w}_{nd} \in \mathcal{C}_{nd}) \\
&= \mathbb{E}_{w_{ndc}}[\prod_{i \neq c} P(w_{ndi} < w_{ndc} | w_{ndc})] \\
&= \mathbb{E}_u[\prod_{i \neq c} P(u_i < u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi} | u)] \\
&= \mathbb{E}_u[\prod_{i \neq c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi})].
\end{aligned} \tag{B.2}$$

Now we calculate the expectations for components $i \neq c$ as

$$
\begin{aligned}
\hat{w}_{ndi} &= \mathcal{Z}_{nd}^{-1} \int_{-\infty}^{\infty} \mathrm{N}(w_{ndc}|\hat{\xi}_{ndc}, 1) \int_{-\infty}^{w_{ndc}} w_{ndi} \mathrm{N}(w_{ndi}|\hat{\xi}_{ndi}, 1) \\
&\quad \times \prod_{j \neq i,k} \int_{-\infty}^{w_{ndc}} \mathrm{N}(w_{ndj}|\hat{\xi}_{ndj}, 1) \, \mathrm{d}w_{ndj} \, \mathrm{d}w_{ndi} \, \mathrm{d}w_{ndc} \\
&= \mathcal{Z}_{nd}^{-1} \mathbb{E}_u \Big[\int_{-\infty}^{u+\hat{\xi}_{ndc}-\hat{\xi}_{ndi}} (u_i + \hat{\xi}_{ndi}) \phi(u_i) \, \mathrm{d}u_i \prod_{j \neq i,c} \int_{-\infty}^{u+\hat{\xi}_{ndc}-\hat{\xi}_{ndj}} \phi(u_j) \, \mathrm{d}u_j \Big] \\
&\overset{(\star)}{=} \hat{\xi}_{ndi} + \mathcal{Z}_{nd}^{-1} \mathbb{E}_u \Big[- \int_{-\infty}^{u+\hat{\xi}_{ndc}-\hat{\xi}_{ndi}} \phi'(u_i) \, \mathrm{d}u_i \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj}) \Big] \\
&= \hat{\xi}_{ndi} - \mathcal{Z}_{nd}^{-1} \mathbb{E}_u [\phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi}) \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})]. \qquad \text{(B.3)}
\end{aligned}
$$

In $(\star)$ we used the result in (B.2) and the easily verifiable identity $-x\phi(x) = \phi'(x)$, where ϕ is the probability density function of a $\mathrm{N}(0, 1)$ distribution.

Then we calculate the expectation for the component c as

$$
\begin{aligned}
\hat{w}_{ndc} &= \mathcal{Z}_{nd}^{-1} \int_{-\infty}^{\infty} w_{ndc} \mathrm{N}(w_{ndc}|\hat{\xi}_{ndc}, 1) \prod_{j \neq c} \int_{-\infty}^{w_{ndc}} \mathrm{N}(w_{ndc}|\hat{\xi}_{ndj}, 1) \, \mathrm{d}w_{ndj} \, \mathrm{d}w_{ndc} \\
&= \mathcal{Z}_{nd}^{-1} \mathbb{E}_u [(u + \hat{\xi}_{ndc}) \prod_{j \neq c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})] \\
&= \hat{\xi}_{ndc} + \mathcal{Z}_{nd}^{-1} \mathbb{E}_u [u \prod_{j \neq c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})] \\
&\overset{(\star)}{=} \hat{\xi}_{ndc} + \mathcal{Z}_{nd}^{-1} \sum_{l \neq c} \mathbb{E}_u [\phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndl}) \prod_{j \neq l,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})] \\
&\overset{(\star\star)}{=} \hat{\xi}_{ndc} + \sum_{l \neq c} (\hat{\xi}_{ndl} - \hat{w}_{ndl}),
\end{aligned}
$$

where we used:

- $(\star)$ The fact that for any differentiable function g and random variable $u \sim \mathrm{N}(0, 1)$ it holds that $\mathbb{E}[ug(u)] = \mathbb{E}[g'(u)]$, assuming that $\lim_{u \to \pm\infty} \phi(u)g(u) = 0$. This is easily proven by the identity $-x\phi(x) = \phi'(x)$ and integration by parts.
- $(\star\star)$ The result that was calculated in (B.3).

We still need the second moments for each of the components of $\mathbf{w}_{nd}$ as they appear in the variational distribution $q(\mathbf{Z})$. The principle for calculating them is the same as above, but some additional intermediate steps are needed.

First for $i \neq c$ we have

$$
\begin{aligned}
\hat{w^2}_{ndi} &= \mathcal{Z}_{nd}^{-1} \int_{-\infty}^{\infty} \mathrm{N}(w_{ndc}|\hat{\xi}_{ndc}, 1) \int_{-\infty}^{w_{ndc}} w_{ndi}^2 \mathrm{N}(w_{ndi}|\hat{\xi}_{ndi}, 1) \\
&\quad \times \prod_{j \neq i,c} \int_{-\infty}^{w_{ndc}} \mathrm{N}(w_{ndj}|\hat{\xi}_{ndj}, 1)\, \mathrm{d}w_{ndj}\, \mathrm{d}w_{ndi}\, \mathrm{d}w_{ndc} \\
&= \mathcal{Z}_{nd}^{-1} \mathbb{E}_u \Big[\int_{-\infty}^{u+\hat{\xi}_{ndc}-\hat{\xi}_{ndi}} (u_i + \hat{\xi}_{ndi})^2 \phi(u_i)\, \mathrm{d}u_i \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj}) \Big] \\
&= \hat{\xi}_{ndi}^2 + \mathcal{Z}_{nd}^{-1} \mathbb{E}_u \Big[\int_{-\infty}^{u+\hat{\xi}_{ndc}-\hat{\xi}_{ndi}} (u_i^2 + 2\hat{\xi}_{ndi} u_i) \phi(u_i)\, \mathrm{d}u_i \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj}) \Big] \\
&\overset{(\star)}{=} \hat{\xi}_{ndi}^2 + \mathcal{Z}_{nd}^{-1} \mathbb{E}_u \Big[\Big[- \int_{-\infty}^{u+\hat{\xi}_{ndc}-\hat{\xi}_{ndi}} u_i \phi'(u_i)\, \mathrm{d}u_i - 2\hat{\xi}_{ndi} \phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi}) \Big] \\
&\quad \times \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj}) \Big] \\
&\overset{(\star\star)}{=} \hat{\xi}_{ndi}^2 + 1 + \mathcal{Z}_{nd}^{-1} \mathbb{E}_u \Big[- (u + \hat{\xi}_{ndc} + \hat{\xi}_{ndi}) \phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi}) \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj}) \Big] \\
&\overset{(\star\star\star)}{=} \hat{\xi}_{ndi}^2 + 1 - (\hat{\xi}_{ndc} + \hat{\xi}_{ndi})(\hat{\xi}_{ndi} - \hat{w}_{ndi}) \\
&\quad - \mathcal{Z}_{nd}^{-1} \mathbb{E}_u [u \phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndi}) \prod_{j \neq i,c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})],
\end{aligned}
$$

where we used:

- $(\star)$ Identity $-x\phi(x) = \phi'(x)$ for u_i^2 and u_i.
- $(\star\star)$ Integration by parts and rearranging terms.
- $(\star\star\star)$ Result in (B.3).

The second moment for w_{ndc} is then calculated as

$$\begin{aligned}
\hat{w^2}_{ndc} &= \mathcal{Z}_{nd}^{-1} \int_{-\infty}^{\infty} w_{ndc}^2 \mathrm{N}(w_{ndc}|\hat{\xi}_{ndc}, 1) \prod_{j\neq c} \int_{-\infty}^{w_{ndc}} \mathrm{N}(w_{ndj}|\hat{\xi}_{ndj}, 1)\, \mathrm{d}w_{ndj} \mathrm{d}w_{ndc} \\
&= \mathcal{Z}_{nd}^{-1} \mathbb{E}_u \Big[(u + \hat{\xi}_{ndc})^2 \prod_{j\neq c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj}) \Big] \\
&\overset{(\star)}{=} \hat{\xi}_{ndc}^2 + \mathcal{Z}_{nd}^{-1} \mathbb{E}_u [u^2 \prod_{j\neq c} \Phi(u + \hat{\xi}_{ndc} - \hat{\xi}_{ndj})] + 2\hat{\xi}_{ndc} \sum_{l\neq c} (\hat{\xi}_{ndl} - \hat{w}_{ndl}),
\end{aligned}$$

where in $(\star)$ we used results in (B.2) and (B.3) to simplify the expression.

Lower bound terms

The bound for $\mathbf{W}$ is

$$\begin{aligned}
&\mathbb{E}_q[\ln p(\mathbf{W}|\mathbf{Z}, \boldsymbol{\Xi})] - \mathbb{E}_q[\ln q(\mathbf{W})] \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} \sum_{d=1}^{D_M} \mathbb{E}_q[z_{nk}] \Big[-\frac{C_d}{2} \ln(2\pi) - \frac{1}{2} \mathbb{E}_q[(\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})^T (\mathbf{w}_{nd} - \boldsymbol{\xi}_{kd})] \Big] \\
&\quad - \sum_{n=1}^{N} \sum_{d=1}^{D_M} \Big[\mathbb{E}_q[\ln \mathbb{1}_{\{\mathbf{w}_{nd} \in \mathcal{C}_{nd}\}}] - \ln \mathcal{Z}_{nd} - \frac{C_d}{2} \ln(2\pi) - \frac{1}{2} \mathbb{E}_q[(\mathbf{w}_{nd} - \hat{\boldsymbol{\xi}}_{nd})^T (\mathbf{w}_{nd} - \hat{\boldsymbol{\xi}}_{nd})] \Big].
\end{aligned}$$

The expectations are easy to calculate as the required moments with respect to the variational distribution are known. Especially

$$\mathbb{E}_q[\ln \mathbb{1}_{\{\mathbf{w}_{nd} \in \mathcal{C}_{nd}\}}] = -\infty \cdot Q(\mathbf{w}_{nd} \notin \mathcal{C}_{nd}) + 0 \cdot Q(\mathbf{w}_{nd} \in \mathcal{C}_{nd}) = 0, \qquad \text{(B.4)}$$

where Q is the variational measure. Since the variational distribution for $\mathbf{w}_{nd}$ is truncated so that $\mathbf{w}_{nd} \in \mathcal{C}_{nd}$ almost surely (that is $Q(\mathbf{w}_{nd} \in \mathcal{C}_{nd}) = 1$), the above expectation simplifies to zero[1]. After further simplifications we have the bound as

$$\sum_{n,d} \ln \mathcal{Z}_{nd} + \sum_{n,d,c,k} \hat{r}_{nk} \hat{w}_{ndc} \hat{o}_{kdc} - \sum_{n,d,c} \hat{w}_{ndc} \hat{\xi}_{ndc} - \frac{1}{2} \sum_{d,c,k} \hat{R}_k (\hat{o}_{kdc}^2 + \hat{\delta}_k^{-1}) + \frac{1}{2} \sum_{n,d,c} \hat{\xi}_{ndc}^2.$$

[1] We define $0 \cdot \pm\infty$ to be 0, as is usually done in probability theory.

For $\boldsymbol{\Xi}$ the bound is given by

$$\mathbb{E}_q[\ln p(\boldsymbol{\Xi})] - \mathbb{E}_q[\ln q(\boldsymbol{\Xi})] = \frac{1}{2}\sum_{k=1}^{K}\sum_{d=1}^{D_M}\Big[C_d(1+\ln\frac{\delta_{C_d}}{\hat{\delta}_k}) - \delta_{C_d}\sum_{c=1}^{C_d}(\hat{o}_{kdc}^2 + \hat{\delta}_k^{-1})\Big],$$

and finally the bound for $\mathbf{X}^{(M)}$ is

$$\mathbb{E}_q[\ln p(\mathbf{X}^{(M)}|\mathbf{W})] = \sum_{n=1}^{N}\sum_{d=1}^{D_M}\mathbb{E}_q[\ln \mathbb{1}_{\{\mathbf{w}_{nd}\in\mathcal{C}_{nd}\}}] = 0,$$

by the same argument as in (B.4).