Fully Bayesian Learning with Markov Chain Monte Carlo Methods for Asymmetric Generalized Gaussian Mixture and Hidden Markov Models

Ravi Teja Vemuri

A Thesis

in

The Concordia Institute

for

Information Systems Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Quality Systems Engineering) at

Concordia University

Montréal, Québec, Canada

December 2021

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Ravi Teja Vemuri**

Entitled: **Fully Bayesian Learning with Markov Chain Monte Carlo Methods for Asymmetric Generalized Gaussian Mixture and Hidden Markov Models**

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

Master of Applied Science (Quality Systems Engineering)

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

Signed by the Final Examining Committee:

______________________ Chair, Examiner
Dr. Chadi Assi

______________________ Examiner
Dr. Farnoosh Naderkhani

______________________ Supervisor
Dr. Nizar Bouguila

______________________ Co-supervisor
Dr. Zachary Patterson

Approved by ______________________
Dr. Mohammad Mannan, Graduate Program Director

__________ 2021

____
Dr. Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Abstract

Fully Bayesian Learning with Markov Chain Monte Carlo Methods for Asymmetric Generalized Gaussian Mixture and Hidden Markov Models

Ravi Teja Vemuri

A unique and efficient Bayesian learning framework is proposed for the learning of asymmetric generalized Gaussian mixtures and hidden Markov models. This framework is based on Markov chain Monte Carlo (MCMC) sampling with hybrid Metropolis-Hastings within Gibbs sampling as the fundamental learning algorithm. The algorithm is integrated with the reversible jump MCMC (RJMCMC) technique to achieve a fully Bayesian learning framework for proposed models. A fully Bayesian learning framework allows self-adaptive learning where the two major challenges of mixture modelling, parameter estimation and model selection, are done automatically thereby making the learning process autonomous. Furthermore, feature selection is explored and incorporated in the learning process to enhance the capability of the models to weight and pick relevant features in multi-dimensional data. The proposed framework is tested through a wide range of applications: activity recognition, speaker recognition etc., and its performance is evaluated with multiple performance metrics to display the robustness of the approach.

Acknowledgments

"Guru devo bhava"

I want to offer my profound gratitude to Prof. Nizar Bouguila, my supervisor. It all began in 2019 when I was working as a full-stack developer and aiming to be a machine learning engineer. With no prior knowledge of artificial intelligence (AI), I attempted a couple of online classes; all I learned was how to implement a machine learning model without first understanding it. After recognizing that I needed a master's degree to go deeper into AI, I applied to various universities and was refused by all of them. I was losing hope, and obtaining an AI degree from a reputable university had become a pipe dream. Concordia University was my final attempt; I found Prof. Nizar's study interesting and expressed my enthusiasm in an email to him. I am here with my thesis. I also want to express my gratitude to him for his unwavering support and guidance during my master's program and for providing me opportunities of all-round development by placing me on various industrial projects. Additionally, I want to express my heartfelt appreciation to my co-supervisor, Prof. Zachary Patterson, for believing in me and helping me with my study, and giving me a chance by involving me in a highly innovative industrial project that provided valuable experience.

I would like to express my gratitude to Dr. Muhammad Azam, a friend, mentor, and colleague, for supporting and motivating me throughout my research and master's journey. I'd want to express my gratitude to all of my lab mates for all of the knowledge sharing and fun we had. Ultimately, I want to express my gratitude to my family for their constant support. I could not have accomplished what I did without the help and support of everyone I've listed. Many thanks!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Due to the data generated as a result of numerous technological processes, Artificial Intelligence (AI) has become a vital aspect of many technical solutions in the modern day [24]. In general, the term "AI" is used to refer to any system or computer capable of self-optimization, learning, or inference. Additionally, it can be characterised as a computer system that has been trained to execute cognitive activities similar to those performed by humans, such as image recognition [3], natural language processing [26], and speech processing [71]. Machine learning (ML) [4, 23] is a subset of artificial intelligence that enables computers to perform tasks beyond their capabilities through the use of a variety of programming approaches and algorithms. In machine learning, enormous volumes of data are analysed to assist the machine in evolving with each iteration, which is the learning element. As data is the core component, an orderly data flow is required for every machine learning algorithm to perform properly. The data must be clean, varied, and machine-readable in order for the learning process to be efficient. Data mining [45] on the other hand is used to uncover patterns in data, enabling machines to make decisions autonomously and forecast future patterns.

Data modelling is a difficult undertaking in and of itself, and numerous statistical data mining methodologies have been advocated for managing and extracting information [73] and discovering patterns [41] from ever-increasing volumes of data. On the other hand, numerous generative and discriminative approaches in machine learning have been created to enhance the effectiveness of the learning process. These approaches can be broadly classified into two categories: supervised and unsupervised learning. In the former approach, prior to modelling, classes in the data are identified, and the model is trained on them in order to assign new data to one of these classes. In the latter technique, no prior knowledge about the data is assumed, and models are expected to cluster the data autonomously. The term ”classification” is frequently used to refer to supervised learning activities, whereas ”clustering” is used to refer to unsupervised learning tasks. In either case, the learning process is extremely challenging and complex [64]. Inadequate and insufficient data, over- and under-fitting, as well as other learning biases, can all contribute to models performing badly, resulting in erroneous predictions.

Approaches to mixture modelling [17] try to resolve the problem of learning model parameters and calculating the number of clusters (M) that best characterise a dataset. This thesis examines Bayesian approaches such as Markov chain Monte Carlo (MCMC) methods that are used in the proposed work in combination with mixture models and hidden Markov models. In Bayesian inference, parameters are randomly picked from several posterior probability distributions that represent the prior knowledge about the data and the model's parameters. Throughout this study, a hybrid MCMC strategy referred to as Metropolis-Hastings [46] inside Gibbs sampling [43] is employed for learning. This hybrid strategy is used due to the difficulty of implementing classic MCMC approaches in situations when direct sampling is impractical. By employing this hybrid approach, proposal distributions for mixture parameters are formed iteratively and decisions are made after each iteration using acceptance ratio computations. Finally, following convergence, the ideal parameter values are determined. With the primary goal of effectively modelling data, mixtures of Gaussian distributions have become a popular technique due to the isotropic character of the Gaussian distribution, or its ability to represent data using a mean and covariance.

Due to its symmetric nature around the mean and fixed form, the Gaussian distribution (GD) has its own restrictions. In reality, data may be non-Gaussian, and so a Gaussian mixture may have downsides. To overcome the limits of Gaussian distributions, a few researchers have fitted non-Gaussian data using the Generalized Gaussian distribution (GGD) [1,28]. In comparison to the GD, the GGD has an additional parameter (β) that influences the distribution's form. As the value of β increases, the curve becomes flatter, and vice versa. While the GGD is more flexible than the GD, it remains a symmetric distribution around the mean and is hence ineffective at fitting non-symmetric data. As a result, mixtures of asymmetric Gaussian distributions (AGD) [32, 33, 37–40] have been proposed to fit non-symmetric data by dividing the variance parameter (σ) of the GD into left and right variance. It has the same constraints on the shape of the distribution as GGD, except that GGD tackles only the constraint of GD's rigidity of shape, whereas AGD addresses GD's inability to fit asymmetric data. Finally, the research presented in [30] succeeded in proposing an asymmetric generalised Gaussian distribution (AGGD) that overcomes both of the GDs's limitations by adding three additional parameters $\beta, \sigma_l, \sigma_r$ to provide shape flexibility and the ability to fit asymmetric data [13]. The parameters σ_l, σ_r define the skewed nature of the peak, while β determines its kurtosis and can be used to make it sharp or flat by adjusting its value. With this flexibility, proposed model is superior for fitting asymmetric data and this Bayesian framework effectively estimates model parameters, as demonstrated in the remainder of the thesis through demanding real-world applications.

Later, following [16], a completely Bayesian framework for an asymmetric generalised Gaussian mixture model (AGGMM) is employed in light of the Reversible Jump (RJMCMC) algorithm, which permits changing the number of states from one iteration to the next in response to a change in the dimension of the parameter space. Additionally, the approach enables estimation of the joint posterior distribution for the total number of states and all parameters. Later in the study, feature selection [62] is incorporated into the model to optimise the learning process by minimising the time complexity by choosing relevant features from high dimensional data sets.

In the thesis, hidden Markov models (HMM) which are advanced statistical models that employ a Markov process for system modelling are explored. These procedures make the assumption that an observable series of data is dependent on hidden patterns and attempt to infer them from visible observations. HMMs are effective in speech recognition applications as well as in any other type of sequence or time series analysis. Due to the fact that HMMs are typically used to represent dependent heterogeneous events, they are applied in a variety of fields, including econometrics [44], biology [49], genetics [27], speech processing [53], and finance [59]. For this research, [58] and [56] are followed to develop a Bayesian approach incorporating mixture models in learning and employ

RJMCMC making it a fully Bayesian learning framework.

1.1 Contributions

The contributions of this thesis are as follows:

- ☞ **A Bayesian Sampling Framework for Asymmetric Generalized Gaussian Mixture Models Learning**:

  In the first study in chapter 2, a novel unsupervised Bayesian framework for the learning of a finite mixture of asymmetric generalised Gaussian distributions (AGGD) is proposed. The parameters are computed using a hybrid Markov Chain Monte Carlo (MCMC) technique that combines Metropolis-Hastings and Gibbs sampling. By introducing new parameters, this research aims to increase the modelling flexibility of the Gaussian distribution and its capacity to fit asymmetric and non-Gaussian data. To evaluate the suggested model's performance, multiple experiments are done and the model is evaluated on a variety of real-world applications. The model's performance is evaluated using a variety of evaluation methodologies and compared to the performance of benchmark models in order to determine its goodness of fit. This work [68] was published in the *Neural Computing and Applications: Special Issue on Emerging trends in Artificial Intelligence and Machine Learning*.

- ☞ **Model Selection and Feature Selection in Asymmetric Generalized Gaussian Mixtures**:

  In the second part of chapter 2, an advanced unsupervised Bayesian framework for learning a finite mixture of asymmetric generalised Gaussian distributions (AGGD) using Markov Chain Monte Carlo (MCMC) is proposed, which incorporates Metropolis-Hastings within Gibbs sampling for estimating mixture parameters and reversible jump MCMC (RJMCMC) to dynamically select the number of model components M. To facilitate learning, feature selection methods that weight features according to their importance were devised. Numerous experiments are conducted to assess the model's performance in a range of real-world scenarios. The performance of the proposed model is compared to that of other benchmark models in order to assess the proposed learning approach's importance. This work was submitted to

IEEE IES ICIT 2022: 23rd IEEE International Conference on Industrial Technology.

☞ **Bayesian Inference of Hidden Markov Models using Dirichlet Mixtures**:

In this work from the chapter 3, an effective unsupervised learning strategy for Hidden Markov Model (HMM) learning using a Bayesian framework is presented and a novel method for parameter estimation based on Markov Chain Monte Carlo (MCMC) sampling is demonstrated. Additionally, reversible jump MCMC (RJMCMC) is built to highlight how dynamic model selection can be used to boost modelling capabilities. The Dirichlet mixture model is integrated into HMM as a critical phase in the learning process in order to enhance its modelling power. Experiments were performed with the proposed model utilising real-world video and audio data to demonstrate its effectiveness in comparison to other benchmark models. This work is accepted as a book chapter in: *Hidden Markov Models and Applications*.

1.2 Thesis Overview

❑ In chapter 1, machine learning, various approaches to machine learning, existing models, and the proposed models are briefly introduced. Then the contributions are described.

❑ In chapter 2, Bayesian learning, finite mixtures, proposed asymmetric generalized Gaussian mixture model (AGGMM) and its MCMC based learning algorithm: Metropolis-Hastings within Gibbs sampling are described in detail. Moreover, experiments and discussions about the proposed model's performance on various real world applications are mentioned. In addition to MCMC, a fully Bayesian approach to AGGMM via reversible jump MCMC is introduced. Finally, feature selection employed.

❑ In chapter 3, HMMs and their learning via a Bayesian approach is introduced: RJMCMC is proposed for parameter estimation and model selection making it a fully Bayesian learning framework. Mixture modelling approach in the proposed model is introduced by using a Dirichlet mixture model as part of the learning algorithm. As part of RJMCMC, the split/combine and birth/death moves responsible for dynamic model selection in HMM are explored in detail. Finally, experiments are conducted on the presented learning approach with well-known image and video data sets and evaluate the results to display the robustness of the proposed model in comparison to other benchmark models.

❑ Finally, chapter 4 outlines and summarises the thesis and suggests areas for future exploration.

Chapter 2

A Bayesian Sampling Framework for Asymmetric Generalized Gaussian Mixture Models Learning

This chapter presents a unique unsupervised Bayesian framework for recognition tasks. Here we introduce and elaborate our proposed work on AGGMM. We explain our model and its learning approach which is based on a hybrid MCMC: Metropolis-Hastings within Gibbs sampling. We also showcase the performance of our model by experimenting it on various real world applications. We evaluate and record the results and compare them with other base models. Furthermore, we also propose a full Bayesian approach using RJMCMC for AGGMM and also incorporate feature selection into our models learning to improve its performance. Finally, we perform experiments using our model in this framework with similar applications as before and evaluate its performance with benchmark models.

2.1 Finite Asymmetric Generalized Gaussian Mixture Model

2.1.1 Finite Mixture Model

Mixture models are probabilistic models that aim to fit the data with clusters that follow different or same probabilistic distributions and these are often represented by mixtures of M components, where each component fits a cluster of the whole data. A d-dimensional random variable $\vec{X} = [X_1, ..., X_d]^T$ is said to be following a mixture of M components if its probability density function takes following form:

$$p(\vec{X}|\Theta) = \sum_{j=1}^{M} p_j p(\vec{X}|\xi_j) \tag{1}$$

where ξ_j is the set of parameters of mixture distributions for jth component, p_j are the mixing probabilities, which are always positive and sum to 1. Θ, the set of all parameters is represented as $\Theta = \{p_1, ..., p_M, \xi_1, ..., \xi_M\}$. Here, M represents the number of components in the mixture and is always greater or equal to 1. For our case, we assume that the data follows a mixture of AGGD which we call AGGMM. In AGGMM, each component follows an AGGD whose density is represented as $p(\vec{X}|\xi_j)$ and given as follows [30] :

$$p(\vec{X}|\xi_j) = \prod_{k=1}^{d} \begin{cases} \frac{\beta_{jk}\left[\frac{\Gamma(3/\beta_{jk})}{\Gamma(1/\beta_{jk})}\right]^{1/2}}{(\sigma_{l_{jk}}+\sigma_{r_{jk}})\Gamma(1/\beta_{jk})} \exp\left[-A(\beta_{jk})\left(\frac{\mu_{jk}-X_k}{\sigma_{l_{jk}}}\right)^{\beta_{jk}}\right] & \text{if} \quad X_k < \mu_{jk} \\ \frac{\beta_{jk}\left[\frac{\Gamma(3/\beta_{jk})}{\Gamma(1/\beta_{jk})}\right]^{1/2}}{(\sigma_{l_{jk}}+\sigma_{r_{jk}})\Gamma(1/\beta_{jk})} \exp\left[-A(\beta_{jk})\left(\frac{X_k-\mu_{jk}}{\sigma_{r_{jk}}}\right)^{\beta_{jk}}\right] & \text{if} \quad X_k \geq \mu_{jk} \end{cases} \tag{2}$$

where $A(\beta_{jk}) = \left[\frac{\Gamma(3/\beta_{jk})}{\Gamma(1/\beta_{jk})}\right]^{\beta_{jk}/2}$ and $\Gamma(.)$ is the Gamma function given by $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}\, dt$, $x > 0$. Here $\xi_j = (\vec{\mu}_j, \vec{\beta}_j, \vec{\sigma}_{l_j}, \vec{\sigma}_{r_j})$ are the set of parameters for component j, each given as, mean $\vec{\mu}_j = (\mu_{1j}, ..., \mu_{jd})$, left standard deviation, $\vec{\sigma}_{l_j} = (\sigma_{l_{j1}}, ..., \sigma_{l_{jd}})$ and right standard deviation $\vec{\sigma}_{r_j} = (\sigma_{r_{j1}}, ..., \sigma_{r_{jd}})$ of a d-dimensional AGGD. The parameter $\vec{\beta}_j = (\beta_{j1}, ..., \beta_{jd})$ controls the shape of the probability density function (PDF) and determines whether the peak of the curve is pointed or flat with a smaller $\vec{\beta}_j$ resulting in a sharper peak. The AGGD is chosen because of its flexible nature to fit symmetric or asymmetric data by adjusting left and right standard deviations. Using Eq. 2, the PDF of each observation is computed and later used to compute the posterior probabilities of the observation to belong to the different clusters.

For the finite asymmetric generalized Gaussian mixture [13], let $\mathcal{X} = (\vec{X}_1, ..., \vec{X}_N)$ be a set of N independent vectors which are identically distributed and follow a finite AGGMM with M components. Their likelihood can be represented as:

$$p(\mathcal{X}|\Theta) = \prod_{i=1}^{N} \sum_{j=1}^{M} p_j p(\vec{X}_i|\xi_j) \tag{3}$$

where the set of parameters of the mixture with M components is given by $\Theta = (\vec{\mu}_1, ..., \vec{\mu}_M, \vec{\sigma}_{l_1}, ..., \vec{\sigma}_{l_M}, \vec{\sigma}_{r_1}, ..., \vec{\sigma}_{r_M}, \vec{\beta}_1, ..., \vec{\beta}_M, p_1, ..., p_M)$.

As an important step in mixture modelling, we produce an M-dimensional membership vector [30] $\vec{Z}_i$ for each data point $\vec{X}_i \in \mathcal{X}$, such that $\vec{Z}_i = (Z_{i1}, ..., Z_{iM})$ in which entries are defined as [40]:

$$Z_{ij} = \begin{cases} 1 & \text{if } \vec{X}_i \text{ belongs to } j^{th} \text{ component} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

which means that for an observation $\vec{X}_i$, $Z_{ij} = 1$ only for one of the M components according to the highest assignment probability and $Z_{ij} = 0$ for the rest of the $M - 1$ components. Therefore, the complete probability density function can be obtained by combining Eq. 3 and Eq. 4 and is given as follows:

$$p(\mathcal{X}, Z|\Theta) = \prod_{i=1}^{N} \sum_{j=1}^{M} (p_j p(\vec{X}_i|\xi_j))^{Z_{ij}} \tag{5}$$

2.1.2 Bayesian Learning

The Bayesian approach in machine learning has been a well known technique for more than a decade [21, 22, 29]. It mainly respects Bayes's theorem [65] in order to produce a posterior distribution by taking into account prior information and information available from a data set. Prior information is our belief about the unknown parameters before considering the data and the posterior distribution summarizes our assumptions about the parameters after considering and analyzing the data. For our case, we use the Markov Chain Monte Carlo [2] technique for generating Bayesian estimates. Since the main goal is to find the posterior distribution $\pi(\Theta|\mathcal{X}, Z)$ [28] by combining the prior information about the unknown parameters $\pi(\Theta)$ with analyzed data $p(\Theta|\mathcal{X}, Z)$ the Bayes formula can be given as follows [30]:

$$\pi(\Theta|\mathcal{X}, Z) = \frac{\pi(\Theta)p(\mathcal{X}, Z|\Theta)}{\int_\Theta \pi(\Theta)p(\Theta|\mathcal{X}, Z)d\Theta} \propto \pi(\Theta)p(\mathcal{X}, Z|\Theta) \tag{6}$$

Here, $\mathcal{X}$ and Z together represent the complete data. Using a combination of Metropolis Hastings [25] and Gibbs sampling [42, 43] techniques we can estimate the unknown parameters and thereby find its posterior distribution.

Bayesian Learning Algorithm for AGGMM

We use MH-within-Gibbs [19] learning for AGGMM. The algorithm will be summarized at the end of this section. First, we simulate Z with the posterior distribution $\pi(Z|\Theta, \mathcal{X})$ from a Multinomial distribution of order 1 as $Z_i^{(t)} \sim \mathcal{M}(1; \hat{Z}_{i1}^{(t-1)}, ..., \hat{Z}_{iM}^{(t-1)})$. Following [28] this can be represented as follows:

$$Z^{(t)} \sim \pi(Z|\Theta^{(t-1)}, \mathcal{X}) \tag{7}$$

Then the number of observations assigned to the jth cluster can be calculated using $Z^{(t)}$ as:

$$n_j^{(t)} = \sum_{i=1}^{N} Z_{ij} (j = 1, ..., M) \tag{8}$$

So, the number of observations allocated to the M components can be represented by $n^{(t)} = (n_1^{(t)}, ..., n_M^{(t)})$. Since the mixture weights satisfy the conditions $(0 < p_j \leq 1$ and $\sum_{j=1}^{M} p_j = 1)$, the natural choice for the prior distribution is the Dirichlet distribution which satisfies the same condition:

$$\pi(p_1, ..., p_M) \sim \mathcal{D}(\delta_1, ..., \delta_M) \sim \frac{\Gamma(\sum_{k=1}^{M} \delta_k)}{\prod_{k=1}^{M} \Gamma(\delta_k)} \prod_{k=1}^{M} p_k^{\delta_k - 1} \tag{9}$$

where δ_j is a known hyper-parameter. The posterior distribution of the mixing weights p_j is given as follows by taking into account the number of observations per cluster, obtained from Eq. 8 and Eq. 7. It is also sampled from a Dirichlet distribution because it is a conjugate prior:

$$p(p_1, ..., p_M|Z^{(t)}) \sim D(\delta_1 + n_1^{(t)}, ..., \delta_M + n_M^{(t)}) \tag{10}$$

We use the Metropolis-Hastings [25] method to sample the other parameters from the respective proposal distributions since direct sampling can be a very complicated process. So, we choose a proposal distribution $\xi^{(t)} \sim q(\xi^{(t)}|\xi^{(t-1)})$ for each parameter μ, σ_l, σ_r and β as follows [40]:

$$\mu_j^{(t)} \sim \mathcal{N}_d(\mu_j^{(t-1)}, \Sigma), \quad \beta_j^{(t)} \sim \Gamma_d(\alpha_j^{(t-1)}, \Delta) \sim \frac{\Delta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\Delta x} \tag{11}$$

$$\sigma_{l_j}^{(t)} \sim \mathcal{N}_d(\sigma_{l_j}^{(t-1)}, \Sigma), \quad \sigma_{r_j}^{(t)} \sim \mathcal{N}_d(\sigma_{r_j}^{(t-1)}, \Sigma) \sim \frac{1}{\Sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\sigma}{\Sigma}\right)^2\right) \tag{12}$$

For the parameters μ, σ_l, σ_r, the proposal distributions are d-dimensional Gaussian distributions with Σ as $d \times d$ identity matrix. For β the proposal distribution is given by a d-dimensional Gamma distribution with Δ as a $d \times d$ identity matrix and follows the PDF in Eq. 12. As a key feature of the MH, for every iteration, the parameters set $\pi(\Theta)$ needs to be updated with a new set of parameters based on an acceptance ratio r. It is the value of r that decides whether to accept and update the parameters or reject and discard them. So, r needs to be computed after every iteration and expressed as:

$$r = \frac{p(\mathcal{X}|\Theta^{(t)})\pi(\Theta^{(t)})q(\Theta^{(t-1)}|\Theta^{(t)})}{p(\mathcal{X}|\Theta^{(t-1)})\pi(\Theta^{(t-1)})q(\Theta^{(t)}|\Theta^{(t-1)})} \tag{13}$$

Here, $p(\mathcal{X}|\Theta^{(t)})$ is the PDF of AGGM model at iteration t and $\pi(\Theta)$ is the prior distribution of the parameters that can be represented as d-dimensional Gaussian distributions for the parameters μ, σ_l, σ_r as $\mu \sim \mathcal{N}_d(\eta, \Sigma)$ and $\sigma_l, \sigma_r \sim \mathcal{N}_d(\tau, \Sigma)$ with known hyper-parameters η, τ and d-dimensional Gamma distribution for β as $\beta \sim \Gamma_d(\gamma, \Delta)$. With the assumption that mixture parameters are independent of each other the derivation of $\pi(\Theta)$ is simplified as follows:

$$\pi(\Theta) = \pi(p, \xi) = \pi(\xi) = \prod_{j=1}^{M} \pi(\mu_j)\pi(\sigma_{l_j})\pi(\sigma_{r_j})\pi(\beta_j) \tag{14}$$

$$= \prod_{j=1}^{M} \mathcal{N}_d(\mu_j|\eta, \Sigma)\mathcal{N}_d(\sigma_{l_j}|\tau, \Sigma)\mathcal{N}_d(\sigma_{r_j}|\tau, \Sigma)\Gamma_d(\alpha_j|\gamma, \Delta)$$

Mixture weight p in Eq. 14 is avoided because it is generated from a Gibbs sampling method with an acceptance probability always equal to 1 with the same rule applied to the proposal distribution as follows:

$$\begin{aligned} q(\Theta^{(t)}|\Theta^{(t-1)}) &= q(\xi^{(t)}|\xi^{(t-1)}) \\ &= \prod_{j=1}^{M} \mathcal{N}_d(\mu_j^{(t)}|\mu_j^{(t-1)}, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t)}|\sigma_{l_j}^{(t-1)}, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t)}|\sigma_{l_j}^{(t-1)}, \Sigma)\Gamma_d(\alpha_j^{(t)}|\alpha_j^{(t-1)}, \Delta) \end{aligned} \tag{15}$$

Eq. 13 can be expanded as follows:

$$\begin{aligned} r &= \frac{p(\mathcal{X}|\Theta^{(t)})\pi(\Theta^{(t)})q(\Theta^{(t-1)}|\Theta^{(t)})}{p(\mathcal{X}|\Theta^{(t-1)})\pi(\Theta^{(t-1)})q(\Theta^{(t)}|\Theta^{(t-1)})} \\ &= \prod_{i=1}^{N}\prod_{j=1}^{M}\left(\frac{p(X_i|\mu_j^{(t)}, \sigma_{l_j}^{(t)}, \sigma_{r_j}^{(t)}, \beta_j^{(t)})}{p(X_i|\mu_j^{(t-1)}, \sigma_{l_j}^{(t-1)}, \sigma_{r_j}^{(t-1)}, \beta_j^{(t-1)})}\right) \\ &\times \frac{\mathcal{N}_d(\mu_j^{(t)}|\eta, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t)}|\tau, \Sigma)\mathcal{N}_d(\sigma_{r_j}^{(t)}|\tau, \Sigma)\Gamma_d(\alpha_j^{(t)}|\gamma, \Delta)}{\mathcal{N}_d(\mu_j^{(t-1)}|\eta, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t-1)}|\tau, \Sigma)\mathcal{N}_d(\sigma_{r_j}^{(t-1)}|\tau, \Sigma)\Gamma_d(\alpha_j^{(t-1)}|\gamma, \Delta)} \\ &\times \frac{\mathcal{N}_d(\mu_j^{(t-1)}|\mu_j^{(t)}, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t-1)}|\sigma_{l_j}^{(t)}, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t-1)}|\sigma_{l_j}^{(t)}, \Sigma)\Gamma_d(\alpha_j^{(t-1)}|\alpha_j^{(t)}, \Delta)}{\mathcal{N}_d(\mu_j^{(t)}|\mu_j^{(t-1)}, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t)}|\sigma_{l_j}^{(t-1)}, \Sigma)\mathcal{N}_d(\sigma_{l_j}^{(t)}|\sigma_{l_j}^{(t-1)}, \Sigma)\Gamma_d(\alpha_j^{(t)}|\alpha_j^{(t-1)}, \Delta)} \end{aligned} \tag{16}$$

Once we compute the acceptance ratio r from Eq. 16, we can calculate the acceptance probability with $\alpha = min[1, r]$. Then, we randomly sample $u \sim \mathcal{U}_{[0,1]}$, if $\alpha < u$, the proposed move is accepted and the parameters are updated to $p^{(t)}$ and $\xi^{(t)}$. Otherwise we discard the parameters $p^{(t)}$ and $\xi^{(t)}$, and retain the parameters of the previous iteration $p^{(t)} = p^{(t-1)}, \xi^{(t)} = \xi^{(t-1)}$. The entire MH-within-Gibbs learning for AGGMM can be summarized as follows:

Input: Observations $\mathcal{X}$ witn M number of components

Output: AGGMM parameter set

(1) Initialization

(2) Step at time t: For t=1......n

Gibbs sampling part

- Generate $Z^{(t)}$ from Eq. 7
- Generate $n_j^{(t)}$ from Eq. 8
- Generate $p_j^{(t)}$ from Eq. 9

Metropolis-Hastings part

- Sample $\xi_j^{(t)}(\mu_j^{(t)}, \sigma_{l_j}^{(t)}, \sigma_{r_j}^{(t)}, \beta_j^{(t)})$ from Eqs. 11, 12
- Compute acceptance ratio r from Eq. 16
- Generate $\alpha = min[1, r]$ and $u \sim \mathcal{U}_{[0,1]}$
- if $u \geq \alpha$ then $\xi^{(t)} = \xi^{(t-1)}$

2.2 Experimental Results

In this section, the Bayesian AGGMM is applied on real data sets corresponding to various applications in human activity recognition, vehicle recognition, pedestrian detection and malaria detection. In order to compare the performance of our model, other relevant Bayesian mixture models like the Gaussian Mixture Model (BGMM) and Asymmetric Gaussian Mixture Model (BAGMM), with the same number of components, are considered. Then, clustering is performed and the results are evaluated and compared using multiple performance measures.

2.2.1 Experimental Setup

All the applications chosen to evaluate the performance of our model involve data in the form of images and videos, so it is necessary to have relevant computer vision setup to translate an image or a video into a model-understandable format using feature extraction and representation techniques. In our applications, we have two types of setups, one for the videos and the other for the images.

The video and image setups differ in one additional step, namely converting the video data to images by extracting frames out of it. The remaining setup is as follows: first, each image is loaded and converted to grey scale to reduce the complexity of the image. This is done when the colors in the image are not relevant in our task and then a feature extraction technique is selected, which in our case is SIFT (Scale Invariant Feature Transform) [50] to extract required features from the image followed by the construction of Bags Of Visual Words (BoVW) [70]. The implementation of BoVW is an extension of Bag Of Words (BoW) as in [72] for images where each column represents an extracted feature similar to a word in BoW.

Generally, a BoVW is constructed by applying clustering algorithms with K known clusters, where each cluster represents a common feature in the set of images considered. K-Means with 9 clusters, is the clustering algorithm in our case. Finally, the generated vectors of features are given as an input to the clustering framework developed with proposed model. The model parameters are initialized using Gaussian mixtures [55] and are iterated for specific numbers of iterations chosen for the application. For each iteration, our learning algorithm is applied and some parameters are accepted while others are rejected according to the computed acceptance ratios.

We have selected 12 performance measures [66] to benchmark the performance of Bayesian AGGMM (BAGGMM) with other relevant Bayesian models like the Bayesian Asymmetric Gaussian Mixture Model (BAGMM) and the Bayesian Gaussian Mixture Model (BGMM). The performance measures include, accuracy: the proportion of all correctly predicted labels, precision: proportion of positive predictions that are actually correct, recall: the probability of actual positives that were identified, the F1 score: the harmonic mean of precision and recall, balanced accuracy: the arithmetic mean of the true positive and negative rates, log loss: cross-entropy loss and the Jaccard score: a similarity indicator that measures the closeness of the predicted and actual labels, the G-mean 1: the geometric mean of sensitivity and precision, the G-mean 2: the geometric mean of sensitivity and specificity, the false positive rate: the probability of falsely rejecting the null hypothesis, specificity: a measure of the ability of the model to recognize negative examples, the Matthews correlation coefficient (MCC): the correlation coefficient used to measure the quality of the binary classification. Here, balanced accuracy is also used along with accuracy measure because balanced accuracy works well in the case of imbalanced data and for binary classification. For the convergence of our learning algorithm, since it is a sampling technique, we consider the last 30% of the accepted moves and the average of the estimated parameters and compute the performance measures with the averaged parameters after the model fitting.

2.2.2 Human Activity Recognition

Human activity recognition is an important application, for getting machines to identify human activities and utilize this knowledge in many other applications across many domains. Here, our goal is to cluster human activities using the experimental framework mentioned in the previous

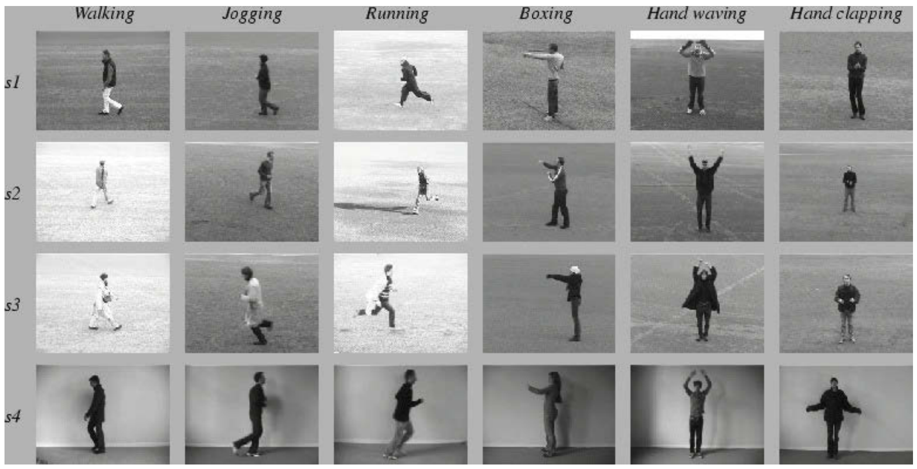

Figure 2.1: KTH data set

section. The data for this application are videos. We use two well-known data sets, KTH [60] and UCF 101 [63]. Each contains different kinds of human activities. In our experiment, we have considered two groups of activities from each data set: indoor and outdoor activities.

Indoor activity recognition

As of today, we find almost every home, school or office equipped with a camera in order to ensure safety and well-being. Our motivation for working on detecting indoor activities was to explore this possibility in using such footage to detect human activities which could further be used for various applications such as smart homes and living, indoor behaviour analysis etc.

This experiment was aimed to cluster indoor activities of various types observed in common households and other enclosed places. The UCF101 [63] from Fig. 2.2, as the name suggests contain 101 human actions divided into 25 categories and each category has 4-7 videos of each action and the action categories can be divided as follows: Human-Object Interaction, Body-Motion Only, Human-Human Interaction, Playing Musical Instruments, Sports.

Here we have considered two indoor activities: baby crawling and cutting in kitchen, which are from the UCF101 data set [63]. In order to recognise the activities we selected 700 images for each class and extracted features from the videos of selected activities. The goal of this experiment was to know the capability of our model in understanding two different indoor activities and compare

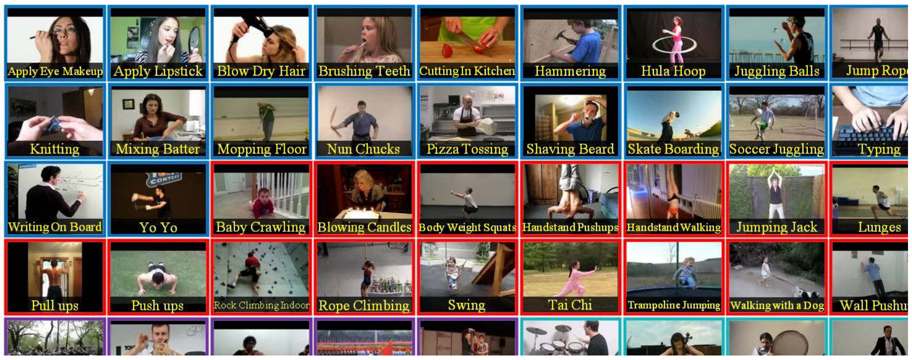

Figure 2.2: UCF data set

Table 2.1: Indoor Activity Recognition Results

Metric	Acc	Pre	Recall	F1	B.Acc	L.Los	Jaccard	FPR	G.M1	G.M2	MCC	Spec.
BAGGMM	84.00	87.03	83.71	86.20	83.71	5.52	75.75	30.61	85.35	76.21	70.66	69.38
BAGMM	76.66	76.61	76.61	75.52	76.61	8.05	66.67	22.29	76.61	77.15	53.23	77.79
BGMM	63.40	64.34	63.62	66.78	63.62	12.64	50.13	47.84	63.98	57.60	27.96	52.15

the clustering efficiency with other relevant models like BAGMM and BGMM. As part of the experiment the learning algorithm was applied to the extracted features for specific number for 1500 iterations and performance metrics mentioned in the previous sections were computed. The results of our model along with other models are listed in Table 2.1. Now, if we observe the scores for accuracy (84%), precision (87.03%) and recall (83.71%) in specific, we can understand the significance of the model and its effectiveness in correctly classifying the activities. Our proposed model performs better compared to BAGMM and BGMM for most of the performance measures and successfully clusters the data into relevant classes. Better performance in a clustering task through our proposed algorithm can help create more robust recognition applications in real-world scenarios.

Outdoor activity recognition

Clustering human activities out of doors can be of great importance in several ways and our interest is to know how recognizing human activities can benefit the current way of handling situations. The main requirement of any object-recognizing problem is to have well captured images or video feed.

For outdoor activities two actions namely, boxing and hand waving with 700 images in each

Table 2.2: Outdoor Activity Recognition Results

Metric	Acc	Pre	Recall	F1	B.Acc	L.Los	Jaccard	FPR	G.M1	G.M2	MCC	Spec.
BAGGMM	92.50	92.86	92.92	92.46	92.92	2.59	85.98	13.08	92.89	89.86	85.78	86.91
BAGMM	82.33	85.47	83.02	84.08	83.02	6.10	72.53	37.84	84.34	84.24	75.22	68.35
BGMM	79.00	81.02	79.25	81.08	79.25	7.25	68.18	33.31	80.13	72.68	60.25	66.66

class were selected from the well known KTH data set. The KTH data set [60] in Fig. 2.1 contains 6 types of human activities: walking, jogging, running, boxing, hand waving and hand clapping. Each of these activities are performed several times by 25 people in different scenarios having similar backgrounds with a static camera with 25 frames per second (fps). Each sequence has a spatial resolution of 160×120 pixels and is 4 seconds long on an average.

In the same way as the previous section, the extracted data from each class are passed through the proposed model. In order to know the efficiency of our model as well as the BAGMM and BGMM in distinguishing outdoor human activities, the learning algorithm was applied for 2000 iterations . The performance measures were computed with estimated parameters and listed in Table 2.2. As we can see from the results in Table 2, our model performed exceptionally well in recognizing different activities compared to the benchmark models. Specific measures like log loss (2.59) and FPR (13.08) from Table 2.2 indicate low error rates and therefore demonstrate our model's high performance compared to BAGMM and BGMM in a similar experimental setting. The proposed model has thereby successfully demonstrated its effectiveness in clustering the video data into particular activities, which opens the door to further apply this approach in real-world scenarios for recognition and AI-assisted surveillance in an unsupervised manner.

2.2.3 Surveillance System

Surveillance systems have been an interesting topic ever since the development of surveillance video cameras. Mainly capturing traffic and street footage for security, safety and maintenance reasons have been a key focus of interest. Due to the growing surveillance applications in many areas around the globe, there has been a drastic reduction in criminal activities [69] such as theft, vandalism etc. It also created a scope for the development of remote monitoring, alerting and immediate responding solutions. With the development of computer vision and learning technologies, object

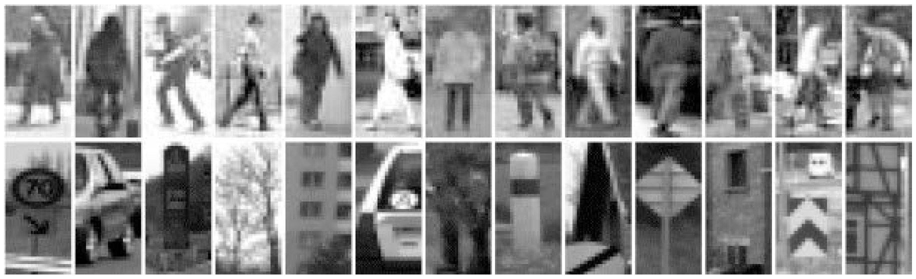

Figure 2.4: Pedestrian data set

detection and tracking have also been key focus areas in all surveillance related applications.

In many recent developments relating to road transportation, footage from surveillance cameras can be used to identify, track and count vehicles used in traffic analysis. Many malls and public places are using surveillance footage to detect, track and count people to know the number of visitors and visiting times used in shopping analysis. Here, we choose to validate our model with vehicle and pedestrian recognition tasks that play an important role in object tracking and related applications.

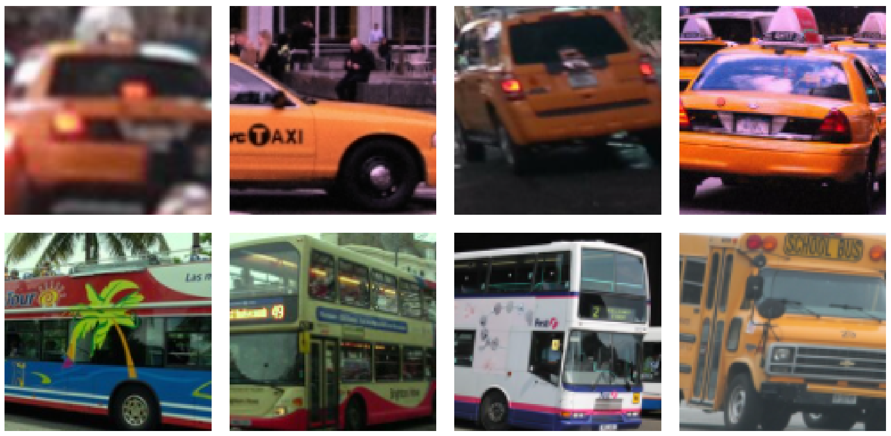

Figure 2.3: Vehicle data set

Vehicle Recognition

In this section, our aim is to test our model's ability to identify different types of vehicles on the road. For this task we have considered images of different types of vehicles from the TAU vehicle type recognition competition [47]. The data set contains vehicles of 17 categories, and for our experiment 700 images of taxis and busses displayed in Fig. 2.3 were selected. Features were

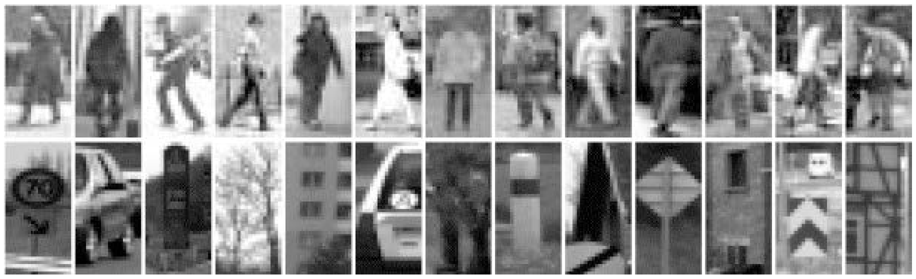

Figure 2.4: Pedestrian data set

detection and tracking have also been key focus areas in all surveillance related applications.

In many recent developments relating to road transportation, footage from surveillance cameras can be used to identify, track and count vehicles used in traffic analysis. Many malls and public places are using surveillance footage to detect, track and count people to know the number of visitors and visiting times used in shopping analysis. Here, we choose to validate our model with vehicle and pedestrian recognition tasks that play an important role in object tracking and related applications.

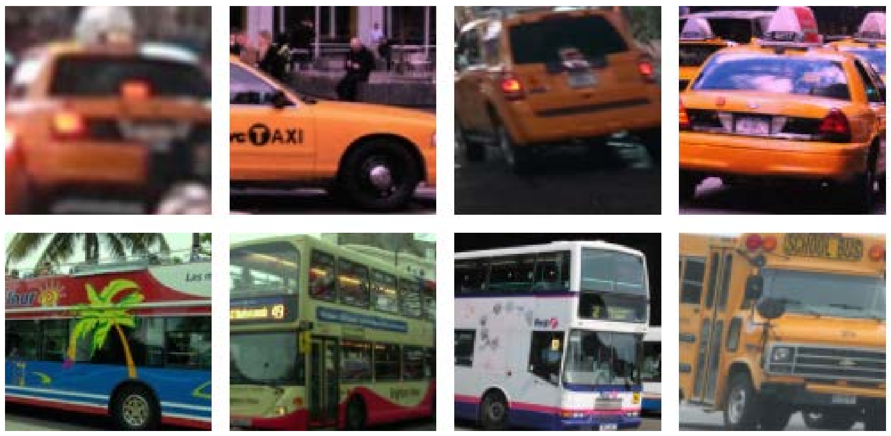

Figure 2.3: Vehicle data set

Vehicle Recognition

In this section, our aim is to test our model's ability to identify different types of vehicles on the road. For this task we have considered images of different types of vehicles from the TAU vehicle type recognition competition [47]. The data set contains vehicles of 17 categories, and for our experiment 700 images of taxis and busses displayed in Fig. 2.3 were selected. Features were

Table 2.3: Vehicle Recognition Results

Metric	**Acc**	**Pre**	**Recall**	**F1**	**B.Acc**	**L.Los**	**Jaccard**	**FPR**	**G.M1**	**G.M2**	**MCC**	**Spec.**
BAGGMM	79.20	73.15	79.01	79.20	79.01	7.25	65.57	20.40	79.00	79.30	58.01	79.59
BAGMM	76.45	76.68	75.83	78.18	75.83	7.25	64.18	32.65	76.25	71.42	52.50	67.34
BGMM	71.51	72.02	70.46	74.33	70.76	10.01	59.15	40.81	71.39	64.17	42.77	59.18

extracted using the previously described experimental setup and the BoVW was generated for each selected vehicle. The importance of this task can be observed in extended applications such as vehicle tracking, traffic analysis and vehicle categorization that might be done in smart surveillance systems.

The learning algorithm is applied with 2,500 iterations on the BoVW. Upon convergence, the estimated parameters are used to analyse the efficiency of the clustering compared to benchmark models. The performance measures were computed for each model and are listed in Table 2.3. From the results in Table 2.3, it can be observed that our model is better at recognizing vehicles compared to BAGMM and BGMM. Observing the closeness of balance accuracy (79.01%) and Accuracy (79.20%), the classes are well balanced. The F1 score (79.01%) summarizes the effectiveness of the classification. From all the performance measures, it is observed that BAGGMM has demonstrated its efficacy in clustering the vehicle images as compared to BAGMM and BGMM. The proposed approach has a potentially significant impact on real-world recognition applications in transportation. The success of the proposed approach in this unsupervised learning application can be applied to improve the performance of existing recognition systems further.

Pedestrian Recognition

Pedestrian recognition is a very important application, which can improve the performance of many surveillance and monitoring systems. Here, we tried to validate using pedestrian data, with the aim of identifying pedestrians and distinguishing between their presence and absence.

The pedestrian data set mentioned in Fig. 2.4 [52] contains a collection of 4,000 pedestrian and 5,000 non-pedestrian, cut out from videos and scaled to a size of 18×36 pixels. For this data set, as the images were very small to extract features, we scaled the image 2 times and applied a sharpen and blur filter to improve the feature extraction. 700 images from each class were selected

Table 2.4: Pedestrian Recognition Results

Metric	Acc	Pre	Recall	F1	B.Acc	L.Los	Jaccard	FPR	G.M1	G.M2	MCC	Spec.
BAGGMM	73.00	75.26	72.68	76.92	72.68	9.32	62.50	42.81	73.96	64.44	47.88	57.14
BAGMM	69.00	64.20	69.04	69.30	69.04	10.70	53.03	33.33	69.05	67.05	38.18	66.66
BGMM	64.00	63.98	63.98	64.70	63.98	12.43	47.82	36.73	63.90	63.62	27.97	63.26

and fed to the model after feature extraction. The proposed learning algorithm is then applied for 2500 iterations and the best parameters selected after convergence.

In order to understand the capability and evaluate the effectiveness of our model, 12 performance measures were computed. They are listed in Table 2.4. From the results in Table 2.4, it can be observed that the Jaccard score (62.50%), the G-mean 1 (73.96%), and all other measures indicate that our model recognized pedestrians better than BAGMM and BGMM. The effectiveness of our proposed model in pedestrian recognition highlights the fact that a surveillance and monitoring system can improve its recognition ability by adopting BAGGMM. The results also showcase acceptable levels of confidence in clustering. Hence, this application can be further extended to address other use cases.

2.2.4 Malaria Detection

Medical diagnosis using machine learning has been a very useful contribution in the medical domain. Our motivation in conducting this experiment was to test the validity of our model in correctly diagnosing malaria infection from medical imagery. Here we choose a malaria data set to categorize a blood cell as malaria-parasitized or uninfected based on the images of thin blood slide images. The data set [54] mentioned in Fig. 2.5 contains a total of 27,558 cell images equally labelled as parasitized or uninfected. A sample of the data set is considered with 5,000 images in each class and fed to the model after feature extraction and the learning algorithm is applied 2500 iterations and the best parameters are selected after convergence.

In order to benchmark our model, we run the same setup for BAGMM and BGMM. After convergence, we computed the performance measures to evaluate the efficiency of our model with estimated parameters of our model and the benchmark models. The results are listed in Table 2.5. Here the goal of our model is to identify infected cells from normal cells. The scores of G-mean

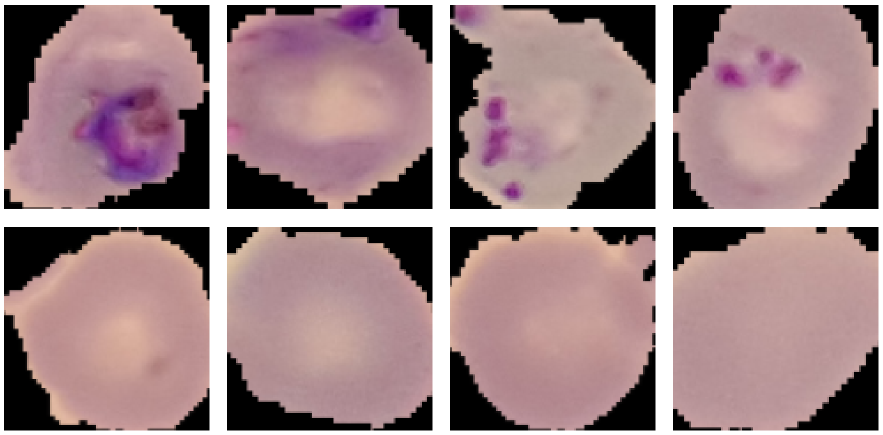
Figure 2.5: Malaria data set

Table 2.5: Malaria Detection Results

Metric	Acc	Pre	Recall	F1	B.Acc	L.Los	Jaccard	FPR	G.M1	G.M2	MCC	Spec.
BAGGMM	84.125	85.93	83.58	86.24	83.58	5.48	75.80	27.82	84.75	77.67	69.48	72.17
BAGMM	78.24	83.97	77.51	82.39	77.51	7.51	70.06	43.38	8.67	66.24	61.14	56.61
BGMM	72.25	72.14	72.18	73.75	72.18	9.58	58.42	28.27	72.16	71.65	44.32	71.1

2 (77.67%) and Specificity (72.17%) indicate the preciseness in classifying negative and positive examples. The experimental results presented in Table 2.5 demonstrate a clear picture of the success of BAGGMM in clustering the normal and infected cells as compared to BAGMM and BGMM in a similar set of experiments. As such, it could be applied in assisting medical practitioners in the diagnosis process. It must be noted that this framework is developed in an unsupervised manner, and it does not require any label during the learning process, which is a great success in this experimental setting and application.

Based on the model's results and analysis and flexibility to fit data of different kinds, the proposed model can also be considered for other applications where the distribution of data follows Gaussian or is close to Gaussian. Therefore, we can clearly state that AGGMM can fit the data in a better way. Hence we can consider applying the model to a wide range of data and expect good results.

2.3 Reversible Jump Markov Chain Monte Carlo

The standard MH-within-Gibbs approach implies that the number of components M is fixed and constant during the learning process, which limits the AGGM model's ability to adapt to new situations. However, because of poor initialization or leaking of information, M may be wrong or unknown. The RJMCMC algorithm has its advantages in this situation since it provides an additional four independent phases (birth/death steps and combine/split steps) into the learning process, which could modify component number M, and thus, adds more generalities into the learning process.

2.3.1 Split and Combine Moves

In RJMCMC a state at a given point of time is choosing to split or combine with probabilities b_m and $d_m = 1 - b_m$ respectively. As $d_1 = b_{mMax} = 0$. We use $b_m = d_m = \frac{1}{2}$ for $M = 2, 3, ..., M_{max} - 1$. Since we consider M as one of the parameters of the model we choose uniform distribution as its prior: $p(\kappa) \sim U$. For our case we initially assume $\kappa = 4$.

In combine move we suppose the current state of MCMC algorithm is $\tilde{x}$ with σ_l, σ_r and β etc., as parameters with a total of $m + 1$ states. Then we randomly choose a pair (j_1, j_2) which are two adjacent states, and combine them to a single new state $j\prime$ resulting in an MCMC with x states and m components. For the combine move, the parameters are updated according to Eqs. 17, 18, 19, 20.

$$pj\prime = pj_1 + pj_2 \tag{17}$$

$$pj\prime\mu_{j\prime} = pj_1\mu_{j_1} + pj_2\mu_{j_2} \tag{18}$$

$$pj\prime(\mu_{j\prime}^2 + \sigma_{j\prime l}^2) = pj_1(\mu_{j_1}^2 + \sigma_{j_1 l}^2) + pj_2(\mu_{j_2}^2 + \sigma_{j_2 l}^2) \tag{19}$$

$$pj\prime(\mu_{j\prime}^2 + \sigma_{j\prime r}^2) = pj_1(\mu_{j_1}^2 + \sigma_{j_1 r}^2) + pj_2(\mu_{j_2}^2 + \sigma_{j_2 r}^2) \tag{20}$$

In split move, with three degrees of freedom $(u_1 \sim Beta(2, 2), u_2 \sim Beta(2, 2),$

$u_3 \sim Beta(1,1))$, we split state j' into two (j_1 and j_2), and so increase m to $m+1$ in the reversal of the combining step. As a result, Eq. 21 can be used to calculate the mixture parameters for split components:

$$\begin{aligned}
pj_1 = pj'u_1, pj_2 = pj'u_2 \\
\mu_{j_1} = \mu_{j'} - \tfrac{u_2\left(\sigma_{j'l}+\sigma_{j'r}\right)}{2}\sqrt{\tfrac{pj_2}{pj_1}} \\
\mu_{j_2} = \mu_{j'} + \tfrac{u_2\left(\sigma_{j'l}+\sigma_{j'r}\right)}{2}\sqrt{\tfrac{pj_1}{pj_2}} \\
\sigma^2_{j_1 l} = u_3\left(1-u_2^2\right)\sigma^2_{j'l}\tfrac{pj'}{pj_1} \\
\sigma^2_{j_1 r} = u_3\left(1-u_2^2\right)\sigma^2_{j'r}\tfrac{pj'}{pj_1} \\
\sigma^2_{j_2 l} = \left(1-u_3\right)\left(1-u_2^2\right)\sigma^2_{j'l}\tfrac{pj'}{pj_2} \\
\sigma^2_{j_2 r} = \left(1-u_3\right)\left(1-u_2^2\right)\sigma^2_{j'r}\tfrac{pj'}{pj_2}
\end{aligned} \tag{21}$$

The acceptance probability [56,58] decides whether or not to accept the combine or split move. The following formula can be used to calculate the acceptance probability:

$$\begin{aligned}
\mathcal{A} = & \frac{p\left(\mathcal{X}, Z \mid \Theta'\right)}{p(\mathcal{X}, Z \mid \Theta)} \frac{m'\mathcal{P}\left(m' \mid \kappa\right)}{\mathcal{P}(m \mid \kappa)} \frac{p_{j_1}^{\delta-1+n_1} p_{j_2}^{\delta-1+n_2}}{p_{j'}^{\delta-1+n_1+n_2}\operatorname{Beta}(\delta, m\delta)} \\
& \times \frac{d_{m'}}{b_m P_{alloc}}\left[\operatorname{Beta}\left(u_1 \mid 2,2\right)\operatorname{Beta}\left(u_2 \mid 2,2\right)\operatorname{Beta}\left(u_3 \mid 1,1\right)\right]^{-1} \\
& \times \frac{p_{j'}\left|\mu_{j_1}-\mu_{j_2}\right| \sigma^2_{j_1 l}\sigma^2_{j_1 r}\sigma^2_{j_2 l}\sigma^2_{j_2 r}}{u_2\left(1-u_2^2\right) u_3\left(1-u_3\right)\sigma^2_{j'l}\sigma^2_{j'r}}
\end{aligned} \tag{22}$$

where Θ and $m' = m+1$ signify the mixture parameters set and the component number prior to or following split steps, respectively. Additionally, P_{alloc} is the likelihood that this particular allocation will occur. As a result, the acceptance probability for the combine step is $min(1, A)$, while the acceptance probability for the split step is $min(1, A^{-1})$.

2.3.2 Birth and Death Moves

In comparison to combine and split moves, birth and death steps are very simple because the newborn and dead components are empty, eliminating the requirement for parameter recalculation. The weight of the mixture p_{new} in the birth step may be determined by sampling from the Beta distribution $p_{new} \sim Beta(1, m)$, and the mixture parameters can be computed as follows [56]:

$$\mu \sim \mathcal{N}_d(\eta, \Sigma) \qquad \sigma_l, \sigma_r \sim \mathcal{N}_d(\tau, \Sigma) \qquad \beta \sim \Gamma_d(\gamma, \Delta) \tag{23}$$

For the death step, a random empty component should be chosen and deleted from among the existing components, if any. If there are no empty components, this step will be skipped. Following the birth and death steps, the mixture weights p_j should be rescaled so that they add to 1. Acceptance probability is also necessary for birth and death phases, as well as combine and split steps, which are defined as follows:

$$\begin{aligned} \mathcal{A}' = & \frac{\mathcal{P}(m' \mid \kappa)}{\mathcal{P}(m \mid \kappa)} \frac{1}{\text{Beta}(m\delta, \delta)} p_{j'}^{\delta-1} \times \left(1 - p_{j'}\right)^{N+m\delta-m} \\ & \times m' \frac{d_{m'}}{(m_0+1)\, b_m} \frac{1}{\text{Beta}\left(p_{j'} \mid 1, m\right)} \times \left(1 - p_{j'}\right)^{m} \end{aligned} \tag{24}$$

where m_0 is the number of unused components. Thus, the odds of birth and death are $min(1, A')$ and $min(1, A'^{-1})$ respectively.

2.4 Feature Selection

The AGGM model stated in Eq. 2 implies that all of an observation's d features have the same weight of relevance and include essential information, which is not necessarily the case, and many of those features may be irrelevant for clustering purposes. To address this issue and establish the relevance and significance of characteristics, feature selection strategies [12, 20, 31, 51] should be considered. By indicating background Gaussian distributions for every d features with parameter $\Psi = \{\mu'_1, \dots, \mu'_d, \sigma'_1, \dots, \sigma'_d\}$, where μ' and σ' denote the mean and standard deviation of the distribution, respectively. Then, Eq. 1 can be reformulated with the feature relevancy approach suggested in [15, 31, 36, 38] as follows:

$$p(\mathcal{X} \mid \Theta, \Psi, \Phi) = \prod_{i=1}^{N} \sum_{j=1}^{M} p_j \prod_{k=1}^{d} p\left(X_{ik} \mid \xi_{jk}\right)^{\phi_k} \mathcal{N}\left(X_{ik} \mid \psi_k\right)^{1-\phi_k} \tag{25}$$

where $\mathcal{N}(X_{ik} \mid \psi_k)$ is the background distribution, and $\psi_k = (\mu'_k, \sigma'_k)$ are the background distribution's parameters $\Phi = (\phi_1, \ldots, \phi_d)$ is a binary relevance vector in which ϕ_k equals to 1 if the k^{th} characteristic is significant and $\phi_k = 0$, otherwise. If we take the relevancy vector Φ to be a latent variable, the following is the whole likelihood function of the AGGM model with the full parameter set:

$$p(\mathcal{X} \mid \Theta') = \prod_{i=1}^{N} \sum_{j=1}^{M} p_j \prod_{k=1}^{d} \left[\omega_k p(X_{ik} \mid \xi_{jk}) + (1-\omega_k) \mathcal{N}(X_{ik} \mid \psi_k)\right] \tag{26}$$

where $\Theta' = (\Theta, \Psi, \Omega)$ and $\Omega = (\omega_1, ..., \omega_d)$ is the relevancy weight with value range of $0 \leq \omega_d \leq 1$ which denotes the probability that k^{th} feature is important. Finally, relevancy weight [14] ω_k is given as follows

$$\omega_k = \frac{\prod_{i=1}^{N} \sum_{j=1}^{M} p_j p(X_{ik} \mid \xi_{jk})}{\prod_{i=1}^{N} \sum_{j=1}^{M} p_j p(X_{ik} \mid \xi_{jk}) + \prod_{i=1}^{N} \mathcal{N}(X_{ik} \mid \psi_k)} \tag{27}$$

As a result, irrelevant features contribute minimally to the clustering process, extending the AGGM model's applicability to more common and challenging instances such as high-dimensional noisy applications.

The full Bayesian learning approach with feature selection can be summarized as follows:

Input: Observations $\mathcal{X}$ witn M number of components

Output: AGGMM parameter set

(1) Initialization

(2) Step at time t: For t=1......n

Gibbs sampling part

- Generate $Z^{(t)}$ from Eq. 7
- Generate $n_j^{(t)}$ from Eq. 8
- Generate $p_j^{(t)}$ from Eq. 9

Metropolis-Hastings part

- Sample $\xi_j^{(t)}(\mu_j^{(t)}, \sigma_{l_j}^{(t)}, \sigma_{r_j}^{(t)}, \beta_j^{(t)})$ from Eqs. 11, 12

- Calculate relevancy weight $\omega_k^{(t)}$ from Eq. 27
- Generate background Gaussian parameters $\Psi_k^{(t)}$ by random walk
- Compute acceptance ratio r from Eq. 16
- Generate $\alpha = min[1, r]$ and $u \sim \mathcal{U}_{[0,1]}$
- if $u \geq \alpha$ then $\xi^{(t)} = \xi^{(t-1)}$

RJMCMC part

- Generate $u' \sim U_{[0,1]}$. If $b_m >= u'$, perform split or birth step, then calculate acceptance probability $\mathcal{A}$. If the step is accepted, set $m = m + 1$.
- Generate $u' \sim U_{[0,1]}$. If $d_m >= u'$, perform combine or death step, then calculate acceptance probability $\mathcal{A}'$. If the step is accepted, set $m = m - 1$.

2.5 Validation of the RJMCMC with feature selection approach

This section applies Bayesian AGGMM to real data sets for human activity recognition, vehicle recognition, pedestrian detection, and malaria detection. Other relevant Bayesian mixture models initially with the same number of components are explored to compare our model's performance. The outcomes of clustering are then compared using several performance measures.

2.5.1 Setup

All the applications are chosen to evaluate our model's performance using images and videos; therefore, a computer vision setup is required to convert an image or video into a vector of features using feature extraction and representation approaches. In this case, we use SIFT (Scale Invariant Feature Transform) ([50]) to extract the required features from the image, then we build Bags Of Visual Words (BoVW) [70] by applying clustering methods to K known clusters of images. Our clustering approach is K-Means with 9 clusters. Finally, the resulting feature vectors are fed into the proposed model's clustering architecture.

2.5.2 Human Activity Recognition

Human activity identification is a critical application because it enables machines to recognize human behaviors and use this information to a variety of other applications across a variety of areas. The purpose of this section is to cluster human activities using the proposed framework discussed previously. A video data set KTH [60] is used for this experiment. It contains a variety of human activities.

Two actions were chosen from the KTH data set: boxing and hand waving, each with 700 photos. We employed the setup mentioned previously and feed the resultant BoVW to our model. Then the model is run for a fixed number of iterations to complete learning. The performance of our model, BAGMM and BGMM, in identifying human activities are listed Table 2.6.

From the results in Table 2.6, by noticing the values of evaluated performance metrics, it can be observed that as M value drops, the model's accuracy drastically improves from 61.72% to 83.22%. The same can be observed in the case of all other performance metrics. This indicates that: as the model approaches the convergence point, it can predict the optimal number of clusters/components in the data. The high accuracy of the model indicates that the best parameters were estimated.

The results also showcase the performance of model selection and parameter estimation of the proposed model in comparison with the benchmark models: BAGM and BGMM.

Table 2.6: Human activity recognition results

States	BAGGMM				BAGMM				BGMM			
M	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
4	61.72	62.96	61.72	54.70	57.77	58.93	57.77	64.21	52.05	52.19	52.05	57.46
3	75.33	75.55	75.33	74.12	66.77	68.81	66.77	60.23	59.83	65.20	59.83	42.84
2	83.22	85.72	83.22	85.18	74.83	82.13	74.83	79.67	67.05	68.86	67.05	61.01

2.5.3 Automobile Recognition

The purpose of this section is to evaluate our model's capacity to recognize various types of automobiles on the road. We used photos of automobiles like buses and taxis from the TAU vehicle type recognition competition [47] for this challenge.

The previously mentioned experimental setup was used to extract features, and the BoVW was constructed for each selected car.

The BoVW, in this case, contains 800 images of each of both automobiles. The learning algorithm is applied to the BoVW for a fixed number of iterations. The performance of the model learning is evaluated using four metrics: accuracy, precision, recall, F1. The values of which for the proposed model and other models are recorded in Table 2.7.

Observing the values of evaluated performance measures in Table 2.7, it can be seen that when the M value decreases, the model accuracy improves dramatically, from 49.07% to 82.92%. The same may be said for all other performance indicators. This means that when the model gets closer to convergence, it can estimate the best number of components in the data. And the model's excellent accuracy indicates that the best parameters were estimated.

Compared to the benchmark models BAGM and BGMM, the results reveal that the suggested model is more efficient in model selection and parameter estimation.

Table 2.7: Vehicle recognition results

States	BAGGMM				BAGMM				BGMM			
M	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
4	49.07	49.07	49.87	49.32	47.28	47.21	47.28	42.79	43.52	28.12	43.92	43.34
3	72.35	72.38	72.35	71.89	64.74	66.65	64.14	69.97	61.71	65.92	61.71	69.54
2	82.92	83.97	82.92	81.28	72.35	72.38	72.35	71.89	66.42	74.16	66.42	53.18

2.5.4 Walker Recognition

Pedestrian recognition is a critical application that has the potential to significantly improve the performance of numerous surveillance and monitoring systems.

The pedestrian data set [52] contains 4,000 pedestrian and 5,000 non-pedestrian images extracted from movies and scaled to an 18x36 pixel resolution. After feature extraction, 700 images from each class were chosen and fed into the model. The proposed learning algorithm is run for a fixed number of iterations to determine optimal parameters.

In order to understand the capability and evaluate the effectiveness of our model, four performance measures were computed. They are listed in Table 2.8.

By examining the values of evaluated performance metrics in Table 2.8, it is clear that when the

M value decreases, the model's accuracy improves dramatically, increasing from $63.2\%$to 78.7%. The same is true for all other performance metrics. This indicates that the model is capable of predicting the ideal number of clusters in the data as it approaches the convergence point. The model's excellent accuracy indicates that the optimal parameters were estimated.

Additionally, the results demonstrate the proposed model's efficiency in model selection and parameter estimation compared to two benchmark models: BAGM and BGMM.

Table 2.8: Pedestrian Recognition Results

States	BAGGMM				BAGMM				BGMM			
M	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
4	63.2	79.01	63.22	70.83	56.81	55.29	56.81	39.21	47.53	47.22	47.51	39.74
3	72.64	72.72	72.64	66.83	68.85	72.97	68.88	53.04	61.71	65.92	61.71	69.54
2	78.79	80.95	78.76	79.13	74.66	79.63	74.66	62.77	66.63	74.19	66.62	44.17

2.5.5 Malaria Detection

Machine learning-based medical diagnosis has made a significant contribution to the medical realm. The purpose of this experiment was to determine the validity of our model's ability to correctly diagnose malaria infection using medical images.

In this section, we use a malaria data set to classify blood cells as malaria-parasitized or uninfected based on images of thin blood slide images. The data set [54] described comprises a total of 27,558 images of parasitized and uninfected cells. After feature extraction, a sample of the data set is evaluated with 5,000 images in each class.

The learning technique is applied for a fixed number of iterations, and the best parameters are to be selected after convergence. Four performance indicators were computed to have a better understanding of our model's capabilities and efficacy. They are described in detail in Table 2.9.

By studying the results of the analyzed performance metrics in Table 2.9, it is evident that when the M value falls, the model's accuracy increases significantly, increasing from 63.11% to 82.11%. Similarly, this is true for all other performance indicators. This demonstrates that the model is capable of predicting the optimal number of clusters in the data as it nears convergence. Additionally, the model's great accuracy implies that the model's parameters were approximated to their ideal values.

In addition, the results illustrate the proposed model's efficiency compared to two benchmark models: BAGM and BGMM in terms of model selection and parameter estimation.

Table 2.9: Malaria detection results

States	BAGGMM				BAGMM				BGMM			
M	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
4	62.11	64.44	62.11	68.45	56.55	59.91	56.55	38.71	50.88	68.40	50.88	66.99
3	74.83	82.13	74.83	79.67	67.55	68.86	67.05	61.02	62.83	70.61	62.83	71.56
2	82.11	82.14	82.11	82.40	74.83	82.13	74.83	79.67	75.33	75.55	75.33	74.12

Chapter 3

Bayesian Inference of Hidden Markov Models using Dirichlet Mixtures

Here, we present HMMs and their learning using a Bayesian approach: RJMCMC is presented for parameter estimation and model selection, making it a fully Bayesian learning framework. We use the Dirichlet mixture model as part of the learning process to incorporate the mixture modelling approach in our proposed model. We investigate in depth the split/combine and birth/death moves that are responsible for dynamic model selection in HMM [34, 35] . Finally, we run experiments using well-known image and video data sets using the provided learning approach and assess the results to demonstrate the robustness of our model in contrast to existing benchmark models.

3.1 The Learning Model

In this section, we present the proposed model, which is a combination of various components, and we elaborate on each component and its contribution to the parameter estimation and model selection processes. As we proceed, the following sections and subsections will give more details about our modeling approach.

3.1.1 The Bayesian Model

As previously mentioned in chapter 2, we choose to implement a Bayesian approach [57] in our modeling, in which θ represents a vector of parameters describing the model. For a given data set y, Bayes' theorem is:

$$p(\theta|y) \propto p(y|\theta)p(\theta) \tag{28}$$

where $p(y|\theta)$ is the likelihood and $p(\theta)$ is the prior distribution of the parameter set. Later on, in this section, we discuss in detail the prior distributions of our parameter set and the complete hierarchical model, which is the heart of the proposed modeling approach where the joint probability is computed.

3.1.2 Mixture Model

Mixture model, presented in chapter 2, Eq. 1 can be represented by a d-dimensional random variable $\vec{y} = [y_1, ..., y_d]^T$ with k components as follows:

$$\phi(\vec{y}; \Theta) = \sum_{j=1}^{k} \pi_i \phi(\vec{y_t}; \xi_i) \tag{29}$$

where $\xi_i = \{\mu_i, \alpha_i\}$ is the set of parameters for the ith component, π_i are the mixing probabilities, which are always positive and sum to 1. $\mu_i = (\mu)_{i=1}^{k}$ is the mean and $\alpha_i = (\alpha)_{i=1}^{k}$ is the sharpness parameter. Θ, the set of all parameters is represented as $\Theta = \{\pi_1, ..., \pi_k, \xi_1, ..., \xi_k\}$. Here, k represents the number of components in the mixture and is always greater or equal to 1.

Using this approach the prior distributions for μ_i and α_i are defined in the next sections. $\phi(y_t; \mu_i, \alpha_i)$ in Eq. 29 is the density of the Dirichlet distribution:

$$\phi(y_t; |\boldsymbol{\alpha}\ |, \mu) = \frac{\Gamma(|\boldsymbol{\alpha}|)}{\prod_{i=1}^{k} \Gamma\left(\mu_i |\boldsymbol{\alpha}|\right)} \prod_{i=1}^{k} y_i^{\mu_i|\alpha|-1} \tag{30}$$

Where $\sum_{i=1}^{d} y_i = 1$ and $|\boldsymbol{\alpha}| = \sum_{i=1}^{k} \alpha_i,\ \alpha_i > 0\ \forall i = 1, \ldots, k$. Given α the mean and variance of the Dirichlet distribution can be given as follows:

$$\mu_i = E(y_i) = \frac{\alpha_i}{|\alpha|} \tag{31}$$

$$\sigma_i^2 = Var(y_i) = \frac{\alpha_i(|\alpha| - \alpha_i)}{|\alpha|^2(|\alpha| + 1)} \tag{32}$$

3.1.3 Hidden Markov Model

Given, $y = (y_t)_{t=1}^{T}$ which are the vectors of observations with respect to time T, HMMs assume that the distribution of each data point y_t depends on hidden states, which are unobserved and are denoted by s_t and can take values from 1 to k. The hidden variable $s = (s_t)_{t=1}^{T}$ is often called a "regime" or "state" - we adopt the former word throughout the rest of the chapter. In HMMs that follow the Markov chain property, it is assumed that the hidden state variable s_t always depends on past realizations of y and s as shown in Eq.33:

$$p(s_t = j | s_{t-1} = i) = a_{ij} \tag{33}$$

where a_{ij} is the element of a transition probability matrix denoted by $A = (a_{ij})$. A transition probability matrix A, where each row is a vector of stationary probabilities given by π and satisfies $\pi^{'} A = \pi^{'}$, and stationary probabilities decide the initial state of the model from which, with time, the state transition takes place. As it is assumed in HMMs that, for every state change s_t an observation y_t is noticed, which follows a marginal probability distribution given in Eq. 29. The same equation can also be represented as :

$$y_t | \pi, \mu, \alpha \sim \Sigma_{i=1}^{k} \pi_i \phi(y_t; \mu_i, \alpha_i) \tag{34}$$

Eq. 34 can also be expressed as follows involving s_t:

$$y_t | s, \pi, \mu, \alpha \sim \Sigma_{i=1}^{k} \pi_i \phi(y_t; \mu_{s_t}, \alpha_{s_t}) \tag{35}$$

Here, we assume that the number of components k (i.e., the number of states) is unknown and subject to inference and we can observe that for $k = 1$ the model in Eq. 34 reduces to a simple random walk with drift.

Prior distributions

In any Bayesian modeling approach [16, 57, 58], prior information is one's belief about an unknown quantity before considering any evidence about it. Usually, prior information would be a probability distribution describing the unknown parameters in a model. Since the prior information is a probability distribution describing a parameter, the parameters of such a prior distribution are called hyper-parameters. In our case, we have three prior distributions for three unknown parameters $(\mu_i, \alpha_i, a_{ij})$ of the model, found in Eqs. 36, 37 and 38 as follows:

$$\mu_i \sim \mathcal{D}(\delta_1, ..., \delta_k) \sim \frac{\Gamma(\sum_{j=1}^{k} \delta_j)}{\prod_{j=1}^{k} \Gamma(\delta_j)} \prod_{j=1}^{k} \mu_{ij}^{\delta_j - 1} \tag{36}$$

$$a_{ij} \sim \mathcal{D}(\eta_1, ..., \eta_k) \tag{37}$$

where the mean μ_i and each row of the transition probability matrix a_{ij} has a Dirichlet distribution as prior with $\delta = \{\delta_1, ..., \delta_k\}$ and $\eta = \{\eta_1, ..., \eta_k\}$ as the hyper-parameters. The sharpness parameter $\alpha = \{\alpha_1, ..., \alpha_k\}$ has an inverse Gamma as a prior as follows:

$$\alpha_i \sim |\alpha_i|^{-3/2} \exp\left(-1/(2|\alpha_i|)\right) \tag{38}$$

Complete hierarchical model

The joint probability distribution like in [58] for all the variables including their hyper-parameters can be represented according to Eq. 39 as follows:

$$p(k, A, \mu, \alpha, s, y) = p(k)p(A|k, \eta)p(\mu|\delta, k)p(\alpha)p(s|A)p(y|\mu, \alpha, s) \tag{39}$$

$$\text{where} \qquad p(s|A) = p(s_1|A) \prod_{t=2}^{T} p(s_t|s_{t-1}, A) \tag{40}$$

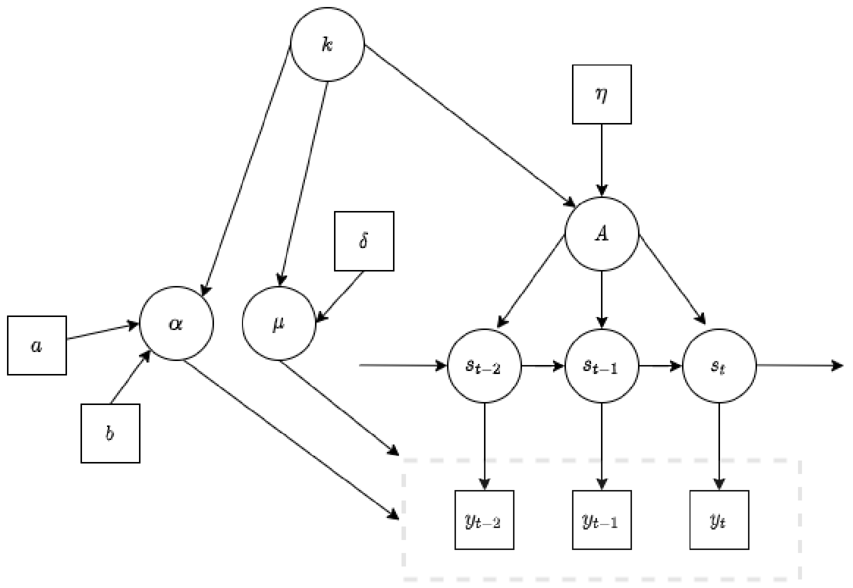

Figure 3.1: Directed acyclic graph for the complete hierarchical model

The term $p(s_t|s_{t-1}, A)$ from Eq. 40 is given by Eq. 33 and $p(s_1 = i|A) = \pi_i$, and from Eq. 39.

$$p(y|s, \mu, \alpha) = \prod_{t=1}^{T} \phi(y_t; \mu_{s_t}, \alpha_{s_t}) \tag{41}$$

Fig. 3.1 is a directed acyclic graph (DAG) representing the complete hierarchical model in which the usual convention is followed where the square boxes represent fixed or observed quantities and the circles represent the unknowns.

3.2 Markov Chain Monte Carlo Methodology

The mixture model [17] approach taken in this chapter is a fairly complex one and requires MCMC techniques to approximate the posterior distribution. A detailed description of these computational techniques can be found in [18, 58]. To get the realizations from the posterior joint distribution with all the parameters, we use the following moves at each sweep of the MCMC algorithm:

1. update the transition probability matrix A

2. update the allocations s

3. update the mean μ

4. update the sharpness parameter α

5. update standard deviation σ

6. consider split or combine moves

7. consider birth or death moves

3.2.1 Gibbs moves

Moves from (1 - 5) presented in the algorithm are called Gibbs moves and follow [58]. In move 1, the ith row of A is sampled from a Dirichlet distribution $D(\eta + n_{i1}, ..., \eta + n_{ik})$ where:

$$n_{ij} = \sum_{t=1}^{T-1} I\{s_t = i, s_{t+1} = j\} \tag{42}$$

is the number of jumps from state i to state j. In move 2, the state allocations $s_1, ...s_T$ are sampled one at a time from $t = 1, .., T$ by drawing new values from the fully conditional distribution in Eq. 43. For $t = 1$, the first factor is replaced by the stationary probability π_i, and for $t = T$, the rightmost factor is replaced by 1. Here $\phi(y_t; \mu_i, \alpha_i)$ is the density of a Dirichlet random variable with mean μ_i and sharpness parameter α_i.

$$p(s_t = i|S) \sim a_{s_{t-1}i}\phi(y_t; \mu_i, \alpha_i)a_{is_{t+1}} \tag{43}$$

where $S = \{s_1, ..., s_T\}$. In move 3, the mean μ_i is updated by sampling from a log normal distribution whose mean is the natural log of the mean from the previous iteration with a transformation $\mu^*_{il} = \log(\frac{\mu_{il}}{(1-\mu_{il})})$ and with Σ^2 as co-variance matrix where $\Sigma^2 = diag[0.01, ..., 0.01]$. The whole equation is represented in Eq. 44.

$$\mu_i^* \sim \mathcal{LN}(log(\mu_i^{*(t-1)}), \Sigma^2) \tag{44}$$

In move 4, α_i is updated in a similar fashion like μ_i with the only difference being where the parameters of the log-normal distribution are changed, and Σ^2 is replaced by σ^2 whose value is 0.01, the equation is given as Eq. 45. As α_i is updated, in move 5, σ is updated using the new values of α according to Eq. 32.

$$|\alpha_i| \sim \mathcal{LN}(log(|\alpha_i|^{t-1}), \sigma^2) \tag{45}$$

3.2.2 Split and Combine moves

In the above mentioned moves, 6 and 7 are considered as reversible jump MCMC moves which allow the number of components to increase or decrease by 1. In move 6, a state at a given point of time is chosen to split or combine with probabilities b_k and $d_k = 1 - b_k$, respectively. As $d_1 = b_{kmax} = 0$. We use $b_k = d_k = \frac{1}{2}$ for $k = 2, 3, ..., k_{max} - 1$.

In the combine move we suppose the current state of the MCMC algorithm is $\tilde{x}$ with $\tilde{a}_{ij}$ etc., as parameters with a total of $k + 1$ states. Then, we randomly choose a pair (j_1, j_2) which are two adjacent states, and combine them to a single new state j_* resulting in an MCMC with x states and k components.

For the combined move, the parameters are updated as follows Eqs. 46, 47, and 48:

$$\mu_{j^*} = \frac{\tilde{\pi}_{j_1}\mu_{j_1} + \tilde{\pi}_{j_2}\mu_{j_2}}{\tilde{\pi}_{j_1} + \tilde{\pi}_{j_2}} \tag{46}$$

$$\mu_{j^*}^2 + \sigma_{j^*}^2 = \frac{\tilde{\pi}_{j_1}(\mu_{j_1}^2 + \sigma_{j_1}^2) + \tilde{\pi}_{j_2}(\mu_{j_2}^2 + \sigma_{j_2}^2)}{\tilde{\pi}_{j_1} + \tilde{\pi}_{j_2}} \tag{47}$$

$$\begin{aligned} a_{j^*j} =& \frac{\tilde{\pi}_{j_1}}{\tilde{\pi}_{j_1} + \tilde{\pi}_{j_2}}\tilde{a}_{j_1j} + \frac{\tilde{\pi}_{j_2}}{\tilde{\pi}_{j_2} + \tilde{\pi}_{j_2}}\tilde{a}_{j_2j} \qquad \text{for } j \neq j*, \\ a_{ij_*} =& \tilde{a}_{ij_1} + \tilde{a}_{ij_2} \qquad \text{for } i \neq j*, \end{aligned} \tag{48}$$

and for any t with $\tilde{s}_t$ equal to $j_1 or j_2$, s_t is set to j_* and the remaining $\tilde{s}_t$ are simply copied. Similarly, a state j_* is selected at random in the split move and split into two new components j_1 and j_2.

In the split move, a state j_* is randomly selected and split into two new states j_1 and j_2. In the old representation, we assume that x is the current state with a total of k states and after the move is executed the system is represented with $\tilde{x}$ with a total of $k+1$ states. The goal of the split move is to split j_* in a way that the stationary probabilities for the chain of hidden states satisfy the following: $\tilde{\pi}_j = \pi_j \;\; for\ j \neq j_1, j_2, \tilde{\pi}_{j_1} = u_0 \pi_{j_*} \;\; and \;\; \tilde{\pi}_{j_2} = (1-u_0)\pi_{j_*}$. This can be achieved by sampling $u_0 \sim Be(2,2), u_j \sim Be(r,s) \; for\ each\ j \neq j_1, j_2$ and $v_i \sim Be(r,s) \; for\ each\ i \neq j_1, j_2$. The parameters of the Beta distribution r and s are given from Eq. 50. The transition probabilities $\tilde{A}$ after the split are updated according to the Eq. 49 where $K_i = \frac{\pi_i}{\pi_{j_*}}$:

$$\begin{aligned} &\tilde{a}_{j_1 j} = \frac{u_j}{u_0} a_{j_* j}, \qquad \tilde{a}_{j_2 j} = \frac{1-u_j}{1-u_0} a_{j_* j} \qquad \text{for } j \neq j_1, j_2, \\ &\tilde{a}_{i j_1} = v_i a_{ij*}, \qquad \tilde{a}_{ij_2} = (1-v_i) a_{ij_*} \qquad \text{for } i \neq j_1, j_2, \\ &\qquad \tilde{a}_{j_1 j_2} = u_1 \left(1 - \sum_{i \neq j_*} \frac{u_j}{u_0} a_{j_* j}\right), \\ &\tilde{a}_{j_2 j_1} = \left\{(1-u_1) \sum_{j \neq j_*} u_j a_{j_* j} + u_0 u_1 - \sum_{i \neq j_*} K_i v_i a_{ij_*}\right\} \Big/ (1-u_0) \end{aligned} \tag{49}$$

$$\begin{aligned} r &= \frac{1-u_0(1+c^2)}{c^2}, \quad s = r\frac{1-u_0}{u_0} \qquad if\ u_0 \leq \frac{1}{2} \\ s &= \frac{1-(1-u_0)(1+c^2)}{c^2}, \quad r = s\frac{u_0}{1-u_0} \qquad if\ u_0 > \frac{1}{2} \end{aligned} \tag{50}$$

Here c^2 is known as the squared coefficient of variation of the Beta distribution, and for the reasons mentioned in [58] we assume it to be $c^2 = 0.5$ for numerical result stability. We discuss u_1, which is the range for $\tilde{A}$: $[u_1^L, u_1^U]$ and is given as follows:

$$\begin{aligned} u_1^L &= max\left(1 - \frac{1 - \sum_{i \neq j_1, j_2} K_i/u_0 \times \tilde{a}_{ij_1}}{1 - \sum_{j \neq j_1, j_2} \tilde{a}_{j_1, j}}, 0\right), \\ u_1^U &= min\left\{1 - \frac{1 - \sum_{i \neq j_1, j_2} K - i/u_0 \times \tilde{a}_{ij_1} - (1-u_0)/u_0 \times (1 - \sum_{j \neq j_1, j_2} \tilde{a}_{j_2 j})}{1 - \sum_{j \neq j_1, j_2} \tilde{a}_{j_1, j}}, 1\right\} \end{aligned} \tag{51}$$

During the split move if $u_1^L > u_1^U$, it means that there is no valid range for $\tilde{A}$ and the move is rejected. If $u_1^L < u_1^U$, the move is not rejected and we can get the u_1 by $u_1 \sim u_1^L + (u_1^U - u_1^L)Be(1,1)$.

In the split move after splitting new parameters μ_{j_*}, σ_{j_*} are computed as follows:

$$\begin{aligned} \tilde{\mu}_{j_1} &= \mu_{j_*} - z_1\sigma_{j_*}\sqrt{\frac{\tilde{\pi}_{j_2}}{\tilde{\pi}_{j_1}}}, \quad \tilde{\mu}_{j_2} = \mu_{j_*} - z_1\sigma_{j_*}\sqrt{\frac{\tilde{\pi}_{j_1}}{\tilde{\pi}_{j_2}}} \\ \tilde{\sigma}^2_{j_1} &= z_2(1-z_1^2)\sigma^2_{j_*}\frac{\pi_{j_*}}{\tilde{\pi}_{j_1}}, \quad \tilde{\sigma}^2_{j_2} = (1-z_2)(1-z_1^2)\sigma^2_{j_*}\frac{\pi_{j_*}}{\tilde{\pi}_{j_2}} \end{aligned} \tag{52}$$

In this process of splitting, we make use of a two-dimensional random vector z which is sampled from a Beta distribution as $z_1 \sim z_1^U Be(1,1)$ and $z_2 \sim Be(1,1)$ and z_1^U is given in Eq. 53, which is the upper bound for z_1 and in which $\mu_i' s$ are properly sorted.

$$z_1^U = min\left\{\frac{\mu_{j_*} - \mu_{j_*-1}}{\sigma_{j_*}}\sqrt{\frac{\tilde{\pi}_{j_1}}{\tilde{\pi}_{j_2}}} \ , \ \frac{\mu_{j_*+1} - \mu_{j_*}}{\sigma_{j_*}}\sqrt{\frac{\tilde{\pi}_{j_2}}{\tilde{\pi}_{j_1}}} \ , \ 1\right\} \tag{53}$$

At last, we choose to reallocate the observation that belongs to $s_t = j_*$ before splitting to j_1 and j_2. We achieve this by using a restricted backward algorithm. Let us assume that $s_t = j_*$ for $t_1 \le t \le t_2$ with $s_{t_{1-1}} \ne j_*$ and $s_{t_2+1} \ne j_*$. Then we sample $\tilde{s}_{t_1}, ..., \tilde{s}_{t_2}$ one at a time from $t = t_1$ to $t = t_2$ with conditional probabilities given as follows:

$$p(\tilde{s}_t = j|\Delta) \sim \tilde{a}_{\tilde{s}_{t-1,j}}\phi(y_t; \tilde{\mu}_j, \tilde{\sigma}_j^2)b_t(i) \quad \text{for } j = j_1, j_2 \tag{54}$$

where Δ = $\{y, \tilde{s}_{t_1-1}, \tilde{s}_{t_1}, ..., \tilde{s}_{t_1+1} \in [j_1, j_2], ..., \tilde{s}_{t1} \in [j_1, j_2], \tilde{s}_{t_2+1}, \tilde{A}, \tilde{\mu}, \tilde{\sigma}\}$ and
$b_t(i) = p(y_{t+1}, ..., y_{t_2}, \tilde{s}_{t_1+1} \in [j_1, j_2], ..., \tilde{s}_{t1} \in [j_1, j_2], \tilde{s}_{t_2+1}|\tilde{s}_t = j, \tilde{A}, \tilde{\mu}, \tilde{\sigma})$ and for $j = j_1, j_2$

$$b_{t_2}(i) = \tilde{a}_{i,\tilde{s}_{t_2+1}} \tag{55}$$

For $t = t_2 - 1, ..., t_1$, $b_t(i)$ is given as follows:

$$b_t(i) = \sum_{j=i_1,i_2} b_{t+1}(j)\tilde{a}_{ij}\phi(y_{t+1}; \tilde{\mu}_j, \tilde{\sigma}_j^2) \tag{56}$$

when $t_1 = 1$, the $a_{\tilde{s}_{t-1},j}$ from Eq. 54 is replaced by $\tilde{\pi}_j$ which is the stationary probability and when $t_2 = T$, $\tilde{a}_{i,\tilde{s}_{t_2+1}}$ from Eq. 55 is replaced by 1.

As per the reverse jump algorithm, the acceptance probability of a split move is given as $min(1, R)$, and it is $min(1, R^{-1})$ for the combined move.

$$R = \frac{p(y|\tilde{s}, \tilde{\mu}, \tilde{\alpha})}{p(y|s, \mu, \alpha)} \times \frac{p(k+1)}{p(k)} \times \frac{p(\tilde{A}|k+1, \eta)}{p(A|k, \eta)} \times \frac{p(\tilde{s}|\tilde{A})}{p(s|A)} \times \frac{d_{k+1}}{b_k P_{\text{alloc}}} \times J \tag{57}$$
$$\times \Big[\frac{1}{z_1^U} g_{1,1}\Big(\frac{z_1}{z_1^U}\Big) g_{1,1}(z_2) g_{2,2}(u_0) \frac{1}{u_1^U - u_1^L} g_{1,1}\Big(\frac{u_1 - u_1^L}{u_1^U - u_1^L}\Big) \prod_j g_{r,s}(u_j) \prod_i g_{r,s}(v_i)\Big]^{-1}$$

where $g_{r,s}$ is the $Be(r, s)$ density, P_{alloc} is the probability of allocation for $\tilde{s}_t$ and J is the Jacobian determinant (explained in the Appendix).

3.2.3 Birth and Death Moves

Now we talk about birth and death moves as our final step in the RJMCMC algorithm. In this move, we randomly choose between birth and death with probabilities b_k and d_k, respectively. An empty state is selected at random in the death move among all the empty states and deleted. Then the remaining rows A are normalized, and the s_t is not changed.

In birth move the aim is to create a new state j_*. To do this we sample j_* row which will be a new row of A from a Dirichlet prior $D(\delta, ..., \delta)$. Then we draw v_i $i \neq j_*$ from $Be(1, k)$ and set:

$$\tilde{a}_{ij} = (1 - v_i)a_{ij} \qquad for j \neq j_*, \tag{58}$$
$$a_{ij_*} = v_i$$

The new parameters for this state are generated in the same way as previously mentioned, and the s_t remains untouched as the new state is empty. Similar to split and combine move the acceptance probability is computed to satisfy the rule of reversible jump where at any time t, the number of states can be increased or reduced. So, the acceptance probability for these moves is: $min(1, R)$

for birth and $min(1, R^{-1})$ for death.

$$R = \frac{p(k+1)}{p(k)} \times k^k \times \frac{p(\tilde{s}|\tilde{A})}{p(s|A)} \times (k+1) \times \frac{d_{k+1}}{b_k(k_0+1)} \Big\{ \prod_i g_{1,k}(v_i) \Big\}^{-1} \times J \tag{59}$$

where k_0 is the number of states before birth and J is the Jacobian determinant given by:

$$J = \sum_{i \neq j_*} (1 - v_i)^{k-1} \tag{60}$$

The entire MH-within-Gibbs learning for HMM-DMM can be summarized as follows:

Input: Observations $\mathcal{X}$ with k number of components

Output: HMM-DMM parameter set, k components

(1) Initialization

(2) Step at time t: For $t = \{1, ..., n\}$

Gibbs sampling part

- Generate s from Eq. (43)
- Generate $n_i j$ from Eq. (42)
- Generate a_{ij} from Eq. (37)

Metropolis-Hastings & RJMCMC

- Sample μ_i, α_i from Eq. (44), (45)
- compute σ_i from Eq. (32)
- compute acceptance ratio (R) for split and combine move from Eq. (57)
- compute acceptance ratio (R) for birth and death move from Eq. (59)

3.3 Experiments

In this section, we provide experiments to validate the proposed model with real-world applications. According to the literature [53], in many cases, HMMs are known to work well with sequential or time series data. In this chapter, we conducted experiments using our model with video and speech data sets which are both time-series in nature.

3.3.1 Human Activity Recognition

The outcome of this experiment is to recognize various human activities, cluster the appropriate activities, and check the appropriateness of the clustering process with various available metrics.

The motivation behind choosing this application as our clustering task is to highlight the importance of recognizing human activities in daily life and its applications in various real-life scenarios. Much learning and knowledge can be derived from this task. Its application can be further extended to trending research areas such as human behavior analysis, criminal activity recognition, gait recognition, etc.

We choose two well-known activity recognition data sets, namely: KTH [60] (See Fig. 3.2) and UCF101 [63] (See Fig. 3.3). Both data-sets contain human activities of different kinds. KTH contains actors performing six types of outdoor activities. Each of these activities is performed by 25 people in a similar background setup captured with a static camera with 25 frames per second (fps). Each image video sequence has a resolution of 160 x 120 pixels with an average length of 4 seconds. In UCF101, we have 101 human actions of 25 categories. Each action category has around 4-7 videos.

For our model to effectively cluster the human activities, we process the video data according to the following experimental setup: First, we extract the frames from the video using video prepossessing techniques, then a feature extraction technique called SIFT [50] is applied to the extracted frames which are images to generate BoVW (Bag of Visual Words).

For this experiment, we consider four actions from the KTH data-set: walking, jogging, running, and boxing. From UCF data set, we consider pull-ups, push-ups, swing, and haircut. To begin, a BoVW [70,72] is generated for each of these actions and fed to the model by combining BoVW for all the actions belonging to one data-set without disturbing the sequence, i.e., avoiding the shuffle. The model is then left to iterate until it converges, that is, until the average value of the latest batch of iterations for the parameters is approximately equal to zero or remains constant. Each iteration computes all seven stages of the MCMC algorithm, including the reversible jump, and the

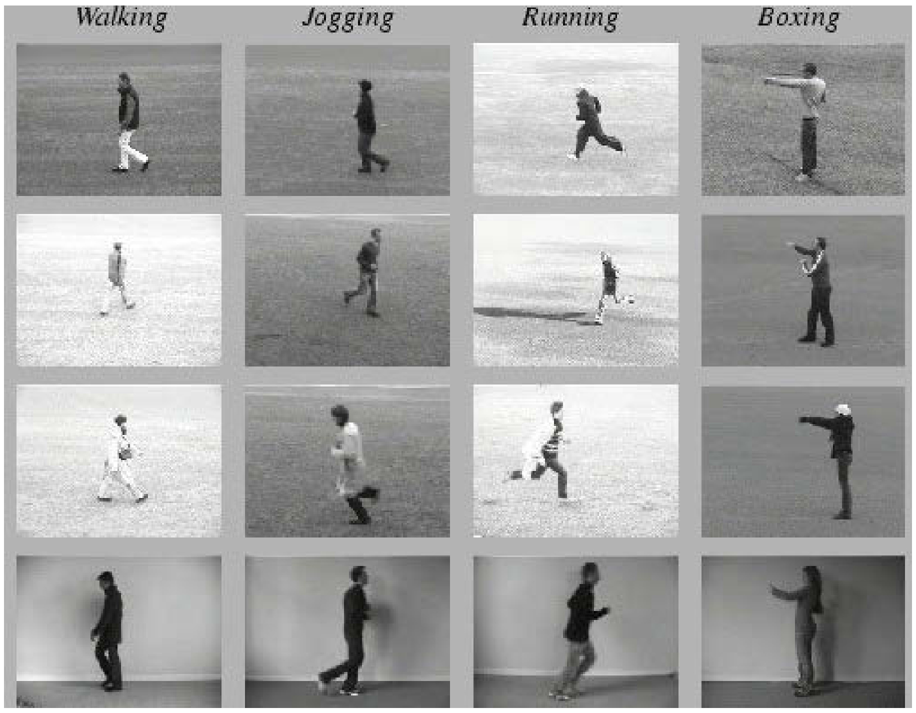

Figure 3.2: KTH data set

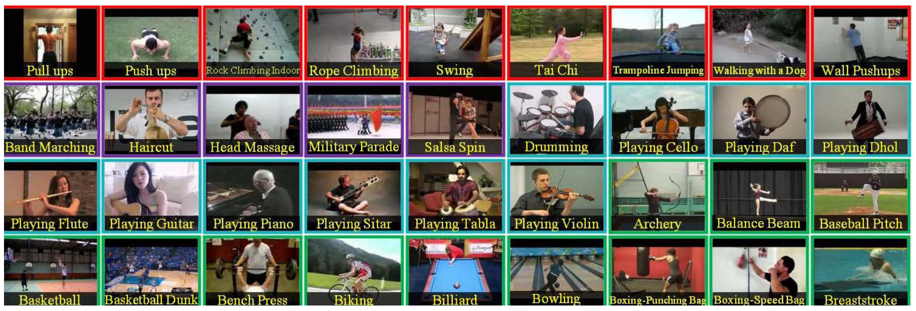

Figure 3.3: UCF data set

parameters are used to evaluate the model's performance using the four performance metrics listed in Table 3.1 [5, 10]. Then for the same BoVW, we use HMM with GMM [55] and standard normal distributions as base models to compare our results.

In Tables 3.1 & 3.2, we have displayed the relevant results of our model for the learning task of activity recognition. As can be observed, we started the model with two states and left the learning model to figure out the right number of states. After many iterations, it is clearly visible that our model could arrive at the correct number of states with the best parameters and produce better results than the other benchmark models. From Tables 3.1 & 3.2, it can be seen that our model score for a number of states gradually increases from a non-optimal number of states to the optimal number of states, with the following accuracies: 49.07% and: $72.94\%, 82.92\%$ and $54.94\%, 67.27\%, 83.22\%$ respectively for KTH and UCF activities. It can also be observed that our model outperforms the benchmark models in both cases.

Table 3.1: Activity Recognition with KTH

States	HMM-DMM				HMM-GMM				HMM-Std.Norm			
K	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
2	49.07	49.07	49.87	49.32	47.28	47.21	47.28	42.79	43.52	28.12	43.92	43.34
3	72.35	72.38	72.35	71.89	64.74	66.65	64.14	69.97	61.71	65.92	61.71	69.54
4	82.92	83.97	82.92	81.28	72.35	72.38	72.35	71.89	66.42	74.16	66.42	53.18

Table 3.2: Activity Recognition with UCF

States	HMM-DMM				HMM-GMM				HMM-Std.Norm			
K	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
2	54.94	61.30	54.94	67.23	52.55	74.61	52.55	67.79	49.55	47.44	49.55	65.31
3	67.27	79.47	67.27	75.24	62.72	76.15	62.72	72.55	59.27	59.34	59.27	57.40
4	83.22	85.72	83.22	85.18	75.88	78.93	75.88	79.25	68.94	72.62	68.94	61.09

3.3.2 Speaker Recognition

Speaker recognition is the task of automatically detecting the speaker by exploiting the speaker-specific information included in speech waves to validate the identities claimed by persons accessing systems; in other words, it enables voice access control of various services. Voice dialing, banking over a telephone network, telephone shopping, database access services, information and reservation services, voice mail, security control for private information, and remote computer access are all applicable services. Another key use for speech recognition technology is as a forensic tool. Speaker recognition also has several significant advantages over other types of identity identification, such as iris scans, facial recognition, and fingerprint scans. To begin, because all phones have microphones, it is commonly utilized for verification on mobile phones. Second, it is inexpensive to incorporate into other devices like home appliances and automobiles; third, because of the rapid proliferation of IoT devices, it is convenient and familiar to most users. Finally, it has been demonstrated to be extremely accurate in some conditions.

The goal of this experiment is to cluster and identify various voices in a speech sample. In this process, we take several steps to make the speech into a machine-understandable format to be fed to the model.

According to the literature, HMMs have proved their prominence in efficiently processing and clustering speech data on multiple occasions. This is our main motivation to experiment with speech data. Another reason to work with speech data is to showcase the learning efficiency of the model and thereby establish a scope for the model to extend applications to advanced research domains such as emotion detection, speech verification [5, 8, 11] and speech classification [**?**], automatic speech recognition (ASR) [6, 7, 9], and automatic audio transcription [71], etc.

In order to facilitate our experimentation, we have selected a prominent leaders speech data set [48] which contains speeches prominent leaders like Benjamin Netanyahu, Jens Stoltenberg, Julia Gillard, Margaret Thatcher, and Nelson Mandela as folder names. Each audio sample is of one-second 16,000 sample rate PCM encoded.

For this experiment, we have selected four speakers as mentioned in Fig. 3.4 as part of audio prepossessing for each speaker sample; we employ Mel-frequency cepstral coefficients (MFCCs) [67] for feature extraction and perform voice activity detection (VAD) [61] to eliminate pauses in the speech sample prior to the feature extraction step.

As a result of audio processing for each speaker sample, a feature matrix is obtained and is given as input to our model after excluding the labels for the clustering process without disturbing the sequence of the feature vectors.

The model is then left to iterate until it converges, that is, until the average value of the latest

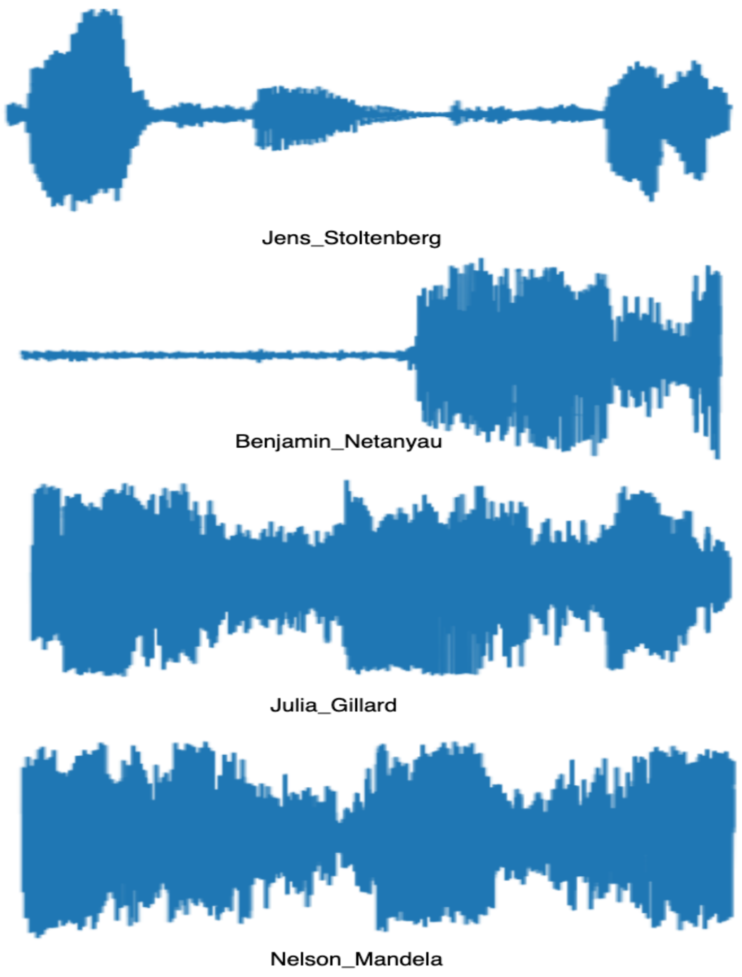

Figure 3.4: Speaker speech samples

batch of iterations for the parameters is approximately equal to zero or remains constant. Each iteration computes all seven stages of the MCMC algorithm, including the reversible jump, and the parameters are used to evaluate the model's performance using the four performance metrics listed in Table 3.1. Then for the same feature vector of speech samples, we use HMM with GMM [55] and standard normal distributions as base models to compare our results.

The model is then run for a set number of iterations until it converges, that is, until the average value of the latest batch of iterations for the parameters is about equal or does not change: all seven stages of the MCMC, including the reverse jump component, are computed in each iteration, and the parameters are utilized to verify the model for performance using the four performance metrics shown in Table 3.3. Then, using the same feature vector of voice samples, we compare our findings using HMM with GMM [55] and HMM with standard normal distributions as base models.

In Table 3.3, in this speaker recognition learning problem, we have shown the appropriate findings of our model. As can be seen, we started the model with two states and left the learning model to determine the optimal number of states. After many iterations, it is evident that our model could reach the proper number of states with the appropriate parameters and produce better outcomes than other benchmark models.

From Table 3.3, it can be observed that our model performed poorly for non optimal number of states with low accuracy: 55.07%, precision: 57.28% and we can also notice that the performance of the model gradually improved while approaching the optimal number of states and finally reaching a maximum accuracy and precision of 79.46% and 79.47%, respectively, outperforming the benchmark models.

Table 3.3: Speaker Recognition

States	HMM-DMM				HMM-GMM				HMM-Std.Norm			
K	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
2	55.07	57.28	55.07	0.08	52.50	53.00	52.50	49.96	51.12	51.29	51.12	46.22
3	64.73	64.61	64.73	55.78	61.91	65.48	61.91	37.13	58.80	58.25	58.80	40.56
4	79.46	79.47	79.46	75.17	71.45	71.38	71.45	67.72	63.93	63.81	63.93	54.19

Chapter 4

Conclusion

In this thesis, main contributions were in three important learning tasks: parameter estimation, model selection and feature selection, for two models: AGGMM and HMM with Dirichlet mixture model.

In chapter 2, by combining Gibbs sampling and Metropolis-Hastings approaches, a Bayesian learning framework for learning asymmetric generalised Gaussian mixture model parameters was proposed. Several recognition and learning applications, including human activity, vehicle, pedestrian, and infected malaria cell recognition, are used to validate the proposed system. Furthermore, a novel Reversible Jump MCMC technique was investigated for accurately computing the number of components in an AGGMM, as well as a feature selection approach was explored to further improve the learning. In chapter 3, a Bayesian learning framework was proposed for HMMs to learn model parameters effectively. Furthermore, a new RJMCMC technique was investigated for determining the number of components in HMMs. By introducing Dirichlet mixtures, a mixture modelling method was applied to improve the model's learning capacity.

In both chapters 2, 3, experiments were conducted on well-known data sets from the video and audio domains to demonstrate the model's utility in a variety of tasks. Throughout this experiment, a variety of pre-processing and domain-specific feature extraction strategies were explored to aid the learning process of the model. Moreover, the models' learning efficiency was evaluated by comparing their outcomes and performance using well-known performance metrics, which revealed that the proposed models outperformed current benchmark models.

As part of the future work, for chapter 2, a more advanced mixture modelling technique could be employed to help the parameter estimation, model selection and feature selection. For chapter 3, feature selection techniques could be integrated within the HMM model to improve its generalisation capabilities.

Appendix A

Appendix

In this section, we explain the computation of Jacobian determinant which is a part of the acceptance ratio of split and combine move:

Table A.1 presents a Jacobian matrix which has partly block diagonal structure, and our goal is to find out its determinant. For that, first, we identify sub determinants across the diagonal and evaluate the sub determinants individually and finally multiply the resultant to obtain the determinant of the whole matrix, and the same is shown below:

$$J_1 = \begin{vmatrix} \operatorname{diag}(v_i) & \operatorname{diag}(1-v_i) \\ \operatorname{diag}(\tilde{a}_{ij_*}) & -\operatorname{diag}(\tilde{a}_{ij_*}) \end{vmatrix} = \begin{vmatrix} \boldsymbol{I} & \operatorname{diag}(1-v_i) \\ \mathbf{0} & -\operatorname{diag}(\tilde{a}_{ij_*}) \end{vmatrix} = \textstyle\prod_{i\neq j_*} \tilde{a}_{ij_*}$$

$$J_2 = \begin{vmatrix} \operatorname{diag}\left(\frac{u_j}{u_0}\right) & \operatorname{diag}\left(\frac{1-u_j}{1-u_0}\right) & -\operatorname{col}\left(\frac{u_1 u_j}{u_0}\right) & \operatorname{col}\left(\frac{(1-u_1)u_j - \sum_{i\neq j_* v_i \tilde{a}_{ii^\star}} \partial\gamma_i/\partial\tilde{a}_{j_* j}}{1-u_0}\right) \\ \operatorname{diag}\left(\frac{\tilde{a}_{j_* j}}{u_0}\right) & -\operatorname{diag}\left(\frac{\tilde{a}_{j_* j}}{1-u_0}\right) & -\operatorname{col}\left(\frac{u_1\tilde{a}_{j_* j}}{u_0}\right) & \operatorname{col}\left(\frac{(1-u_1)\tilde{a}_{j_*}}{1-u_0}\right) \\ -\operatorname{row}\left(\frac{u_j\tilde{a}_{j_* j}}{u_0^2}\right) & \operatorname{row}\left(\frac{(1-u_j)\tilde{a}_{j_* j}}{(1-u_0)^2}\right) & \frac{u_1\left(1-\tilde{\tilde{a}}_{i_1 i_1}-\tilde{\tilde{a}}_{i_1 i_2}\right)}{u_0} & \frac{u_1+\tilde{\tilde{a}}_{i_2 i_1}}{1-u_0} \\ 0 & 0 & \tilde{\tilde{a}}_{i_1 i_1}+\tilde{\tilde{a}}_{i_1 i_2} & \frac{u_0\left(\tilde{\tilde{a}}_{i_1 i_1}+\tilde{\tilde{a}}_{i_1 i_2}\right)}{1-u_0} \end{vmatrix}$$

$$J_3 = \begin{vmatrix} 1 & 1 & 0 & 0 \\ -\sigma_{j_*}\sqrt{\frac{1-u_0}{u_0}} & \sigma_{j_*}\sqrt{\frac{u_0}{1-u_0}} & \frac{2z_1u_0}{\sigma_{j_*}^2 z_2\left(1-z_1^2\right)^2} & \frac{2z_1(1-u_0)}{\sigma_{j_*}^2(1-z_2)\left(1-z_1^2\right)^2} \\ \frac{z_1\sigma_{j_*}^3}{2}\sqrt{\frac{1-u_0}{u_0}} & -\frac{z_1\sigma_{j_*}^3}{2}\sqrt{\frac{u_0}{1-u_0}} & \frac{u_0}{z_2\left(1-z_1^2\right)} & \frac{(1-u_0)}{(1-z_2)\left(1-z_1^2\right)} \\ 0 & 0 & \frac{u_0}{\sigma_{j_*}^2\left(1-z_1^2\right)} & \frac{1-u_0}{\sigma_{j_*}^2(1-z_2)^2\left(1-z_1^2\right)} \end{vmatrix} = \frac{\sqrt{u_0(1-u_0)}}{\sigma_{j_* z_2^2(1-z_2)^2\left(1-z_1^2\right)^3}}$$

J_2 here is evaluated in the same way as shown in [58].

Table A.1: Table of partial derivatives

	$\tilde{a}_{jj_1}$	$\tilde{a}_{jj_2}$	$\tilde{a}_{j_1j}$	$\tilde{a}_{j_2j}$	$\tilde{a}_{ij_2}$	$\tilde{a}_{j_2j_1}$	$\tilde{\mu}_{j_1}$	$\tilde{\mu}_{j_2}$	$\tilde{\sigma}_{j_1}$	$\tilde{\sigma}_{j_2}$
$\tilde{a}_{ij_*}$	x	x	0	0	0	x	0	0	0	0
v_i	x	x	0	0	0	x	0	0	0	0
$\tilde{a}_{j_*j}$	0	0	x	x	x	x	0	0	0	0
u_j	0	0	x	x	x	x	0	0	0	0
u_0	0	0	x	x	x	x	x	x	x	x
u_1	0	0	0	0	x	x	0	0	0	0
$\tilde{\mu}_{j_*}$	0	0	0	0	0	0	x	x	0	0
z_1	0	0	0	0	0	0	x	x	x	x
$\tilde{\sigma}_{j_*}$	0	0	0	0	0	0	x	x	x	x
z_2	0	0	0	0	0	0	0	0	x	x

Bibliography

[1] M. S. Allili, N. Bouguila, and D. Ziou. Finite general Gaussian mixture modeling and application to image and video foreground segmentation. *Journal of Electronic Imaging*, 17(1):1 – 13, 2008.

[2] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to mcmc for machine learning. *Machine Learning*, 50(1):5–43, 2003.

[3] G. J. Awcock and R. Thomas. *Applied image processing*. Macmillan International Higher Education, 1995.

[4] T. O. Ayodele. Types of machine learning algorithms. *New advances in machine learning*, 3:19–48, 2010.

[5] M. Azam, B. Alghabashi, and N. Bouguila. Multivariate bounded asymmetric Gaussian mixture model. In *Mixture Models and Applications*, pages 61–80. Springer, 2020.

[6] M. Azam and N. Bouguila. Unsupervised keyword spotting using bounded generalized Gaussian mixture model with ica. In *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1150–1154. IEEE, 2015.

[7] M. Azam and N. Bouguila. Blind source separation as pre-processing to unsupervised keyword spotting via an ica mixture model. In *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 833–836. IEEE, 2018.

[8] M. Azam and N. Bouguila. Speaker verification using adapted bounded Gaussian mixture model. In *2018 IEEE International Conference on Information Reuse and Integration*, pages 300–307. IEEE, 2018.

[9] M. Azam and N. Bouguila. Bounded generalized Gaussian mixture model with ICA. *Neural Processing Letters*, 49(3):1299–1320, 2019.

[10] M. Azam and N. Bouguila. Multivariate bounded support laplace mixture model. *Soft Computing*, 24:1–30, 2020.

[11] M. Azam and N. Bouguila. Multivariate-bounded Gaussian mixture model with minimum message length criterion for model selection. *Expert Systems*, 38(5):e12688, 2021.

[12] 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, 2016.

[13] N. Bouguila. Non-gaussian mixture image models prediction. In *Proceedings of the International Conference on Image Processing, ICIP 2008, October 12-15, 2008, San Diego, California, USA*, pages 2580–2583. IEEE, 2008.

[14] N. Bouguila. On multivariate binary data clustering and feature weighting. *Comput. Stat. Data Anal.*, 54(1):120–134, 2010.

[15] N. Bouguila, K. Almakadmeh, and S. Boutemedjet. A finite mixture model for simultaneous high-dimensional clustering, localized feature selection and outlier rejection. *Expert Syst. Appl.*, 39(7):6641–6656, 2012.

[16] N. Bouguila and T. Elguebaly. A fully bayesian model based on reversible jump MCMC and finite beta mixtures for clustering. *Expert Syst. Appl.*, 39(5):5946–5959, 2012.

[17] N. Bouguila and W. Fan. *Mixture models and applications*. Springer, 2020.

[18] 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.

[19] N. Bouguila, D. Ziou, and R. I. Hammoud. On bayesian analysis of a finite generalized Dirichlet mixture via a metropolis-within-gibbs sampling. *Pattern Analysis and Applications*, 12(2):151–166, 2009.

[20] 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. Curran Associates, Inc., 2007.

[21] S. Boutemedjet, D. Ziou, and N. Bouguila. Model-based subspace clustering of non-gaussian data. *Neurocomputing*, 73(10-12):1730–1739, 2010.

[22] G. E. Box and G. C. Tiao. *Bayesian inference in statistical analysis*, volume 40. John Wiley & Sons, 2011.

[23] J. Burrell. How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society*, 3(1), 2016.

[24] E. Charniak. *Introduction to artificial intelligence*. Pearson Education India, 1985.

[25] S. Chib and E. Greenberg. Understanding the metropolis-hastings algorithm. *The american statistician*, 49(4):327–335, 1995.

[26] G. G. Chowdhury. Natural language processing. *Annual review of information science and technology*, 37(1):51–89, 2003.

[27] G. A. Churchill. Accurate restoration of dna sequences. In *Case Studies in Bayesian Statistics, Volume II*, pages 90–148. Springer, 1995.

[28] T. Elguebaly and N. Bouguila. Bayesian learning of finite generalized Gaussian mixture models on images. *Signal Processing*, 91:801–820, 2011.

[29] T. Elguebaly and N. Bouguila. A bayesian approach for the classification of mammographic masses. In *Sixth International Conference on Developments in eSystems Engineering, DeSE 2013, Abu Dhabi, United Arab Emirates, December 16-18, 2013*, pages 99–104. IEEE, 2013.

[30] T. Elguebaly and N. Bouguila. Finite asymmetric generalized Gaussian mixture models learning for infrared object detection. *Computer Vision and Image Understanding*, 117:1659–1671, 2013.

[31] T. Elguebaly and N. Bouguila. Simultaneous bayesian clustering and feature selection using rjmcmc-based learning of finite generalized Dirichlet mixture models. *Signal Processing*, 93(6):1531–1546, 2013.

[32] T. Elguebaly and N. Bouguila. Background subtraction using finite mixtures of asymmetric Gaussian distributions and shadow detection. *Mach. Vis. Appl.*, 25(5):1145–1162, 2014.

[33] T. Elguebaly and N. Bouguila. Simultaneous high-dimensional clustering and feature selection using asymmetric Gaussian mixture models. *Image Vis. Comput.*, 34:27–41, 2015.

[34] E. Epaillard and N. Bouguila. Hidden markov models based on generalized dirichlet mixtures for proportional data modeling. In *Artificial Neural Networks in Pattern Recognition - 6th IAPR TC 3 International Workshop, ANNPR 2014, Montreal, QC, Canada, October 6-8, 2014. Proceedings*, volume 8774 of *Lecture Notes in Computer Science*, pages 71–82. Springer, 2014.

[35] E. Epaillard and N. Bouguila. Proportional data modeling with hidden markov models based on generalized dirichlet and beta-liouville mixtures applied to anomaly detection in public areas. *Pattern Recognit.*, 55:125–136, 2016.

[36] W. Fan, N. Bouguila, and X. Liu. A nonparametric bayesian learning model using accelerated variational inference and feature selection. *Pattern Anal. Appl.*, 22(1):63–74, 2019.

[37] S. Fu and N. Bouguila. A bayesian intrusion detection framework. In *2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. IEEE.

[38] S. Fu and N. Bouguila. Asymmetric Gaussian-based statistical models using Markov chain monte carlo techniques for image categorization. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1205–1208. IEEE, 2018.

[39] S. Fu and N. Bouguila. Asymmetric Gaussian mixtures with reversible jump MCMC. In *2018 IEEE Canadian Conference on Electrical & Computer Engineering, CCECE 2018, Quebec, QC, Canada, May 13-16, 2018*, pages 1–4. IEEE, 2018.

[40] S. Fu and N. Bouguila. A soft computing model based on asymmetric Gaussian mixtures and bayesian inference. *Soft Computing*, 24(7):4841–4853, 2019.

[41] K. Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[42] A. E. Gelfand. Gibbs sampling. *Journal of the American statistical Association*, 95(452):1300–1304, 2000.

[43] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, PAMI-6(6):721–741, 1984.

[44] J. D. Hamilton. A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica: Journal of the econometric society*, pages 357–384, 1989.

[45] D. J. Hand. Principles of data mining. *Drug Safety*, 30(7):621–622, 2007.

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

[47] H. Huttunen. Tau vehicle type recognition. https://www.kaggle.com/c/vehicle/data, 2020.

[48] E. Kiplagat. Speaker recognition dataset. https://www.kaggle.com/kongaevans/speaker-recognition-dataset/version/1, 2019.

[49] B. G. Leroux and M. L. Puterman. Maximum-penalized-likelihood estimation for independent and Markov-dependent mixture models. *Biometrics*, 48(2):545–558, 1992.

[50] G. Lowe. Sift-the scale invariant feature transform. *International Journal*, 2(91-110):2, 2004.

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

[52] S. Munder and D. M. Gavrila. An experimental study on pedestrian classification. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1863–1868, 2006.

[53] L. Rabiner and B. Juang. An introduction to hidden Markov models. *ieee assp magazine*, 3(1):4–16, 1986.

[54] S. Rajaraman, S. K. Antani, M. Poostchi, K. Silamut, M. A. Hossain, R. J. Maude, S. Jaeger, and G. R. Thoma. Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images. *PeerJ*, 6, 2018.

[55] D. A. Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, 741:659–663, 2009.

[56] S. Richardson and P. J. Green. On bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society: series B (statistical methodology)*, 59(4):731–792, 1997.

[57] C. Robert. *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media, 2007.

[58] C. P. Robert, T. Ryden, and D. M. Titterington. Bayesian inference in hidden Markov models through the reversible jump Markov chain monte carlo method. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(1):57–75, 2000.

[59] T. Rydén, T. Teräsvirta, and S. Åsbrink. Stylized facts of daily return series and the hidden Markov model. *Journal of applied econometrics*, 13(3):217–244, 1998.

[60] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 32–36 Vol.3. IEEE, 2004.

[61] J. Sohn, N. S. Kim, and W. Sung. A statistical model-based voice activity detection. *IEEE signal processing letters*, 6(1):1–3, 1999.

[62] Z. Song, S. Ali, and N. Bouguila. Bayesian inference for infinite asymmetric Gaussian mixture with feature selection. *Soft Comput.*, 25(8):6043–6053, 2021.

[63] K. Soomro, A. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *Clinical Orthopaedics and Related Research*, 2012.

[64] M. Steinbach, L. Ertöz, and V. Kumar. The challenges of clustering high dimensional data. In *New directions in statistical physics*, pages 273–309. Springer Berlin Heidelberg, 2004.

[65] R. Swinburne. Bayes' theorem. *Revue Philosophique de la France Et de l'Etranger*, 194(2):250–251, 2004.

[66] A. Tharwat. Classification assessment methods. *Applied Computing and Informatics*, 17(1):168–192, 2020.

[67] V. Tiwari. Mfcc and its applications in speaker recognition. *International journal on emerging technologies*, 1(1):19–22, 2010.

[68] R. T. Vemuri, M. Azam, N. Bouguila, and Z. Patterson. A bayesian sampling framework for asymmetric generalized Gaussian mixture models learning. *Neural Computing and Applications*, 09 2021.

[69] B. C. Welsh and D. P. Farrington. Surveillance for crime prevention in public space: Results and policy choices in britain and america. *Criminology & Public Policy*, 3(3):497–526, 2004.

[70] Y. Yang and S. Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, GIS '10, pages 270–279, 2010.

[71] D. Yu and L. Deng. *Automatic Speech Recognition.* Springer, 2016.

[72] Y. Zhang, R. Jin, and Z.-H. Zhou. Understanding bag-of-words model: a statistical framework. volume 1, pages 43–52. Springer, 2010.

[73] X. Zhu and I. Davidson. *Knowledge Discovery and Data Mining: Challenges and Realities: Challenges and Realities*. Igi Global, 2007.

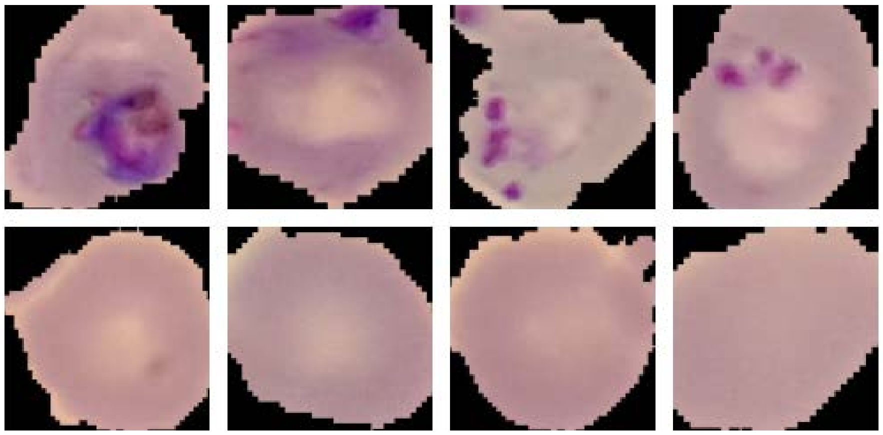

Figure 2.5: Malaria data set

Table 2.5: Malaria Detection Results

Metric	Acc	Pre	Recall	F1	B.Acc	L.Los	Jaccard	FPR	G.M1	G.M2	MCC	Spec.
BAGGMM	84.125	85.93	83.58	86.24	83.58	5.48	75.80	27.82	84.75	77.67	69.48	72.17
BAGMM	78.24	83.97	77.51	82.39	77.51	7.51	70.06	43.38	8.67	66.24	61.14	56.61
BGMM	72.25	72.14	72.18	73.75	72.18	9.58	58.42	28.27	72.16	71.65	44.32	71.1

2 (77.67%) and Specificity (72.17%) indicate the preciseness in classifying negative and positive examples. The experimental results presented in Table 2.5 demonstrate a clear picture of the success of BAGGMM in clustering the normal and infected cells as compared to BAGMM and BGMM in a similar set of experiments. As such, it could be applied in assisting medical practitioners in the diagnosis process. It must be noted that this framework is developed in an unsupervised manner, and it does not require any label during the learning process, which is a great success in this experimental setting and application.

Based on the model's results and analysis and flexibility to fit data of different kinds, the proposed model can also be considered for other applications where the distribution of data follows Gaussian or is close to Gaussian. Therefore, we can clearly state that AGGMM can fit the data in a better way. Hence we can consider applying the model to a wide range of data and expect good results.

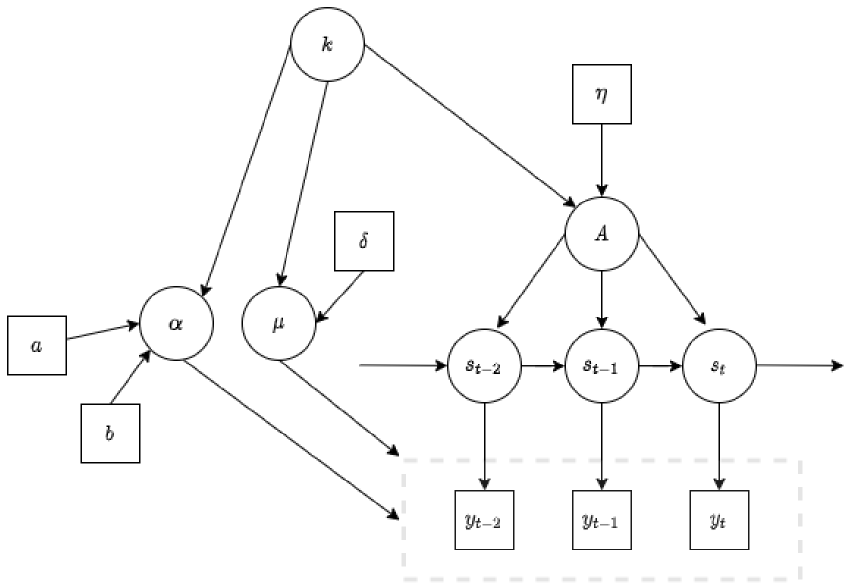

Figure 3.1: Directed acyclic graph for the complete hierarchical model

The term $p(s_t|s_{t-1}, A)$ from Eq. 40 is given by Eq. 33 and $p(s_1 = i|A) = \pi_i$, and from Eq. 39.

$$p(y|s, \mu, \alpha) = \prod_{t=1}^{T} \phi(y_t; \mu_{s_t}, \alpha_{s_t}) \tag{41}$$

Fig. 3.1 is a directed acyclic graph (DAG) representing the complete hierarchical model in which the usual convention is followed where the square boxes represent fixed or observed quantities and the circles represent the unknowns.

3.2 Markov Chain Monte Carlo Methodology

The mixture model [17] approach taken in this chapter is a fairly complex one and requires MCMC techniques to approximate the posterior distribution. A detailed description of these computational techniques can be found in [18, 58]. To get the realizations from the posterior joint distribution with all the parameters, we use the following moves at each sweep of the MCMC algorithm:

1. update the transition probability matrix A

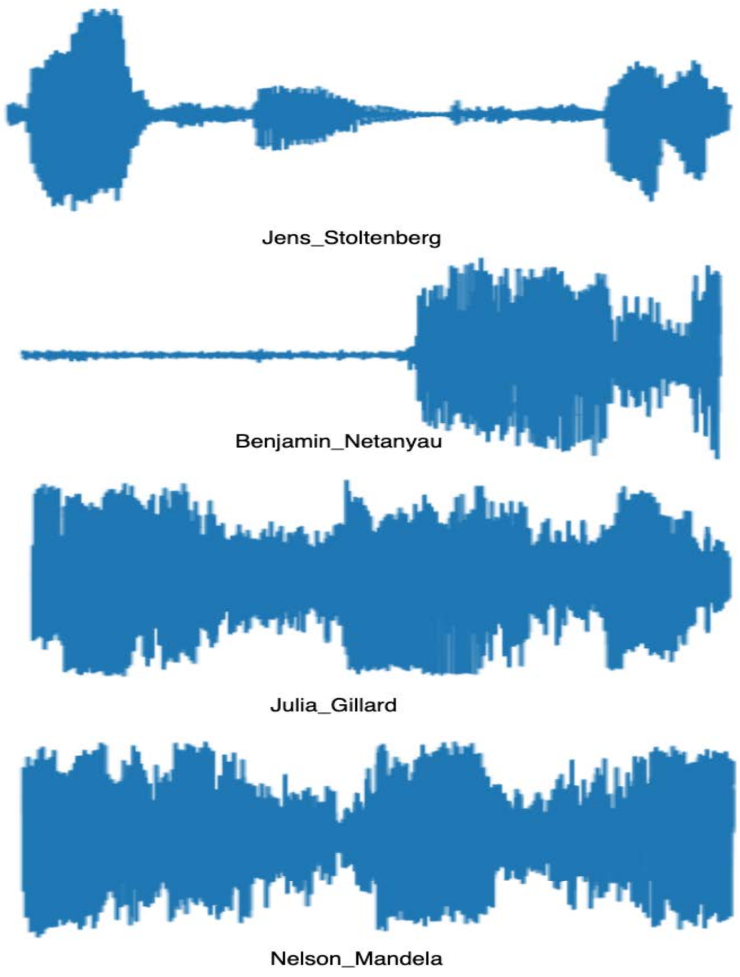

Figure 3.4: Speaker speech samples