A STUDY ON VARIATIONAL COMPONENT SPLITTING APPROACH FOR MIXTURE MODELS

KAMAL MAANICSHAH MATHIN HENRY

A THESIS
IN
CONCORDIA INSTITUTE FOR INFORMATION SYSTEMS ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF APPLIED SCIENCE
(INFORMATION SYSTEMS SECURITY)
CONCORDIA UNIVERSITY
MONTRÉAL, QUÉBEC, CANADA

AUGUST 2019

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Kamal Maanicshah Mathin Henry**

Entitled: **A Study on Variational Component Splitting Approach for Mixture Models**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science
(Information Systems Security)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

Dr. Roch H. Glitho _____ Chair

Dr. Nizar Bouguila _____ Supervisor

Dr. Jamal Bentahar _____ CIISE Examiner

Dr. Fuzhan Nasiri _____ External Examiner

Approved _____

Dr. Mohammad Mannan, Graduate Program Director

2019.07.23 _____

Dr. Amir Asif, Dean
Faculty of Engineering and Computer Science

Abstract

A Study on Variational Component Splitting Approach for Mixture Models

Kamal Maanicshah Mathin Henry

Increase in use of mobile devices and the introduction of cloud-based services have resulted in the generation of enormous amount of data every day. This calls for the need to group these data appropriately into proper categories. Various clustering techniques have been introduced over the years to learn the patterns in data that might better facilitate the classification process. Finite mixture model is one of the crucial methods used for this task. The basic idea of mixture models is to fit the data at hand to an appropriate distribution. The design of mixture models hence involve finding the appropriate parameters of the distribution and estimating the number of clusters in the data.

We use a variational component splitting framework to do this which could simultaneously learn the parameters of the model and estimate the number of components in the model. The variational algorithm helps to overcome the computational complexity of purely Bayesian approaches and the over fitting problems experienced with Maximum Likelihood approaches guaranteeing convergence. The choice of distribution remains the core concern of mixture models in recent research. The efficiency of Dirichlet family of distributions for this purpose has been proved in latest studies especially for non-Gaussian data. This led us to study the impact of variational component splitting approach on mixture models based on several distributions.

Hence, our contribution is the application of variational component splitting approach to design finite mixture models based on inverted Dirichlet, generalized inverted Dirichlet and inverted Beta-Liouville distributions. In addition, we also incorporate a simultaneous feature selection approach for generalized inverted Dirichlet mixture model along with component splitting as another experimental contribution. We evaluate the performance of our models with various real-life applications such as object, scene, texture, speech and video categorization.

Acknowledgments

I would like to express my deepest gratitude to my supervisor Prof. Nizar Bouguila, who has widened my view of machine learning research with his vast knowledge on the subject. During the course of two years working together, he has always been a wise, witty, patient and cool mentor. Despite my slow start, his constant motivation has inspired me to move forward to complete the milestones. I will always be grateful for his relentless support and guidance.

I express my profound gratitude to my co-supervisor Dr. Wentao Fan for his technical advise without which the completion of this work would have been very hard.

A special thanks to Mr. Muhammad Azam who was like a big brother to me here. He always made sure that I was in high spirits and kept me motivated when I felt low.

I am also grateful to Eddy who has been the center of a number of discussions which helped me understand the theory easily.

I have been lucky to have such awesome lab mates who have made two years unforgettable. I would like to thank Hieu, Jaspreet, Narges, Meeta, Maryam, Samr, Omar, Basim, Huda, Rim, Fatma, Divya, Shuai and other lab members, who have always shared their vast knowledge as well as taken the time and effort to explain various concepts.

I would like to extend my gratitude to my roommates and all other friends who have supported me with all their love and support. Last but not least, I am deeply grateful to my parents who encouraged me to work hard and sent loads of love over phone which made me work at my best.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Cluster Analysis via Finite Mixture Models

Given a data set containing data from different categories, the data points belonging to each category possess similar properties quite different from those that belong to other categories. Clustering algorithms exploit this similarity to group data points belonging to different categories. Use of mixture models is one of the methods for clustering data. Finite mixture models assume that the data can be represented as a mixture of multiple distributions. Finite mixture models based on various distributions have been proposed in recent years. Especially, Gaussian mixture models are used widely in the industry and have a number of applications [1–4]. However, it has been found that Gaussian mixture models cannot fit all types of data. It has been proved that different distributions such as scaled Dirichlet [5], generalized inverted Dirichlet [6], Beta [7], inverted Beta-Liouville [8], etc. can be used to model non-Gaussian distributions. Choice of the distribution hence proves to be an important stage in mixture model design. For our study, we choose inverted Dirichlet, Generalized inverted Dirichlet and inverted Beta-Liouville distribution as they have proved to be efficient in recent studies. Inverted Dirichlet distribution performs really well for semi bounded positive vectors [9, 10]. Generalized inverted Dirichlet has more degrees of freedom as compared to inverted Dirichlet distribution and hence fits the data better [11,12]. Also, inverted Beta-Liouville distribution has been proved to be an efficient supplement as well [8]. This choice helps us to study three different distributions with varying properties.

One of the important stages in designing a mixture model is the parameters estimation. Usually this is done by deterministic methods such as maximum likelihood estimation as

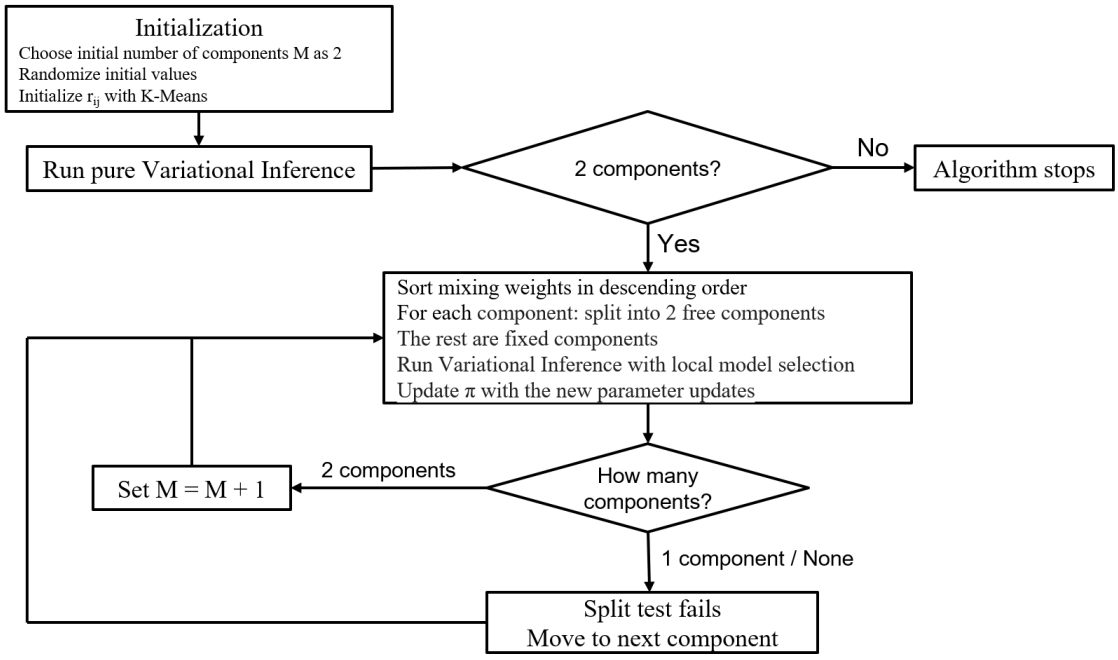


Figure 1: Flowchart for the component splitting algorithm

in [13]. Though these methods are found to be useful, they suffer a pitfall which is that they might converge to a local maximum which might be farther from the global one [14]. This results in wrong selection of the number of components. The issue has been addressed in various ways, for example, in [13], [15] and [16] the minimum message length criterion has been used for model selection. Some researchers used a completely Bayesian approach as in [11, 17]. On the downside the computation cost of these fully Bayesian approaches is very high and convergence is not guaranteed.

In order to gain a middle ground between the pitfalls of entirely deterministic and Bayesian methods the variational inference approach was proposed [18]. This method has then been tested on various models [19,20]. The variational approach approximates the true posterior distribution instead of computing it. This reduces the complexity of the model and furthermore, convergence is guaranteed. More information about the variational approach is given in [21]. The concept of variational parameter estimation will be explained briefly in the next chapter. In our work we follow the variational algorithm proposed in [22]. This algorithm used a global model selection process. Component splitting on the other hand is a local model selection method. An interesting technique based on component splitting has been proposed in [20] which was then followed in [23]. This algorithm follows an incremental approach initially starting with one or two clusters and then uses a split and merge

criterion to determine the optimal number of components. The splitting continues until all clusters fail the split test. This is very efficient as it takes place within the variational learning algorithm. A flow chart of the algorithm is shown in Fig. 1.

The algorithm will be explained in detail in Chapter 2. The authors in [23] have developed an efficient application of this model to finite Dirichlet mixture model. Our intention is to study the efficiency of this model when applied to inverted Dirichlet (ID), generalized inverted Dirichlet (GID) and inverted Beta-Liouville (IBL) mixture models. We have applied these models to a wide variety of applications such as multimedia (image, video, audio and texture) categorization, software defect detection, spam image detection in emails and occupancy estimation in smart homes.

1.2 Contributions

The main objective of this thesis is to study the efficiency of ID, GID and IBL mixture models when integrated with the variational component splitting algorithm. The contributions are listed as follows:

- ☞ **Data Clustering using Finite Inverted Dirichlet Mixture Models with a Variational Component Splitting Approach**

  We propose a finite Inverted Dirichlet mixture model for unsupervised learning using variational inference with a component splitting algorithm. We illustrate our model and learning algorithm with synthetic data and some real applications for occupancy estimation in smart homes and topic learning in images and videos. This work has been submitted to *Journal of Applied Intelligence (Springer)* and is under revision.

- ☞ **Data Clustering using Finite Generalized Inverted Dirichlet Mixture Models with a Variational Component Splitting Approach**

  A finite generalized inverted Dirichlet mixture model with a variational learning method integrated with component splitting is proposed. Efficiency of proposed model is tested for image categorization tasks. This contribution has been accepted in the 28^{th} *International Symposium on Industrial Electronics*. Incorporating a variational feature selection approach, this work has been extended

and also accepted by the 16^{th} *International Conference on Image Analysis and Recognition*.

☞ **Data Clustering using Finite Inverted Beta-Liouville Mixture Models with a Variational Component Splitting Approach**

We introduce a finite mixture model based on Inverted Beta-Liouville distribution which provides a better fit for the data with a component splitting approach for model selection. We evaluate our model against some challenging applications like image clustering, speech clustering, spam image detection and software defect detection. This work has been accepted as a book chapter in the book titled *Mixture Models and Applications*.

1.3 Thesis Overview

❑ Chapter 1 introduces the concepts of clustering and a brief overview of various concepts related to the work. We also explain clearly the motivations behind the conducted research work.

❑ In Chapter 2, we explain in detail the variational inference learning for inverted Dirichlet mixture models with the component splitting approach. The efficiency of the proposed model in solving various problems such us occupancy estimation in smart homes, image and video clustering is also explained.

❑ In chapter 3, we integrate variational component splitting approach with generalized inverted Dirichlet mixture models along with variational feature selection. The experiments with various applications like image clustering, image texture analysis and video texture analysis is described in detail.

❑ Chapter 4 describes the application of variational component splitting approach to inverted Beta-Liouville mixture models. The model has been tested with challenging applications such as software defect detection, image spam detection in emails, image clustering and speech clustering.

❑ In conclusion, we briefly summarize our contributions.

Chapter 2

Finite Inverted Dirichlet Mixture Model with Variational Component Splitting

In this chapter, we detail the design of component splitting algorithm applied to inverted Dirichlet mixture models within the variational framework. We illustrate our model and learning algorithm with synthetic data and some real applications for occupancy estimation in smart homes and topic learning in images and videos. Extensive comparisons with comparable recent approaches have shown the merits of our proposed model.

2.1 Model Specification

The first part of this section will give a brief description of finite Inverted Dirichlet mixture model and the next part will apply local model selection on the model using component splitting approach.

2.1.1 Finite Inverted Dirichlet mixture model

Considering we have N samples of independent identically distributed vectors generated from Inverted Dirichlet distribution $\mathcal{X} = \big(\vec{X}_1, \vec{X}_2, ..., \vec{X}_N\big)$, each sample is a D-dimensional vector $\vec{X}_i = \big(X_{i1}, X_{i2}, ...X_{iD}\big)$ having a probability density function given by [24]

$$ID\big(\vec{X}_i \mid \vec{\alpha}_j\big) = \frac{\Gamma\big(\sum_{l=1}^{D+1} \alpha_{jl}\big)}{\prod_{l=1}^{D+1} \Gamma\big(\alpha_{jl}\big)} \prod_{l=1}^{D} X_{il}^{\alpha_{jl}-1} \Big(1 + \sum_{l=1}^{D} X_{il}\Big)^{-\sum_{l=1}^{D+1} \alpha_{jl}} \tag{2.1}$$

where, X_{il} is positive for $l = 1, ..., D$ and $\vec{\alpha}_j = \big(\alpha_{j1}, \alpha_{j2}, ...\alpha_{jD+1}\big), \alpha_{jl} > 0$ for $l = 1, ..., D+1$. The mean and variance of the Inverted Dirichlet distribution is given by,

$$\mathbb{E}\big[X_l\big] = \frac{\alpha_l}{\alpha_{D+1} - 1}, \;\; var\big(X_l\big) = \frac{\alpha_l\big(\alpha_j + \alpha_{D+1} - 1\big)}{\big(\alpha_{D+1} - 1\big)^2\big(\alpha_{D+1} - 2\big)} \tag{2.2}$$

Considering an Inverted Dirichlet mixture model with M components, $ID\big(\vec{X}_i \mid \vec{\alpha}_j\big)$ represents the distribution for j^{th} component with parameter $\vec{\alpha}_j$. $\vec{\alpha} = \big(\vec{\alpha}_1, \vec{\alpha}_2, ..., \vec{\alpha}_j\big)$ is the parameter vector for the respective M components. According to this terminology, for each sample, the mixture model can be represented as,

$$p\big(\vec{X}_i \mid \vec{\pi}, \vec{\alpha}\big) = \sum_{j=1}^{M} \pi_j ID\big(\vec{X}_i \mid \vec{\alpha}_j\big) \tag{2.3}$$

$\vec{\pi} = \big(\pi_1, \pi_2, ..., \pi_M\big)$ is the set of mixing coefficients, such that, $\sum_{j=1}^{M} \pi_j = 1$. So the likelihood function for N samples is,

$$p\big(\mathcal{X} \mid \vec{\pi}, \vec{\alpha}\big) = \prod_{i=1}^{N} \left[\sum_{j=1}^{M} \pi_j ID\big(\vec{X}_i \mid \vec{\alpha}_j\big) \right] \tag{2.4}$$

In order to calculate the maximum likelihood, we define a latent variable $\mathcal{Z}$ which represents the expectation for each sample to belong to one of the M components. Mathematically, $\vec{Z}_i = \big(Z_{i1}, Z_{i2}, Z_{iM}\big)$ where Z_{ij} is 1 if the expectation that $\vec{X}_i$ belongs to the component j is higher than the other components and 0 elsewhere. The conditional probability for the N latent variables $\mathcal{Z} = \big(\vec{Z}_1, \vec{Z}_2, ..., \vec{Z}_N\big)$ given $\vec{\pi}$ is,

$$p\big(\mathcal{Z} \mid \vec{\pi}\big) = \prod_{i=1}^{N} \prod_{j=1}^{M} \pi_j^{Z_{ij}} \tag{2.5}$$

From this equation we can write the conditional probability of the data given the class labels $\mathcal{Z}$ as,

$$p\big(\mathcal{X} \mid \mathcal{Z}, \vec{\alpha}\big) = \prod_{i=1}^{N} \prod_{j=1}^{M} ID\big(\vec{X}_i \mid \vec{\alpha}_j\big)^{Z_{ij}} \tag{2.6}$$

2.1.2 Component splitting for model selection

Model selection is one of the major problems with mixture models. Component splitting is an interesting approach to alleviate this problem. The efficiency of this method on mixture

models has been studied in [20] and [23] for Gaussian and finite Dirichlet mixture models, respectively. We use this approach on finite inverted Dirichlet mixture model. The component splitting algorithm involves partitioning the components in the mixture model based on the split criteria. The partitions are made by splitting each component in the mixture model and among the split components, one set is called the free components and the remaining set of components are considered as the fixed components. The algorithm performs calculations on the free components based on the assumption that the fixed components already fit the data. In other words, if we are estimating s components considering them as free components, we assume that the remaining $M - s$ components fit to the data already. Based on this idea we can rewrite the equation $\big(2.5\big)$ as,

$$p\big(\mathcal{Z} \mid \vec{\pi}, \vec{\pi}^*\big) = \prod_{i=1}^{N} \left[\prod_{j=1}^{s} \pi_j^{Z_{ij}} \prod_{j=s+1}^{M} \pi_j^{*Z_{ij}} \right] \tag{2.7}$$

where $\{\pi_j\}$ represents the mixing coefficients of the free components and $\{\pi_j^*\}$ represents the mixing coefficients of fixed components with the constraint,

$$\sum_{j=1}^{s} \pi_j + \sum_{j=s+1}^{M} \pi_j^* = 1 \tag{2.8}$$

In our case $\vec{\pi}_j$ is considered more like a parameter rather than a random variable, whereas $\vec{\pi}_j^*$ is considered as random variable. Due to this reason we fit a prior distribution over the mixing coefficients of the fixed components which is based on the free components. Our objective here is to find the conditional probability of fixed components given the free components. This will help us estimate the maximum likelihood of the free components without affecting the fixed components. We choose the distribution to be a nonstandard Dirichlet distribution which is also the conjugate prior of the fixed coefficients as mentioned in [20],

$$p\big(\vec{\pi}^* \mid \vec{\pi}\big) = \left(1 - \sum_{k=1}^{s} \pi_k\right)^{-M+s} \frac{\Gamma\big(\sum_{j=s+1}^{M} c_j\big)}{\prod_{j=s+1}^{M} \Gamma\big(c_j\big)} \prod_{j=s+1}^{M} \left(\frac{\pi_j^*}{1 - \sum_{k=1}^{s} \pi_k}\right)^{c_j - 1} \tag{2.9}$$

Next, we have to find a prior for the $\vec{\alpha}$ parameters. Unfortunately, a conjugate prior does not exist. Thus, we use a Gamma prior as an approximation assuming that the parameters are statistically independent. So, the probability density function of α_{jl} is now given by,

$$p\big(\alpha_{jl}\big) = \mathcal{G}\big(\alpha_{jl} \mid u_{jl}, \nu_{jl}\big) = \frac{\nu_{jl}^{u_{jl}}}{\Gamma\big(u_{jl}\big)} \alpha_{jl}^{u_{jl}-1} e^{-\nu_{jl}\alpha_{jl}} \tag{2.10}$$

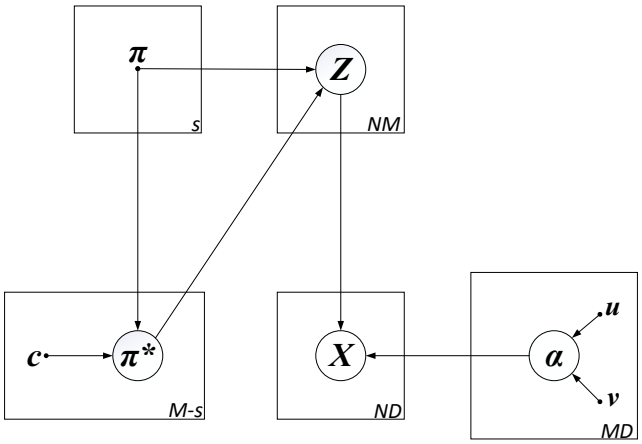


Figure 2.1: Graphical model of finite inverted Dirichlet mixture with component splitting. Circles represent the random variables and parameters. Plates denote repetitions. Number in the lower right corners of the plates indicate the number of repetitions. The conditional dependencies of the variables are represented by the arcs.

Here, $\{u_{jl}\}$ and $\{\nu_{jl}\}$ are hyperparameters such that $u_{jl} > 0$ and $\nu_{jl} > 0$. Now considering $\vec{\alpha}$ we can write,

$$p(\vec{\alpha}) = \prod_{j=1}^{M}\prod_{l=1}^{D} p(\alpha_{jl}) \tag{2.11}$$

Now, using all the details available to us, the joint distribution of all the random variables is given by

$$\begin{aligned}
p(\mathcal{X}, \mathcal{Z}, \vec{\alpha}, \vec{\pi}^* \mid \vec{\pi}) =& p(\mathcal{X} \mid \mathcal{Z}, \vec{\alpha}) p(Z \mid \vec{\pi}, \vec{\pi}^*) p(\vec{\pi}^* \mid \vec{\pi}) p(\vec{\alpha}) \\
=& \prod_{i=1}^{N}\prod_{j=1}^{M}\left[\frac{\Gamma\left(\sum_{l=1}^{D+1}\alpha_{jl}\right)}{\prod_{l=1}^{D+1}\Gamma(\alpha_{jl})}\prod_{l=1}^{D} X_{il}^{\alpha_{jl}-1}\left(1+\sum_{l=1}^{D} X_{il}\right)^{-\sum_{l=1}^{D+1}\alpha_{jl}}\right]^{Z_{ij}} \\
&\times \prod_{i=1}^{N}\left[\prod_{j=1}^{s}\pi_j^{Z_{ij}}\prod_{j=s+1}^{M}\pi_j^{*Z_{ij}}\right] \times \left(1-\sum_{k=1}^{s}\pi_k\right)^{-M+s} \\
&\times \frac{\Gamma\left(\sum_{j=s+1}^{M} c_j\right)}{\prod_{j=s+1}^{M}\Gamma(c_j)}\prod_{j=s+1}^{M}\left(\frac{\pi_j^*}{1-\sum_{k=1}^{s}\pi_k}\right)^{c_j-1} \\
&\times \frac{\nu_{jl}^{u_{jl}}}{\Gamma(u_{jl})}\alpha_{jl}^{u_{jl}-1}e^{-\nu_{jl}\alpha_{jl}}
\end{aligned} \tag{2.12}$$

A graphical representation of this model is shown in Fig. 2.1.

2.2 Variational inference

Here we introduce the variational inference method for parameter estimation and then we describe the algorithm for our model based on variational learning.

2.2.1 Variational learning

We estimate the parameters for our model based on the approach followed in [25]. For easy reference let's say $\Theta = \{\mathcal{Z}, \vec{\alpha}, \vec{\pi}^*\}$. Our main objective is to find the posterior distribution $p(\Theta \mid \mathcal{X}, \vec{\pi})$. Variational learning helps us by estimating a distribution $Q(\Theta)$ which is an approximation to the real posterior distribution $p(\Theta \mid \mathcal{X}, \vec{\pi})$. To determine this approximation, we compute the Kullback-Leibler (KL) divergence between $Q(\Theta)$ and $p(\Theta \mid \mathcal{X}, \vec{\pi})$ given by,

$$KL(Q \mid\mid P) = -\int Q(\Theta) \ln \left(\frac{p(\Theta \mid \mathcal{X}, \vec{\pi})}{Q(\Theta)} \right) d\Theta \tag{2.13}$$

Modifying this equation we can write

$$KL(Q \mid\mid P) = \ln\ p(\mathcal{X} \mid \vec{\pi}) - \mathcal{L}(Q) \tag{2.14}$$

where,

$$\mathcal{L}(Q) = \int Q(\Theta) \ln \left(\frac{p(\mathcal{X}, \Theta \mid \vec{\pi})}{Q(\Theta)} \right) d\Theta \tag{2.15}$$

The KL divergence being a similarity measure follows the conditions $KL(Q \mid\mid P) \geq 0$ and $KL(Q \mid\mid P) = 0$ when $Q(\Theta) = p(\Theta \mid \mathcal{X})$. From (2.14) we can say $\mathcal{L}(Q)$ is the lower bound of $p(\mathcal{X} \mid \vec{\pi})$. Maximizing this lower bound means we are minimizing the KL divergence and hence approximating the true posterior distribution. However, the true posterior distribution cannot be used directly for variational inference as it is computationally intractable. For this reason we use the method of mean-field approximation [26] [21] [27] by which we factorize $Q(\Theta)$, such that,

$$Q(\Theta) = Q(\mathcal{Z}) Q(\vec{\alpha}) Q(\vec{\pi}^*) \tag{2.16}$$

Now to maximize the lower bound we find the variational solution for $\mathcal{L}(Q)$ with respect to each of the parameters [25]. The variational solution for a specific parameter $Q_k(\Theta_k)$ is

$$Q_k(\Theta_k) = \frac{exp\langle \ln\ p(\mathcal{X}, \Theta) \rangle_{\neq k}}{\int exp\langle \ln\ p(\mathcal{X}, \Theta) \rangle_{\neq k} d\Theta} \tag{2.17}$$

where $\left\langle \cdot \right\rangle_{\neq k}$ is the expectation with respect to all the parameters other than Θ_k.

Next we find the solutions for the optimal variational posteriors using $\big(2.17\big)$ as derived in Appendix A. The solutions are given by

$$Q\big(\mathcal{Z}\big) = \prod_{i=1}^{N} \left[\prod_{j=1}^{s} r_{ij}^{Z_{ij}} \prod_{j=s+1}^{M} r_{ij}^{*Z_{ij}} \right] \tag{2.18}$$

$$Q\big(\vec{\pi}^*\big) = \left(1 - \sum_{k=1}^{s} \pi_k\right)^{-M+s} \frac{\Gamma\big(\sum_{j=s+1}^{M} c_j^*\big)}{\prod_{j=s+1}^{M} \Gamma\big(c_j^*\big)} \prod_{j=s+1}^{M} \left(\frac{\pi_j^*}{1 - \sum_{k=1}^{s} \pi_k}\right)^{c_j^*-1} \tag{2.19}$$

$$Q\big(\vec{\alpha}\big) = \prod_{j=1}^{M} \prod_{l=1}^{D} \mathcal{G}\big(\alpha_{jl} \mid u_{jl}^*, \nu_{jl}^*\big) \tag{2.20}$$

provided,

$$r_{ij} = \frac{\tilde{r}_{ij}}{\sum_{j=1}^{s} \tilde{r}_{ij} + \sum_{j=s+1}^{M} \tilde{r}_{ij}^*}, r_{ij}^* = \frac{\tilde{r}_{ij}^*}{\sum_{j=1}^{s} \tilde{r}_{ij} + \sum_{j=s+1}^{M} \tilde{r}_{ij}^*} \tag{2.21}$$

$$\tilde{r}_{ij} = exp\Bigg\{ \ln \pi_j + \tilde{R}_j + \sum_{l=1}^{D} \big(\overline{\alpha}_{jl} - 1\big) \ln X_{il} - \sum_{l=1}^{D+1} \overline{\alpha}_{jl} \ln\Big(1 + \sum_{l=1}^{D} X_{il}\Big) \Bigg\} \tag{2.22}$$

$$\tilde{r}_{ij}^* = exp\Bigg\{ \big\langle \ln \pi_j^* \big\rangle + \tilde{R}_j + \sum_{l=1}^{D} \big(\overline{\alpha}_{jl} - 1\big) \ln X_{il} - \sum_{l=1}^{D+1} \overline{\alpha}_{jl} \ln\Big(1 + \sum_{l=1}^{D} X_{il}\Big) \Bigg\} \tag{2.23}$$

$$\begin{aligned}
\tilde{R}_j = \ln & \frac{\Gamma\big(\sum_{l=1}^{D+1} \overline{\alpha}_{jl}\big)}{\prod_{l=1}^{D+1} \Gamma\big(\overline{\alpha}_{jl}\big)} \\
& + \sum_{l=1}^{D+1} \overline{\alpha}_{jl} \left[\psi\left(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \right) - \psi\big(\overline{\alpha}_{jl}\big) \right] \Big[\big\langle \ln \alpha_{jl} \big\rangle - \ln \overline{\alpha}_{jl} \Big] \\
& + \frac{1}{2} \sum_{l=1}^{D+1} \overline{\alpha}_{jl}^2 \left[\psi'\left(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \right) - \psi'\big(\overline{\alpha}_{jl}\big) \right] \Big\langle \big(\ln \alpha_{jl} - \ln \overline{\alpha}_{jl} \big)^2 \Big\rangle \\
& + \frac{1}{2} \sum_{a=1}^{D+1} \sum_{b=1}^{D+1} \overline{\alpha}_{ja}\, \overline{\alpha}_{jb} \left[\psi'\left(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \right) \Big(\big\langle \ln \alpha_{ja} \big\rangle - \ln \overline{\alpha}_{ja} \Big) \right. \\
& \left. \times \Big(\big\langle \ln \alpha_{jb} \big\rangle - \ln \overline{\alpha}_{jb} \Big) \right]
\end{aligned} \tag{2.24}$$

$$c_j^* = \sum_{i=1}^{N} r_{ij}^* + c_j \tag{2.25}$$

$$u_{jl}^{*} = u_{jl} + \varphi_{jl}, \;\; \nu_{jl}^{*} = \nu_{jl} - \vartheta_{jl} \tag{2.26}$$

$$\varphi_{jl} = \sum_{i=1}^{N} \langle Z_{ij} \rangle \overline{\alpha}_{jl} \Bigg[\psi\Bigg(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \Bigg) - \psi\big(\overline{\alpha}_{jl}\big) + \sum_{s \neq l}^{D+1} \psi'\Bigg(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \Bigg) \times \overline{\alpha}_{js} \Big(\big\langle \ln \alpha_{js} \big\rangle - \ln \overline{\alpha}_{js} \Big) \Bigg] \tag{2.27}$$

$$\vartheta_{jl} = \sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle \Bigg[\ln X_{il} - \ln \Bigg(1 + \sum_{l=1}^{D+1} X_{il} \Bigg) \Bigg] \tag{2.28}$$

$\psi\big(.\big)$ and $\psi'\big(.\big)$ in the above equations represent the digamma and trigamma functions. The expectation of values mentioned in the equations above is given by,

$$\big\langle Z_{ij} \big\rangle = \begin{cases} r_{ij}, & \text{for } j = 1, ..., s \\ r_{ij}^{*}, & \text{for } j = s+1, ..., M \end{cases} \tag{2.29}$$

$$\overline{\alpha}_{jl} = \big\langle \alpha_{jl} \big\rangle = \frac{u_{jl}^{*}}{\nu_{jl}^{*}}, \quad \big\langle \ln \alpha_{jl} \big\rangle = \psi\big(u_{jl}^{*}\big) - \ln \nu_{jl}^{*} \tag{2.30}$$

$$\Big\langle \big(\ln \alpha_{jl} - \ln \overline{\alpha}_{jl} \big)^2 \Big\rangle = \Big[\psi\big(u_{jl}^{*}\big) - \ln u_{jl}^{*} \Big]^2 + \psi'\big(u_{jl}^{*}\big) \tag{2.31}$$

$$\big\langle \pi_{j}^{*} \big\rangle = \Bigg(1 - \sum_{k=1}^{s} \pi_{k} \Bigg) \frac{\sum_{i=1}^{N} r_{ij}^{*} + c_{j}}{\sum_{i=1}^{N} \sum_{k=s+1}^{M} r_{ik}^{*} + c_{k}} \tag{2.32}$$

$$\big\langle \ln \pi_{j}^{*} \big\rangle = \ln \Bigg(1 - \sum_{k=1}^{s} \pi_{k} \Bigg) + \psi\Bigg(\sum_{i=1}^{N} r_{ij}^{*} + c_{j} \Bigg) - \psi\Bigg(\sum_{i=1}^{N} \sum_{k=s+1}^{M} r_{ik}^{*} + c_{k} \Bigg) \tag{2.33}$$

Logically, we can see that the expectation of π and π^{*} are coupled with each other as they sum to 1. Hence, the parameters are to be iteratively estimated. Generally, the variational approach involves initializing the variational solutions $Q_k\big(\Theta_k\big)$ suitably, and then iteratively updating the values based on equation $\big(2.17\big)$ using the current values of rest of the variables. This way we maximize the lower bound $\mathcal{L}$ with respect to $Q\big(\mathcal{Z}\big), Q\big(\vec{\pi}^{*}\big), Q\big(\vec{\alpha}\big)$ in each iteration and in addition update the weights of fixed components with respect to

that of free components. To find an equation reach this goal we equate the differentiation of the lower bound with respect to π_j to zero which gives

$$\pi_j = \left(1 - \sum_{k=s+1}^{M} \left\langle \pi_k^* \right\rangle\right) \frac{\sum_{i=1}^{N} r_{ij}}{\sum_{i=1}^{N} \sum_{k=1}^{s} r_{ik}} \tag{2.34}$$

The updates are inculcated within the variational estimation algorithm which forms a good method of model selection for practical applications. During this process the mixing weights of some free components might move closer to zero. These components can be eliminated with an appropriate threshold.

2.2.2 Incremental algorithm using component splitting

We follow the method proposed in [20] for component splitting which is based on the split and merge algorithm described in [28]. The basic assumption of this algorithm is that the model has more than one component. So we check this assumption. Thus, we first run the variational estimation without component splitting. At convergence if there is only one component, then the process is stopped as there is no point in splitting the components. If the model is found to have two components then we initiate the component splitting algorithm. Here, we select one of the mixture components and split them into two different components. The components that have been split will be called the free components and the rest of the components are called fixed components as explained earlier. We then estimate the parameters based on the variational solutions derived in section 3.1. Doing this we might end up with three different scenarios. First, the free components might be a good fit for the data meaning the mixing weights of the components have considerable values. In this case split is a success and the components are retained. Second, one of the free components might end up with a considerable value at convergence and the other might be infinitesimal. Here the split is considered a failure and the components are restored to values they had before the split. Lastly, in rare cases, both the components might tend to become infinitesimal at convergence. This is due to the presence of outliers in the data. As this is an undesirable situation we check for this condition and ignore the split when it occurs. However, we can remove this component once the component splitting algorithm terminates as it is the resultant of outliers in the data. The basic idea is to apply this splitting test to every component sequentially until all the components fail the splitting test. Each time there is a successful split the number of components increases by one and the split

process is re-iterated for all the components. The proposed algorithm can be described as follows:

1. Initialization:
    - Set the number of components M to be 2
    - Initiate $\{u_{jl}\}, \{\nu_{jl}\}$ and c_j with all ones for u and c; and 0.1 for ν.
2. Apply the variational algorithm for inverted Dirichlet mixture models without component splitting for model selection as described in [29].
3. Terminate if only one component is estimated.
4. Let J be the set of M components in the mixture model.
5. Sort J in descending order with respect to the mixing weights.
6. For each element $j \, \epsilon \, J$ follow the split and merge algorithm as follows:
    - Split component j into j_1 and j_2
    - Set $\pi_{j_1} = \pi_{j2} = \pi_j/2, u_{jl_1} = u^*_{jl}, u_{jl_2} = u^*_{jl}, \nu_{jl_1} = \nu^*_{jl}$ and $\nu_{jl_2} = \nu^*_{jl}$
    - Consider $F = \{j_1, j_2\}$ the free components and F^* the other fixed components.
    - Set $c_j = \sum_{i=1}^{N} r^*_{ij}$ for $j \epsilon F^*$.
    - Iteratively update $Q\big(\mathcal{Z}\big), Q\big(\vec{\pi}^*\big), Q\big(\vec{\alpha}\big)$ using equations $\big(2.18\big), \big(2.19\big)$ and $\big(2.20\big)$ until convergence to achieve variational optimization of the model with local model selection.
    - Calculate mixing weights of free components according to equation $\big(2.34\big)$
    - If only one component is retained based on the split criteria $\big($say, π_j close to $0\big)$ then the split is deemed failure. Go to step 6 to test next component.
    - If both the components are eliminated, consider the split a failure. Go to step 6 to test next component.
    - If both components have a considerable mixing weight at convergence, set M = M+1.
7. Repeat steps 4-6 until all the components fail the splitting test.

According to [30] the variational bound is convex with respect to the other factors and hence convergence is guaranteed. Though we use first and second order Taylor expansion during the estimation of the lower bound the convexity of the approximated functions have been proved in [31]. All we have to do is to evaluate $\mathcal{L}\big(Q\big)$ at the end of each iteration and terminate the algorithm when the lower bound does not change much with respect to the lower bound in the previous iteration. The lower bound $\mathcal{L}\big(Q\big)$ in our model is given by $\big(2.15\big)$ as,

$$
\begin{aligned}
\mathcal{L}\big(Q\big) = & \Big\langle \ln p\big(\mathcal{X} \mid \mathcal{Z}, \vec{\alpha}\big) \Big\rangle + \Big\langle \ln p\big(\mathcal{Z} \mid \vec{\pi}, \vec{\pi}^*\big) \Big\rangle + \Big\langle \ln p\big(\vec{\pi}^* \mid \vec{\pi}\big) \Big\rangle \\
& + \Big\langle \ln p\big(\vec{\alpha}\big) \Big\rangle - \Big\langle \ln Q\big(\mathcal{Z}\big) \Big\rangle - \Big\langle \ln Q\big(\vec{\pi}^*\big) \Big\rangle - \Big\langle \ln Q\big(\vec{\alpha}\big) \Big\rangle \\
= & \sum_{i=1}^{N} \sum_{j=1}^{M} \big\langle Z_{ij} \big\rangle \left[\tilde{R}_j + \sum_{l=1}^{D} \Big(\overline{\alpha}_{jl} - 1 \Big) \ln X_{il} - \left(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \right) \ln \left(1 + \sum_{l=1}^{D} X_{il} \right) \right] \\
& + \sum_{i=1}^{N} \left[\sum_{j=1}^{s} r_{ij} \, \ln \pi_j + \sum_{j=s+1}^{M} r_{ij}^* \big\langle \ln \pi_{ij}^* \big\rangle \right] + \ln \Gamma \left(\sum_{j=s+1}^{M} c_j \right) \\
& - \sum_{j=s+1}^{M} \ln \Gamma\big(c_j\big) + \sum_{j=s+1}^{M} \big(c_j - 1\big) \left[\big\langle \ln \pi_j^* \big\rangle - \ln \left(1 - \sum_{k=1}^{s} \pi_k \right) \right] \\
& + \sum_{j=1}^{M} \sum_{l=1}^{D+1} \left[u_{jl} \, \ln \nu_{jl} - \ln \Gamma\big(u_{jl}\big) + \big(u_{jl} - 1\big) \big\langle \ln \alpha_{jl} \big\rangle - \nu_{jl} \, \overline{\alpha}_{jl} \right] \\
& - \sum_{i=1}^{N} \left[\sum_{j=1}^{s} r_{ij} \, \ln r_j + \sum_{j=s+1}^{M} r_{ij}^* \, \ln r_{ij}^* \right] - \ln \Gamma \left(\sum_{j=s+1}^{M} c_j^* \right) \\
& - \sum_{j=s+1}^{M} \ln \Gamma\big(c_j^*\big) + \sum_{j=s+1}^{M} \big(c_j^* - 1\big) \left[\big\langle \ln \pi_j^* \big\rangle - \ln \left(1 - \sum_{k=1}^{s} \pi_k \right) \right] \\
& - \sum_{j=1}^{M} \sum_{l=1}^{D+1} \left[u_{jl}^* \, \ln \nu_{jl}^* - \ln \Gamma\big(u_{jl}^*\big) + \big(u_{jl}^* - 1\big) \big\langle \ln \alpha_{jl} \big\rangle - \nu_{jl}^* \, \overline{\alpha}_{jl} \right]
\end{aligned}
\tag{2.35}
$$

2.3 Experimental results

In order to test the performance, we evaluate our model with synthetic data, a smart home dataset and then apply it for clustering of images and videos. Evaluating synthetic data helps us to see how accurately our model is able to estimate the parameters. For occupancy estimation in smart homes and the clustering of images and videos we compare

the accuracy of our variational inverted Dirichlet mixture model $\big($denoted from now on as varIDMM$\big)$ with Gaussian mixture model using maximum likelihood estimation $\big($denoted from now on as GMM$\big)$ and variational Gaussian mixture model $\big($denoted from now on as varGMM$\big)$. We then compare the efficiency of the model selection approach for all these experiments. In our experiments, we use the initial values of hyper parameters u and c as 1 and ν is initiated to be 0.1. These choices proved to give best results in our experiments.

2.3.1 Synthetic data

We first test the effectiveness of our model with three synthetic datasets having different sizes (300, 500 and 1000 respectively). All the datasets are generated to have three components, but the composition of the data from each component varies in proportion. This helps us to test the efficiency of our model with datasets of varying proportions. We take the number of dimensions to be three for ease of representation. The model was tested to be accurate for higher number of dimensions as well. The results of this experiment are shown in Table $\big(2.1\big)$. The estimated parameters in the table is an average of the estimates obtained by running the experiments 10 times. We can see that the parameters estimated by our varIDMM model are very close to the original ones and also the estimated number of components and their respective mixing weights were accurate as well. This proves the efficiency of our model with respect to parameters estimation.

In the next experiment we evaluate the performance of model selection with respect to different datasets. Table $\big(2.2\big)$ shows the properties of the different datasets X1, X2, X3, X4 and X5 generated with varying dimensions and number of components. Datasets X4 and X5 especially have the same dimensions and number of components but the number of data points in the datasets are less for each component. This would help us determine the robustness of our model with respect to small datasets. Table $\big(2.3\big)$ shows the estimated number of components of the datasets X1, X2, X3, X4 and X5 respectively. Similar to the previous case we take the average of estimates over ten different observations. It clearly shows that our model is able to find the number of components accurately in all the cases. The estimated parameters in all these cases were found to be accurate as well, but they are not tabulated to avoid prolixity.

Table 2.1: Real and estimated parameters of different datasets. N denotes the total number of data points, N_j denotes the number of data points in the cluster j. $\alpha_{j1}, \alpha_{j3}, \alpha_{j3}$ and π_j are the real parameters and $\hat{\alpha}_{j1}, \hat{\alpha}_{j3}, \hat{\alpha}_{j3}$ and $\hat{\pi}_j$ are the parameters estimated by our proposed model.

Data set	N_j	j	α_{j1}	α_{j2}	α_{j3}	π_j	$\hat{\alpha}_{j1}$	$\hat{\alpha}_{j2}$	$\hat{\alpha}_{j3}$	$\hat{\pi}_j$
S1	100	1	5	12	23	0.33	4.8	11.73	22.06	0.330
$\big(N = 300\big)$	100	2	10	25	13	0.34	10.47	26.28	13.58	0.341
	100	3	1	19	17	0.33	0.99	18.63	16.62	0.329
S2	150	1	5	12	23	0.30	5.09	13.47	22.35	0.294
$\big(N = 500\big)$	150	2	10	25	13	0.30	10.34	25.75	13.22	0.309
	200	3	1	19	17	0.40	1.00	18.64	16.64	0.397
S1	200	1	5	12	23	0.20	5.1	12.24	23.62	0.198
$\big(N = 1000\big)$	200	2	10	25	13	0.20	9.7	24.33	12.52	0.206
	600	3	1	19	17	0.60	1.02	19.51	17.58	0.596

2.3.2 Energy Saving in Smart Homes

Recent development in smart home technologies has attracted a lot of people to this industry. In addition to making life easy for residents with automated control, these advancements also help to reduce the energy consumption by limiting the electricity used for heating elements and other equipments. The primary motivation behind occupancy estimation in smart buildings is to increase energy performance, indeed, occupants who run a smart building are a crucial component of its intelligence. Thus, identifying the number of occupants in the housing unit is an important task [32, 33]. The authors in [34] worked on this problem based on an experimental setup in an of office at Grenoble Institute of Technology. The office was fitted with a network of connected sensors measuring the temperature, luminance, humidity, motion, CO2 concentrations, power consumption, door and window positions, acoustic pressure from microphones, etc. The setup also has two video cameras to find the original number of occupants. The data recorded from the sensors are transferred using EnOcean protocol (A standardized protocol for applications related to smart buildings and similar) to a centralized database and can be monitored by a web application. Our basic goal in this experiment is to estimate the number of occupants

Table 2.2: Properties of the synthetic datasets X1, X2, X3, X4, X5. N denotes the number of data points, D represents the dimension of the dataset and M is the number of components.

Data set	N	D	M
X1	400	5	4
X2	800	15	8
X3	1000	10	10
X4	750	25	15
X5	1500	25	15

Table 2.3: Estimated number of components $\hat{M}$ for the datasets X1, X2, X3, X4 and X5

Data set	X1	X2	X3	X4	X5
$\hat{M}$	4.2	7.6	10	14.5	14.6

currently in the building. So our model should be able to cluster the number of occupants. This helps us limit the power spent on unnecessary appliances when there are no occupants in the housing unit. Basing on the experimental results of [34] we take the most important features having a high correlation with the number of occupants in the building. These features are:

Motion data: A PIR sensor generating binary data with 1 when motion is detected and 0 if not; for each time instance.

Acoustic pressure: This sensor records data based on the RMS (Root Mean Square) value of the signals generated by a microphone.

Power consumption: This feature represents the power consumption of four laptops that are regularly used in the office.

Door position: Again a binary value of 0 or 1 for each time instance indicating the door is closed or open respectively.

The two video cameras provide the ground truth of the occupants originally present (no occupant, 1 occupant, two occupants, three or more occupants). Obviously, these labels

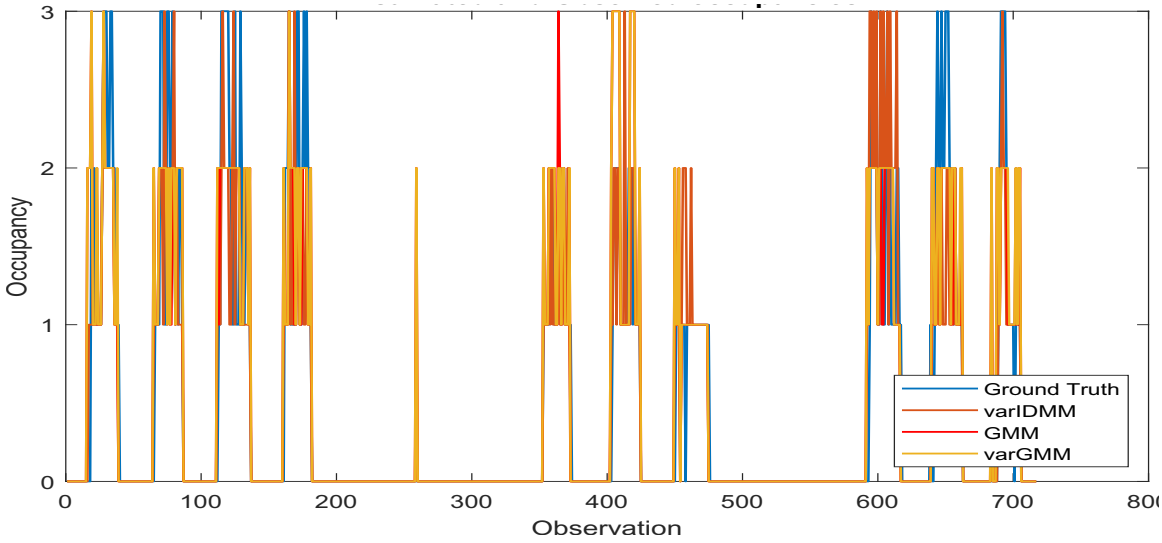


Figure 2.2: Comparison between actual occupancy and estimated occupancies using different approaches.

were removed when given as input to our model as we follow an unsupervised approach. It is used for validation purpose, and defining the meaning of each cluster. The original data set was in fact labeled, but in our experiments, we have removed completely the labels to make the learning problem completely unsupervised.

The comparison of estimation between varIDMM, GMM and varGMM is shown in Fig. 2.2 along with the ground truth, where both actual occupancy and the estimated occupancy are plotted as a graph of number of occupants with quantum time is equal to 30 minutes. Since the main objective of our application is to reduce energy consumption it is important to have a minimal mean absolute error (MAE):

$$MAE = \frac{\sum_{i=1}^{N} \mid (observed\ occupancy)_i - (estimated\ occupancy)_i \mid}{N} \tag{2.36}$$

The observed occupancy is the ground truth we obtain from the video cameras. This measure is important because the closer we estimate with respect to the original number of occupants, the energy consumption can be regulated accordingly with more accuracy. The estimated MAE for each of the models is shown in Table 2.4. Our model has a lower error rate compared to GMM and varGMM models. In addition to these models we found that one of the K-Means algorithms variants proved to be efficient recently in [35] performs worse for our case. This shows the supremacy of our model compared to the rest of approaches.

Table 2.4: MAE of different models for smart home data

Model	MAE(%)
varIDMM	18.13
GMM	19.20
varGMM	20.08
K-Means	29.71

2.3.3 Image Clustering

Pattern recognition in images has been an important application of machine learning techniques in recent years. Identifying patterns in images using unsupervised learning plays a major role in extraction of features or topics that help greatly in image classification, image retrieval and many other applications [36–39]. In our case we use our model to learn from a set of images. Our main objective here is, given a set of images our model should be able to properly cluster them. Generally, when categorizing images the first step is to extract the feature descriptors from the images. This can be done by various methods like SIFT [40], SURF [41], HOG [42], etc. Then we construct a bag of visual words from these descriptors. Bag of visual words approach has been an important tool for image and video categorization in a wide variety of applications [43–45]. We evaluate the efficiency of our model for image categorization with two different datasets with varying properties as follows:

Experiment 1

The data we use for this experiment was derived from three different datasets. We selected equal number of samples from three varied classes to test the performance of our model for balanced datasets. We choose 333 images from a flowers dataset taken from kaggle[1]. Another 333 images were chosen from the Zurich buildings dataset [46] and another 333 images were chosen from the Food-5K dataset[2]. The Food-5K dataset consists of two categories, 2500 images of food and 2500 images which are not food. We chose images of people from the non food images as they had a lot of variations as compared to the other

[1] `https://www.kaggle.com/alxmamaev/flowers-recognition`.
[2] `https://mmspg.epfl.ch/food-image-datasets`

Figure 2.3: Sample images from Flowers dataset

Figure 2.4: Sample images from Zurich Buildings dataset

class. Some sample images from these datasets are shown in figures 2.3,2.4 and 2.5.

In our case we first detect the SIFT descriptors in each image using Difference of Gaussian (DoG) interest point detectors as in [40]. Considering the descriptors extracted from all the images, we use K-means to cluster these feature vectors. We can then represent the images as frequency histograms of visual words. Finally, we use our varIDMM model on this data for clustering. We also compare our output with varGMM and GMM to show the efficiency of our algorithm in clustering the data according to detected topics. Figure 2.6 shows the confusion matrix obtained by using our varIDMM model on our image dataset. The accuracy of our model was found to be 77.98% which is better as compared to the GMM and varGMM as shown in Table 2.5. We also observed that the other models have

Figure 2.5: Sample images from Food-5K dataset

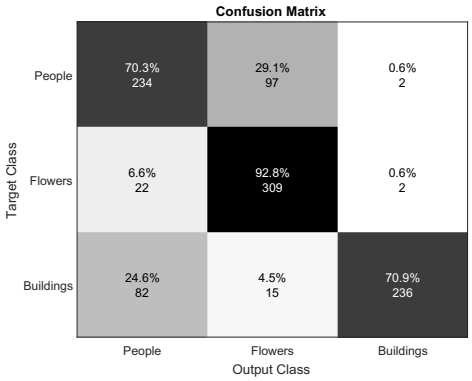


Figure 2.6: Confusion Matrix for the image dataset

Table 2.5: Accuracy of different models for the image dataset

Method	Accuracy(%)
varIDMM	77.98
GMM	66.87
varGMM	66.66

not performed well in detecting images of people whereas our model was able to achieve more than 70% accuracy for all the categories.

Experiment 2

In our second experiment, we choose more image categories and use imbalanced number of images from each of the categories to test the robustness of our model. We use the 15 scenes dataset [47] for this experiment. The dataset has 15 categories ranging from mountains, forests, etc. to homes, skyscrapers, etc. We choose 6 categories from this set namely: "Living Room", "Sea", "Forest", "Building", "Snowy Mountain", "Stores". We choose 200 images of living room, 300 images of the sea, 230 images of forests, 200 images of buildings, 250 images of snowy mountains and 150 images of stores contributing 1330 images on the whole. Some sample images from the dataset are shown in Fig. 2.7. Same as our previous experiment we extract the SIFT features for each of the images in our database and create a bag of visual words representation. The confusion matrix obtained by using our model is shown in Fig. 2.8. Analyzing the confusion matrix we see that, the living room and snowy mountains are hard for our model to distinguish. The same effect was

Figure 2.7: Sample images from different categories of 15 Scenes dataset

Table 2.6: Accuracy of different models for 15 scenes dataset

Method	Accuracy(%)
varIDMM	76.24
GMM	57.81
varGMM	52.93

observed in GMM and varGMM models as well. However, these two models were able to distinguish less than 10% of the images from the two classes accurately which lowered the accuracy of the models greatly. Table 2.6 shows the accuracy of different models on this particular dataset.

2.3.4 Video Clustering

Similar to images, videos clustering is of prime importance as well. It has a variety of applications in event analysis, video retrieval, video indexing [48], etc. The objective of our experiment is to test how well our model is able to represent topics in videos. For this, we evaluate the efficiency of our model on the well renowned KTH dataset [49]. The KTH dataset consists of 599 video files, each from one of the six different categories: boxing, hand clapping, hand waving, jogging, running and walking. Each category has 100 video

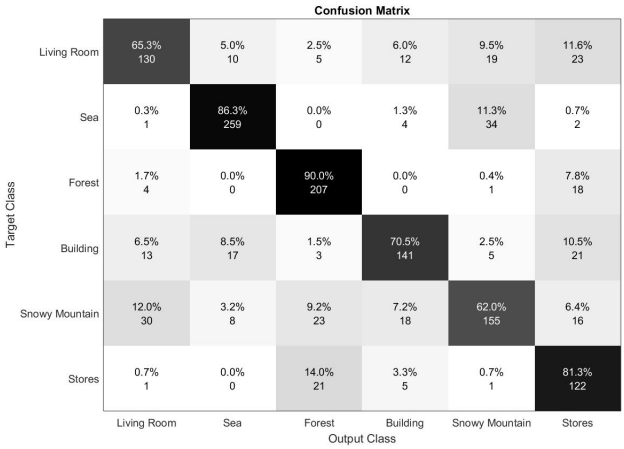


Figure 2.8: Confusion Matrix for the 15 Scenes dataset

Figure 2.9: Sample images from KTH dataset

samples except the hand clap category which has only 99 video samples. The videos were recorded by a still camera for 25 test subjects performing the respective task. The videos were captured with a frame rate of 25fps in 4 different scenarios for each of the test subject: outdoors (S1), outdoors with scale variation (S2), outdoors with different clothing (S3) and indoors (S4). Few sample frames from the dataset are shown in figure 2.9.

We follow a similar approach as in the case of images to form bag of visual words. To do this we use Lukas-Kanade (LK) optical flow [50] a method which tracks the variations in pixel intensities in the x and y directions along with their orientation and magnitude. Optical flow has proved to be an efficient tool for video analysis in many applications [51–53]. For our application we also tried using 3D-SIFT [54], however, the output for

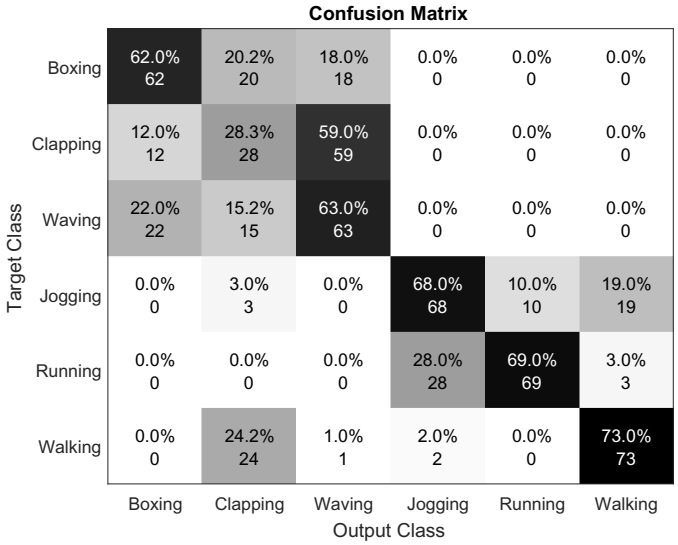


Figure 2.10: Confusion Matrix for KTH dataset

Table 2.7: Accuracy of different models for KTH dataset

Method	Accuracy(%)
varIDMM	60.66
GMM	48.08
varGMM	48.24

our model was not as good compared to optical flow. We also use derivative of Gaussian before extracting the optical flow attributes. We use the data which represents the change in magnitude obtained by LK optical flow estimation for 100 frames in each video to build our bag of words. Figure 2.10 shows the confusion matrix obtained using our model for topic learning. We see that except the hand clapping class, all other classes are clustered with an accuracy greater than 60%. However in the case of GMM and varGMM, the accuracy for all the clusters where less than 50% which shows that our model is able to provide a better approximation. Table 2.7 shows the comparison of our model with GMM and varGMM. We see that our model has a higher overall accuracy than the other two models.

Chapter 3

Finite Generalized Inverted Dirichlet Mixture Model with Variational Component Splitting and Variational Feature Selection

In this chapter, we detail our findings when component splitting appraoch is applied to generalized inverted Dirichlet mixture models. We also expand our study to determine the efficiency of the component splitting algorithm when applied along side a variational feature selection framework. This helps us estimate the complexity of the data efficiently concomitantly eliminating the irrelevant features. We evaluate the performance of our models with applications such as image categorization and dynamic texture analysis.

3.1 Model specification

Let $\mathcal{Y} = \left(\vec{Y}_1, \vec{Y}_2, ..., \vec{Y}_N\right)$ be a set of N independent and identically distributed vectors where each vector $\vec{Y}_i$ is given by $\vec{Y}_i = \left(Y_{i1}, Y_{i2}, ..., Y_{iD}\right)$. Here D is dimension of the vector. Considering that the underlying mixture within $\mathcal{Y}$ is GID, we can define the probability density function $p\left(\vec{Y}_i \mid \vec{\alpha}_j, \vec{\beta}_j\right)$ with respect to the j_{th} component as,

$$p\left(\vec{Y}_i \mid \vec{\alpha}_j, \vec{\beta}_j\right) = \prod_{d=1}^{D} \frac{\Gamma\left(\alpha_{jd} + \beta_{jd}\right)}{\Gamma\left(\alpha_{jd}\right)\Gamma\left(\beta_{jd}\right)} \frac{Y_{id}^{\alpha_{jd}-1}}{\left(1 + \sum_{l=1}^{d} Y_{il}\right)^{\gamma_{id}}} \tag{3.1}$$

where the parameters of GID is defined by $\vec{\alpha}_j = \big(\alpha_{j1}, \alpha_{j2}, ..., \alpha_{jd}\big)$ and $\vec{\beta}_j = \big(\beta_{j1}, \beta_{j2}, ..., \beta_{jd}\big)$ such that, $\alpha_{jd} > 0$ and $\beta_{jd} > 0$. γ_{jd} is defined as $\gamma_{jd} = \beta_{jd} + \alpha_{jd} - \beta_{j(d+1)}$. Assuming the model now consists of M different components [27] we can write the GID mixture model as,

$$p\big(\vec{Y}_i \mid \vec{\pi}, \vec{\alpha}, \vec{\beta}\big) = \sum_{j=1}^{M} \pi_j p\big(\vec{Y}_i \mid \vec{\alpha}_j, \vec{\beta}_j\big) \tag{3.2}$$

where the parameters of the GID distribution pertaining to each component j is represented by $\vec{\alpha} = \big(\vec{\alpha}_1, \vec{\alpha}_2, ..., \vec{\alpha}_M\big)$ and $\vec{\beta} = \big(\vec{\beta}_1, \vec{\beta}_2, ..., \vec{\beta}_M\big)$. Similarly, $\vec{\pi} = \big(\pi_1, \pi_2, ..., \pi_M\big)$ represents the mixing coefficient of the corresponding components, such that, $\sum_{j=1}^{M} \pi_j = 1$. We can write GID as a product of inverted Beta distribution as it does not change the underlying model. This is proved in [6]. Hence equation (3.2) becomes,

$$p\big(\mathcal{X} \mid \pi, \alpha, \beta\big) = \prod_{i=1}^{N} \left(\sum_{j=1}^{M} \pi_j \prod_{l=1}^{D} p_{iBeta}\big(X_{il} \mid \alpha_{jl}, \beta_{jl}\big) \right) \tag{3.3}$$

given $\mathcal{X} = \big(X_1, X_2, ..., X_N\big)$ where $\vec{X}_i = \big(X_{i1}, X_{i2}, ..., X_{iD}\big), X_{i1} = Y_{i1}$ and $X_{il} = \frac{Y_{il}}{1+\sum_{k=1}^{l-1} Y_{ik}}$ for $l > 1$. $p_{iBeta}\big(X_{il} \mid \alpha_{jl}, \beta_{jl}\big)$ in the above equation represents the inverted Beta distribution defined by parameters α_{jl} and β_{jl} and is given by,

$$p_{iBeta}\big(X_{il} \mid \alpha_{jl}, \beta_{jl}\big) = \frac{\Gamma\big(\alpha_{jl} + \beta_{jl}\big)}{\Gamma\big(\alpha_{jl}\big)\Gamma\big(\beta_{jl}\big)} \frac{X_{il}^{\alpha_{jl}-1}}{\big(1 + X_{il}\big)^{\alpha_{jl}+\beta_{jl}}} \tag{3.4}$$

According to this design, estimating the parameters of the model represented by equation (3.3) is the same as estimating the parameters of the model given by equation (3.2). $\mathcal{Z} = \big(Z_1, Z_2, ...Z_N\big)$ represents the latent variable, where $Z_i = \big(Z_{i1}, Z_{i2}, ..., Z_{iM}\big)$ complying to the conditions, $Z_{ij} \epsilon \{0, 1\}$ and $\sum_{j=1}^{M} Z_{ij} = 1$.

To integrate the component splitting algorithm we use equations (2.7) and (2.9). Since GID is a distribution from the exponential family, we choose the prior of the parameters to be Gamma distribution assuming the parameters are independent as well. Thus the prior distribution of α_{jl} and β_{jl} is written as,

$$p\big(\alpha_{jl}\big) = \mathcal{G}\big(\alpha_{jl} \mid u_{jl}, \nu_{jl}\big) = \frac{\nu_{jl}^{u_{jl}}}{\Gamma\big(u_{jl}\big)} \alpha_{jl}^{u_{jl}-1} e^{-\nu_{jl}\alpha_{jl}} \tag{3.5}$$

$$p\big(\beta_{jl}\big) = \mathcal{G}\big(\beta_{jl} \mid g_{jl}, h_{jl}\big) = \frac{h_{jl}^{g_{jl}}}{\Gamma\big(g_{jl}\big)} \beta_{jl}^{g_{jl}-1} e^{-h_{jl}\beta_{jl}} \tag{3.6}$$

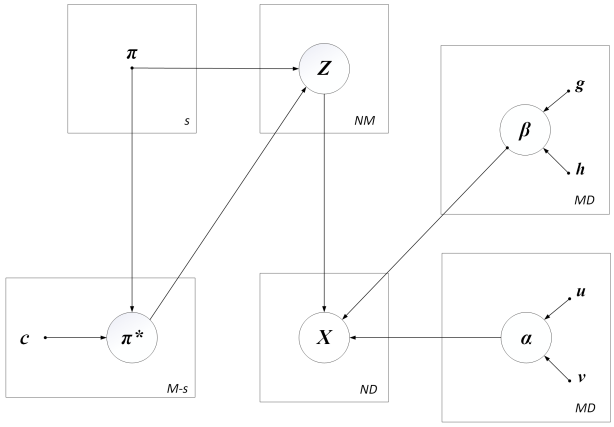


Figure 3.1: Graphical representation of GID mixture model with component splitting. The circles represent the random variables and model parameters, and plates denote repetitions. Number in the lower right corners of the plates indicate the number of repetitions. The conditional dependencies of the variables are represented by the arcs.

where $\mathcal{G}(.)$ indicates a Gamma distribution. Using all these information available to us, we can write the joint distribution of all the random variables in our model as,

$$\begin{aligned}
p(\mathcal{X},\Theta\,|\,\vec{\pi}) =& p(\mathcal{X}\mid \mathcal{Z},\vec{\alpha},\vec{\beta})p(Z\mid \vec{\pi},\vec{\pi}^*)p(\vec{\pi}^*\mid\vec{\pi})p(\vec{\alpha})p(\vec{\beta}) \\
=& \prod_{i=1}^{N}\prod_{j=1}^{M}\left[\frac{\Gamma(\alpha_{jl}+\beta_{jl})}{\Gamma(\alpha_{jl})\Gamma(\beta_{jl})}\frac{X_{il}^{\alpha_{jl}-1}}{(1+X_{il})^{\alpha_{jl}+\beta_{jl}}}\right]^{Z_{ij}} \\
&\times\prod_{i=1}^{N}\left[\prod_{j=1}^{s}\pi_j^{Z_{ij}}\prod_{j=s+1}^{M}\pi_j^{*Z_{ij}}\right]\times\left(1-\sum_{k=1}^{s}\pi_k\right)^{-M+s} \\
&\times\frac{\Gamma(\sum_{j=s+1}^{M}c_j)}{\prod_{j=s+1}^{M}\Gamma(c_j)}\prod_{j=s+1}^{M}\left(\frac{\pi_j^*}{1-\sum_{k=1}^{s}\pi_k}\right)^{c_j-1} \\
&\times\frac{\nu_{jl}^{u_{jl}}}{\Gamma(u_{jl})}\alpha_{jl}^{u_{jl}-1}e^{-\nu_{jl}\alpha_{jl}}\times\frac{h_{jl}^{g_{jl}}}{\Gamma(g_{jl})}\beta_{jl}^{g_{jl}-1}e^{-h_{jl}\beta_{jl}}
\end{aligned}$$

where $\Theta=\{\mathcal{Z},\vec{\alpha},\vec{\beta},\vec{\pi}^*\}$ is the set of unknown parameters. The graphical representation of our model is shown in Fig. 3.1. Following the variational approach we can write the variational variational solutions for our model as,

$$Q(\mathcal{Z}) = \prod_{i=1}^{N}\left[\prod_{j=1}^{s}r_{ij}^{Z_{ij}}\prod_{j=s+1}^{M}r_{ij}^{*Z_{ij}}\right] \tag{3.7}$$

$$Q(\vec{\pi}^*)=(1-\sum_{k=1}^{s}\pi_k)^{-M+s}\frac{\Gamma(\sum_{j=s+1}^{M}c_j^*)}{\prod_{j=s+1}^{M}\Gamma(c_j^*)}\prod_{j=s+1}^{M}(\frac{\pi_j^*}{1-\sum_{k=1}^{s}\pi_k})^{c_j^*-1} \tag{3.8}$$

$$Q(\vec{\alpha}) = \prod_{j=1}^{M}\prod_{l=1}^{D}\mathcal{G}(\alpha_{jl}\,|\,u^*_{jl},\nu^*_{jl}), Q(\vec{\beta}) = \prod_{j=1}^{M}\prod_{l=1}^{D}\mathcal{G}(\beta_{jl}\,|\,g^*_{jl},h^*_{jl}) \tag{3.9}$$

given,

$$r_{ij} = \frac{\tilde{r}_{ij}}{\sum_{j=1}^{s}\tilde{r}_{ij}+\sum_{j=s+1}^{M}\tilde{r}^*_{ij}}, r^*_{ij} = \frac{\tilde{r}^*_{ij}}{\sum_{j=1}^{s}\tilde{r}_{ij}+\sum_{j=s+1}^{M}\tilde{r}^*_{ij}} \tag{3.10}$$

$$\ln\tilde{r}_{ij} = \ln\pi_j + \sum_{l=1}^{D}\tilde{\mathcal{R}}_{jl} + \left(\overline{\alpha}_{jl}-1\right)\ln X_{il} - \left(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\right)\ln\left(1+X_{il}\right) \tag{3.11}$$

$$\ln\tilde{r}^*_{ij} = \left\langle\ln\pi^*_j\right\rangle + \sum_{l=1}^{D}\tilde{\mathcal{R}}_{jl} + \left(\overline{\alpha}_{jl}-1\right)\ln X_{il} - \left(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\right)\ln\left(1+X_{il}\right) \tag{3.12}$$

$$\begin{aligned}\tilde{\mathcal{R}} = &\ln\frac{\Gamma(\overline{\alpha}+\overline{\beta})}{\Gamma(\overline{\alpha})\Gamma(\overline{\beta})} + \overline{\alpha}\left[\psi(\overline{\alpha}+\overline{\beta})-\psi(\overline{\alpha})\right]\left(\left\langle\ln\alpha\right\rangle-\ln\overline{\alpha}\right)\\ &+\overline{\beta}\left[\psi(\overline{\alpha}+\overline{\beta})-\psi(\overline{\beta})\right]\left(\left\langle\ln\beta\right\rangle-\ln\overline{\beta}\right)\\ &+0.5\overline{\alpha}^2\left[\psi'(\overline{\alpha}+\overline{\beta})-\psi'(\overline{\alpha})\right]\left\langle(\ln\alpha-\ln\overline{\alpha})^2\right\rangle\\ &+0.5\overline{\beta}^2\left[\psi'(\overline{\alpha}+\overline{\beta})-\psi'(\overline{\beta})\right]\left\langle(\ln\beta-\ln\overline{\beta})^2\right\rangle\\ &+\overline{\alpha}\overline{\beta}\psi'(\overline{\alpha}+\overline{\beta})\left(\left\langle\ln\alpha\right\rangle-\ln\overline{\alpha}\right)\left(\left\langle\ln\beta\right\rangle-\ln\overline{\beta}\right)\end{aligned} \tag{3.13}$$

$$\begin{aligned}u^*_{jl} = &u_{jl} + \sum_{i=1}^{N}\left\langle Z_{ij}\right\rangle\overline{\alpha}_{jl}\Bigg[\psi\left(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\right)-\psi\left(\overline{\alpha}_{jl}\right)\\ &+\overline{\beta}_{jl}\psi'\left(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\right)\left(\left\langle\ln\beta_{jl}\right\rangle-\ln\overline{\beta}_{jl}\right)\Bigg]\end{aligned} \tag{3.14}$$

$$\nu^*_{jl} = \nu_{jl} - \sum_{i=1}^{N}\left\langle Z_{ij}\right\rangle\ln\frac{X_{il}}{1+X_{il}} \tag{3.15}$$

$$\begin{aligned}g^*_{jl} = &g_{jl} + \sum_{i=1}^{N}\left\langle Z_{ij}\right\rangle\overline{\beta}_{jl}\Bigg[\psi\left(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\right)-\psi\left(\overline{\beta}_{jl}\right)\\ &+\overline{\alpha}_{jl}\psi'\left(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\right)\left(\left\langle\ln\alpha_{jl}\right\rangle-\ln\overline{\alpha}_{jl}\right)\Bigg]\end{aligned} \tag{3.16}$$

$$h^*_{jl} = h_{jl} - \sum_{i=1}^{N}\left\langle Z_{ij}\right\rangle\ln\frac{1}{1+X_{il}} \tag{3.17}$$

in the above equations $\psi(.)$ and $\psi'(.)$ denote the digamma and trigamma functions, respectively. Since $\mathcal{R} = \left\langle \ln \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \right\rangle$ is intractable, we estimate the lower bound by the second order Taylor's expansion as in [6] to get $\tilde{\mathcal{R}}$ in equation (3.13). The expectation of values mentioned in the equations is given by,

$$\left\langle Z_{ij} \right\rangle = \begin{cases} r_{ij}, & \text{for } j = 1, ..., s \\ r^*_{ij}, & \text{for } j = s+1, ..., M \end{cases} \tag{3.18}$$

$$\overline{\alpha}_{jl} = \left\langle \alpha_{jl} \right\rangle = \frac{u^*_{jl}}{\nu^*_{jl}}, \quad \left\langle \ln \alpha_{jl} \right\rangle = \psi\big(u^*_{jl}\big) - \ln \nu^*_{jl} \tag{3.19}$$

$$\overline{\beta}_{jl} = \left\langle \beta_{jl} \right\rangle = \frac{g^*_{jl}}{h^*_{jl}}, \quad \left\langle \ln \beta_{jl} \right\rangle = \psi\big(g^*_{jl}\big) - \ln h^*_{jl} \tag{3.20}$$

$$\left\langle \pi^*_j \right\rangle = \left(1 - \sum_{k=1}^{s} \pi_k\right) \frac{\sum_{i=1}^{N} r^*_{ij} + c_j}{\sum_{i=1}^{N} \sum_{k=s+1}^{M} r^*_{ik} + c_k} \tag{3.21}$$

$$\left\langle \ln \pi^*_j \right\rangle = \ln\big(1 - \sum_{k=1}^{s} \pi_k\big) + \psi\big(\sum_{i=1}^{N} r^*_{ij} + c_j\big) - \psi\big(\sum_{i=1}^{N} \sum_{k=s+1}^{M} r^*_{ik} + c_k\big) \tag{3.22}$$

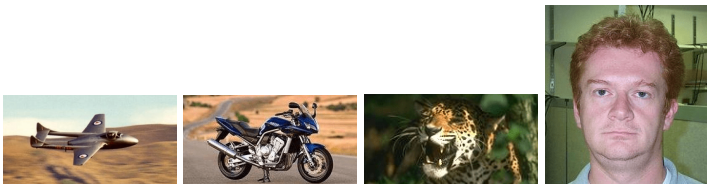
During the variational optimization process, these parameters are updated iteratively until convergence. The component splitting approach as used in [55] is built within this variational framework.

3.2 Experimental Results

We evaluate the built variational GID mixture model (varGIDMM) using two different image datasets focused on image categorization. We compare the effectiveness of the model with mixture models based on variational Inverted Dirichlet (varIDMM), Dirichlet (varDMM) and Gaussian distribution(varGMM). We also compare the results with Gaussian mixture models (GMM) with log likelihood estimation. The hyperparameters u, g and c are initiated to be 1 and ν and h are initiated to be 0.05.

Image Categorization

Image categorization plays an important role in industrial automation and has a wide range of multimedia applications as well [56–58]. Identifying the pattern in these images are



Airplane Motorbikes Leopards Faces

Figure 3.2: Sample images from different categories of Caltech 101 dataset

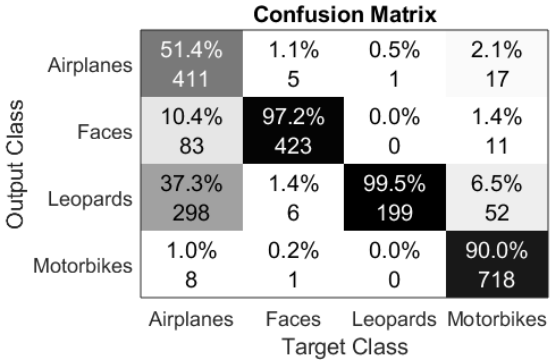


Figure 3.3: Confusion matrix of Caltech 101 dataset with varGIDMM

vital for any image related application. Our application involves two challenging datasets; Caltech 101 objects [59] and VisTex dataset from MIT Media Lab [1].

The first dataset we used to evaluate our model is the caltech 101 dataset. Among the 101 categories we chose four categories: airplanes, faces, leopards and Motorbikes. Among these categories, airplanes have 800 images, faces have 435 images, leopards have 200 images and motorbikes have 798 images. Sample images from these categories are shown in Fig. 3.2. We chose these four categories as they help to test the performance of our model with imbalanced data set. Also, the airplanes class in our model has a similar landscape when compared to many images in the motorbikes and leopards class. We made this selection to evaluate the robustness of our model when it comes to similar looking classes. In order to use our model on the images we have to first create a bag of visual words model [60, 61]. In order to create a bag of visual words model we have to first extract some kind of descriptors from the images. The frequently used descriptors are SIFT [40], SURF [41], HOG [42], etc. In our case we found the SIFT descriptors to be an efficient choice. Hence, we first extract the SIFT features from the images and then perform K-means clustering over the extracted SIFT descriptors to form the bag of words feature

[1] http://vismod.media.mit.edu/vismod/imagery/VisionTexture/

Table 3.1: Accuracy of different models for Caltech 101 dataset

Method	Accuracy(%)
varGIDMM	78.41
varIDMM	68.29
varDMM	68.6
varGMM	70.8
GMM	72.55

Fabric Flowers Food Grass Paintings Tiles Water

Figure 3.4: Sample images from different categories of VisTex dataset

vector for each image. This is used as input to our model for clustering. The confusion matrix constructed out of the output from our model is shown in Fig. 3.3. The airplanes class had less accuracy when compared to the other classes because the images looked similar. The other models were actually worse in distinguishing between the classes. Table 3.1 shows the performance our model compared to the other models. We can see that varGIDMM performed better than the rest of the models.

In order to make things more demanding, we choose another challenging dataset to evaluate our model, which is VisTex. The VisTex data set consisted of a number of classes where the images looked similar. Hence distinguishing them would be an interesting evaluation of our model. For this experiment we split each of the 512×512 images into 64×64 $child\ images$. In this case each 512×512 $mother\ image$ contributes to 64 child images in the database. So, all the 64 images should be classified into a single class. In our experiment we use images from seven different groups namely, “fabric”, “flowers”, “food”, “grass”, “paintings”, “tiles” and “water”. The database consisted of 1344 child images, with 4 mother images each from flowers, food, and water contributing 768 images, 3 mother images from fabric class contributing 192 images and 2 mother images each from grass, paintings and tiles contributing 384 images. Fig. 3.4 shows some sample images from the different categories chosen for the experiment. When it comes to textures we intend to obtain features that would represent prevalent pattern in an image rather than the spatial details. Hence, we use cooccurance matrix [62] which uses the joint probability

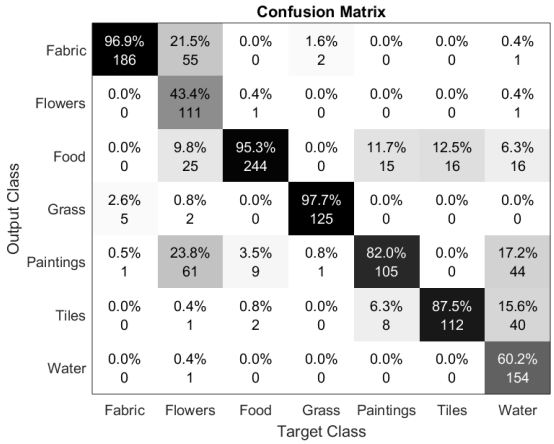


Figure 3.5: Confusion matrix of VisTex dataset with varGIDMM

Table 3.2: Accuracy of different models for VisTex dataset

Method	Accuracy(%)
varGIDMM	77.16
varIDMM	75.37
varDMM	63.69
varGMM	68.37
GMM	69.86

functions of two picture elements in an image at some given relative position [7]. We use the cooccurance matrices obtained for each image as the features for our model. The confusion matrix obtained by using our model is shown in Fig. 3.5. From the confusion matrix we see that the flowers class in our model exhibited lower accuracy due to the similarity with some of the images in fabric and paintings classes. The performance with other models were worse with this pair as well. However, in addition, the other models were poor in distinguishing between paintings and water which decreased their accuracy. The accuracy compared with other models is shown in Table. 3.2. These applications stand proof for the efficiency of our model. During our experiments we also noticed that some of the small categories are indistinguishable when using Gaussian and Dirichlet mixture models which elevates our model in terms of imbalanced datasets.

3.3 The Mathematical Model with Feature Selection

Feature selection is an essential process in a mixture model as some features in the data do not necessarily have importance in clustering. The performance of the model is better when these features are removed. It is to be noted that the configurations of GID model are followed as above. In this work we use the approach proposed in [63] where we approximate the irrelevant features by considering a distribution over it. Hence, the features follow the following distribution:

$$p(X_{il} \mid \phi_{il}, \alpha_{il}, \beta_{il}, \lambda_l, \tau_l) \simeq \big(iBeta(X_{il} \mid \alpha_{jl}, \beta_{jl})\big)^{\phi_{il}} \big(iBeta(X_{il} \mid \lambda_l, \tau_l)\big)^{1-\phi_{il}} \quad (3.23)$$

Here, $\phi_{il} = 0$ if feature l is irrelevant for j^{th} and 1 if relevant. In our case we consider the irrelevant features to follow an inverted beta distribution $iBeta(X_{il} \mid \lambda_l, \tau_l)$. Since ϕ_{il} is a binary latent variable we can write the prior distribution of $\vec{\phi}$ as:

$$p\big(\vec{\phi} \mid \vec{\epsilon}\big) = \prod_{i=1}^{N} \prod_{l=1}^{D} \epsilon_{l_1}^{\phi_{il}} \epsilon_{l_2}^{1-\phi_{il}} \quad (3.24)$$

where, $\epsilon_{l_1} = p(\phi_{il} = 1)$ and $\epsilon_{l_2} = p(\phi_{il} = 0)$ since ϕ_{il} is a Bernoulli variable. $\vec{\epsilon} = (\vec{\epsilon}_1, \vec{\epsilon}_2, ...\vec{\epsilon}_D)$ represent the probabilities that the features are relevant or not (i.e. feature saliencies), where $\vec{\epsilon}_l = (\epsilon_{l_1}, \epsilon_{l_2})$ and $\epsilon_{l_1} + \epsilon_{l_2} = 1$. In this model, the irrelevant features are modeled globally and model selection is done locally. As a final step, we choose a prior distribution to model the parameters $\vec{\lambda}$ and $\vec{\tau}$. Gamma distribution is a perfect choice as GID is also from an exponential family. Hence, assuming the parameters are independent we define the priors for the parameters as, $p\big(\vec{\lambda}\big) = \mathcal{G}\big(\vec{\lambda} \mid \vec{g}, \vec{h}\big)$ and $p\big(\vec{\tau}\big) = \mathcal{G}\big(\vec{\tau} \mid \vec{s}, \vec{t}\big)$, where $\mathcal{G}\big(\vec{x} \mid \vec{a}, \vec{b}\big) = \frac{b^a}{\Gamma\big(a\big)} x^{a-1} e^{-bx}$. All the hyperparameter vectors $\vec{g}, \vec{h}, \vec{s}$ and $\vec{t}$ are positive in the above equations. Summarizing all the unknown variables, we introduce

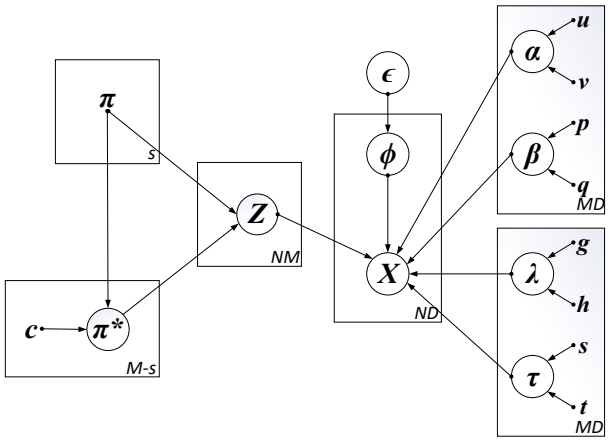


Figure 3.6: Graphical representation of finite GID mixture model with feature selection and component splitting. The circles denote the random variables and the conditional dependencies between the variables are indicated by the arcs. The number in the bottom right corner of the platesindicates the dimension of the variables inside

$\Theta = \{\mathcal{Z}, \vec{\alpha}, \vec{\beta}, \vec{\lambda}, \vec{\tau}, \vec{\phi}, \vec{\pi}^*\}$. Now, the joint distribution is given by:

$$
\begin{aligned}
p\big(\mathcal{X}, \Theta \mid \vec{\pi}, \vec{\epsilon}\big) =& p\big(\mathcal{X} \mid \mathcal{Z}, \vec{\alpha}, \vec{\beta}, \vec{\lambda}, \vec{\tau}, \vec{\phi}\big) p\big(\vec{\phi} \mid \vec{\epsilon}\big) p\big(Z \mid \vec{\pi}, \vec{\pi}^*\big) \\
& \times p\big(\vec{\pi}^* \mid \vec{\pi}\big) p\big(\vec{\alpha}\big) p\big(\vec{\beta}\big) p\big(\vec{\lambda}\big) p\big(\vec{\tau}\big) \\
=& \prod_{i=1}^{N} \prod_{j=1}^{M} \Bigg\{ \prod_{l=1}^{D} \left[\frac{\Gamma\big(\alpha_{jl} + \beta_{jl}\big)}{\Gamma\big(\alpha_{jl}\big)\Gamma\big(\beta_{jl}\big)} \frac{X_{il}^{\alpha_{jl}-1}}{\big(1 + X_{il}\big)^{\alpha_{jl}+\beta_{jl}}} \right]^{\phi_{il}} \\
& \times \left[\frac{\Gamma\big(\lambda_l + \tau_l\big)}{\Gamma\big(\lambda_l\big)\Gamma\big(\tau_l\big)} \frac{X_{il}^{\lambda_l - 1}}{\big(1 + X_{il}\big)^{\lambda_l + \tau_l}} \right]^{1-\phi_{il}} \Bigg\}^{Z_{ij}} \times \prod_{i=1}^{N} \prod_{l=1}^{D} \epsilon_{l_1}^{\phi_{il}} \epsilon_{l_2}^{1-\phi_{il}} \\
& \times \prod_{i=1}^{N} \left[\prod_{j=1}^{s} \pi_j^{Z_{ij}} \prod_{j=s+1}^{M} \pi_j^{*Z_{ij}} \right] \times \left(1 - \sum_{k=1}^{s} \pi_k \right)^{-M+s} \\
& \times \frac{\Gamma\big(\sum_{j=s+1}^{M} C_j\big)}{\prod_{j=s+1}^{M} \Gamma\big(C_j\big)} \prod_{j=s+1}^{M} \left(\frac{\pi_j^*}{1 - \sum_{k=1}^{s} \pi_k} \right)^{C_j - 1} \\
& \times \frac{\nu_{jl}^{u_{jl}}}{\Gamma\big(u_{jl}\big)} \alpha_{jl}^{u_{jl}-1} e^{-\nu_{jl}\alpha_{jl}} \times \frac{q_{jl}^{p_{jl}}}{\Gamma\big(p_{jl}\big)} \beta_{jl}^{p_{jl}-1} e^{-q_{jl}\beta_{jl}} \\
& \times \frac{h_l^{g_l}}{\Gamma\big(g_l\big)} \lambda_{jl}^{g_l - 1} e^{-h_l \lambda_l} \times \frac{t_l^{s_l}}{\Gamma\big(s_l\big)} \tau_l^{s_l - 1} e^{-t_l \tau_l}
\end{aligned}
\tag{3.25}
$$

Fig. 3.6 shows the graphical model of the dependencies between the different parameters. By following the variational equation (2.17), we can write the variational solutions for our

model as:

$$\mathcal{Q}(\mathcal{Z}) = \prod_{i=1}^{N}\left[\prod_{j=1}^{s} r_{ij}^{Z_{ij}} \prod_{j=s+1}^{M} r_{ij}^{*Z_{ij}}\right],\ \mathcal{Q}(\vec{\phi}) = \prod_{j=1}^{M}\prod_{l=1}^{D} f_{il}^{\phi_{il}}\big(1-f_{il}\big)^{1-\phi_{il}} \tag{3.26}$$

$$\mathcal{Q}(\vec{\pi}^*) = (1-\sum_{k=1}^{s}\pi_k)^{-M+s}\frac{\Gamma(\sum_{j=s+1}^{M}c_j^*)}{\prod_{j=s+1}^{M}\Gamma(c_j^*)}\prod_{j=s+1}^{M}(\frac{\pi_j^*}{1-\sum_{k=1}^{s}\pi_k})^{c_j^*-1} \tag{3.27}$$

$$\mathcal{Q}(\vec{\alpha}) = \prod_{j=1}^{M}\prod_{l=1}^{D}\mathcal{G}\big(\alpha_{jl} \mid u_{jl}^*, \nu_{jl}^*\big),\ \mathcal{Q}(\vec{\beta}) = \prod_{j=1}^{M}\prod_{l=1}^{D}\mathcal{G}\big(\beta_{jl} \mid p_{jl}^*, q_{jl}^*\big) \tag{3.28}$$

$$\mathcal{Q}(\vec{\lambda}) = \prod_{l=1}^{D}\mathcal{G}\big(\lambda_l \mid g_l^*, h_l^*\big),\ \mathcal{Q}(\vec{\tau}) = \prod_{l=1}^{D}\mathcal{G}\big(\tau_l \mid s_l^*, t_l^*\big) \tag{3.29}$$

provided,

$$r_{ij} = \frac{\tilde{r}_{ij}}{\sum_{j=1}^{s}\tilde{r}_{ij}+\sum_{j=s+1}^{M}\tilde{r}_{ij}^*},\ r_{ij}^* = \frac{\tilde{r}_{ij}^*}{\sum_{j=1}^{s}\tilde{r}_{ij}+\sum_{j=s+1}^{M}\tilde{r}_{ij}^*} \tag{3.30}$$

$$\begin{aligned}\ln\tilde{r}_{ij} =& \ln\pi_j + \sum_{l=1}^{D}\Bigg\{\big\langle\phi_{il}\big\rangle\Big[\tilde{\mathcal{R}}_{jl} + \big(\overline{\alpha}_{jl}-1\big)\ln X_{il} - \big(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\big)\ln\big(1+X_{il}\big)\Big] \\ &+ \big\langle 1-\phi_{il}\big\rangle\Big[\tilde{\mathcal{F}}_l + \big(\overline{\lambda}_l-1\big)\ln X_{il} - \big(\overline{\lambda}_l+\overline{\tau}_{jl}\big)\ln\big(1+X_{il}\big)\Big]\Bigg\}\end{aligned} \tag{3.31}$$

$$\begin{aligned}\ln\tilde{r}_{ij}^* =& \big\langle\ln\pi_j^*\big\rangle + \sum_{l=1}^{D}\Bigg\{\big\langle\phi_{il}\big\rangle\Big[\tilde{\mathcal{R}}_{jl} + \big(\overline{\alpha}_{jl}-1\big)\ln X_{il} - \big(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\big)\ln\big(1+X_{il}\big)\Big] \\ &+ \big\langle 1-\phi_{il}\big\rangle\Big[\tilde{\mathcal{F}}_l + \big(\overline{\lambda}_l-1\big)\ln X_{il} - \big(\overline{\lambda}_l+\overline{\tau}_{jl}\big)\ln\big(1+X_{il}\big)\Big]\Bigg\}\end{aligned} \tag{3.32}$$

$$c_j^* = \sum_{i=1}^{N} r_{ij}^* + c_j, \quad f_{il} = \frac{f_{il}^{(\phi_{il})}}{f_{il}^{(\phi_{il})}+f_{il}^{(1-\phi_{il})}} \tag{3.33}$$

$$f_{il}^{(\phi_{il})} = exp\Bigg\{\big\langle\ln\epsilon_{l_1}\big\rangle + \sum_{j=1}^{M}\big\langle Z_{ij}\big\rangle\Big[\tilde{\mathcal{R}}_{jl} + \big(\overline{\alpha}_{jl}-1\big)\ln X_{il} - \big(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\big)\ln\big(1+X_{il}\big)\Big]\Bigg\} \tag{3.34}$$

$$f_{il}^{(1-\phi_{il})} = exp\Bigg\{\big\langle\ln\epsilon_{l_2}\big\rangle + \Big[\tilde{\mathcal{F}}_l + \big(\overline{\lambda}_l-1\big)\ln X_{il} - \big(\overline{\lambda}_l+\overline{\tau}_l\big)\ln\big(1+X_{il}\big)\Big]\Bigg\} \tag{3.35}$$

$$\begin{aligned}\tilde{\mathcal{R}} =& \ln\frac{\Gamma(\overline{\alpha}+\overline{\beta})}{\Gamma(\overline{\alpha})\Gamma(\overline{\beta})} + \overline{\alpha}\big[\psi(\overline{\alpha}+\overline{\beta}) - \psi(\overline{\alpha})\big]\big(\big\langle \ln\alpha\big\rangle - \ln\overline{\alpha}\big) \\ &+ \overline{\beta}\big[\psi(\overline{\alpha}+\overline{\beta}) - \psi(\overline{\beta})\big]\big(\big\langle \ln\beta\big\rangle - \ln\overline{\beta}\big) \\ &+ 0.5\overline{\alpha}^2\big[\psi'(\overline{\alpha}+\overline{\beta}) - \psi'(\overline{\alpha})\big]\big\langle(\ln\alpha - \ln\overline{\alpha})^2\big\rangle \\ &+ 0.5\overline{\beta}^2\big[\psi'(\overline{\alpha}+\overline{\beta}) - \psi'(\overline{\beta})\big]\big\langle(\ln\beta - \ln\overline{\beta})^2\big\rangle \\ &+ \overline{\alpha}\overline{\beta}\psi'(\overline{\alpha}+\overline{\beta})\big(\big\langle \ln\alpha\big\rangle - \ln\overline{\alpha}\big)\big(\big\langle \ln\beta\big\rangle - \ln\overline{\beta}\big) \end{aligned} \tag{3.36}$$

$$\begin{aligned}\tilde{\mathcal{F}} =& \ln\frac{\Gamma(\overline{\lambda}+\overline{\tau})}{\Gamma(\overline{\lambda})\Gamma(\overline{\tau})} + \overline{\lambda}\big[\psi(\overline{\lambda}+\overline{\tau}) - \psi(\overline{\lambda})\big]\big(\big\langle \ln\lambda\big\rangle - \ln\overline{\lambda}\big) \\ &+ \overline{\tau}\big[\psi(\overline{\lambda}+\overline{\tau}) - \psi(\overline{\tau})\big]\big(\big\langle \ln\tau\big\rangle - \ln\overline{\tau}\big) \\ &+ 0.5\overline{\lambda}^2\big[\psi'(\overline{\lambda}+\overline{\tau}) - \psi'(\overline{\tau})\big]\big\langle(\ln\lambda - \ln\overline{\lambda})^2\big\rangle \\ &+ 0.5\overline{\tau}^2\big[\psi'(\overline{\lambda}+\overline{\tau}) - \psi'(\overline{\tau})\big]\big\langle(\ln\tau - \ln\overline{\tau})^2\big\rangle \\ &+ \overline{\lambda}\,\overline{\tau}\,\psi'(\overline{\lambda}+\overline{\tau})\big(\big\langle \ln\lambda\big\rangle - \ln\overline{\lambda}\big)\big(\big\langle \ln\tau\big\rangle - \ln\overline{\tau}\big) \end{aligned} \tag{3.37}$$

$$\begin{aligned}u^*_{jl} =& u_{jl} + \sum_{i=1}^{N}\big\langle Z_{ij}\big\rangle\big\langle\phi_{il}\big\rangle\overline{\alpha}_{jl}\Bigg[\psi\big(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\big) - \psi\big(\overline{\alpha}_{jl}\big) \\ &+ \overline{\beta}_{jl}\psi'\big(\overline{\alpha}_{jl}+\overline{\beta}_{jl}\big)\big(\big\langle \ln\beta_{jl}\big\rangle - \ln\overline{\beta}_{jl}\big)\Bigg] \end{aligned} \tag{3.38}$$

$$\nu^*_{jl} = \nu_{jl} - \sum_{i=1}^{N}\big\langle Z_{ij}\big\rangle\big\langle\phi_{il}\big\rangle\ln\frac{X_{il}}{1+X_{il}} \tag{3.39}$$

Similar to the calculation of u^*_{jl} and ν^*_{jl} we can calculate the hyperparameters $p^*_{jl}, q^*_{jl}, g^*_l, h^*_l, s^*_l$ and t^*_l as well. $\psi\big(.\big)$ and $\psi'\big(.\big)$ denote the digamma and trigamma functions, in the equations above. $\tilde{\mathcal{R}}$ and $\tilde{\mathcal{F}}$ in equation (3.36) and (3.37) are the taylor series approximation of $\mathcal{R} = \big\langle \ln\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\big\rangle$ and $\mathcal{F} = \big\langle \ln\frac{\Gamma(\lambda+\tau)}{\Gamma(\lambda)\Gamma(\tau)}\big\rangle$ since these equations are intractable [6]. The expected values mentioned in the equations above are given by:

$$\big\langle Z_{ij}\big\rangle = \begin{cases} r_{ij}, & \text{for } j = 1, ..., s \\ r^*_{ij}, & \text{for } j = s+1, ..., M \end{cases} \tag{3.40}$$

$$\overline{\alpha}_{jl} = \frac{u^*_{jl}}{\nu^*_{jl}},\ \big\langle \ln\alpha_{jl}\big\rangle = \psi\big(u^*_{jl}\big) - \ln\nu^*_{jl} \tag{3.41}$$

$$\big\langle\big(\ln\alpha_{jl} - \ln\overline{\alpha}_{jl}\big)^2\big\rangle = \big[\psi\big(u^*_{jl}\big) - \ln u^*_{jl}\big]^2 + \psi'\big(u^*_{jl}\big) \tag{3.42}$$

$$\langle \phi_{il} \rangle = f_{il},\ \langle 1 - \phi_{il} \rangle = 1 - f_{il} \tag{3.43}$$

$$\langle \pi_j^* \rangle = \left(1 - \sum_{k=1}^{s} \pi_k\right) \frac{\sum_{i=1}^{N} r_{ij}^* + c_j}{\sum_{i=1}^{N} \sum_{k=s+1}^{M} r_{ik}^* + c_k} \tag{3.44}$$

$$\langle \ln \pi_j^* \rangle = \ln\big(1 - \sum_{k=1}^{s} \pi_k\big) + \psi\big(\sum_{i=1}^{N} r_{ij}^* + c_j\big) - \psi\big(\sum_{i=1}^{N} \sum_{k=s+1}^{M} r_{ik}^* + c_k\big) \tag{3.45}$$

We can derive similar equations like in (3.41) and (3.42), for $\vec{\beta}, \vec{\lambda}$ and $\vec{\tau}$. According to our algorithm, the irrelevant features will have lower probabilities and hence will not be used in the clustering process. These features are eliminated in the learning process which increases the efficiency of the clustering algorithm. The model selection method using component splitting approach on the other hand takes care of the problem of model selection. The efficiency of the model lies in the fact that the model selection approach is applied only to the components of the relevant features which saves time.

3.4 Experimental Results

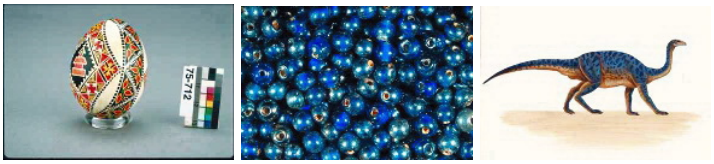
To evaluate our model we use two challenging datasets; the dynamic texture dataset (Dyntex) [64] and the Corel 10K dataset[2] for image categorization. We compare the results of our proposed variational GID mixture model (*varGIDMM*) with the standard benchmark of Gaussian mixture models based on maximum likelihood estimation (*GMM*) and variational approximation (*varGMM*). The initial values of the hyperparameters u, p, g, s and c is set to 1, that of ν and q is set to be 0.09 and that of h and t is set to be 0.06. These initiations were found to give the best results in our experiments.

3.4.1 Image Clustering

There has been a huge increase in the amount of images generated in recent years. With the increase in the volume of images, the need to categorize them based on analyzed patterns has been on the rise as well. Clustering the images hence plays a predominant role in categorizing the images. The efficiency of the use of bag of visual words features [45] is also imminent in recent years. To get the bag of visual words we first have to extract

[2] `http://www.ci.gxnu.edu.cn/cbir/Dataset.aspx`

Playing Cards Dolls Steam Tractors Paintings



Easter Eggs Beads Dinosaurs

Figure 3.7: Sample images from different categories of Corel 10K dataset

Table 3.3: Accuracy of different models on Corel 10K dataset

Method	Accuracy(%)
varGIDMM	87.41
varGMM	60.42
GMM	57.42

feature descriptors (scale invariant feature transform (SIFT) [40], histogram of Gaussians (HOG) [42], Speeded-up robust features (SURF) [41], etc.) from the images. We then use k-means clustering on the extracted descriptors with the k value indicating the number of features. The Corel 10K dataset which we choose for our application has about a 100 classes with 100 images per class. We choose 7 image classes from them corresponding to "Playing Cards","Dolls","Steam Tractors","Paintings","Easter Eggs","Beads" and "Dinosaurs". Sample images from the dataset are shown in Fig. 3.7. It is to be noted that the use of seven categories is ease of representation. In our case we first extract SIFT feature descriptors from the images as it is found to give better results and then generate bag of visual words features from the descriptors. We feed this data as input to our model. The Confusion matrix pertaining to our model is shown in Fig. 3.8. Table 3.3 shows the accuracy of different models compared with ours. It clearly shows that our model outperforms *GMM* models by a large margin.

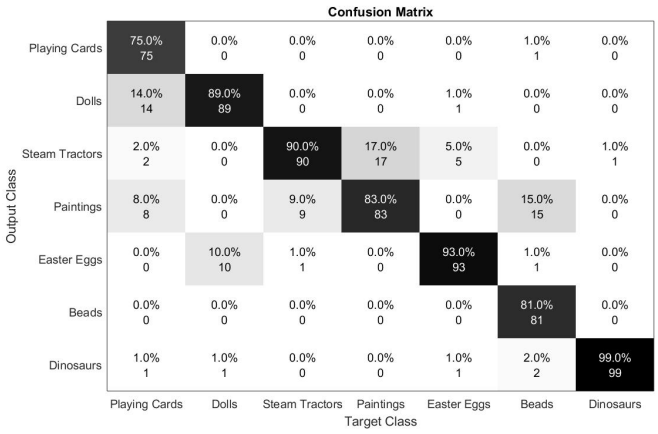


Figure 3.8: Confusion matrix of Corel 10K dataset with varGIDMM

Flags Flowers Sea Trees

Figure 3.9: Sample snapshots from different categories of DynTex dataset

3.4.2 Dynamic Texture Clustering

Dynamic textures refers to textures in the temporal dimension. For example, videos of burning fire, turbulence, sea waves, etc. Dynamic textures play an important role in various applications such as dynamic background subtraction, video completion, etc. Hence clustering them is of prime importance as well. In the case of dynamic textures extracting local binary pattern (LBP) features makes more sense because LBP mainly divides an image into cells and constructs a histogram of features by comparing each cell with its neighboring cells. In our experiment we use 4 classes from the DynTex dataset, which are: Flags, Flowers, Sea and Trees. Examples of the four classes are shown in Fig. 3.9. We extract LBP features from each frame of every video in a class. This is used as input to our model. The confusion matrix indicating the results obtained with our model is shown in Fig. 3.10. The accuracy of the different models is shown in Table 3.4. The results show that the *varGIDMM* is better than the *GMM*models. Based on the number of frames assigned to a particular cluster we can predict to which cluster the video belongs to.

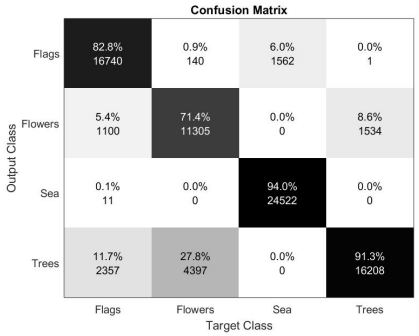


Figure 3.10: Confusion matrix of DynTex dataset with varGIDMM

Table 3.4: Accuracy of different models on DynTex dataset

Method	Accuracy(%)
varGIDMM	86.10
varGMM	84.42
GMM	84.87

Chapter 4

Finite Inverted Beta-Liouville Mixture Model with Variational Component Splitting

In this chapter, we introduce a finite mixture model based on Inverted Beta-Liouville distribution which provides a better fit for the data. We use a variational learning framework to estimate the parameters which decreases the computational complexity of the model. We handle the problem of model selection with a component splitting approach which is an added advantage as it is done within the variational framework. We evaluate our model against some challenging applications like image clustering, speech clustering, spam image detection and software defect detection.

4.1 The Statistical Model

Consider a $D-$dimensional vector $\vec{X}_i = \left(X_1, X_2, ..., X_D\right)$ drawn from a set of N independent and identically distributed data samples $\mathcal{X} = \left(\vec{X}_1, \vec{X}_2, ..., \vec{X}_N\right)$ generated from an inverted Beta-Liouville (IBL) distribution [46]. Then, the probability density function

$p\big(\vec{X}_i \mid \alpha_1, ..., \alpha_D, \alpha, \beta, \lambda\big)$ is given by:

$$p\big(\vec{X}_i \mid \alpha_{i1}, ..., \alpha_{iD}, \alpha, \beta, \lambda\big) = \frac{\Gamma\big(\sum_{l=1}^{D} \alpha_l\big)\Gamma\big(\alpha+\beta\big)}{\Gamma\big(\alpha\big)\Gamma\big(\beta\big)} \prod_{l=1}^{D} \frac{X_{il}^{\alpha_l - 1}}{\Gamma\big(\alpha_l\big)} \times \lambda^{\beta} \left(\sum_{l=1}^{D} X_{il}\right)^{\alpha - \sum_{l=1}^{D} \alpha_l} \left(\lambda + \sum_{l=1}^{D} X_{il}\right)^{-(\alpha+\beta)} \tag{4.1}$$

with the conditions $X_{il} > 0$ for $l = 1, .., D$, $\alpha > 0$, $\beta > 0$ and $\lambda > 0$. The mean, variance and covariance of IBL distribution is given by:

$$E(X_{il}) = \frac{\lambda\alpha}{\beta - 1} \frac{\alpha_l}{\sum_{l=1}^{D} \alpha_l} \tag{4.2}$$

$$Var(X_{il}) = \frac{\lambda^2\alpha(\alpha+1)}{(\beta-1)(\beta-2)} \frac{\alpha_l(\alpha+1)}{\sum_{l=1}^{D} \alpha_l(\sum_{l=1}^{D} \alpha_l + 1)} \frac{\lambda^2\alpha^2}{(\beta-1)^2} \frac{\alpha_l^4}{(\sum_{l=1}^{D} \alpha_l)^4} \tag{4.3}$$

$$Cov(X_{im}, X_{in}) = \frac{\alpha_m \alpha_n}{\sum_{l=1}^{D} \alpha_l} \left[\frac{\lambda^2\alpha(\alpha+1)}{(\beta-1)(\beta-2)(\sum_{l=1}^{D} \alpha_l + 1)} - \frac{\lambda^2\alpha^2}{(\beta-1)^2(\sum_{l=1}^{D} \alpha_l)} \right] \tag{4.4}$$

If we assume that each sample X_i is picked from a mixture of IBL distributions then the mixture model is represented as:

$$p\big(\mathcal{X} \mid \vec{\pi}, \Theta\big) = \sum_{i=1}^{N} \sum_{j=1}^{M} \pi_j p\big(\vec{X}_i \mid \theta_j\big) \tag{4.5}$$

where M is the number of components in the mixture model and $\Theta = (\theta_1, \theta_2, ..., \theta_M)$. $p\big(\vec{X}_i \mid \theta_j\big)$ denotes the conditional probability of the data sample with respect to each component, $\theta_j = (\alpha_{j1}, ..., \alpha_{jD}, \alpha_j, \beta_j, \lambda_j)$ represents the parameter with respect to the component j and $\vec{\pi} = (\pi_1, ..., \pi_M)$ is the set of mixing parameters and follows the conditions $\sum_{j=1}^{M} \pi_j = 1$ and $0 \leq \pi_j \leq 1$. We now introduce an indicator matrix $\mathcal{Z} = (\vec{Z}_1, ..., \vec{Z}_N)$ which indicates to which component each data sample is assigned to. Here $\vec{Z}_i = (Z_{i1}, ..., Z_{iM})$. $\vec{Z}_i$ is a binary vector that satisfies the conditions $Z_{ij} \epsilon \{0, 1\}$ and $\sum_{j=1}^{M} Z_{ij} = 1$ and is defined by:

$$Z_{ij} = \begin{cases} 1, & \text{if } \vec{X}_i \epsilon j \\ 0, & \text{otherwise} \end{cases} \tag{4.6}$$

The conditional distribution of $\mathcal{Z}$ can thus be defined as:

$$p\big(\mathcal{Z} \mid \vec{\pi}\big) = \prod_{i=1}^{N} \prod_{j=1}^{M} \pi_j^{Z_{ij}} \tag{4.7}$$

Based on this equation we can write the conditional distribution of a data set $\mathcal{X}$ with respect to the clusters as:

$$p\big(\mathcal{X} \mid \mathcal{Z}, \Theta\big) = \sum_{i=1}^{N}\sum_{j=1}^{M} p\big(\vec{X}_i \mid \theta_j\big)^{Z_{ij}} \tag{4.8}$$

As we know that all the parameters are positive it would be good choice to model them using Gamma priors. Hence the priors are defined by:

$$p\big(\alpha_{jl}\big) = \mathcal{G}\big(\alpha_{jl} \mid u_{jl}, \nu_{jl}\big) = \frac{\nu_{jl}^{u_{jl}}}{\Gamma\big(u_{jl}\big)}\alpha_{jl}^{u_{jl}-1}e^{-\nu_{jl}\alpha_{jl}} \tag{4.9}$$

$$p\big(\alpha_j\big) = \mathcal{G}\big(\alpha_j \mid p_j, q_j\big) = \frac{q_j^{p_j}}{\Gamma\big(p_j\big)}\alpha_j^{p_j-1}e^{-q_j\alpha_j} \tag{4.10}$$

$$p\big(\beta_j\big) = \mathcal{G}\big(\beta_j \mid g_j, h_j\big) = \frac{h_j^{g_j}}{\Gamma\big(g_j\big)}\beta_j^{g_j-1}e^{-h_j\beta_j} \tag{4.11}$$

$$p\big(\lambda_j\big) = \mathcal{G}\big(\lambda_j \mid s_j, t_j\big) = \frac{t_j^{s_j}}{\Gamma\big(s_j\big)}\lambda_j^{s_j-1}e^{-t_j\lambda_j} \tag{4.12}$$

where $\mathcal{G}\big(\cdot\big)$ represents a Gamma distribution and all the hyperparameters in the above priors are positive. Based on the component splitting design we can write the joint distribution for our model as:

$$p\big(\mathcal{X}, \mathcal{Z}, \Theta, \vec{\pi}^* \mid \vec{\pi}\big) = p\big(\mathcal{X} \mid \mathcal{Z}, \Theta\big)p\big(\mathcal{Z} \mid \vec{\pi}, \vec{\pi}^*\big)p\big(\vec{\pi}^* \mid \vec{\pi}\big)p\big(\alpha\big)p\big(\vec{\alpha}\big)p\big(\vec{\beta}\big)p\big(\vec{\lambda}\big) \tag{4.13}$$

$$\begin{aligned}
&= \prod_{i=1}^{N}\prod_{j=1}^{M}\Bigg[\frac{\Gamma\big(\sum_{l=1}^{D}\alpha_{jl}\big)\Gamma\big(\alpha_j+\beta_j\big)}{\Gamma\big(\alpha_j\big)\Gamma\big(\beta_j\big)}\prod_{l=1}^{D}\frac{X_{il}^{\alpha_{jl}-1}}{\Gamma\big(\alpha_{jl}\big)} \\
&\quad\times \lambda_j^{\beta_j}\Bigg(\sum_{l=1}^{D}X_{il}\Bigg)^{\alpha_j-\sum_{l=1}^{D}\alpha_{jl}}\Bigg(\lambda_j+\sum_{l=1}^{D}X_{il}\Bigg)^{-(\alpha_j+\beta_j)}\Bigg]^{Z_{ij}} \\
&\quad\times \prod_{i=1}^{N}\Bigg[\prod_{j=1}^{s}\pi_j^{Z_{ij}}\prod_{j=s+1}^{M}\pi_j^{*Z_{ij}}\Bigg]\times\Bigg(1-\sum_{k=1}^{s}\pi_k\Bigg)^{-M+s} \\
&\quad\times \frac{\Gamma\big(\sum_{j=s+1}^{M}c_j\big)}{\prod_{j=s+1}^{M}\Gamma\big(c_j\big)}\prod_{j=s+1}^{M}\Bigg(\frac{\pi_j^*}{1-\sum_{k=1}^{s}\pi_k}\Bigg)^{c_j-1} \\
&\quad\times \prod_{j=1}^{M}\prod_{l=1}^{D}\Bigg[\frac{\nu_{jl}^{u_{jl}}}{\Gamma\big(u_{jl}\big)}\alpha_{jl}^{u_{jl}-1}e^{-\nu_{jl}\alpha_{jl}}\times\frac{q_j^{p_j}}{\Gamma\big(p_j\big)}\alpha_j^{p_j-1}e^{-q_j\alpha_j} \\
&\quad\times \frac{h_j^{g_j}}{\Gamma\big(g_j\big)}\beta_j^{g_j-1}e^{-h_j\beta_j}\times\frac{t_j^{s_j}}{\Gamma\big(s_j\big)}\lambda_j^{s_j-1}e^{-t_j\lambda_j}\Bigg]
\end{aligned} \tag{4.14}$$

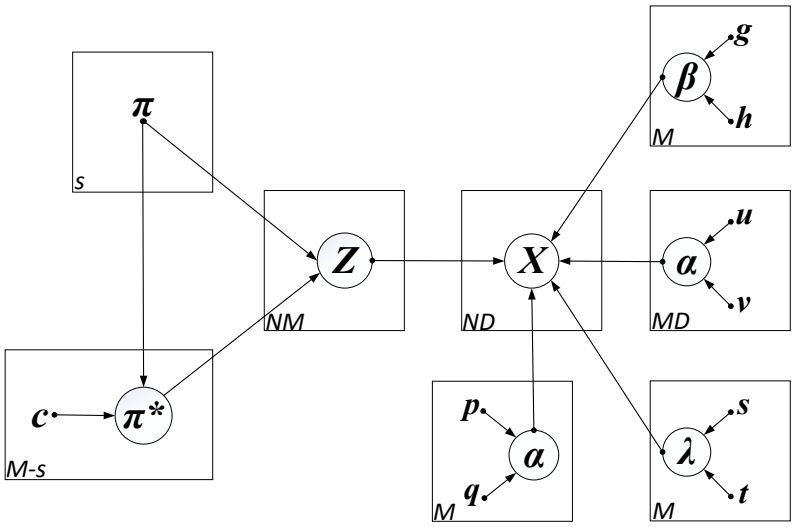


Figure 4.1: Graphical representation of IBL mixture model with component splitting. The circles indicate the random variables and model parameters, and plates point out the repetitions with the number in the lower left corners indicating the number of repetitions. The arcs specify the conditional dependencies of the variables.

Fig. 4.1 shows the graphical representation of the model. Similar to the previous cases, we can derive the variational solutions for our model as shown in Appendix B as:

$$Q(\mathcal{Z}) = \prod_{i=1}^{N} \left[\prod_{j=1}^{s} r_{ij}^{Z_{ij}} \prod_{j=s+1}^{M} r_{ij}^{*Z_{ij}} \right] \tag{4.15}$$

$$Q(\vec{\pi}^*) = \left(1 - \sum_{k=1}^{s} \pi_k\right)^{-M+s} \frac{\Gamma\left(\sum_{j=s+1}^{M} c_j^*\right)}{\prod_{j=s+1}^{M} \Gamma(c_j^*)} \prod_{j=s+1}^{M} \left(\frac{\pi_j^*}{1 - \sum_{k=1}^{s} \pi_k}\right)^{c_j^*-1} \tag{4.16}$$

$$Q(\alpha) = \prod_{j=1}^{M} \prod_{l=1}^{D} \mathcal{G}(\alpha_{jl} \mid u_{jl}^*, \nu_{jl}^*), \qquad Q(\vec{\alpha}) = \prod_{j=1}^{M} \mathcal{G}(\alpha_j \mid p_j^*, q_j^*) \tag{4.17}$$

$$Q(\vec{\beta}) = \prod_{j=1}^{M} \mathcal{G}(\beta_j \mid g_j^*, h_j^*), \qquad Q(\vec{\lambda}) = \prod_{j=1}^{M} \mathcal{G}(\lambda_j \mid s_j^*, t_j^*) \tag{4.18}$$

where:

$$r_{ij} = \frac{\tilde{r}_{ij}}{\sum_{j=1}^{s} \tilde{r}_{ij} + \sum_{j=s+1}^{M} \tilde{r}_{ij}^*}, \qquad r_{ij}^* = \frac{\tilde{r}_{ij}^*}{\sum_{j=1}^{s} \tilde{r}_{ij} + \sum_{j=s+1}^{M} \tilde{r}_{ij}^*} \tag{4.19}$$

$$\tilde{r}_{ij} = \exp\Bigg\{ \ln \pi_j + R_j + S_j + \Big(\overline{\alpha}_j - \sum_{l=1}^{D} \overline{\alpha}_{jl}\Big) \ln\Big(\sum_{l=1}^{D} X_{il}\Big) + \overline{\beta}_j \langle \ln \lambda_j \rangle + \sum_{l=1}^{D} \Big[\big(\overline{\alpha}_{jd} - 1\big) \ln X_{id}\Big] - \big(\overline{\alpha} + \overline{\beta}\big) T_{ij} \Bigg\} \tag{4.20}$$

$$\tilde{r}^{*}_{ij} = \exp\Bigg\{ \langle \ln \pi^{*}_j \rangle + R_j + S_j + \Big(\overline{\alpha}_j - \sum_{l=1}^{D} \overline{\alpha}_{jl}\Big) \ln\Big(\sum_{l=1}^{D} X_{il}\Big) + \overline{\beta}_j \langle \ln \lambda_j \rangle + \sum_{l=1}^{D} \Big[\big(\overline{\alpha}_{jd} - 1\big) \ln X_{id}\Big] - \big(\overline{\alpha} + \overline{\beta}\big) T_{ij} \Bigg\} \tag{4.21}$$

$$\begin{aligned} R_j = {} & \ln \frac{\Gamma\big(\sum_{l=1}^{D} \overline{\alpha}_{jl}\big)}{\prod_{l=1}^{D} \Gamma\big(\overline{\alpha}_{jl}\big)} + \sum_{l=1}^{D} \overline{\alpha}_{jl} \Bigg[\psi\Bigg(\sum_{l=1}^{D} \overline{\alpha}_{jl}\Bigg) - \psi\big(\overline{\alpha}_{jl}\big)\Bigg] \Big[\big\langle \ln \alpha_{jl} \big\rangle - \ln \overline{\alpha}_{jl}\Big] \\ & + \frac{1}{2} \sum_{l=1}^{D} \overline{\alpha}^{2}_{jl} \Bigg[\psi'\Bigg(\sum_{l=1}^{D} \overline{\alpha}_{jl}\Bigg) - \psi'\big(\overline{\alpha}_{jl}\big)\Bigg] - \Big\langle \big(\ln \alpha_{jl} - \ln \overline{\alpha}_{jl}\big)^2 \Big\rangle \\ & + \frac{1}{2} \sum_{a=1}^{D} \sum_{b=1}^{D} \overline{\alpha}_{ja}\, \overline{\alpha}_{jb} \Bigg[\psi'\Bigg(\sum_{l=1}^{D} \overline{\alpha}_{jl}\Bigg) \Big(\big\langle \ln \alpha_{ja} \big\rangle - \ln \overline{\alpha}_{ja}\Big) \\ & \times \Big(\big\langle \ln \alpha_{jb} \big\rangle - \ln \overline{\alpha}_{jb}\Big)\Bigg] \end{aligned} \tag{4.22}$$

$$\begin{aligned} S = {} & \ln \frac{\Gamma(\overline{\vec{\alpha}} + \overline{\vec{\beta}})}{\Gamma(\overline{\vec{\alpha}})\Gamma(\overline{\vec{\beta}})} + \overline{\vec{\alpha}}\big[\psi(\overline{\vec{\alpha}} + \overline{\vec{\beta}}) - \psi(\overline{\vec{\alpha}})\big]\big(\big\langle \ln \vec{\alpha} \big\rangle - \ln \overline{\vec{\alpha}}\big) \\ & + \overline{\vec{\beta}}\big[\psi(\overline{\vec{\alpha}} + \overline{\vec{\beta}}) - \psi(\overline{\vec{\beta}})\big]\big(\big\langle \ln \vec{\beta} \big\rangle - \ln \overline{\vec{\beta}}\big) \\ & + 0.5\overline{\vec{\alpha}}^{2}\big[\psi'(\overline{\vec{\alpha}} + \overline{\vec{\beta}}) - \psi'(\overline{\vec{\alpha}})\big]\big\langle (\ln \vec{\alpha} - \ln \overline{\vec{\alpha}})^2 \big\rangle \\ & + 0.5\overline{\vec{\beta}}^{2}\big[\psi'(\overline{\vec{\alpha}} + \overline{\vec{\beta}}) - \psi'(\overline{\vec{\beta}})\big]\big\langle (\ln \vec{\beta} - \ln \overline{\vec{\beta}})^2 \big\rangle \\ & + \overline{\vec{\alpha}}\overline{\vec{\beta}}\psi'(\overline{\vec{\alpha}} + \overline{\vec{\beta}})\big(\big\langle \ln \vec{\alpha} \big\rangle - \ln \overline{\vec{\alpha}}\big)\big(\big\langle \ln \vec{\beta} \big\rangle - \ln \overline{\vec{\beta}}\big) \end{aligned} \tag{4.23}$$

$$T_{ij} = \ln\Big[\overline{\lambda}_j + \sum_{l=1}^{D} X_{il}\Big] + \frac{\overline{\lambda}_j}{\overline{\lambda}_j + \sum_{l=1}^{D} X_{il}} \Big[\big\langle \ln \lambda_j \big\rangle - \ln \overline{\lambda}_j\Big] \tag{4.24}$$

$$c^{*}_{j} = \sum_{i=1}^{N} r^{*}_{ij} + c_j \tag{4.25}$$

$$u_{jl}^{*} = \; u_{jl} + \sum_{i=1}^{N} \langle Z_{ij} \rangle \overline{\alpha}_{jl} \left[\psi\Big(\sum_{l=1}^{D} \overline{\alpha}_{jl} \Big) - \psi\big(\overline{\alpha}_{jl}\big) + \psi'\Big(\sum_{l=1}^{D} \overline{\alpha}_{jl} \Big) \sum_{d \neq l}^{D} \Big(\langle \ln \alpha_{jl} \rangle - \ln \overline{\alpha}_{jl} \Big) \overline{\alpha}_{jl} \right] \tag{4.26}$$

$$\nu_{jl}^{*} = \nu_{jl} - \sum_{i=1}^{N} \langle Z_{ij} \rangle \left[\ln X_{il} - \ln \Big(\sum_{l=1}^{D} X_{il} \Big) \right] \tag{4.27}$$

$$p_{j}^{*} = \; p_{j} + \sum_{l=1}^{D} \langle Z_{ij} \rangle \left[\psi\big(\vec{\overline{\alpha}}_{j} + \vec{\overline{\beta}}_{j}\big) - \psi\big(\vec{\overline{\alpha}}_{j}\big) + \vec{\overline{\beta}}_{j} \psi'\big(\vec{\overline{\alpha}}_{j} + \vec{\overline{\beta}}_{j}\big) \Big(\langle \ln \beta_{j} \rangle - \vec{\overline{\beta}}_{j} \Big) \right] \vec{\overline{\alpha}}_{j} \tag{4.28}$$

$$q_{j}^{*} = q_{j} - \sum_{i=1}^{N} \langle Z_{ij} \rangle \ln \Big(\sum_{l=1}^{D} X_{id} \Big) + \sum_{i=1}^{N} \langle Z_{ij} \rangle T_{ij} \tag{4.29}$$

$$g_{j}^{*} = g_{j} + \sum_{l=1}^{D} \langle Z_{ij} \rangle \left[\psi\big(\overline{\alpha}_{j} + \overline{\beta}_{j}\big) - \psi\big(\overline{\beta}_{j}\big) + \overline{\alpha}_{j} \psi'\big(\overline{\alpha}_{j} + \overline{\beta}_{j}\big) \Big(\langle \ln \alpha_{j} \rangle - \overline{\alpha}_{j} \Big) \right] \overline{\beta}_{j}$$

$$h_{j}^{*} = h_{j} + \sum_{i=1}^{N} \langle Z_{ij} \rangle \Big[T_{ij} - \langle \ln \lambda_{j} \rangle \Big] \tag{4.30}$$

$$s_{j}^{*} = s_{j} + \sum_{i=1}^{N} \langle Z_{ij} \rangle \overline{\beta}_{j} \tag{4.31}$$

$$t_{j}^{*} = t_{j} + \sum_{i=1}^{N} \langle Z_{ij} \rangle \frac{\overline{\alpha}_{j} + \overline{\beta}_{j}}{\overline{\lambda}_{j} + \sum_{l=1}^{D} X_{il}} \tag{4.32}$$

The first and second derivatives of the Gamma function is given by the digamma and trigamma functions, $\psi(\cdot)$ and $\psi'(\cdot)$ respectively. The values of the expectations mentioned in the above equations are given by:

$$\langle Z_{ij} \rangle = \begin{cases} r_{ij}, & \text{for} j = 1, ..., s \\ r_{ij}^{*}, & \text{otherwise} \end{cases} \tag{4.33}$$

$$\overline{\alpha}_{jl} = \langle \alpha_{jl} \rangle = \frac{u_{jl}}{\nu_{jl}}, \quad \overline{\alpha}_{j} = \langle \alpha_{j} \rangle = \frac{p_{j}}{q_{j}}, \quad \overline{\beta}_{j} = \langle \beta_{j} \rangle = \frac{g_{j}}{h_{j}}, \quad \overline{\lambda}_{j} = \langle \lambda_{j} \rangle = \frac{s_{j}}{t_{j}} \tag{4.34}$$

$$\langle \ln \alpha_{jl} \rangle = \psi(u_{jl}^{*}) - \ln \nu_{jl}^{*}, \qquad \langle \ln \alpha_{j} \rangle = \psi(p_{j}^{*}) - \ln q_{j}^{*}, \tag{4.35}$$

$$\langle \ln \beta_j \rangle = \psi(g_j^*) - \ln h_j^*, \qquad \langle \ln \lambda_j \rangle = \psi(s_j^*) - \ln t_j^* \tag{4.36}$$

$$\left\langle \left(\ln \alpha_{jl} - \ln \overline{\alpha}_{jl} \right)^2 \right\rangle = \left[\psi\left(u_{jl}^*\right) - \ln u_{jl}^* \right]^2 + \psi'\left(u_{jl}^*\right) \tag{4.37}$$

$$\left\langle \left(\ln \alpha_j - \ln \overline{\alpha}_j \right)^2 \right\rangle = \left[\psi\left(p_j^*\right) - \ln p_j^* \right]^2 + \psi'\left(p_j^*\right) \tag{4.38}$$

$$\left\langle \left(\ln \beta_j - \ln \overline{\beta}_j \right)^2 \right\rangle = \left[\psi\left(g_j^*\right) - \ln g_j^* \right]^2 + \psi'\left(g_j^*\right) \tag{4.39}$$

$$\left\langle \pi_j^* \right\rangle = \left(1 - \sum_{k=1}^{s} \pi_k \right) \frac{\sum_{i=1}^{N} r_{ij}^* + c_j}{\sum_{i=1}^{N} \sum_{k=s+1}^{M} r_{ik}^* + c_k} \tag{4.40}$$

$$\left\langle \ln \pi_j^* \right\rangle = \ln \left(1 - \sum_{k=1}^{s} \pi_k \right) + \psi\left(\sum_{i=1}^{N} r_{ij}^* + c_j \right) - \psi\left(\sum_{i=1}^{N} \sum_{k=s+1}^{M} r_{ik}^* + c_k \right) \tag{4.41}$$

Based on the above update equations, we can calculate the lower bound by:

$$\mathcal{L}(Q) = \langle \ln p(\mathcal{X} \mid \mathcal{Z}, \Theta) \rangle + \langle \ln p(\mathcal{Z} \mid \vec{\pi}, \vec{\pi}^*) \rangle + \langle \ln p(\vec{\pi}^* \mid \vec{\pi}) \rangle \tag{4.42}$$

$$- \langle \ln p(\Theta) \rangle - \langle \ln Q(\mathcal{Z}) \rangle - \langle \ln Q(\Theta) \rangle \tag{4.43}$$

Based on these equations we run our component splitting algorithm to achieve efficient results.

4.2 Experimental Results

We present the experimental results we have obtained with our model in this section. We compare our variational IBL mixture model (IBLMM) with Gaussian mixture models with maximum likelihood estimation (GMM) and variational Gaussian mixture models (varGMM) since these are the standards nowadays. We evaluate our model against 4 challenging applications: object categorization, speech categorization, spam image categorization and software defect categorization. We try varying combinations involving imbalanced data to check the robustness of our model even when little data is available. We start with more or less equally weighted data sets to highly imbalanced data. The results are as follows:

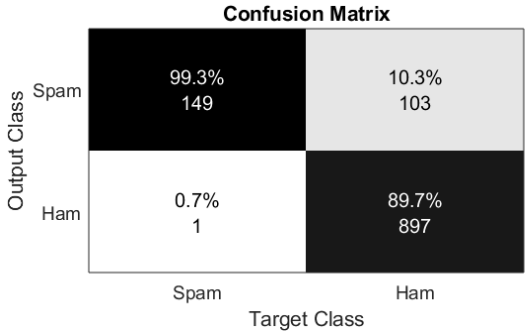


Figure 4.2: Confusion matrix of TSP speech data set with varIBLMM

4.2.1 Speech Categorization

With voice recognition based automation and control taking over in recent time, efficient categorization of speech signals becomes an important task. To evaluate our model, we took a simple task of clustering between male and female speakers in the TSP speech data set [65]. The TSP data set consists of speech utterances of 10 speakers where 5 are male and 5 are female. There are 60 speech utterances for each of the speakers. We take 500 samples from each category for our experiment. The pre-processing step for speech data involves, removal of non-speech parts like momentary pauses as a first step. This is done by voice activity detection (VAD) which removes the empty signals so that our model doesn't get trained on unnecessary pause signals. We now extract Mel Frequency Cepstral Coefficients (MFCC) which has been widely used for speech recognition tasks [66, 67]. The MFCC feature descriptors are 39 dimensional. Each speech utterance is sampled with a frame rate of 25ms with a window shift of 10ms. By this method a number of feature descriptors can be obtained from a single speech utterance file. We use bag of words feature model to create a histogram of the extracted MFCC features. This data serves as input to our model. The confusion matrix for our model is shown in Fig. 4.2. Table 4.1 shows the accuracy of IBLMM compared to GMM and varGMM. It shows that IBLMM improves the accuracy of GMM and varGMM.

4.2.2 Image Categorization

Pattern recognition from images is an important part of applications related to computer vision [36–38, 68]. It plays a major role in image retrieval, automated machinery, robot navigation, etc. For us to apply our model for image clustering, we have to extract feature

Table 4.1: Accuracy of different models for TSP speech data set

Method	Accuracy(%)
varIBLMM	86.6
varGMM	85.9
GMM	85.2

descriptors from the images. Some of the commonly used methods for feature extraction are: Scale Invariant feature Transform (SIFT) [40], Histogram of Gaussians (HoG) [42], Speeded-Up Robust Features (SURF) [41], etc. Once the features are extracted we have to represent each image in terms of these features. The best way to do that would be to use the bag of visual words representation [43–45]. The idea fo the bag of visual words approach is to cluster all the feature descriptors extracted from all the images using k-means creating a histogram of unique features for each of the image. These data will act as input to our model.

Images Clustering

For our first experiment we use the Ghim dataset [1] to evaluate the efficiency of our model. The Ghim dataset has 20 categories with 500 images in each class. Each of the images is 400×300 or 300×400. All images are in JPEG format. We use only four classes for ease of representation. Sample image from each of the four classes is shown in Fig. 4.3 we choose 500 images from the fireworks class, 150 images from cars, 250 images from Chinese buildings and 275 from dragon flies contributing 1175 images on the whole. It should be noted that the data points belonging to each class is varied over the 4 classes. We extract SIFT features from these images and use it to create bag of visual words features. Fig. 4.4 shows the confusion matrix obtained by using IBLMM on this data. The comparison of accuracy with GMM and varGMM model is shown in Table 4.2. It is clearly seen that the accuracy with our model is higher than the other two.

Spam Images Clustering

Usage of email services has become a quotidian task of everyday life nowadays. This also leaves us as a target to multiple ad agencies and fraudsters who send repeated ads and

[1] http://www.ci.gxnu.edu.cn/cbir/Dataset.aspx

(a) Fireworks (b) Cars (c) Chinese Buildings (d) Dragon Flies

Figure 4.3: Sample images from Ghim dataset

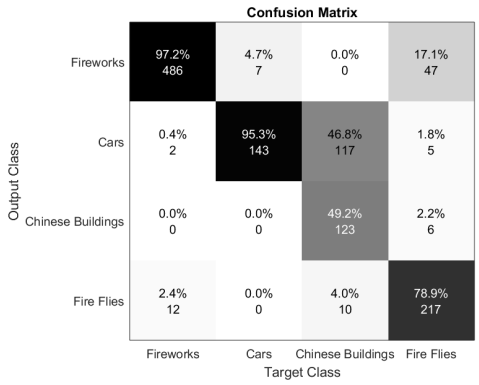


Figure 4.4: Confusion matrix of Ghim data set with varIBLMM

fake ones to trick us to reveal our personal information. Spam mails have also become a source of threats over the recent years. Hence it is very important to isolate the spam mails from the legitimate ones. However, it is also a very challenging task as the amount of spam data available is less when compared to the real ones in real world applications due to the presence of repeated images. Due to this reason detecting Spam images could be a good application to test the robustness our model. So we choose the spam data set created in [7] which consisted of threes sets of images. One is the ham data which contains normal images obtained from personal mails of people and considered useful. The other two sets contain spam images from spam archive created in [69] and a set of spam images taken from personal spam emails. In addition to this we also use images form the Princeton spam benchmark data set [2]. All the images used are taken from real emails and hence is a good representation of the real world scenario. However, all the three spam data sets contained a number of duplicate images. We took 150 varied spam images between the three spam data sets and 1000 images from the ham data. Sample images form the spam and ham sets are shown in Fig. 4.5 and Fig. 4.6 respectively. We can see that the spam data accounts

[2]http://www.cs.princeton.edu/cass/spam/

Table 4.2: Accuracy of different models for Ghim dataset

Method	Accuracy(%)
varIBLMM	82.46
varGMM	74.21
GMM	74.04

Figure 4.5: Sample images from the spam collection

for only 15% of the total data. We then extract SIFT features from these images and then create visual bag of words feature histogram from it. The confusion matrix for this data with our model is shown in Fig. 4.7. In applications based on security it is important that the false negative rate (FNR) is low because even if one malicious image is allowed in the network it might result in compromising the entire network in the worst case. On the other hand it is also important that the important images that is intended for the user is delivered as well; hence the false positive rates (FPR) should be low. Due to this reason both FPR and FNR have to be low for a good spam categorization model. We enforce the following

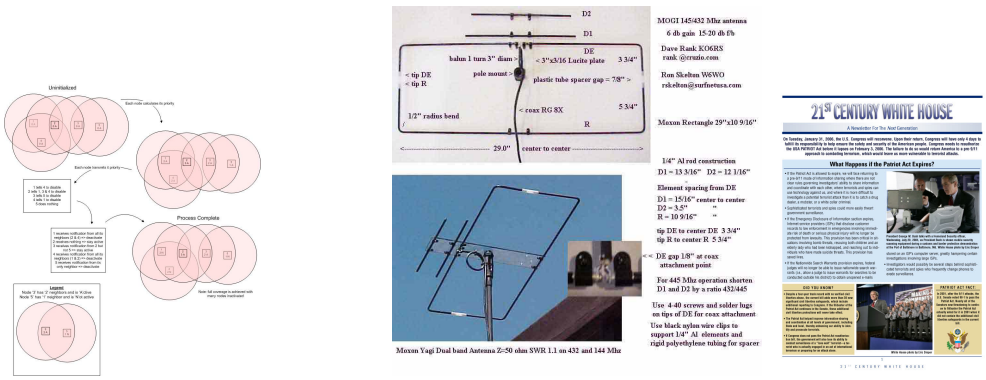


Figure 4.6: Sample images from the ham collection

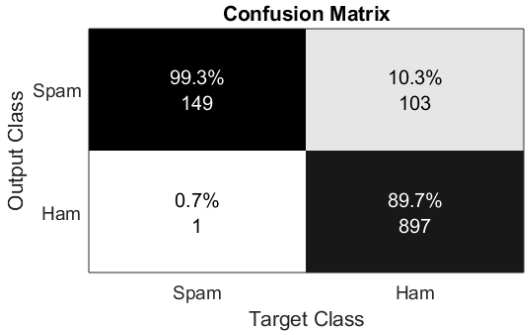


Figure 4.7: Confusion matrix of spam image data set with varIBLMM

Table 4.3: Performance measures of different models for spam image data set

Method	Accuracy(%)	Precision	Recall	FPR	FNR
varIBLMM	90.96	59.1	99.33	0.10	0.006
varGMM	81.22	40.8	98.00	0.21	0.020
GMM	79.73	39.1	98.66	0.23	0.013

performance measures to evaluate our model:

$$Precision = \frac{TP}{TP + FP} \tag{4.44}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.45}$$

$$FalsePositiveRate(FPR) = \frac{FP}{FP + TN} \tag{4.46}$$

$$FalseNegativeRate(FNR) = \frac{FN}{FN + TP} \tag{4.47}$$

where, true positives (TP) is the number of spam images correctly predicted as spam; false positives (FP) is the number of non spam images predicted as spam; true negatives (TN) is the number of non spam images correctly predicted as not spam and false negative (FN) is then number of spam images that have been classified as not spam. Table 4.3 shows the comparison of different performance measures for IBLMM, GMM and varGMM respectively. The FNR and FPR values are both low compared to GMM and varGMM which highlights the capability of our model to cluster imbalanced data sets.

4.2.3 Software Defect Categorization

Identification of software defects is an important part of software testing. Using machine learning techniques helps to identify defects in a short time and helps reduce the manual workforce for testing [70–72]. We validate our model against 5 data sets from the Promise software engineering repository [73] namely CM 1, JM1, KC1, KC2 and PC1. CM1, JM1 and PC1 are written in C and KC1 and KC2 are written in C++. CM1 is a software written for a NASA spacecraft instrument, JM1 is a real-time predictive ground system, KC1 and KC2 are storage management systems for processing ground data and PC1 is a flight software for earth orbiting satellites. McCabe and Halstead features are considered to describe the source code of these software to create these data sets. The performance metrics used are the same as in previous subsection. In the case of software defects we are more concerned about the false negatives and hence FNR is the most important measure. From Table 4.4 we are able to see that the False negative rate of IBLMM is very much higher than GMM and varGMM for all the data sets. The ratio between the defect and the non defect class was around 1:10 in some cases and we found in our experiments that varGMM and GMM are unable to distinguish the data into two classes in these scenarios.

Table 4.4: Results on defect detection using different models

Data Set	Model (%)	Accuracy	Precision	Recall	FNR
CM1	varIBL	67.87	17.50	61.22	0.39
	varGMM	72.69	1.13	2.04	0.98
	GMM	71.29	1.04	2.04	0.98
JM1	varIBL	66.09	0.29	52.16	0.48
	varGMM	74.10	0.23	15.59	0.84
	GMM	74.10	0.23	15.59	0.84
KC1	varIBL	69.41	30.28	75.15	0.25
	varGMM	73.06	32.81	70.85	0.29
	GMM	72.68	32.39	70.56	0.29
KC2	varIBL	77.78	47.40	79.43	0.21
	varGMM	49.04	4.04	6.50	0.93
	GMM	74.90	7.14	1.87	0.98
PC1	varIBL	68.80	11.68	53.24	0.47
	varGMM	89.35	2.32	1.3	0.99
	GMM	89.35	2.32	1.3	0.99

Chapter 5

Conclusion

Clustering has become an inevitable part of pattern recognition tasks. We have explored component splitting algorithm for different distributions and evaluated its efficiency.

In chapter 2, we introduce a novel unsupervised learning approach based on finite inverted Dirichlet mixture model. With the addition of component splitting approach for model selection our model was able to predict the number of clusters in the data with an impressive accuracy. The use of variational approach for model learning minimizes the computational complexity making it a better choice to pure Bayesian methods. The performance measures from synthetic data, occupancy estimation, images and videos by the application of our proposed model validates the effectiveness of our algorithm. In the case of synthetic data our model was able to find good estimates for the parameters closer to the real values and also the estimation of number of clusters was accurate. Occupancy estimation in smart homes proved to be another good application for our model. The results obtained with images and videos are quite promising. Our algorithm was able to correctly estimate the number of clusters in all the experiments compared to the remaining models. The performance of our model paves way for a number of new applications as well.

Then, in chapter 3, we describe the design of generalized inverted Dirichlet mixture model with the component splitting algorithm. The component splitting approach acts as an added advantage to the model as the estimation of number of clusters was accurate in all our experiments. Furthermore, variational optimization decreases the computational complexity of the model as well. The application to image categorization validates the effectiveness of the algorithm. Our model outperformed all the other models including Gaussian and variational Gaussian mixture models which are industry standards right now.

We then put forth an extended model integrating variational feature selection. Our model performed better than the GMM models by a large margin of over 25 percent in image categorization. The results with dynamic texture categorization also shows that our model is significantly better than the standard GMM models. The performance of the model is encouraging and can be applied to many other applications such as image segmentation and video categorization.

Finally, in chapter 4, we have proposed an efficient mixture model for clustering based on Inverted Beta Liouville mixtures. The variational framework combined with component splitting approach is found to be effective in model selection. The robustness of our model is evident from the experiments which involved data sets with varied weights. The first experiment with equal weight exhibited good results. The second experiment for object image clustering showed the effectiveness of our model in terms of mixed weights and also proved the efficiency of model selection. With spam image clustering we were able to achieve 90% accuracy against GMM and $varGMM$ which had only around 80% accuracy along with low FPR and FNR. In the last experiment, our model proved to be better than the other two models in identifying the defect class. It is to be noted that some data sets in this experiment had less than 10% of total data for the defect class.

The experiments with proposed frameworks are motivating and proves to be a better solution than Gaussian mixture models for appropriate data. Future works might include making the variational approach online to efficiently handle large amount of data.

Bibliography

[1] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proc. 17th Int. Conf. Pattern Recognition ICPR 2004*, volume 2, pages 28–31 Vol.2, August 2004.

[2] K. A. B. Lima, K. R. T. Aires, and F. W. P. D. Reis. Adaptive method for segmentation of vehicles through local threshold in the gaussian mixture model. In *Proc. Brazilian Conf. Intelligent Systems (BRACIS)*, pages 204–209, November 2015.

[3] Y. Li, C. Xiong, Y. Yin, and Y. Liu. Moving object detection based on edged mixture gaussian models. In *Proc. Int. Workshop Intelligent Systems and Applications*, pages 1–5, May 2009.

[4] D. Reynolds. Gaussian mixture models. *Encyclopedia of Biometrics*, January 2015.

[5] B. S. Oboh and N. Bouguila. Unsupervised learning of finite mixtures using scaled dirichlet distribution and its application to software modules categorization. In *2017 IEEE International Conference on Industrial Technology (ICIT)*, pages 1085–1090, March 2017.

[6] T. Bdiri, N. Bouguila, and D. Ziou. Variational bayesian inference for infinite generalized inverted dirichlet mixtures with feature selection and its application to clustering. *Applied Intelligence*, 44(3):507–525, Apr 2016.

[7] N. Bouguila and D. Ziou. Unsupervised learning of a finite discrete mixture model based on the multinomial dirichlet distribution: Application to texture modeling. In *Pattern Recognition in Information Systems*, pages 118–127, 2004.

[8] C. Hu, W. Fan, J. Du, and N. Bouguila. A novel statistical approach for clustering positive data based on finite inverted beta-liouville mixture models. *Neurocomputing*, 333:110 – 123, 2019.

[9] N. Bouguila and D. Ziou. Dirichlet-based probability model applied to human skin detection [image skin detection]. In *Proc. and Signal Processing 2004 IEEE Int. Conf. Acoustics, Speech*, volume 5, pages V–521, May 2004.

[10] N. Bouguila and D. Ziou. A powerful finite mixture model based on the generalized dirichlet distribution: unsupervised learning and applications. In *Proc. 17th Int. Conf. Pattern Recognition ICPR 2004*, volume 1, pages 280–283 Vol.1, August 2004.

[11] S. Bourouis, M. Mashrgy, and N. Bouguila. Bayesian learning of finite generalized inverted dirichlet mixtures: Application to object classification and forgery detection. *Expert Syst. Appl.*, 41(5):2329–2336, 2014.

[12] M. Mashrgy, T. Bdiri, and N. Bouguila. Robust simultaneous positive data clustering and unsupervised feature selection using generalized inverted dirichlet mixture models. *Knowl.-Based Syst.*, 59:182–195, 2014.

[13] T. Bdiri and N. Bouguila. Positive vectors clustering using inverted dirichlet finite mixture models. *Expert Syst. Appl.*, 39(2):1869–1882, February 2012.

[14] M. Gori and A. Tesi. On the problem of local minima in backpropagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1):76–86, January 1992.

[15] N. Bouguila and D. Ziou. High-dimensional unsupervised selection and estimation of a finite generalized dirichlet mixture model based on minimum message length. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1716–1731, October 2007.

[16] N. Zamzami and N. Bouguila. Model selection and application to high-dimensional count data clustering - via finite EDCM mixture models. *Appl. Intell.*, 49(4):1467–1488, 2019.

[17] N. Bouguila, J. H. Wang, and A. B. Hamza. Software modules categorization through likelihood and bayesian analysis of finite dirichlet mixtures. *Journal of Applied Statistics*, 37(2):235–252, 2010.

[18] H. Attias. Inferring parameters and structure of latent variable models by variational bayes. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 21–30. Morgan Kaufmann Publishers Inc., 1999.

[19] Y. W. Teh, D. Newman, and M. Welling. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Advances in neural information processing systems*, pages 1353–1360, 2007.

[20] C. Constantinopoulos and A. Likas. Unsupervised learning of gaussian mixtures based on variational component splitting. *IEEE Transactions on Neural Networks*, 18(3):745–755, May 2007.

[21] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, Nov 1999.

[22] W. Fan, N. Bouguila, and D. Ziou. A variational statistical framework for object detection. In *Neural Information Processing*, pages 276–283, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[23] W. Fan, N. Bouguila, and D. Ziou. Variational learning of finite dirichlet mixture models using component splitting. *Neurocomputing*, 129:3–16, 2014.

[24] G. G. Tiao and I. Cuttman. The inverted dirichlet distribution with applications. *Journal of the American Statistical Association*, 60(311):793–805, 1965.

[25] A. Corduneanu. and C. M. Bishop. Variational bayesian model selection for mixture distributions. In *Proceedings Eighth International Conference on Artificial Intelligence and Statistics*, 2001.

[26] M. Opper and D. Saad. *Tutorial on Variational Approximation Methods*. MITP, 2001.

[27] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, January 2006.

[28] H. X. Wang, Bin B. Luo, Q. B. Zhang, and S. Wei. Estimation for the number of components in a mixture model using stepwise split-and-merge em algorithm. *Pattern Recognition Letters*, 25(16):1799–1809, 2004.

[29] P. Tirdad, N. Bouguila, and D. Ziou. Variational learning of finite inverted dirichlet mixture models and applications. *Artificial Intelligence Applications in Information and Communication Technologies*, January 2015.

[30] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[31] W. Fan, N. Bouguila, and D. Ziou. Variational learning for finite dirichlet mixture models and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 23(5):762–774, May 2012.

[32] H. T. Malazi and M. Davari. Combining emerging patterns with random forest for complex activity recognition in smart homes. *Appl. Intell.*, 48(2):315–330, 2018.

[33] Z. Liouane, T. Lemlouma, P. Roose, F. Weis, and H. Messaoud. An improved extreme learning machine model for the prediction of human scenarios in smart homes. *Appl. Intell.*, 48(8):2017–2030, 2018.

[34] M. Amayri and S. Ploix. Decision tree and parametrized classifier for estimating occupancy in energy management. In *5th International Conference on Control, Decision and Information Technologies, CoDIT 2018, Thessaloniki, Greece, April 10-13, 2018*, pages 397–402, 2018.

[35] P. Fränti and S. Sieranoja. K-means properties on six clustering benchmark datasets. *Appl. Intell.*, 48(12):4743–4759, 2018.

[36] Y. Chen, J. Z. Wang, and R. Krovetz. An unsupervised learning approach to content-based image retrieval. In *Proc. Seventh Int. Symp. Signal Processing and Its Applications*, volume 1, pages 197–200 vol.1, July 2003.

[37] Y. Chen, J. Z. Wang, and R. Krovetz. Clue: cluster-based retrieval of images by unsupervised learning. *IEEE Transactions on Image Processing*, 14(8):1187–1201, August 2005.

[38] S. M. Zakariya, R. Ali, and N. Ahmad. Combining visual features of an image at different precision value of unsupervised content based image retrieval. In *Proc. IEEE Int. Conf. Computational Intelligence and Computing Research*, pages 1–4, December 2010.

[39] E. Gultepe and M. Makrehchi. Improving clustering performance using independent component analysis and unsupervised feature learning. *Human-centric Computing and Information Sciences*, 8(1):1, August 2018.

[40] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91, November 2004.

[41] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. Similarity Matching in Computer Vision and Multimedia.

[42] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, June 2005.

[43] M. Ravinder and T. Venugopal. Content-based cricket video shot classification using bag-of-visual-features. *Artificial Intelligence and Evolutionary Computations in Engineering Systems*, January 2016.

[44] Q. Zhu, Y. Zhong, B. Zhao, G. Xia, and L. Zhang. Bag-of-visual-words scene classifier with local and global features for high spatial resolution remote sensing imagery. *IEEE Geoscience and Remote Sensing Letters*, 13(6):747–751, June 2016.

[45] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints, 2004.

[46] H. Shao, T. Svoboda, and L. Van Gool. Zubud zurich buildings database for image based recognition. 01 2003.

[47] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision & Pattern Recognition (CPRV '06)*, pages 2169 – 2178, New York, United States, June 2006. IEEE Computer Society.

[48] M. Soleymani, M. Larson, T. Pun, and A. Hanjalic. Corpus development for affective video indexing. *IEEE Transactions on Multimedia*, 16(4):1075–1089, June 2014.

[49] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *Proc. 17th Int. Conf. Pattern Recognition ICPR 2004*, volume 3, pages 32–36 Vol.3, August 2004.

[50] F. Becker, S. Petra, and C. Schnörr. Optical flow. *Handbook of Mathematical Methods in Imaging*, January 2015.

[51] I. Bellamine and H. Tairi. Optical flow estimation based on the structure–texture image decomposition. *Signal, Image and Video Processing*, 9(1):193, December 2015.

[52] H. Miao and Y. Wang. Optical flow based obstacle avoidance and path planning for quadrotor flight. *Proceedings of 2017 Chinese Intelligent Automation Conference*, January 2018.

[53] T. Araújo, G. Aresta, J. Rouco, C. Ferreira, E. Azevedo, and A. Campilho. Optical flow based approach for automatic cardiac cycle estimation in ultrasound images of the carotid. *Image Analysis and Recognition*, January 2015.

[54] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th International Conference on Multimedia*, MULTIMEDIA '07, pages 357–360, New York, NY, USA, 2007. ACM.

[55] W. Fan and N. Bouguila. A variational component splitting approach for finite generalized dirichlet mixture models. In *2012 International Conference on Communications and Information Technology (ICCIT)*, pages 53–57, June 2012.

[56] S. Boutemedjet, D. Ziou, and N. Bouguila. Unsupervised feature selection for accurate recommendation of high-dimensional image data. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 177–184, 2007.

[57] T. Bdiri, N. Bouguila, and D. Ziou. Object clustering and recognition using multi-finite mixtures for semantic classes and hierarchy modeling. *Expert Syst. Appl.*, 41(4):1218–1235, 2014.

[58] N. Bouguila. A model-based discriminative framework for sets of positive vectors classification: Application to object categorization. In *2014 1st International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), Sousse, Tunisia, March 17-19, 2014*, pages 277–282, 2014.

[59] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *2004 Conference on Computer Vision and Pattern Recognition Workshop*, pages 178–178, 2004.

[60] T. Li, T. Mei, I. Kweon, and X. Hua. Contextual bag-of-words for visual categorization. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(4):381–392, April 2011.

[61] N. Bouguila and D. Ziou. Improving content based image retrieval systems using finite multinomial dirichlet mixture. In *Proceedings of the 2004 14th IEEE Signal Processing Society Workshop Machine Learning for Signal Processing, 2004.*, pages 23–32, 2004.

[62] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, Nov 1973.

[63] S. Boutemedjet, N. Bouguila, and D. Ziou. A hybrid feature extraction selection approach for high-dimensional non-gaussian data clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(8):1429–1443, Aug 2009.

[64] R. Péteri, S. Fazekas, and M. J. Huiskes. Dyntex: A comprehensive database of dynamic textures. *Pattern Recognition Letters*, 31(12):1627 – 1632, 2010.

[65] P. Kabal. TSP speech database. Technical report, Department of Electrical & Computer Engineering, McGill University, Montreal, Quebec, Canada, 2002.

[66] F. Zheng, G. Zhang, and Z. Song. Comparison of different implementations of mfcc. *Journal of Computer Science and Technology*, 16(6):582–589, Nov 2001.

[67] V. Tyagi and C. Wellekens. On desensitizing the mel-cepstrum to spurious spectral components for robust speech recognition. In *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 1, pages I/529–I/532 Vol. 1, March 2005.

[68] D. Liu and T. Chen. Unsupervised image categorization and object localization using topic models and correspondences between images. In *Proc. IEEE 11th Int. Conf. Computer Vision*, pages 1–7, October 2007.

[69] G. Fumera, I. Pillai, and F. Roli. Spam filtering based on the analysis of text information embedded into images. *Journal of Machine Learning Research*, 7:2699–2720, 2006.

[70] E. A. Felix and S. P. Lee. Integrated approach to software defect prediction. *IEEE Access*, 5:21524–21547, 2017.

[71] R. Islam and K. Sakib. A package based clustering for enhancing software defect prediction accuracy. In *2014 17th International Conference on Computer and Information Technology (ICCIT)*, pages 81–86. IEEE, 2014.

[72] X. Jing, Z. Zhang, S. Ying, F. Wang, and Y. Zhu. Software defect prediction based on collaborative representation classification. In *Companion Proceedings of the 36th International Conference on Software Engineering*, ICSE Companion 2014, pages 632–633, New York, NY, USA, 2014. ACM.

[73] S. J. Shirabad and T. J. Menzies. The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada, 2005.

[74] Z. Ma and A. Leijon. Bayesian estimation of beta mixture models with variational inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2160–2173, November 2011.

[75] M. W. Woolrich and T. E. Behrens. Variational bayes inference of spatial mixture models for segmentation. *IEEE Transactions on Medical Imaging*, 25(10):1380–1391, October 2006.

Appendix

A Proof of equations $\big(2.18\big), \big(2.19\big), \big(2.20\big)$

The solution for variational inference $Q_s\big(\Theta_s\big)$ is given by $\big(2.17\big)$ as,

$$\ln Q_s\big(\Theta_s\big) = \big\langle \ln p\big(\mathcal{X}, \Theta\big)\big\rangle_{t\neq s} + \text{const} \tag{A.1}$$

where the constant term is the culmination of all the terms that are independent of $Q_s\big(\Theta_s\big)$. The solutions can be easily derived from the logarithm of the joint distribution $p\big(\mathcal{X}, \Theta\big)$ given by,

$$\begin{aligned}
\ln p\big((X), \Theta\big) = &\sum_{i=1}^{N}\sum_{j=1}^{M} Z_{ij}\Bigg[\ln \pi_j + \ln \frac{\Gamma\big(\sum_{l=1}^{D+1}\alpha_{jl}\big)}{\prod_{l=1}^{D+1}\Gamma\big(\alpha_{jl}\big)} + \sum_{l=1}^{D}\big(\alpha_{jl}-1\big)X_{il} \\
&- \Bigg(\sum_{l=1}^{D+1}\alpha_{jl}\Bigg)\, ln\Bigg(1+\sum_{l=1}^{D}X_{il}\Bigg)\Bigg] \\
&+ \sum_{i=1}^{N}\Bigg[\sum_{j=1}^{s} Z_{ij}\ln\pi_j + \sum_{j=s+1}^{M} Z_{ij}\ln\pi_j^*\Bigg] \\
&- \big(M-s\big)\ln\Bigg[1-\sum_{k=1}^{s}\pi_k\Bigg] + \frac{\Gamma\big(\sum_{j=s+1}^{M} c_j\big)}{\prod_{j=s+1}^{M}\Gamma\big(c_j\big)} \\
&+ \sum_{j=s+1}^{M}\big(c_j-1\big)\Bigg[\pi_j^* - \Big(1-\sum_{k=1}^{s}\pi_k\Big)\Bigg] \\
&+ \sum_{j=1}^{M}\sum_{l=1}^{D} u_{jl}\ln\nu_{jl} - \ln\Gamma\big(u_{jl}\big) + \big(u_{jl}-1\big)\ln\alpha_{jl} - \nu_{jl}\alpha_{jl}
\end{aligned} \tag{A.2}$$

A.1 Proof of equation (2.18): variational solution for $Q(\mathcal{Z})$

The logarithm of $p(\mathcal{X}, \Theta)$ with respect to Z is given by,

$$\begin{aligned}
\ln Q(Z_i) =& \big\langle \ln p(\mathcal{X}, \Theta) \big\rangle_{\theta \neq Z_i} \\
=& \sum_{j=1}^{M} Z_{ij} \left[\ln \pi_j + R_j + \sum_{l=1}^{D} (\overline{\alpha}_{jl} - 1) X_{il} - \left(\sum_{l=1}^{D+1} \alpha_{jl} \right) \ln \left(1 + \sum_{l=1}^{D} X_{il} \right) \right] \\
& + \sum_{j=1}^{s} Z_{ij} \ln \pi_j + \sum_{j=s+1}^{M} Z_{ij} \big\langle \ln \pi_j^* \big\rangle + \text{const} \\
=& \sum_{j=1}^{s} Z_{ij} \left[\ln \pi_j + R_j + \sum_{l=1}^{D} (\overline{\alpha}_{jl} - 1) X_{il} - \left(\sum_{l=1}^{D+1} \alpha_{jl} \right) \ln \left(1 + \sum_{l=1}^{D} X_{il} \right) \right] \\
& \sum_{j=s+1}^{M} Z_{ij} \Bigg[\big\langle \ln \pi_j^* \big\rangle + R_j + \sum_{l=1}^{D} (\overline{\alpha}_{jl} - 1) X_{il} \\
& - \left(\sum_{l=1}^{D+1} \alpha_{jl} \right) \ln \left(1 + \sum_{l=1}^{D} X_{il} \right) \Bigg] + \text{const} \qquad \text{(A.3)}
\end{aligned}$$

where,

$$R_j = \left\langle \ln \frac{\Gamma\left(\sum_{l=1}^{D+1} \alpha_{jl}\right)}{\prod_{l=1}^{D+1} \Gamma(\alpha_{jl})} \right\rangle_{\alpha_{j1} \ldots \alpha_{jD+1}}, \overline{\alpha}_{jl} = \big\langle \alpha_{jl} \big\rangle = \frac{u_{jl}}{\nu_{jl}} \qquad \text{(A.4)}$$

Here, R_j is intractable as it has no closed form. In order to make the equation tractable we employ the second-order Taylor expansion of the equation similar to the method followed in [74, 75]. This leads us to the equation (2.24) which is actually approximation of R_j and $(\vec{\alpha}_{j1}, ..., \vec{\alpha}_{jD+1})$ representing the expected values of $\vec{\alpha}_j$. Thus, we can calculate $\tilde{R}_j$ using equation (2.24). This equation is also found to be the strict lower bound of R_j as proved in [29]. Equation $(A.3)$ can be now rewritten as,

$$\ln Q(\mathcal{Z}) = \sum_{i=1}^{N} \left[\sum_{j=1}^{s} Z_{ij} \ln \tilde{r}_{ij} + \sum_{j=s+1}^{M} Z_{ij} \ln \tilde{r}_{ij}^* \right] + \text{const} \qquad \text{(A.5)}$$

where

$$\ln \tilde{r}_{ij} = \ln \pi_j + \tilde{R}_j + \sum_{l=1}^{D} (\overline{\alpha}_{jl} - 1) \ln X_{il} - \left(\sum_{l=1}^{D+1} \alpha_{jl} \right) \ln \left(1 + \sum_{l=1}^{D} X_{il} \right) \qquad \text{(A.6)}$$

and

$$\ln \tilde{r}_{ij}^* = \big\langle \ln \pi_j^* \big\rangle + \tilde{R}_j + \sum_{l=1}^{D} (\overline{\alpha}_{jl} - 1) \ln X_{il} - \left(\sum_{l=1}^{D+1} \alpha_{jl} \right) \ln \left(1 + \sum_{l=1}^{D} X_{il} \right) \qquad \text{(A.7)}$$

It can be seen that equation $\big(A.5\big)$ is the logarithmic form of equation $\big(2.7\big)$ ignoring the constant. Exponentiating both the sides of equation $\big(2.7\big)$, we get,

$$Q\big(\mathcal{Z}\big) \propto \prod_{i=1}^{N} \left[\prod_{j=1}^{s} \tilde{r}_{ij}^{Z_{ij}} \prod_{j=s+1}^{M} \tilde{r}_{ij}^{*Z_{ij}} \right] \tag{A.8}$$

Normalizing this equation we can write the variational solution of $Q\big(\mathcal{Z}\big)$ as,

$$Q\big(\mathcal{Z}\big) \propto \prod_{i=1}^{N} \left[\prod_{j=1}^{s} r_{ij}^{Z_{ij}} \prod_{j=s+1}^{M} r_{ij}^{*Z_{ij}} \right] \tag{A.9}$$

where r_{ij} and r_{ij}^{*} can be obtained from equation $\big(2.22\big)$ and $\big(2.23\big)$. Also, we can say that $\big\langle Z_{ij} \big\rangle = r_{ij}$ for $j = 1, ..., s$ and $\big\langle Z_{ij}^{*} \big\rangle = r_{ij}$ for $j = s+1, ..., M$

A.2 Proof of equation $\big(2.19\big)$: variational solution of $Q(\vec{\pi}^{*})$

Similarly, the logarithm of the variational solution $Q\big(\vec{\pi}^{*}\big)$ is given as,

$$\begin{aligned} \ln Q\big(\pi_j^{*}\big) &= \big\langle \ln p\big(\mathcal{X}, \Theta\big) \big\rangle_{\Theta \neq \pi_j^{*}} \\ &= \sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle \ln \pi_j^{*} + \big(c_j - 1\big) \ln \pi_j^{*} + \text{const} \\ &= \ln \pi_j^{*} \left[\sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle + c_j - 1 \right] + \text{const} \end{aligned} \tag{A.10}$$

This equation shows that it has the same logarithmic form as that of equation $\big(2.9\big)$. So we can write the variational solution of $Q\big(\vec{\pi}^{*}\big)$ as,

$$Q\big(\vec{\pi}^{*}\big) = \left(1 - \sum_{k=1}^{s} \pi_k\right)^{-M+s} \frac{\Gamma\big(\sum_{j=s+1}^{M} c_j^{*}\big)}{\prod_{j=s+1}^{M} \Gamma\big(c_j^{*}\big)} \prod_{j=s+1}^{M} \left(\frac{\pi_j^{*}}{1 - \sum_{k=1}^{s} \pi_k} \right)^{c_j^{*}-1} \tag{A.11}$$

where

$$c_j^{*} = \sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle + c_j \tag{A.12}$$

$\big\langle Z_{ij} \big\rangle = r_{ij}^{*}$ in the above equation.

A.3 Proof of equation $\big(2.20\big)$: variational solution of $Q\big(\vec{\alpha}\big)$

As in the other two cases the logarithm of the variational solution $Q\big(\alpha_{jl}\big)$ is given by,

$$\begin{aligned}\ln Q\big(\alpha_{jl}\big) =& \big\langle \ln p\big(\mathcal{X},\Theta\big)\big\rangle_{\Theta\neq\alpha_{jl}} \\ =& \sum_{i=1}^{N}\big\langle Z_{ij}\big\rangle\mathcal{J}\big(\alpha_{jl}\big) + \alpha_{jl}\sum_{i=1}^{N}\big\langle Z_{ij}\big\rangle\ln X_{il} - \alpha_{jl}\ln\left(1+\sum_{l=1}^{D+1}X_{il}\right) \\ &+ \big(u_{jl}-1\big)\ln\alpha_{jl} - \nu_{jl}\alpha_{jl} + \text{const}\end{aligned} \tag{A.13}$$

where,

$$\mathcal{J}\big(\alpha_{jl}\big) = \left\langle \ln\frac{\Gamma\big(\alpha_{jl}+\sum_{s\neq l}^{D+1}\alpha_{js}\big)}{\Gamma\big(\alpha_{jl}\big)\prod_{s\neq l}^{D+1}\Gamma\big(\alpha_{js}\big)}\right\rangle_{\Theta\neq\alpha_{jl}} \tag{A.14}$$

Similar to what we encountered in the case of R_j the equation for $\mathcal{J}\big(\alpha_{jl}\big)$ is also intractable. We solve this problem finding the lower bound for the equation by calculating the first-order Taylor expansion with respect to $\overline{\alpha}_{jl}$. The calculated lower bound is given by,

$$\begin{aligned}\mathcal{L}\big(\alpha_{jl}\big) \geq \overline{\alpha}_{jl}\ln\alpha_{jl}\Bigg[&\psi\left(\sum_{l=1}^{D+1}\overline{\alpha}_{jl}\right) - \psi\big(\overline{\alpha}_{jl}\big) + \sum_{s\neq l}^{D+1}\overline{\alpha}_{js} \\ &\times\psi'\left(\sum_{l=1}^{D+1}\overline{\alpha}_{jl}\right)\big(\big\langle\ln\alpha_{js}\big\rangle - \ln\overline{\alpha}_{js}\big)\Bigg] + \text{const}\end{aligned} \tag{A.15}$$

This approximation is also found to be a strict lower bound of $\mathcal{L}\big(\alpha_{jl}\big)$ and is also proved in [29]. Substituting this equation for lower bound in equation $\big(A.13\big)$

$$\begin{aligned}\ln Q\big(\alpha_{jl}\big) =& \sum_{i=1}^{N}\big\langle Z_{ij}\big\rangle\overline{\alpha}_{jl}\ln\alpha_{jl}\Bigg[\psi\left(\sum_{l=1}^{D+1}\overline{\alpha}_{jl}\right) - \psi\big(\overline{\alpha}_{jl}\big) \\ &+ \sum_{s\neq l}^{D+1}\overline{\alpha}_{js}\psi'\left(\sum_{l=1}^{D+1}\overline{\alpha}_{jl}\right)\big(\big\langle\ln\alpha_{js}\big\rangle - \ln\overline{\alpha}_{js}\big)\Bigg] \\ &+ \alpha_{jl}\sum_{i=1}^{N}\big\langle Z_{ij}\big\rangle\ln X_{il} - \alpha_{jl}\ln\left(1+\sum_{l=1}^{D+1}X_{il}\right) \\ &+ \big(u_{jl}-1\big)\ln\alpha_{jl} - \nu_{jl}\alpha_{jl} + \text{const}\end{aligned} \tag{A.16}$$

This equation can be rewritten as,

$$\ln Q\big(\alpha_{jl}\big) = \ln\alpha_{jl}\big(u_{jl}+\varphi_{jl}-1\big) - \alpha_{jl}\big(\nu_{jl}-\vartheta_{jl}\big) + \text{const} \tag{A.17}$$

where,

$$\varphi_{jl} = \sum_{i=1}^{N} \langle Z_{ij} \rangle \overline{\alpha}_{jl} \Bigg[\psi\Bigg(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \Bigg) - \psi\big(\overline{\alpha}_{jl}\big) + \sum_{s \neq l}^{D+1} \overline{\alpha}_{js} \psi'\Bigg(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \Bigg) \big(\big\langle \ln \alpha_{js} \big\rangle - \ln \overline{\alpha}_{js} \big) \Bigg] \tag{A.18}$$

$$\vartheta_{jl} = \sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle \Bigg[\ln X_{il} - \ln \Bigg(1 + \sum_{l=1}^{D} X_{il} \Bigg) \Bigg] \tag{A.19}$$

Equation $\big(A.17\big)$ is the logarithmic form of a Gamma distribution. If we exponentiate both the sides, we get,

$$Q\big(\alpha_{jl}\big) \propto \alpha_{jl}^{u_{jl}+\varphi_{jl}-1} e^{-\left(\nu_{jl}-\vartheta_{jl}\right)\alpha_{jl}} \tag{A.20}$$

This leaves us with the optimal solution for the hyper-parameters u_{jl} and ν_{jl} given by,

$$u_{jl}^{*} = u_{jl} + \varphi_{jl}, \quad \nu_{jl}^{*} = \nu_{jl} - \vartheta_{jl} \tag{A.21}$$

B Proof of equations (4.15), (4.16), (4.17) and (4.18)

From eq. 4.13 we can write the logarithm of the joint as:

$$
\begin{aligned}
\ln p(\mathcal{X}, \mathcal{Z}) = & \sum_{i=1}^{N}\sum_{j=1}^{M} Z_{ij}\left[\ln \frac{\Gamma\big(\sum_{l=1}^{D}\alpha_{jl}\big)}{\prod_{l=1}^{D}\Gamma\big(\alpha_{jl}\big)} + \ln \frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} + \sum_{l=1}^{D}\big(\alpha_{jl}-1\big)\ln X_{il} \right. \\
& \left. + \beta_j \ln \lambda_j + \Big(\alpha_j - \sum_{i=1}^{D}\alpha_{jl}\Big)\ln\Big(\sum_{i=1}^{D} X_{il}\Big) - \big(\alpha_j+\beta_j\big)\ln\Big(\lambda_j + \sum_{i=1}^{D} X_{il}\Big)\right] \\
& + \sum_{i=1}^{N}\left[\sum_{j=1}^{s} Z_{ij}\ln\pi_j + \sum_{j=s+1}^{M} Z_{ij}\ln\pi_j^*\right] - \big(M-s\big)\ln\left[1-\sum_{k=1}^{s}\pi_k\right] \\
& + \ln\frac{\Gamma\big(\sum_{j=s+1}^{M} c_j\big)}{\prod_{j=s+1}^{M}\Gamma\big(c_j\big)} + \sum_{j=s+1}^{M}\big(c_j-1\big)\ln\left[\pi_j^* - \Big(1-\sum_{k=1}^{s}\pi_k\Big)\right] \\
& + \sum_{j=1}^{M}\sum_{l=1}^{D} u_{jl}\ln\nu_{jl} - \ln\Gamma\big(u_{jl}\big) + \big(u_{jl}-1\big)\ln\alpha_{jl} - \nu_{jl}\alpha_{jl} \\
& + \sum_{j=1}^{M} p_j \ln q_j - \ln\Gamma\big(p_j\big) + \big(p_j-1\big)\ln\alpha_j - q_j\alpha_j \\
& + \sum_{j=1}^{M} g_j \ln h_j - \ln\Gamma\big(g_j\big) + \big(g_j-1\big)\ln\beta_j - h_j\beta_j \\
& + \sum_{j=1}^{M} s_j \ln t_j - \ln\Gamma\big(s_j\big) + \big(s_j-1\big)\ln\lambda_j - t_j\lambda_j
\end{aligned}
\tag{B.1}
$$

To derive the variational solutions of each parameter, we consider the logarithm with respect to each of the parameter assuming the rest of the parameters to be constant. This is explained in the following sub sections.

B.1 Variational Solution for Q(Z) Eq. (4.15)

The logarithm with respect to $Q(Z_i)$ on the joint is given by:

$$\begin{aligned}\ln Q\big(Z_i\big) =& \sum_{j=1}^{M} Z_{ij}\Bigg[R_j + S_j + \sum_{l=1}^{D}\big(\alpha_{jl}-1\big)\ln X_{il} + \beta_j \ln\lambda_j \\ &+ \Big(\alpha_j - \sum_{i=1}^{D}\alpha_{jl}\Big)\ln\Big(\sum_{i=1}^{D} X_{il}\Big) - \big(\alpha_j+\beta_j\big)T_{ij}\Bigg] \\ &+ \Bigg[\sum_{j=1}^{s} Z_{ij}\ln\pi_j + \sum_{j=s+1}^{M} Z_{ij}\ln\pi_j^*\Bigg] \qquad \text{(B.2)} \\ =& \sum_{j=1}^{s}\Bigg[\ln\pi_j + R_j + S_j + \sum_{l=1}^{D}\big(\alpha_{jl}-1\big)\ln X_{il} + \beta_j\ln\lambda_j \\ &+ \Big(\alpha_j - \sum_{i=1}^{D}\alpha_{jl}\Big)\ln\Big(\sum_{i=1}^{D} X_{il}\Big) - \big(\alpha_j+\beta_j\big)T_{ij}\Bigg] \\ &\sum_{j=s+1}^{M}\Bigg[\langle\ln\pi_j^*\rangle + R_j + S_j + \sum_{l=1}^{D}\big(\alpha_{jl}-1\big)\ln X_{il} + \beta_j\ln\lambda_j \\ &+ \Big(\alpha_j - \sum_{i=1}^{D}\alpha_{jl}\Big)\ln\Big(\sum_{i=1}^{D} X_{il}\Big) - \big(\alpha_j+\beta_j\big)T_{ij}\Bigg] \qquad \text{(B.3)}\end{aligned}$$

Where,

$$R_j = \Bigg\langle \ln\frac{\Gamma\big(\sum_{l=1}^{D}\alpha_{jl}\big)}{\prod_{l=1}^{D}\Gamma\big(\alpha_{jl}\big)}\Bigg\rangle,\ S_j = \Bigg\langle \ln\frac{\Gamma(\alpha_j+\beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)}\Bigg\rangle,\ T_{ij} = \Bigg\langle \ln\Big(\lambda_j + \sum_{i=1}^{D} X_{il}\Big)\Bigg\rangle \qquad \text{(B.4)}$$

R_j, S_j and T_{ij} are intractable in the above equations. Due to this reason we use second order Taylor series approximation for R_j and S_j and first order Taylor series approximation for T_{ij}. the equations are given in eq. (4.22), (4.23) and (4.24) respectively. It is a notable fact that B.2 is of the form:

$$\ln Q\big(\mathcal{Z}\big) = \sum_{i=1}^{N}\Bigg[\sum_{j=1}^{s} Z_{ij}\ln\tilde{r}_{ij} + \sum_{j=s+1}^{M} Z_{ij}\ln\tilde{r}_{ij}^*\Bigg] + \text{const} \qquad \text{(B.5)}$$

given

$$\begin{aligned}\ln\tilde{r}_{ij} =& \ln\pi_j + R_j + S_j + \Big(\overline{\alpha}_j - \sum_{l=1}^{D}\overline{\alpha}_{jl}\Big)\ln\Big(\sum_{l=1}^{D} X_{il}\Big) + \overline{\beta}_j\langle\ln\lambda_j\rangle \\ &+ \sum_{l=1}^{D}\Big[\big(\overline{\alpha}_{jd}-1\big)\ln X_{id}\Big] - \big(\overline{\alpha}+\overline{\beta}\big)T_{ij} \qquad \text{(B.6)}\end{aligned}$$

$$\ln \tilde{r}^*_{ij} = \langle \ln \pi^*_j \rangle + R_j + S_j + \Big(\overline{\alpha}_j - \sum_{l=1}^{D} \overline{\alpha}_{jl}\Big) \ln \Big(\sum_{l=1}^{D} X_{il}\Big) + \overline{\beta}_j \langle \ln \lambda_j \rangle$$
$$+ \sum_{l=1}^{D} \Big[\big(\overline{\alpha}_{jd} - 1\big) \ln X_{id}\Big] - \big(\overline{\alpha} + \overline{\beta}\big) T_{ij} \tag{B.7}$$

By taking the exponentiation of eq. (B.2) we can write:

$$Q\big(\mathcal{Z}\big) \propto \prod_{i=1}^{N} \left[\prod_{j=1}^{s} \tilde{r}_{ij}^{Z_{ij}} \prod_{j=s+1}^{M} \tilde{r}_{ij}^{*Z_{ij}} \right] \tag{B.8}$$

Normalizing this equation we can write the variational solution of $Q\big(\mathcal{Z}\big)$ as,

$$Q\big(\mathcal{Z}\big) \propto \prod_{i=1}^{N} \left[\prod_{j=1}^{s} r_{ij}^{Z_{ij}} \prod_{j=s+1}^{M} r_{ij}^{*Z_{ij}} \right] \tag{B.9}$$

where r_{ij} and r^*_{ij} can be obtained from equation $\big(4.20\big)$ and $\big(4.21\big)$. Also, we can say that $\big\langle Z_{ij} \big\rangle = r_{ij}$ for $j = 1, ..., s$ and $\big\langle Z^*_{ij} \big\rangle = r^*_{ij}$ for $j = s+1, ..., M$

B.2 Proof of eq. $\big(4.16\big)$: variational solution of $Q(\vec{\pi}^*)$

Similarly, the logarithm of the variational solution $Q\big(\vec{\pi}^*\big)$ is given as,

$$\begin{aligned}
\ln Q\big(\pi^*_j\big) &= \big\langle \ln p\big(\mathcal{X}, \Theta\big) \big\rangle_{\Theta \neq \pi^*_j} \\
&= \sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle \ln \pi^*_j + \big(c_j - 1\big) \ln \pi^*_j + \text{const} \\
&= \ln \pi^*_j \left[\sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle + c_j - 1 \right] + \text{const}
\end{aligned} \tag{B.10}$$

This equation shows that it has the same logarithmic form as that of equation $\big(2.9\big)$. So we can write the variational solution of $Q\big(\vec{\pi}^*\big)$ as,

$$Q\big(\vec{\pi}^*\big) = \left(1 - \sum_{k=1}^{s} \pi_k\right)^{-M+s} \frac{\Gamma\big(\sum_{j=s+1}^{M} c^*_j\big)}{\prod_{j=s+1}^{M} \Gamma\big(c^*_j\big)} \prod_{j=s+1}^{M} \left(\frac{\pi^*_j}{1 - \sum_{k=1}^{s} \pi_k}\right)^{c^*_j - 1} \tag{B.11}$$

where

$$c^*_j = \sum_{i=1}^{N} \big\langle Z_{ij} \big\rangle + c_j \tag{B.12}$$

$\big\langle Z_{ij} \big\rangle = r^*_{ij}$ in the above equation.

B.3 Proof of equation $\big(4.17\big)$: variational solution of $Q\big(\vec{\alpha}\big)$

As in the other two cases the logarithm of the variational solution $Q\big(\alpha_{jl}\big)$ is given by,

$$\begin{aligned}\ln Q\big(\alpha_{jl}\big) =& \big\langle \ln p\big(\mathcal{X},\Theta\big)\big\rangle_{\Theta\neq\alpha_{jl}} \\ =& \sum_{i=1}^{N}\big\langle Z_{ij}\big\rangle\Big[\mathcal{J}\big(\alpha_{jl}\big)+\alpha_{jl}\ln X_{il}-\alpha_{jl}\ln\Big(\sum_{i=1}^{D}X_{il}\Big)\Big] \\ &+\big(u_{jl}-1\big)\ln\alpha_{jl}-\nu_{jl}\alpha_{jl}+\text{const}\end{aligned} \tag{B.13}$$

where,

$$\mathcal{J}\big(\alpha_{jl}\big)=\left\langle \ln\frac{\Gamma\big(\alpha_{jl}+\sum_{s\neq l}^{D+1}\alpha_{js}\big)}{\Gamma\big(\alpha_{jl}\big)\prod_{s\neq l}^{D+1}\Gamma\big(\alpha_{js}\big)}\right\rangle_{\Theta\neq\alpha_{jl}} \tag{B.14}$$

Similar to what we encountered in the case of R_j the equation for $\mathcal{J}\big(\alpha_{jl}\big)$ is also intractable. We solve this problem finding the lower bound for the equation by calculating the first-order Taylor expansion with respect to $\overline{\alpha}_{jl}$. The calculated lower bound is given by,

$$\begin{aligned}\mathcal{L}\big(\alpha_{jl}\big)\geq\overline{\alpha}_{jl}\ln\alpha_{jl}\Bigg[&\psi\Bigg(\sum_{l=1}^{D+1}\overline{\alpha}_{jl}\Bigg)-\psi\big(\overline{\alpha}_{jl}\big)+\sum_{s\neq l}^{D+1}\overline{\alpha}_{js} \\ &\times\psi'\Bigg(\sum_{l=1}^{D+1}\overline{\alpha}_{jl}\Bigg)\big(\big\langle\ln\alpha_{js}\big\rangle-\ln\overline{\alpha}_{js}\big)\Bigg]+\text{const}\end{aligned} \tag{B.15}$$

This approximation is also found to be a strict lower bound of $\mathcal{L}\big(\alpha_{jl}\big)$. Substituting this equation for lower bound in equation $\big(B.13\big)$

$$\begin{aligned}\ln Q\big(\alpha_{jl}\big) =& \sum_{i=1}^{N}\langle Z_{ij}\rangle\overline{\alpha}_{jl}\ln\alpha_{jl}\Bigg[\psi\Big(\sum_{l=1}^{D}\overline{\alpha}_{jl}\Big)-\psi\big(\overline{\alpha}_{jl}\big) \\ &+\psi'\Big(\sum_{l=1}^{D}\overline{\alpha}_{jl}\Big)\sum_{d\neq l}^{D}\Big(\langle\ln\alpha_{jl}\rangle-\ln\overline{\alpha}_{jl}\Big)\overline{\alpha}_{jl}\Bigg] \\ &+\sum_{i=1}^{N}\alpha_{jl}\langle Z_{ij}\rangle\Bigg[\ln X_{il}-\ln\Big(\sum_{l=1}^{D}X_{il}\Big)\Bigg]+\text{const}\end{aligned} \tag{B.16}$$

This equation can be rewritten as,

$$\ln Q\big(\alpha_{jl}\big)=\ln\alpha_{jl}\big(u_{jl}+\varphi_{jl}-1\big)-\alpha_{jl}\big(\nu_{jl}-\vartheta_{jl}\big)+\text{const} \tag{B.17}$$

where,

$$\varphi_{jl} = \sum_{i=1}^{N} \langle Z_{ij} \rangle \overline{\alpha}_{jl} \Bigg[\psi\Big(\sum_{l=1}^{D} \overline{\alpha}_{jl} \Big) - \psi\big(\overline{\alpha}_{jl}\big) + \psi'\Big(\sum_{l=1}^{D} \overline{\alpha}_{jl} \Big) \sum_{d \neq l}^{D} \Big(\langle \ln \alpha_{jl} \rangle - \ln \overline{\alpha}_{jl} \Big) \overline{\alpha}_{jl} \Bigg] \psi'\Bigg(\sum_{l=1}^{D+1} \overline{\alpha}_{jl} \Bigg) \Big(\big\langle \ln \alpha_{js} \big\rangle - \ln \overline{\alpha}_{js} \Big) \Bigg] \tag{B.18}$$

$$\vartheta_{jl} = \sum_{i=1}^{N} \langle Z_{ij} \rangle \Bigg[\ln X_{il} - \ln \Big(\sum_{l=1}^{D} X_{il} \Big) \Bigg] \tag{B.19}$$

Eq. $\big(B.17\big)$ is the logarithmic form of a Gamma distribution. If we exponentiate both the sides, we get,

$$Q\big(\alpha_{jl}\big) \propto \alpha_{jl}^{u_{jl}+\varphi_{jl}-1} e^{-\big(\nu_{jl}-\vartheta_{jl}\big)\alpha_{jl}} \tag{B.20}$$

This leaves us with the optimal solution for the hyper-parameters u_{jl} and ν_{jl} given by,

$$u_{jl}^{*} = u_{jl} + \varphi_{jl}, \quad \nu_{jl}^{*} = \nu_{jl} - \vartheta_{jl} \tag{B.21}$$

By following the same procedure we can get the variational solutions for $Q(\vec{\alpha}), Q(\vec{\beta})$ and $Q(\vec{\lambda})$.