Variational Learning Frameworks for Generative Models Based on Hierarchical Dirichlet and Pitman-Yor Processes

Ali Baghdadi

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

January 2022

Concordia University

School of Graduate Studies

This is to certify that the thesis prepared:

By: **Ali Baghdadi**

Entitled: **Variational Learning Frameworks for Generative Models Based on Hierarchical Dirichlet and Pitman-Yor Processes**

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:

Dr. Arash Mohammadi _____ Chair and Examiner

Dr. Mohsen Ghafouri _____ Examiner

Dr. Nizar Bouguila _____ Supervisor

Dr. Zachary Patterson _____ Co-supervisor

Approved by _____

Dr. Abdessamad Ben Hamza, Chair
Department of Concordia Institute for Information Systems Engineering

26 January 2022 _____

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

Abstract

Variational Learning Frameworks for Generative Models Based on Hierarchical Dirichlet and Pitman-Yor Processes

Ali Baghdadi

In many unsupervised machine learning algorithms where labelling a large quantity of data is unfeasible and time-consuming, mixture models have been frequently employed as a helpful statistical learning method. Among all the proposed mixture models, the finite Gaussian mixture model is probably the most famous and widely used. However, other approaches, such as mixtures of Dirichlet distributions have recently been demonstrated to yield even more promising outcomes, especially for non-Gaussian data. As a result, we chose to work on Shifted-Scaled Dirichlet (SSD) mixture models that are more flexible than Gaussian and Dirichlet mixture models in fitting proportional data. In this thesis, we propose variational Bayesian frameworks for learning SSD mixture models. This learning approach has some advantages over other techniques, such as tractable learning computations, precise approximations, and guaranteed convergence. First, we propose a novel non-parametric method based on a hierarchical Dirichlet process mixture of SSD distributions. Then, we extend it to a hierarchical Pitman-Yor process mixture of SSD distributions. These Bayesian frameworks present several properties that make the learning process more accurate. Also, simultaneous parameter estimation and model selection (model complexity determination) are possible with the suggested methods. Finally, we propose a novel hidden Markov model (HMM) framework in which the emission density for conventional continuous HMMs is a mixture of SSD distributions. Experiments on challenging applications such as activity recognition, texture clustering, traffic sign detection, vehicle detection, and spam email detection show that our suggested models may deliver effective solutions when compared to existing alternatives.

Acknowledgements

First and foremost, I want to express my sincere gratitude to my supervisor Prof. Nizar Bouguila, whose knowledge and experience was crucial in developing my research. It was my honor to be his student and I will be forever thankful to him for providing me with this great opportunity.

Also, I would like to thank my co-supervisor, Prof. Zachary Patterson who provided me with valuable feedback during my academic work. Thank you for your patient support and encouragement.

I would like to acknowledge Dr. Florin Talos for inspiring me in my professional work. He is always willing to share his valuable experience with me and encourages me to continue learning more in computer science.

I had the pleasure of having Narges by my side while working on my thesis, and I would like to express my gratitude to her for her advice and support. I would not be able to complete this research without her help. Also, I want to thank all my friends; especially, Muhammad, Ravi, Eddy, Milad, and Hamid.

I was able to complete this thesis thanks to Mitacs and Buspas company for their financial support. In addition, I would like to thank my colleagues in Buspas for providing such a friendly environment.

Last but not least, I want to express my heartfelt gratitude to my wife and family for their support and encouragement throughout my studies in Canada. I would not have been able to finish my education without their unconditional support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction and Related Work

Over the past decades, the demand for novel tools to process and analyze large amounts of data has increased dramatically. Advanced statistical methods and computational techniques provide us diverse approaches to deal with complex data. For instance, clustering and classification are two powerful machine learning techniques that can help scientists detect patterns in data [1]. Data clustering is an unsupervised machine learning method for extracting meaningful information from data. This method is useful in a variety of research domains such as anomaly detection [2,3], recommendation systems [4], social network analysis [5, 6], market segmentation [7, 8], and transportation applications [9, 10].

The finite Gaussian mixture model is perhaps the most well-known and commonly applied of all the suggested mixture models [11–14]. However, recent research has shown that other techniques like mixtures of Dirichlet distributions or mixtures of mixtures could have even more promising results, especially for non-Gaussian data. Real-life data are very complicated and may include data points that are not normally-distributed [15]; therefore, using alternative models to overcome the constraints of Gaussian models might be a suitable choice. To be more specific, Dirichlet, scaled Dirichlet (SD), and generalized Dirichlet mixture models have been used as some alternatives to Gaussian mixture models and these approaches have led to remarkable results when data are proportional. It is worth mentioning that a

vector is said to be proportional when all its elements are positive and sum to one. Proportional data are generated by many processes in different domains ranging from computer vision to pattern recognition and natural language processing. Analyzing this kind of data requires algorithms that have capabilities to model them [16–25]. Therefore, following previous efforts, we are motivated to select an alternative for the Gaussian mixture model for modelling proportional data, so we employ mixtures of shifted-scaled Dirichlet (SSD) distributions in our models. The SSD distribution has two more parameters than the traditional Dirichlet distribution and one more than the SD distribution. The additional parameters of the SSD provide more degrees of freedom and therefore flexibility to fit data. Furthermore, most previous research has been based on finite mixture models and therefore required establishing an optimal number of components (i.e. the model's complexity). In this thesis, we develop non-parametric Bayesian models that make it possible to directly address the above-mentioned issues. In these frameworks, the number of components is supposed to be infinite; therefore, each new observation could be treated as a member of a new cluster that was not seen before.

In the first phase of this thesis, we develop hierarchical Dirichlet process (HDP) and hierarchical Pitman-Yor process (HPYP) mixtures of SSD distributions for proportional data modelling. Then, we go one step further and integrate HDP into a hidden Markov model (HMM) framework which has SSD as its emission probability.

The HDP is an extension to the conventional Dirichlet process (DP). It has a few hierarchical DP levels and in each level, the base measure for the DPs is disturbed by another DP. In our work, we focus on a two-level HDP [26, 27].

Also, the HPYP is based on the Pitman–Yor process and has a power-law characteristic, making it more appropriate than the Dirichlet process for natural phenomenon applications such as text analysis and video segmentation [15, 27, 28].

The HMM is a type of probabilistic model in which each data point is in a hidden state produced by a probability distribution called an emission probability [29]. This approach has been used in a variety of time series

applications and sequential data like anomaly detection [30], facial identification [31], speech recognition [32], machine translation [33], financial analysis [34], healthcare [35], human activity recognition [36], and gesture recognition [37]. The mathematical basis of HMM was initially established by Baum and Petrie [38]. Its primary structure is a Markov chain of latent variables. One of the simplest methods to represent sequential patterns in time series data is to use a Markov chain. This method maintains generality while loosening the assumption of independent identically distributed variables [12]. Depending on the type of data (which can be continuous or discrete), HMM and its emission probabilities are continuous or discrete [39].

For HDP and HPYP learning, we applied a variational inference approach with two settings; batch and online. The online setting allows us to feed our models with large-scale streaming data. Although there are other techniques like Markov Chain Monte Carlo (MCMC) and maximum likelihood (ML), MCMC involves high computational complexity and its convergence is hard to assess [15, 40, 41]. ML, on the other hand, while straightforward to implement, generally deviates from the global maximum and converges instead to local maxima. Another problem of ML is that it is very sensitive to initial values. The main solution to overcome these problems is variational learning. In this method, the idea is based on minimizing the Kullback–Leibler divergence between the true posterior distribution and an approximated variant of it. Due to the integrated estimation scheme of variational learning, it can calculate a model's parameters and find the optimal number of components at the same time.

In the past few years, this approach has been employed and shown its effectiveness [42,43]. Also, for training HMM models, the expectation-maximization technique is commonly used. However, because this method includes a summation of all conceivable combinations of hidden states and mixture components, it is computationally intractable. Besides, ML and MCMC have the above-mentioned issues in HMM context as well. Using variational learning provides us the advantage of overcoming the drawbacks of these methods in learning HMM models [42, 43].

As a summary, this thesis consists of six novel models:

- **BV-HDP-SSD**: Batch variational learning of HDP mixtures of SSD distributions.

- **OV-HDP-SSD**: Online variational learning of HDP mixtures of SSD distributions.
- **BV-HPYP-SSD**: Batch variational learning of HPYP mixtures of SSD distributions.
- **OV-HPYP-SSD**: Online variational learning of HPYP mixtures of SSD distributions.
- **SSD-HMM**: Variational learning of SSD-based hidden Markov model.
- **SSD-HDP-HMM**: Variational learning of SSD based hierarchical Dirichlet process hidden Markov model.

1.2 Contributions

The following are the significant contributions of this thesis:

✓ **Hierarchical Dirichlet and Pitman-Yor Process Mixtures of Shifted-Scaled Dirichlet Distributions for Proportional Data Modelling:**

In this part, we present batch and online variational learning of HDP mixtures of SSD distributions. Then, we extend it to batch and online variational learning of HPYP mixtures of SSD distributions. The objective is to create a model that can fit complicated and proportional real-world data correctly. We show that our SSD based models are more flexible in fitting proportional data than the Dirichlet and Gaussian mixture models. We verify this using four challenging applications, namely, spam email detection, texture clustering, traffic sign detection, and vehicle detection. This work has been submitted to the journal " Advances in Computational Intelligence (ADCI)".

✓ **Shifted-Scaled Dirichlet Based Hidden Markov Model and Shifted-Scaled Dirichlet Based Hierarchical Dirichlet Process Hidden Markov Model with Variational Inference Learning:**

In this part, first, we present the variational learning of SSD based hidden Markov model. Then, we expand our model to a hierarchical Dirichlet process with HMM framework. We illustrate that

choosing SSD as the emission probability distribution helps our models to fit data better than the Gaussian mixture model. Two difficult applications, activity recognition and texture clustering, are used to demonstrate this. Also, the model's complexity is determined throughout the learning process in the second model. This research work has been accepted as a book chapter [44].

1.3 Thesis Overview

This thesis is organized as follows:

- ☐ In chapter 2, we describe the hierarchical Dirichlet process. Then we present mixtures of SSD distributions and derive the hyper parameters of the batch and online variational learning approaches for HDP mixtures of SSD distributions. Then, we have an overview of the hierarchical Pitman-Yor process and go over the derivation of the hyper parameters of the variational learning process for HPYP mixtures of SSD distributions. Finally, we present our experimental results for these models.

- ☐ In chapter 3, we discuss hidden Markov models. Then we propose parameter estimation using variational learning for SSD-HMM and SSD-HDP-HMM models. In addition, we represent the results of analyzing our suggested models.

- ☐ In Chapter 4, we review our research works and contributions and make some suggestions for further research.

Chapter 2

Hierarchical Dirichlet and Pitman-Yor Process Mixtures of Shifted-Scaled Dirichlet Distributions for Proportional Data Modelling

2.1 The Hierarchical Dirichlet Process

The hierarchical Dirichlet process is developed based on a hierarchical structure using multiple DPs [26, 27, 45]. This construction has at least two levels where at each level, the base measure of DP is distributed by another DP. In this thesis, for the sake of simplicity, we suppose a two-level HDP model. In the first level (the local level), there is a grouped dataset $\mathcal{X}$ with M sets. Therefore, the proposed model has a G_j for each set; in which, $j \in \{1, \ldots, M\}$. In the second level (the global level), we have G_0 as a base distribution for the indexed set of $\{G_j\}$. G_0 is a global distribution which is shared between all the j groups. G_0 itself is distributed according to a DP with the base distribution H and concentration parameter γ. For each j we have $G_0 \sim \mathrm{DP}(\gamma, H)$ and $G_j \sim \mathrm{DP}(\lambda, G_0)$. We will utilise stick-breaking construction in the development of our model because it is a fairly simple approach in implementation of HDP models [26, 46]. In this thesis, we have two stick-breaking constructions; one for the global level and another one for

the local level. The global distribution G_0 can be represented by:

$$G_0 = \sum_{k=1}^{\infty} \psi_k \delta_{\Omega_k}, \quad \Omega_k \sim H, \tag{2.1}$$
$$\psi_k = \psi'_k \prod_{s=1}^{k-1} (1 - \psi'_s), \quad \psi'_k \sim \text{Beta}(1, \gamma)$$

In these formulas, $\{\Omega_k\}$ is a series of independent random variables taken from H, and δ_{Ω_k} is an atom at Ω_k. As previously mentioned, G_0 is the base distribution for G_j, therefore, the Ω_k atoms are shared among all G_j. In the stick-breaking construction, we have a proportional concept [45]. First, we consider the total weight of clusters as a unit length stick, then break it into an infinite number of fragments ψ_k which should satisfy this condition: $\sum_{k=1}^{\infty} \psi_k = 1$. The size of each piece is proportional to the stick's length. G_0 has H and γ as its base distribution and concentration parameter, respectively. Besides, G_0 and λ are the base distribution and the concentration parameter for G_j, respectively [45, 47]. Therefore, the local level DP (G_j) would be:

$$G_j = \sum_{t=1}^{\infty} \pi_{jt} \delta_{\varpi_{jt}}, \quad \varpi_{jt} \sim G_0, \tag{2.2}$$
$$\pi_{jt} = \pi'_{jt} \prod_{s=1}^{t-1} (1 - \pi'_{js}), \quad \pi'_{jt} \sim \text{Beta}(1, \lambda)$$

Based on the stick-breaking construction, π_{jt} fragments should satisfy this condition $\sum_{k=1}^{\infty} \pi_{jt} = 1$. Furthermore, ϖ_{jt} follows the base distribution G_0; hence, the value Ω_k will be assigned to it with probability ψ_k. Then we consider the indicator variable W_{jtk} which is a binary latent variable. If ϖ_{jt} is associated with Ω_k, W_{jtk} is equal to 1; otherwise, W_{jtk} is 0. Thus, $\varpi_{jt} = \Omega_k^{W_{jtk}}$. Also, $\vec{W} = (W_{jt1}, W_{jt2}, \dots)$ is distributed as:

$$p(\vec{W} \mid \vec{\psi}) = \prod_{j=1}^{M} \prod_{t=1}^{\infty} \prod_{k=1}^{\infty} \psi_k^{W_{jtk}} \tag{2.3}$$

consequently,

$$p(\vec{W} \mid \vec{\psi'}) = \prod_{j=1}^{M} \prod_{t=1}^{\infty} \prod_{k=1}^{\infty} \left[\psi'_k \prod_{s=1}^{k-1} (1 - \psi'_s) \right]^{W_{jtk}} \tag{2.4}$$

To describe the data set $\mathcal{X}$, we consider i as the index of each observation and j as the index of each group. Therefore, by considering θ_{ji} as a variable assigned to observation X_{ji}, $\vec{\theta}_j = (\theta_{j1}, \theta_{j2}, \dots)$ are distributed by G_j for each j. Consequently, we can define the likelihood function as:

$$\begin{aligned} \theta_{ji} \mid G_j &\sim G_j \\ X_{ji} \mid \theta_{ji} &\sim F(\theta_{ji}) \end{aligned} \tag{2.5}$$

$F(\theta_{ji})$ indicates the distribution of X_{ji} given θ_{ji}. In the HDP mixture model, H as the base distribution of G_0 is considered as the prior distribution for θ_{ji} and each group is related to a mixture model. Since Ω_k atoms are shared among $\{G_j\}$, the mixture components are divided among these mixture models. Each θ_{ji} is distributed by a G_j; therefore, it takes the value ϖ_{jt} having the probability of π_{jt}. We consider the indicator variable $Z_{jit} \in \{0, 1\}$ for each θ_{ji}. Z_{jit} is a binary latent variable as follows: $Z_{jit} = 1$ if θ_{ji} is associated with component t and $Z_{jit} = 0$, otherwise. Thus, we have $\theta_{ji} = {\varpi_{jt}}^{Z_{jit}}$. Consequently, θ_{ji} can be obtained by $\theta_{ji} = {\varpi_{jt}}^{Z_{jit}} = {\Omega_k}^{W_{jtk} Z_{jit}}$. Besides, $\vec{Z} = (Z_{ji1}, Z_{ji2}, \dots)$ is distributed as:

$$p(\vec{Z} \mid \vec{\pi}) = \prod_{j=1}^{M} \prod_{i=1}^{N} \prod_{t=1}^{\infty} \pi_{jt}^{Z_{jit}} \tag{2.6}$$

consequently,

$$p(\vec{Z} \mid \vec{\pi}) = \prod_{j=1}^{M} \prod_{i=1}^{N} \prod_{t=1}^{\infty} \left[\pi'_{jt} \prod_{s=1}^{t-1} (1 - \pi'_{js}) \right]^{Z_{jit}} \tag{2.7}$$

2.2 Mixture of Shifted Scaled Dirichlet Distributions

We focus on a specific form of mixture model in this thesis. In our models, each observation within a group is taken from a mixture of SSD distributions as a generalized version of a Dirichlet distribution. Our models can be considered as hierarchical infinite SSD mixture models so we do not have to select the number of components which is supposed to be countably infinite in these models. Moreover, the flexibility and effectiveness of the SSD distribution have been shown in previous research [23, 24]. Let's assume a proportional observation $\vec{X}_i = (x_{i1}, \ldots, x_{iD})$ is drawn from an SSD distribution where $\sum_{l=1}^{D} x_{il} = 1$, $0 \leq x_{il} \leq 1$, and $l = 1, \ldots, D$. Thus, we have:

$$p(\vec{X}_i \mid \vec{\theta}_j) = \frac{\Gamma(\alpha_j+)}{\prod_{l=1}^{D} \Gamma(\alpha_{jl})} \frac{1}{\tau_j^{D-1}} \frac{\prod_{l=1}^{D} \beta_{jl}^{-\frac{\alpha_{jl}}{\tau_j}} X_{il}^{(\frac{\alpha_{jl}}{\tau_j}-1)}}{\left(\sum_{l=1}^{D} (\frac{X_{il}}{\beta_{jl}})^{\frac{1}{\tau_j}} \right)^{\alpha_j+}} \tag{2.8}$$

where the distribution parameters are: $\vec{\alpha}_j = \left(\alpha_{j1}, \ldots, \alpha_{jD}\right) \in \mathbb{R}_+^D$ as shape parameter, $\vec{\beta}_j = \left(\beta_{j1}, \ldots, \beta_{jD}\right) \in \mathbb{S}^D$ as location parameter and $\tau_j \in \mathbb{R}_+$ as a real scalar. These parameters make our distribution amazingly adaptable which allows the model to better fit proportional data. $\Gamma(.)$ indicates the Gamma function and $\alpha_j+ = \sum_{l=1}^{D} \alpha_{jl}$. If we linearly combine two or more SSD distributions, we will have a mixture model of SSD distributions. For a dataset $\mathcal{X} = \left(\vec{X}_1, \ldots, \vec{X}_N\right)$ including N independent distributed observations, each D-dimensional data vector $\vec{X}$ is taken from an SSD mixture model with M components [12]. We assume that all vectors in $\mathcal{X}$ follow a common probability density function:

$$p(\mathcal{X} \mid \Theta) = \sum_{j=1}^{M} \pi_j p(\vec{X}_i \mid \vec{\theta}_j) \tag{2.9}$$

where π_j is the mixing coefficient of component j satisfying two conditions: $0 < \pi_j < 1$ and $\sum_{j=1}^{M} \pi_j = 1$. Also, $\Theta = \{\pi_1, \ldots, \pi_M, \vec{\theta}_1, \ldots, \vec{\theta}_M\}$ is a set

including model parameters and mixing coefficients in which $\vec{\theta_j} = \{\vec{\alpha_j}, \vec{\beta_j}, \vec{\tau_j}\}$ indicates the parameter vector for jth component. Since the observations are independent, the likelihood function of the SSD mixture model is calculated by:

$$p(\mathcal{X} \mid \vec{\pi}, \vec{\theta}) = \prod_{i=1}^{N} \sum_{j=1}^{M} \pi_j p(\vec{X_i} \mid \vec{\theta_j}) \tag{2.10}$$

Finally, the likelihood function of the hierarchical infinite SSD mixture model is:

$$p(\mathcal{X}) = \prod_{j=1}^{M} \prod_{i=1}^{N} \prod_{t=1}^{\infty} \prod_{k=1}^{\infty} p(\vec{X_i} \mid \vec{\theta_k})^{Z_{jit} W_{jtk}} \tag{2.11}$$

2.3 Variational Learning of HDP Mixtures of SSD Distributions

2.3.1 Batch Variational Learning

One of the most important challenges that we have to deal with is model parameter estimation. Variational inference is a deterministic approximation technique which has shown its capabilities in finding estimates for posterior distributions [12, 42]. For learning HDP mixture of SSD model, we consider $\Theta = (Z, W, \psi', \pi', \alpha, \beta, \tau)$ as the set of latent and unknown random variables. In the variational inference technique, we attempt to find $Q(\Theta)$, which is an estimate for the posterior $p(\Theta \mid \mathcal{X})$. To do this, the lower bound of the marginal likelihood $p(\Theta \mid \mathcal{X})$ should be maximized. By doing this, the KL divergence will reach its minimum according to:

$$KL(Q \mid\mid P) = \ln p(\mathcal{X}) - \mathcal{L}(Q), \quad KL(Q \mid\mid P) \geq 0 \tag{2.12}$$

where

$$\mathcal{L}(Q) = \int Q(\Theta) \ln(\frac{p(X \mid \Theta) p(\Theta)}{Q(\Theta)}) d\Theta \tag{2.13}$$

and

$$KL(Q \mid\mid P) = -\int Q(\Theta) \ln(\frac{Q(\Theta)}{p(\Theta \mid X)}) d\Theta. \tag{2.14}$$

We know that $KL(Q \mid\mid P) = 0$ when $Q(\Theta) = p(\Theta \mid \mathcal{X})$. However, the true posterior $p(\Theta \mid \mathcal{X})$ is intractable; therefore, we have to consider $Q(\Theta)$ as a restricted family of distributions to be able to calculate it. We can rewrite $Q(\Theta)$ as disjoint tractable components using mean field theory [48]:

$$Q(\Theta) = \prod_i Q_i(\Theta_i) \tag{2.15}$$

Maximization of the lower bound $\mathcal{L}(Q)$ is done by variational optimization with respect to each of the factors $Q_i(\Theta_i)$. The general equation for a given $Q_s(\Theta_s)$ is:

$$Q_s(\Theta_s) = \frac{exp \left\langle \ln \ p(\mathcal{X}, \Theta) \right\rangle_{j \neq s}}{\int exp \left\langle \ln \ p(\mathcal{X}, \Theta) \right\rangle_{j \neq s} d\Theta} \tag{2.16}$$

where $\langle . \rangle_{j \neq s}$ is the expectation of all the distributions $Q_s(\Theta_s)$ except $j = s$. Next, we employ the truncation technique [49] to determine the truncation levels of the stick-breaking constructions such that:

$$\psi_K{}' = 1, \quad \sum_{k=1}^{K} \psi_k = 1, \quad \psi_k = 0 \quad \text{when} \quad k > K \tag{2.17}$$

$$\pi_{jT}{}' = 1, \quad \sum_{t=1}^{T} \pi_{jt} = 1, \quad \pi_t = 0 \quad \text{when} \quad t > T \tag{2.18}$$

K and T are global and local truncation levels of hierarchical Dirichlet processes, respectively. These variational parameters can be updated automatically within the learning process. Using a factorization assumption and stick-breaking construction, we obtain:

$$Q(\Theta) = Q(\vec{Z}) Q(\vec{W}) Q(\vec{\pi}') Q(\vec{\psi}') Q(\vec{\alpha}) Q(\vec{\beta}) Q(\vec{\tau}) \tag{2.19}$$

Now, we define prior distributions over model parameters. The joint distribution of all the random variables is expressed as:

$$p(\mathcal{X}, \Theta \mid \vec{\pi}) = p(\mathcal{X} \mid \mathcal{Z}, \mathcal{W}, \vec{\alpha}, \vec{\beta}, \vec{\tau}) p(\mathcal{Z} \mid \vec{\pi}) p(\mathcal{W} \mid \vec{\psi}) p(\vec{\alpha}) p(\vec{\beta}) p(\vec{\tau}) \tag{2.20}$$

The conjugate priors for SSD parameters $\vec{\alpha}, \vec{\beta}$, and $\vec{\tau}$ are chosen as follows:

$$p(\alpha_{kl}) = \mathcal{G}(\alpha_{kl} \mid u_{kl}, \nu_{kl}) = \frac{\nu_{kl}^{u_{kl}}}{\Gamma(u_{kl})} \alpha_{kl}^{u_{kl}-1} e^{-\nu_{kl}\alpha_{kl}} \tag{2.21}$$

$$p(\beta_{kl}) = \mathcal{D}(\beta_{kl} \mid \vec{h_k}) = \frac{\Gamma(\sum_{l=1}^{D} h_{kl})}{\prod_{l=1}^{D} \Gamma(h_{kl})} \prod_{l=1}^{D} \beta_{kl}^{h_{kl}-1} \tag{2.22}$$

$$p(\tau_k) = \mathcal{G}(\tau_k \mid q_k, s_k) = \frac{q_k^{s_k}}{\Gamma(q_k)} \tau_k^{q_k-1} e^{-s_k\tau_k} \tag{2.23}$$

Therefore, the prior distributions for SSD parameters will be:

$$p(\vec{\alpha}) = \prod_{k=1}^{\infty} \prod_{l=1}^{D} p(\alpha_{kl}) \tag{2.24}$$

$$p(\vec{\beta}) = \prod_{k=1}^{\infty} \prod_{l=1}^{D} p(\beta_{kl}) \tag{2.25}$$

$$p(\vec{\tau}) = \prod_{k=1}^{\infty} p(\tau_k) \tag{2.26}$$

$\{u_{kl}\}$, $\{\nu_{kl}\}$, $\{h_{kl}\}$, $\{q_k\}$ and $\{s_k\}$ are positive hyper-parameters. We can write the prior distribution of π' and ψ', respectively [45]:

$$p(\pi') = \prod_{j=1}^{M} \prod_{t=1}^{\infty} \lambda_{jt} (1 - \pi_{jt}{}')^{\lambda_{jt}-1} \tag{2.27}$$

$$p(\psi') = \prod_{k=1}^{\infty} \gamma_k (1 - \psi'_k)^{\gamma_k - 1} \tag{2.28}$$

Optimal variational estimation for each factor of $Q(\Theta)$ is calculated by:

$$Q(\vec{Z}) = \prod_{j=1}^{M} \prod_{i=1}^{N} \prod_{t=1}^{T} \rho_{jit}^{Z_{jit}} \tag{2.29}$$

$$Q(\vec{W}) = \prod_{j=1}^{M} \prod_{t=1}^{T} \prod_{k=1}^{K} \vartheta_{jtk}^{W_{jtk}} \tag{2.30}$$

$$Q(\vec{\pi}') = \prod_{j=1}^{M} \prod_{t=1}^{T} Beta(\pi'_{jt} \mid a^{\star}_{jt}, b^{\star}_{jt}) \tag{2.31}$$

$$Q(\vec{\psi}') = \prod_{k=1}^{K} Beta(\psi'_k \mid c^{\star}_k, d^{\star}_k) \tag{2.32}$$

$$Q(\vec{\alpha}) = \prod_{k=1}^{K} \prod_{l=1}^{D} \mathcal{G}(\alpha_{kl} \mid u^{\star}_{kl}, \nu^{\star}_{kl}) \tag{2.33}$$

$$Q(\vec{\beta}) = \prod_{k=1}^{K} \prod_{l=1}^{D} \mathcal{D}(\beta_{kl} \mid h^{\star}_{kl}) \tag{2.34}$$

$$Q(\vec{\tau}) = \prod_{k=1}^{K} \mathcal{G}(\tau_k \mid q^{\star}_k, s^{\star}_k) \tag{2.35}$$

where $\mathcal{G}(.)$ and $\mathcal{D}(.)$ are Gamma and Dirichlet distributions, respectively. Also,

$$\rho_{jit} = \frac{\exp(\tilde{\rho}_{jit})}{\sum_{f=1}^{T} exp(\tilde{\rho}_{jif})} \tag{2.36}$$

where $\tilde{\rho}_{jit}$ can be calculated using the following equation:

$$\tilde{\rho}_{jit} = \sum_{k=1}^{K} \langle W_{jtk} \rangle [\tilde{R}_k - (D-1)\ln \overline{\tau}_k + \sum_{l=1}^{D} [- \frac{\overline{\alpha}_{kl}}{\overline{\tau}_k} \ln \overline{\beta}_{kl} + (\frac{\overline{\alpha}_{kl}}{\overline{\tau}_k} - 1) \ln x_{jil}] \tag{2.37}$$

$$- (\alpha_k +) \ln(\sum_{l=1}^{D} (\frac{x_{jil}}{\overline{\beta}_{kl}})^{\frac{1}{\overline{\tau}_k}}) + \langle \ln \pi'_{jt} \rangle + \sum_{s=1}^{t-1} \langle \ln(1 - \pi'_{js}) \rangle]$$

and $\tilde{R}_k$ is calculated by [43]:

$$\tilde{R}_k =$$

$$\ln \frac{\Gamma\big(\sum_{l=1}^{D} \overline{\alpha}_{kl}\big)}{\prod_{l=1}^{D} \Gamma\big(\overline{\alpha}_{kl}\big)} + \sum_{l=1}^{D} \overline{\alpha}_{kl} \left[\Psi\left(\sum_{l=1}^{D} \overline{\alpha}_{kl} \right) - \Psi\big(\overline{\alpha}_{kl}\big) \right] \times \left[\big\langle \ln \alpha_{kl} \big\rangle - \ln \overline{\alpha}_{kl} \right]$$

$$+ \frac{1}{2} \sum_{l=1}^{D} \overline{\alpha}_{kl}^2 \left[\Psi'\left(\sum_{l=1}^{D} \overline{\alpha}_{kl} \right) - \Psi'\big(\overline{\alpha}_{kl}\big) \right] \times \left\langle \big(\ln \alpha_{kl} - \ln \overline{\alpha}_{kl} \big)^2 \right\rangle$$

$$+ \frac{1}{2} \sum_{a=1}^{D} \sum_{b=1, a \neq b}^{D} \overline{\alpha}_{ka} \, \overline{\alpha}_{kb} \left[\Psi'\left(\sum_{l=1}^{D} \overline{\alpha}_{kl} \right) \times \Big(\big\langle \ln \overline{\alpha}_{ka} \big\rangle - \ln \overline{\alpha}_{ka} \Big) \times \right. \tag{2.38}$$

$$\left. \Big(\big\langle \ln \overline{\alpha}_{kb} \big\rangle - \ln \overline{\alpha}_{kb} \Big) \right]$$

$\Psi(.)$ and $\Psi'(.)$ are the digamma and trigamma functions, respectively. Similarly, ϑ_{jtk} of the factor $Q(W)$ is defined by:

$$\vartheta_{jtk} = \frac{\exp(\tilde{\vartheta}_{jtk})}{\sum_{f=1}^{K} exp(\tilde{\vartheta}_{jtf})} \tag{2.39}$$

where

$$\begin{aligned}\tilde{\vartheta}_{jtk} = \sum_{i=1}^{N} \big\langle Z_{jit} \big\rangle [\tilde{R}_k - (D-1)\ln \overline{\tau}_k + \sum_{l=1}^{D} [-\frac{\overline{\alpha}_{kl}}{\overline{\tau}_k} \ln \overline{\beta}_{kl} + (\frac{\overline{\alpha}_{kl}}{\overline{\tau}_k} - 1)\ln x_{jil}] \\ - (\alpha_k +) \ln(\sum_{l=1}^{D} (\frac{x_{jil}}{\overline{\beta}_{kl}})^{\frac{1}{\overline{\tau}_k}}) + \langle \ln \psi'_k \rangle + \sum_{s=1}^{k-1} \langle \ln(1-\psi'_s) \rangle] \end{aligned} \tag{2.40}$$

The hyper parameters of the variational factors are calculated by:

$$a^{\star}_{jt} = 1 + \sum_{i=1}^{N} \langle Z_{jit} \rangle \tag{2.41}$$

$$b^{\star}_{jt} = \lambda_{jt} + \sum_{i=1}^{N} \sum_{s=t+1}^{T} \langle Z_{jis} \rangle \tag{2.42}$$

$$c^{\star}_{k} = 1 + \sum_{j=1}^{M} \sum_{t=1}^{T} \langle W_{jtk} \rangle \tag{2.43}$$

$$d^{\star}_{k} = \gamma_k + \sum_{j=1}^{M} \sum_{t=1}^{T} \sum_{s=k+1}^{K} \langle W_{jts} \rangle \tag{2.44}$$

$$\begin{aligned} u^{\star}_{kl} = u_{kl} + \sum_{j=1}^{M} \sum_{t=1}^{T} \langle W_{jtk} \rangle \sum_{i=1}^{N} \big\langle Z_{jit} \big\rangle \overline{\alpha}_{kl} \times \\ [\psi(\sum_{s=1}^{D} \overline{\alpha}_{ks}) - \psi\big(\overline{\alpha}_{kl}\big) + \sum_{s \neq d}^{D} \psi'(\sum_{s=1}^{D} \overline{\alpha}_{ks}) \times \overline{\alpha}_{ks}(\big\langle \ln \alpha_{ks} \big\rangle - \ln \overline{\alpha}_{ks})] \end{aligned} \tag{2.45}$$

$$\nu_{kl}^{\star} = \nu_{kl} - \sum_{j=1}^{M}\sum_{t=1}^{T}\langle W_{jtk}\rangle \sum_{i=1}^{N}\left\langle Z_{jit}\right\rangle \times [\frac{1}{\tau_k}\ln\frac{\beta_{kl}}{x_{jil}} + \ln(\sum_{s=1}^{D}(\frac{x_{jis}}{\beta_{ks}})^{\frac{1}{\tau_k}})] \quad (2.46)$$

$$h_{kl}^{\star} = h_{kl} + \sum_{j=1}^{M}\sum_{t=1}^{T}\langle W_{jtk}\rangle \sum_{i=1}^{N}\langle Z_{jit}\rangle \times \quad (2.47)$$
$$[\frac{-\bar{\alpha}_{kl}}{\bar{\tau}_k} + \frac{\bar{\alpha}_{kl}}{\bar{\tau}_k} \times (\frac{x_{jil}}{\bar{\beta}_{kl}})^{\frac{1}{\bar{\tau}_k}} \times \frac{1}{\sum_{s=1}^{D}(\frac{x_{jis}}{\bar{\beta}_{js}})^{\bar{\tau}_k{}^{-1}}}]$$

$$q_k^{\star} = q_k + \sum_{j=1}^{M}\sum_{t=1}^{T}\langle W_{jtk}\rangle \sum_{i=1}^{N} Z_{jit} \times \quad (2.48)$$
$$[1 - D + \frac{(\alpha_k+)}{\tau_k}\frac{\sum_{l=1}^{D}(\frac{x_{jil}}{\beta_{kl}})^{\tau_k^{-1}}\ln(\frac{x_{jil}}{\beta_{kl}})}{\sum_{l=1}^{D}(\frac{x_{jil}}{\beta_{kl}})^{\tau_k^{-1}}}]$$

$$s_k^{\star} = s_k - \sum_{j=1}^{M}\sum_{t=1}^{T}\langle W_{jtk}\rangle \sum_{i=1}^{N} Z_{jit}[\sum_{l=1}^{D}\frac{\alpha_{kl}}{{\tau_k}^2}\ln(\frac{x_{jil}}{\beta_{kl}})] \quad (2.49)$$

Also, their expected values are given by:

$$\bar{\alpha}_{kl} = \langle\alpha_{kl}\rangle = \frac{u_{kl}^{\star}}{v_{kl}^{\star}}, \quad \overline{\beta}_{kl} = \langle\beta_{kl}\rangle = \frac{h_{kl}^{\star}}{\sum_{l=1}^{D} h_{kl}^{\star}}$$
$$\overline{\tau_k} = \langle\tau_k\rangle = \frac{q_k^{\star}}{s_k^{\star}}, \quad \langle Z_{jit}\rangle = \rho_{jit}, \quad \langle W_{jtk}\rangle = \vartheta_{jtk}$$
$$\left\langle \ln\pi'_{jt}\right\rangle = \Psi(a_{jt}^{\star}) - \Psi(a_{jt}^{\star} + b_{jt}^{\star})$$
$$\left\langle \ln(1-\pi'_{jt})\right\rangle = \Psi(b_{jt}^{\star}) - \Psi(a_{jt}^{\star} + b_{jt}^{\star})$$
$$\left\langle \ln\psi'_k\right\rangle = \Psi(c_k^{\star}) - \Psi(c_k^{\star} + d_k^{\star})$$
$$\left\langle \ln(1-\psi'_k)\right\rangle = \Psi(d_k^{\star}) - \Psi(c_k^{\star} + d_k^{\star})$$
$$\langle\ln\alpha_{kl}\rangle = \Psi(u_{kl}^{\star}) - \ln v_{kl}^{\star}$$
$$\langle(\ln\alpha_{kl} - \ln\overline{\alpha}_{kl})^2\rangle = \left[\Psi(u_{kl}^{\star}) - \ln u_{kl}^{\star}\right]^2 + \Psi'(u_{kl}^{\star}) \quad (2.50)$$

The algorithm of our proposed model is presented in Algorithm 1.

Algorithm 1 Batch variational learning of HDP mixtures of SSD distributions

1. Choose the initial values for K, T, $\{\lambda_{jt}\}$, $\{\gamma_k\}$, $\{u_{kl}\}$, $\{\nu_{kl}\}$, $\{h_{kl}\}$, $\{q_k\}$, $\{s_k\}$, and initialize the values of ρ_{jit} by K-means algorithm.
2. **repeat**
3. The variational E-step: Estimate the values of equation (2.50) using (2.41) to (2.49).
4. The variational M-step: Update the variational solutions using (2.29) to (2.35).
5. **until** Convergence criterion is reached.

2.3.2 Online Variational Learning

In this section, we apply online variational inference as an extension to batch variational learning. This approach is more beneficial when data are received dynamically. If we represent the number of current data points with r, the lower bound of Q will be:

$$\mathcal{L}^{(r)}(Q) = \frac{N}{r}\sum_{i=1}^{r}\int Q(\Lambda)\mathrm{d}\Lambda \sum_{\vec{Z}_i} Q(\vec{Z}_i)\ln[\frac{p(\vec{X}_i, \vec{Z}_i \mid \Lambda)}{Q(\vec{Z}_i)}] + \int Q(\Lambda)\ln[\frac{p(\Lambda)}{Q(\Lambda)}]\mathrm{d}\Lambda \tag{2.51}$$

where $\Lambda = (\mathcal{W}, \psi', \pi', \alpha, \beta, \tau)$ and X_r is a new observation. Our goal is maximizing the lower bound, therefore, we will maximize $\mathcal{L}^{(r)}(Q)$ with respect to $Q^{(r)}(Z)$ while other factors will stay constant at their $r-1$ values. We calculate $Q^{(r)}(Z)$ by:

$$Q^{(r)}(Z) = \prod_{j=1}^{M}\prod_{t=1}^{T}\rho_{jtr}^{Z_{jtr}} \tag{2.52}$$

$$\rho_{jtr} = \frac{\exp(\tilde{\rho}_{jtr})}{\sum_{f=1}^{T} exp(\tilde{\rho}_{jfr})} \tag{2.53}$$

$\tilde{\rho}_{jtr}$ can be calculated using the following equation:

$$\tilde{\rho}_{jtr} = \sum_{k=1}^{K} \langle W_{jtk}^{(r-1)} \rangle [\tilde{R}_k^{(r-1)} - (D-1)\ln \overline{\tau}_j] k^{(r-1)}$$
$$+ \sum_{l=1}^{D} [-\frac{\overline{\alpha}_{kl}^{(r-1)}}{\overline{\tau}_k^{(r-1)}} \ln \overline{\beta}_{kl}^{(r-1)} + (\frac{\overline{\alpha}_{kl}^{(r-1)}}{\overline{\tau}_k^{(r-1)}} - 1) \ln x_{jrl}]$$
$$- (\alpha_k +^{(r-1)}) \ln(\sum_{l=1}^{D} (\frac{x_{jrl}}{\overline{\beta}_{kl}^{(r-1)}})^{\overline{\tau}_k^{-(r-1)}}) + \langle \ln \pi_{jt}'^{(r-1)} \rangle + \sum_{s=1}^{t-1} \langle \ln(1 - \pi_{js}'^{(r-1)}) \rangle] \tag{2.54}$$

Then, we maximize the lower bound with respect to $Q^{(r)}(W)$ while other factors will remain constant. Also,

$$Q^{(r)}(W) = \prod_{j=1}^{M} \prod_{t=1}^{T} \prod_{k=1}^{K} \vartheta_{jtk}^{(r) W_{jtk}^{(r)}} \tag{2.55}$$

we update $\vartheta_{jtk}^{(r)}$ by:

$$\vartheta_{jtk}^{(r)} = \vartheta_{jtk}^{(r-1)} + \xi_r \Delta \vartheta_{jtk}^{(r)} \tag{2.56}$$

where $\Delta \vartheta_{jtk}^{(r)}$ is a natural gradient of $\vartheta_{jtk}^{(r)}$. Also, $\xi_r = (\tau + r)^{-\xi}$ is the learning rate with two constraints: $\xi \in (0.5, 1]$ and $\tau \geq 0$. Also,

$$\vartheta_{jtk}^{(r)} = \frac{\exp(\tilde{\vartheta}_{jtk}^{(r)})}{\sum_{f=1}^{K} exp(\tilde{\vartheta}_{jtf}^{(r)})} \tag{2.57}$$

where:

$$\tilde{\vartheta}_{jtk}^{(r)} = N \rho_{jtr} [\tilde{R}_k^{(r-1)} - (D-1) \ln \overline{\tau}_k^{(r-1)} \sum_{l=1}^{D} [-\frac{\overline{\alpha}_{kl}^{(r-1)}}{\overline{\tau}_k^{(r-1)}} \times \ln \overline{\beta}_{kl}^{(r-1)}$$
$$+ (\frac{\overline{\alpha}_{kl}^{(r-1)}}{\overline{\tau}_k^{(r-1)}} - 1) \ln x_{jrl}] - (\alpha_k +^{(r-1)}) \ln(\sum_{l=1}^{D} (\frac{x_{jrl}}{\overline{\beta}_{kl}^{(r-1)}})^{\overline{\tau}_k^{-(r-1)}})$$
$$+ \langle \ln \psi_k'^{(r-1)} \rangle + \sum_{s=1}^{k-1} \langle \ln(1 - \psi_s'^{(r-1)}) \rangle] \tag{2.58}$$

Now, the lower bound can be maximized with respect to $Q^{(r)}(\vec{\pi}')$, $Q^{(r)}(\vec{\psi}')$, $Q^{(r)}(\vec{\alpha})$, $Q^{(r)}\big(\vec{\beta}\big)$, and $Q^{(r)}\big(\vec{\tau}\big)$:

$$Q^{(r)}(\vec{\pi}') = \prod_{j=1}^{M}\prod_{t=1}^{T} Beta(\pi_{jt}^{\prime(r)} \mid a_{jt}^{\star(r)}, b_{jt}^{\star(r)}) \tag{2.59}$$

$$Q^{(r)}(\vec{\psi}') = \prod_{k=1}^{K} Beta(\psi_{k}^{\prime(r)} \mid c_{k}^{\star(r)}, d_{k}^{\star(r)}) \tag{2.60}$$

$$Q^{(r)}(\vec{\alpha}) = \prod_{k=1}^{K}\prod_{l=1}^{D} \mathcal{G}(\alpha_{kl}^{(r)} \mid u_{kl}^{\star(r)}, v_{kl}^{\star(r)}) \tag{2.61}$$

$$Q^{(r)}\big(\vec{\beta}\big) = \prod_{k=1}^{K}\prod_{l=1}^{D} \mathcal{D}\big(\beta_{kl}^{(r)} \mid h_{kl}^{\star(r)}\big) \tag{2.62}$$

$$Q^{(r)}\big(\vec{\tau}\big) = \prod_{k=1}^{K} \mathcal{G}\big(\tau_{k}^{(r)} \mid q_{k}^{\star(r)}, s_{k}^{\star(r)}\big) \tag{2.63}$$

The hyper parameters are given by:

$$a_{jt}^{\star(r)} = a_{jt}^{\star(r-1)} + \xi_r \Delta a_{jt}^{\star(r)}, \quad b_{jt}^{\star(r)} = b_{jt}^{\star(r-1)} + \xi_r \Delta b_{jt}^{\star(r)} \tag{2.64}$$

$$c_{k}^{\star(r)} = c_{k}^{\star(r-1)} + \xi_r \Delta c_{k}^{\star(r)}, \quad d_{k}^{\star(r)} = d_{k}^{\star(r-1)} + \xi_r \Delta d_{k}^{\star(r)} \tag{2.65}$$

$$u_{kl}^{\star(r)} = u_{kl}^{\star(r-1)} + \xi_r \Delta u_{kl}^{\star(r)}, \quad v_{kl}^{\star(r)} = v_{kl}^{\star(r-1)} + \xi_r \Delta v_{kl}^{\star(r)} \tag{2.66}$$

$$h_{kl}^{\star(r)} = h_{kl}^{\star(r-1)} + \xi_r \Delta h_{kl}^{\star(r)}, \quad q_{k}^{\star(r)} = q_{k}^{\star(r-1)} + \xi_r \Delta q_{k}^{\star(r)} \tag{2.67}$$

$$s_{k}^{\star(r)} = s_{k}^{\star(r-1)} + \xi_r \Delta s_{k}^{\star(r)} \tag{2.68}$$

All the natural gradients can be updated by:

$$\Delta a_{jt}^{\star(r)} = 1 + N\rho_{jtr} - a_{jt}^{(r-1)} \tag{2.69}$$

$$\Delta b_{jt}^{\star(r)} = \lambda_{jt} + N \sum_{s=t+1}^{T} \rho_{jsr} - b_{jt}^{(r-1)} \tag{2.70}$$

$$\Delta c_{k}^{\star(r)} = 1 + \sum_{j=1}^{K} \sum_{t=1}^{T} \vartheta_{jtk}^{(r)} - c_{k}^{(r-1)} \tag{2.71}$$

$$\Delta d_{k}^{\star(r)} = \gamma_k + \sum_{j=1}^{K} \sum_{t=1}^{T} \sum_{s=k+1}^{K} \vartheta_{jts}^{(r)} - d_{k}^{(r-1)} \tag{2.72}$$

$$\begin{aligned} \Delta u_{kl}^{\star(r)} = u_{kl} + N \sum_{j=1}^{M} \sum_{t=1}^{T} \vartheta_{jtk}^{(r)} \rho_{jtr} \overline{\alpha}_{kl}^{(r-1)} \times \\ [\psi(\sum_{s=1}^{D} \overline{\alpha}_{ks}^{(r-1)}) - \psi\big(\overline{\alpha}_{kl}^{(r-1)}\big) + \sum_{s \neq d}^{D} \psi'(\sum_{s=1}^{D} \overline{\alpha}_{ks}^{(r-1)}) \overline{\alpha}_{ks}^{(r-1)} \times \\ (\big\langle \ln \alpha_{ks}^{(r-1)} \big\rangle - \ln \overline{\alpha}_{ks}^{(r-1)})] - u_{kl}^{(r-1)} \end{aligned} \tag{2.73}$$

$$\begin{aligned} \Delta \nu_{kl}^{\star(r)} = \nu_{kl} - \nu_{kl}^{(r-1)} - N \sum_{j=1}^{M} \sum_{t=1}^{T} \vartheta_{jtk}^{(r)} \rho_{jtr} \times \\ [\frac{1}{\tau_k^{(r-1)}} \ln \frac{\beta_{kl}^{(r-1)}}{x_{jil}} + \ln (\sum_{s=1}^{D} (\frac{x_{jis}}{\beta_{ks}^{(r-1)}})^{\tau_k^{-(r-1)}})] \end{aligned} \tag{2.74}$$

$$\begin{aligned} \Delta h_{kl}^{\star(r)} = h_{kl} + N \sum_{j=1}^{M} \sum_{t=1}^{T} \vartheta_{jtk}^{(r)} \rho_{jtr} \frac{\bar{\alpha}_{kl}^{(r-1)}}{\tau_k^{(r-1)}} \times \\ [-1 + \frac{\bar{\alpha}_{kl}^{(r-1)} (\frac{x_{jil}}{\beta_{kl}^{(r-1)}})^{\tau_k^{-(r-1)}}}{\tau_k^{(r-1)} \sum_{s=1}^{D} (\frac{x_{jis}}{\beta_{ks}^{(r-1)}})^{\tau_k^{-(r-1)}}}] - h_{kl}^{(r-1)} \end{aligned} \tag{2.75}$$

$$\Delta q_k^{\star(r)} = q_k - q_k^{(r-1)} + N \sum_{j=1}^{M} \sum_{t=1}^{T} \vartheta_{jtk}^{(r)} \rho_{jtr} \times \tag{2.76}$$
$$[1 - D + \frac{\alpha_k^{+(r-1)} \sum_{l=1}^{D} (\frac{x_{jil}}{\beta_{kl}^{(r-1)}})^{\tau_k^{-(r-1)}} \ln (\frac{x_{jil}}{\beta_{kl}^{(r-1)}})}{\tau_k^{(r-1)} \sum_{l=1}^{D} (\frac{x_{jil}}{\beta_{kl}^{(r-1)}})^{\tau_k^{-(r-1)}}}]$$

$$\Delta s_k^{\star(r)} = s_k - N \sum_{j=1}^{M} \sum_{t=1}^{T} \vartheta_{jtk}^{(r)} \rho_{jtr} \times [\sum_{l=1}^{D} \frac{\alpha_{kl}^{(r-1)}}{\tau_k^{(r-1)^2}} \ln (\frac{x_{jil}}{\beta_{kl}^{(r-1)}})] - s_k^{(r-1)} \tag{2.77}$$

The algorithm of online variational learning of HDP mixtures of SSD distributions (OV-HDP-SSD) is presented in Algorithm 2.

Algorithm 2 Online variational learning of HDP mixtures of SSD distributions.

1. Choose the initial truncation level K, T and the values for hyper parameters $\{\lambda_{jt}\}$, $\{\gamma_k\}$, $\{u_{kl}\}$, $\{\nu_{kl}\}$, $\{h_{kl}\}$, $\{q_k\}$, and $\{s_k\}$.
2. **for** $r = 1 \rightarrow N$ **do**
3. Update the variational solution for $Q^{(r)}(\vec{Z})$ using (2.52).
4. Compute learning rate by $\xi_r = (\tau + r)^{-\xi}$ and natural gradient $\Delta\vartheta_{jtk}^{(r)}$.
5. Calculate the natural gradients of remaining hyper parameters using (2.69) to (2.77).
6. Update the variational solutions for $Q^{(r)}(\vec{W})$, $Q^{(r)}(\vec{\pi}')$, $Q^{(r)}(\vec{\psi}')$, $Q^{(r)}(\vec{\alpha})$, $Q^{(r)}(\vec{\beta})$, and $Q^{(r)}(\vec{\tau})$.
7. Repeat the variational until new data is observed.
8. **end for**

2.4 Hierarchical Pitman-Yor Process

In this section, we suggest a hierarchical Pitman–Yor process in which each observation within a group is derived from a mixture of SSD distributions [15, 28]. It is worth mentioning that the SSD mixture model was chosen mostly because of its promising achievements in modelling high dimensional proportional data. The SSD distribution is defined in (2.8).
The hierarchical Pitman–Yor process has a recursive construction in which the base measure for a Pitman–Yor process is obtained from another Pitman–Yor process. Assume we have M groups of grouped data, and each observation inside each group j is distributed according to a Pitman–Yor process G_j. The indexed set of G_j of all groups shares a common base distribution G_0, which follows a Pitman–Yor process: $G_0 \sim \mathrm{PY}(\eta, \gamma, H)$ and $G_j \sim \mathrm{PY}(\varphi, \lambda, G_0)$ where $j \in \{1, \ldots, M\}$ [50].
The stick-breaking construction can also be used to depict the hierarchical Pitman–Yor process like what we did for HDP [26, 46]. Therefore, for the global level (G_0) we have:

$$G_0 = \sum_{k=1}^{\infty} \psi_k \delta_{\Omega_k}, \quad \Omega_k \sim H, \tag{2.78}$$
$$\psi_k = \psi'_k \prod_{s=1}^{k-1} (1 - \psi'_s), \quad \psi'_k \sim \mathrm{Beta}(1 - \eta, \gamma + k\eta)$$

Similarly, we have another stick breaking construction for the local level (G_j):

$$G_j = \sum_{t=1}^{\infty} \pi_{jt} \delta_{\varpi_{jt}}, \quad \varpi_{jt} \sim G_0, \tag{2.79}$$
$$\pi_{jt} = \pi'_{jt} \prod_{s=1}^{t-1} (1 - \pi'_{js}), \quad \pi'_{jt} \sim \mathrm{Beta}(1 - \varphi, \lambda + t\varphi)$$

Then, as an indicator variable for each ω_{jt}, we can define a binary latent variable $W_{jtk} \in \{0, 1\}$ in which $W_{jtk} = 1$, if ω_{jt} maps onto the base-level atom Ω_k and otherwise $W_{jtk} = 0$. The indicator variable $\vec{W} = (W_{jt1}, W_{jt2}, ...)$ is conditionally distributed as (2.3). Therefore, the prior distribution of ψ can

be calculated using [50]:

$$p(\vec{\psi'}) = \prod_{k=1}^{\infty} \frac{\Gamma(1-\eta_k+\gamma_k+k\eta_k)}{\Gamma(1-\eta_k)\Gamma(\gamma_k+k\eta_k)}(1-\psi'_k)^{\gamma_k+k\eta_k-1}\psi_k'^{-\eta_k} \tag{2.80}$$

The hierarchical Pitman–Yor process mixture model's configuration is as follows:
Let's consider θ_{ji} variables that are distributed according to the Pitman–Yor process G_j where $\vec{\theta}_j = (\theta_{j1}, \theta_{j2}, \dots)$, and each variable is a factor corresponding to an observation X_{ji}. The likelihood function can therefore be written as: $\theta_{ji} \mid G_j \sim G_j$ and $X_{ji} \mid \theta_{ji} \sim F(\theta_{ji})$. Here, the distribution of the observation X_{ji} given θ_{ji} is represented by $F(\theta_{ji})$.The base distribution H provides the prior distribution for the θ factors. Each group j is linked with an infinite mixture model. Since each factor θ_{ji} follows G_j's distribution, it takes the value Ω_{jt} with probability π_{jt}. Therefore, for each θ_{ji} we can consider a binary latent variable $Z_{jit} \in \{0,1\}$ in which $Z_{jit} = 1$ if θ_{ji} is linked with component t otherwise $Z_{jit} = 0$. Thus, we have $\theta_{ji} = \varpi_{jt}{}^{Z_{jit}}$. Consequently, θ_{ji} can be obtained by $\theta_{ji} = \Omega_k{}^{W_{jtk}Z_{jit}}$. Moreover, $\vec{Z} = (Z_{ji1}, Z_{ji2}, \dots)$ is conditionally distributed given $\vec{\pi}$ as (2.6).
As a result, the prior distribution of π can be computed using the following formula [50]:

$$p(\vec{\pi'}) = \prod_{j=1}^{M}\prod_{t=1}^{\infty} \frac{\Gamma(1-\varphi_{jt}+\lambda_{jt}+t\varphi_{jt})}{\Gamma(1-\varphi_{jt})\Gamma(\lambda_{jt}+t\varphi_{jt})} \times (1-\pi'_{jt})^{\lambda_{jt}+t\varphi_{jt}-1}\pi_{jt}'^{-\varphi_{jt}} \tag{2.81}$$

2.5 Variational Learning of HPYP Mixtures of SSD Distributions

2.5.1 Batch Variational Learning

Following section 2.3 where we applied a variational learning algorithm to learn the HDP model, in this section we develop another variational learning framework to learn the HPYP model with two different settings, batch and online. We consider the truncation level K for the variational approximation

of G_0: ${\psi_K}' = 1$, $\sum_{k=1}^{K} \psi_k = 1$, and $\psi_k = 0$ when $k > K$. We also truncate the variational approximation of G_j at T: ${\pi_{jT}}' = 1$, $\sum_{t=1}^{T} \pi_{jt} = 1$, and $\pi_t = 0$ when $t > T$. In addition, we use a factorization assumption to factorize $Q\big(\Theta\big)$ into disjoint tractable components, as (2.19). Therefore, we can obtain the update equations using (2.29) to (2.35). Following formulas may be used to calculate the hyper parameters of the variational factors. The expected values are given by (2.50):

$$a^{\star}_{jt} = 1 + \sum_{i=1}^{N} \langle Z_{jit} \rangle - \varphi_{jt} \tag{2.82}$$

$$b^{\star}_{jt} = \lambda_{jt} + \sum_{i=1}^{N} \sum_{s=t+1}^{T} \langle Z_{jis} \rangle + t\varphi_{jt} \tag{2.83}$$

$$c^{\star}_{k} = 1 + \sum_{j=1}^{M} \sum_{t=1}^{T} \langle W_{jtk} \rangle - \eta_k \tag{2.84}$$

$$d^{\star}_{k} = \gamma_k + \sum_{j=1}^{M} \sum_{t=1}^{T} \sum_{s=k+1}^{K} \langle W_{jts} \rangle + k\eta_k \tag{2.85}$$

$$\begin{aligned} u^{\star}_{kl} = u_{kl} + \sum_{j=1}^{M} \sum_{t=1}^{T} \langle W_{jtk} \rangle \sum_{i=1}^{N} \big\langle Z_{jit} \big\rangle \overline{\alpha}_{kl} [\psi(\sum_{s=1}^{D} \overline{\alpha}_{ks}) \\ - \psi\big(\overline{\alpha}_{kl}\big) + \sum_{s \neq d}^{D} \psi'(\sum_{s=1}^{D} \overline{\alpha}_{ks}) \times \overline{\alpha}_{ks}(\big\langle \ln \alpha_{ks} \big\rangle - \ln \overline{\alpha}_{ks})] \end{aligned} \tag{2.86}$$

$$\begin{aligned} \nu^{\star}_{kl} = \nu_{kl} - \sum_{j=1}^{M} \sum_{t=1}^{T} \langle W_{jtk} \rangle \sum_{i=1}^{N} \big\langle Z_{jit} \big\rangle \times \\ [\frac{1}{\tau_k} \ln \frac{\beta_{kl}}{x_{jil}} + \ln (\sum_{s=1}^{D} (\frac{x_{jis}}{\beta_{ks}})^{\frac{1}{\tau_k}})] \end{aligned} \tag{2.87}$$

$$h^{\star}_{kl} = h_{kl} + \sum_{j=1}^{M}\sum_{t=1}^{T}\langle W_{jtk}\rangle \sum_{i=1}^{N}\langle Z_{jit}\rangle \times \tag{2.88}$$
$$[\frac{-\bar{\alpha}_{kl}}{\bar{\tau}_k} + \frac{\bar{\alpha_{kl}}}{\bar{\tau}_k} \times (\frac{x_{jil}}{\bar{\beta_{kl}}})^{\frac{1}{\bar{\tau}_k}} \times \frac{1}{\sum_{s=1}^{D}(\frac{x_{jis}}{\bar{\beta_{js}}})^{\bar{\tau}_k - 1}}]$$

$$q^{\star}_{k} = q_{k} + \sum_{j=1}^{M}\sum_{t=1}^{T}\langle W_{jtk}\rangle \sum_{i=1}^{N} Z_{jit} \times \tag{2.89}$$
$$[1 - D + \frac{(\alpha_k +)}{\tau_k}\frac{\sum_{l=1}^{D}(\frac{x_{jil}}{\beta_{kl}})^{\tau_k^{-1}} \ln(\frac{x_{jil}}{\beta_{kl}})}{\sum_{l=1}^{D}(\frac{x_{jil}}{\beta_{kl}})^{\tau_k^{-1}}}]$$

$$s^{\star}_{k} = s_{k} - \sum_{j=1}^{M}\sum_{t=1}^{T}\langle W_{jtk}\rangle \sum_{i=1}^{N} Z_{jit}[\sum_{l=1}^{D}\frac{\alpha_{kl}}{{\tau_k}^2}\ln(\frac{x_{jil}}{\beta_{kl}})] \tag{2.90}$$

Algorithm 3 shows the procedure of learning our proposed model.

Algorithm 3 Batch variational learning of HPYP mixtures of SSD distributions

---

1. Choose the initial values for K, T,$\{\varphi_{jt}\}$,$\{\eta_k\}$, $\{\lambda_{jt}\}$, $\{\gamma_k\}$, $\{u_{kl}\}$, $\{\nu_{kl}\}$, $\{h_{kl}\}$, $\{q_k\}$, $\{s_k\}$, and initialize the value of ρ_{jit} with the K-means algorithm.

2. **repeat**

3. The variational E-step: Estimate the values of equation (2.50) using (2.82) to (2.90).

4. The variational M-step: Update the variational solutions using (2.29) to (2.35).

5. **until** Convergence criterion is reached.

2.5.2 Online Variational Learning

In this part, we develop an online variational inference approach to learn the proposed HPYP mixture model. As we mentioned before, when working with large-scale or streaming data, online algorithms are more efficient than batch learning algorithms. In online variational inference the goal is to maximize the current variational lower bound for the observed data (r) according to (2.51). In addition, we consider K and T truncation levels like we did in the previous section. Now we maximize $\mathcal{L}^{(r)}(Q)$ with respect to each factor of (2.19) while other factors will stay constant at their $r-1$ values. Therefore, we can calculate $Q^{(r)}(\vec{Z})$, $Q^{(r)}(\vec{W})$, $Q^{(r)}(\vec{\pi}')$, $Q^{(r)}(\vec{\psi}')$, $Q^{(r)}(\vec{\alpha})$, $Q^{(r)}(\vec{\beta})$, and $Q^{(r)}(\vec{\tau})$ using (2.52), and (2.55) to (2.63). Also, the hyper parameters are given by (2.64) to (2.68) and their natural gradients are updated using following equations:

$$\Delta a_{jt}^{\star(r)} = 1 + N\rho_{jtr} - a_{jt}^{(r-1)} - \varphi_{jt} \tag{2.91}$$

$$\Delta b_{jt}^{\star(r)} = \lambda_{jt} + N \sum_{s=t+1}^{T} \rho_{jsr} - b_{jt}^{(r-1)} + t\varphi_{jt} \tag{2.92}$$

$$\Delta c_{k}^{\star(r)} = 1 + \sum_{j=1}^{K}\sum_{t=1}^{T} \vartheta_{jtk}^{(r)} - c_{k}^{(r-1)} - \eta_k \tag{2.93}$$

$$\Delta d_{k}^{\star(r)} = \gamma_k + \sum_{j=1}^{K}\sum_{t=1}^{T}\sum_{s=k+1}^{K} \vartheta_{jts}^{(r)} - d_{k}^{(r-1)} + k\eta_k \tag{2.94}$$

$$\begin{aligned}\Delta u_{kl}^{\star(r)} &= u_{kl} + N \sum_{j=1}^{M}\sum_{t=1}^{T} \vartheta_{jtk}^{(r)}\rho_{jtr}\overline{\alpha}_{kl}^{(r-1)}\times \\ &[\psi(\sum_{s=1}^{D}\overline{\alpha}_{ks}^{(r-1)}) - \psi\big(\overline{\alpha}_{kl}^{(r-1)}\big) + \sum_{s\neq d}^{D}\psi'(\sum_{s=1}^{D}\overline{\alpha}_{ks}^{(r-1)})\times \\ &\overline{\alpha}_{ks}^{(r-1)}(\big\langle \ln \alpha_{ks}^{(r-1)}\big\rangle - \ln \overline{\alpha}_{ks}^{(r-1)})] - u_{kl}^{(r-1)}\end{aligned} \tag{2.95}$$

$$\Delta\nu_{kl}^{\star(r)} = \nu_{kl} - \nu_{kl}^{(r-1)} - N\sum_{j=1}^{M}\sum_{t=1}^{T}\vartheta_{jtk}^{(r)}\rho_{jtr}\times \tag{2.96}$$
$$[\frac{1}{\tau_k^{(r-1)}}\ln\frac{\beta_{kl}^{(r-1)}}{x_{jil}} + \ln(\sum_{s=1}^{D}(\frac{x_{jis}}{\beta_{ks}^{(r-1)}})^{\tau_k^{-(r-1)}})]$$

$$\Delta h_{kl}^{\star(r)} = h_{kl} + N\sum_{j=1}^{M}\sum_{t=1}^{T}\vartheta_{jtk}^{(r)}\rho_{jtr}\frac{\bar{\alpha}_{kl}^{(r-1)}}{\tau_k^{(r-1)}} \tag{2.97}$$
$$\times[-1 + \frac{\bar{\alpha}_{kl}^{(r-1)}(\frac{x_{jil}}{\beta_{kl}^{(r-1)}})^{\tau_k^{-(r-1)}}}{\tau_k^{(r-1)}\sum_{s=1}^{D}(\frac{x_{jis}}{\beta_{ks}^{(r-1)}})^{\tau_k^{-(r-1)}}}] - h_{kl}^{(r-1)}$$

$$\Delta q_k^{\star(r)} = q_k - q_k^{(r-1)} + N\sum_{j=1}^{M}\sum_{t=1}^{T}\vartheta_{jtk}^{(r)}\rho_{jtr}\times$$
$$[1 - D + \frac{\alpha_k^{+(r-1)}\sum_{l=1}^{D}(\frac{x_{jil}}{\beta_{kl}^{(r-1)}})^{\tau_k^{-(r-1)}}\ln(\frac{x_{jil}}{\beta_{kl}^{(r-1)}})}{\tau_k^{(r-1)}\sum_{l=1}^{D}(\frac{x_{jil}}{\beta_{kl}^{(r-1)}})^{\tau_k^{-(r-1)}}}] \tag{2.98}$$

$$\Delta s_k^{\star(r)} = s_k - N\sum_{j=1}^{M}\sum_{t=1}^{T}\vartheta_{jtk}^{(r)}\rho_{jtr}\times \tag{2.99}$$
$$[\sum_{l=1}^{D}\frac{\alpha_{kl}^{(r-1)}}{\tau_k^{(r-1)2}}\ln(\frac{x_{jil}}{\beta_{kl}^{(r-1)}})] - s_k^{(r-1)}$$

The algorithm of online variational learning of HPYP mixtures of SSD distributions is shown in Algorithm 4.

Algorithm 4 Online variational learning of HPYP mixtures of SSD distributions.

1. Choose the initial truncation level K, T and the values for hyper parameters $\{\lambda_{jt}\}$, $\{\gamma_k\}$, $\{u_{kl}\}$, $\{\nu_{kl}\}$, $\{h_{kl}\}$, $\{q_k\}$, and $\{s_k\}$.
2. **for** $r = 1 \rightarrow N$ **do**
3. Update the variational solution for $Q^{(r)}(\vec{Z})$ using (2.52).
4. Compute learning rate by $\xi_r = (\tau + r)^{-\xi}$ and natural gradient $\Delta\vartheta^{(r)}_{jtk}$.
5. Calculate the natural gradients of remaining hyper parameters using (2.91) to (2.99).
6. Update the variational solutions for $Q^{(r)}(\vec{W})$,$Q^{(r)}(\vec{\pi}')$, $Q^{(r)}(\vec{\psi}')$, $Q^{(r)}(\vec{\alpha})$, $Q^{(r)}(\vec{\beta})$, and $Q^{(r)}(\vec{\tau})$.
7. Repeat the variational until new data is observed.
8. **end for**

2.6 Experimental Results

In this section, we investigate the effectiveness of our model by applying it on four real-life applications, namely: spam email detection, traffic sign detection, vehicle detection, and texture clustering. Altogether, we compare the results of seven models including our proposed models: BV-HDP-SSD, OV-HDP-SSD, BV-HPYP-SSD, OV-HPYP-SSD, Gaussian mixture mode (GMM), batch variational learning of HDP mixtures of Dirichlet distributions (BV-HDP-DMM), and the batch variational learning of HDP mixtures of SD distributions (BV-HDP-SDMM). To extract the image features, we applied two different feature extraction approaches, VGG16 [51] and scale-invariant feature transform (SIFT). The VGG16 method is used for the vehicle detection and traffic sign detection datasets because of their complex images and the capacity of VGG16 to extract shapes in images [52]. In addition to this, we employed SIFT [53] and the bag of visual words technique [54] for feature extraction in the texture clustering application. Also, we applied principal component analysis (PCA) to reduce data dimensionality.

As our model is a clustering approach, we first remove the labels to obtain unlabeled datasets and then, with the help of our model, we find the predicted clusters. Finally, we employ four popular metrics to test the performance of our model using the original and predicted clusters:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.100}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.101}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.102}$$

$$F1_{score} = 2 * \frac{precision * recall}{precision + recall} \tag{2.103}$$

where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively.

2.6.1 Spam Email Filtering

In recent years, email has played a major role in most people's lives and business communications. A significant proportion of emails however is spam (junk emails). According to recent research, each user receives 40-50 emails per day and more than 50 percent of them are spam [55]. These emails are undesired messages that are sent *en masse* and contain advertisements, irrelevant content or attempted fraud [56, 57]. Due to ever-improving spamming tactics, filtering such emails is increasingly difficult. In such cases, machine learning has been found to be effective [58]. In this thesis, we used the spambase dataset to test our models [59]. This dataset has 57 features and 4,601 observations where 2,788 observations are labeled as non-spam emails and 1,813 as spam emails. To have a better model with higher accuracy, we balanced data with the same number of observations for each category. We

randomly selected 1,813 samples of non-spam emails. In the online variational setting, we employed a method called mini-batch introduced in [47]. This method helps the stability of the online learning process. For the spambase application, we created sets of 5 observations as mini-batches and fed our models with them in each iteration according to Algorithm 2 and 4. The results of the different tested methods are illustrated in table 2.1. Our models, BV-HDP-SSD, OV-HDP-SSD, BV-HPYP-SSD, and OV-HPYP-SSD have the most promising results with 84.4% , 91.3%, 96%, and 96% accuracy, respectively. As we can see in this table, taking SSD as the parent distribution increased the performance significantly compared to Dirichlet and SD distributions. Also, the HPYP frameworks with SSD mixtures work better than HDP frameworks with the same mixture model. Furthermore, our models outperform the GMM in all of the four metrics.

Table 2.1: Spam email results

Method	***Accuracy***	***Precision***	***Recall***	***F*1*-score***
BV-HDP-SSD	84.4	83.3	82.3	82.79
BV-HDP-SDMM	71	71.07	71.02	71.04
BV-HDP-DMM	63.5	62.41	61.3	61.8
BV-HPYP-SSD	96	92.31	85.7	88.88
OV-HDP-SSD	91.3	91.54	89.88	90.7
OV-HPYP-SSD	96	96	95.9	95.97
GMM	69.54	68.11	66.28	67.18

2.6.2 Traffic Sign Recognition

Traffic sign recognition has a critical role in driver assistance technologies, autonomous vehicle systems, and road maintenance [60]. This task has always been a challenging issue due to contextual environmental factors like background, brightness, and blurriness that can affect the quality of images. The growth of autonomous vehicles has led to greater attention to this application. Developing models to cluster signs would be very helpful in the transportation industry and smart vehicle technology. We used the "German

Traffic Sign Recognition" dataset [61], which contains 51,840 images. This dataset was created by recording video for 10 hours on German roads [62]. We try to cluster stop and speed limit signs. In figure 2.1 samples of the German traffic sign dataset are shown. We randomly selected 270 images of the stop signs and 270 images of the speed limit signs to feed our model with balanced data. 3,000 features were extracted by VGG16 and then, 100 of them selected by PCA. The results of comparing seven methods are presented in table 2.2. For the OV-HDP-SSD and OV-HPYP-SSD models, we used 10 images in each mini-batch. Online variational learning provides the capability of learning streaming data, something very helpful for controlling smart vehicles and providing driver assistance using real time data. According to table 2.2, the accuracy of BV-HDP-SSD, OV-HDP-SSD, BV-HPYP-SSD, and OV-HPYP-SSD models are 94.8%, 96.6%, 95.1%, and 99.25% respectively. Therefore, the performance of HPYP framework is better than HDP framework. In addition, the SSD mixture model fits the data better compared with the SD and Dirichlet mixture models leading to higher accuracy, precision, recall and F1-score. It is worth mentioning that the OV-HDP-SSD and OV-HPYP-SSD outperform GMM on all metrics.

Figure 2.1: Samples of German traffic sign recognition dataset.

Table 2.2: German traffic sign results

Method	***Accuracy***	***Precision***	***Recall***	***F*1*-score***
BV-HDP-SSD	94.8	94.9	94.8	94.8
BV-HDP-SDMM	94.48	94.6	94.48	94.53
BV-HDP-DMM	94.25	94.58	94.25	94.41
BV-HPYP-SSD	95.1	95	95.1	95.1
OV-HDP-SSD	96.6	95.52	94.88	95.19
OV-HPYP-SSD	99.25	99.25	99.26	99.26
GMM	96.1	95.3	93.44	94.36

2.6.3 Vehicle Detection

With traffic increasing the world over, transportation planners are trying to solve traffic problems using different methods and technologies such as smart traffic monitoring systems. Cameras play a central role in the monitoring process [63]. Recognizing vehicle type such as bus and car using images taken by cameras can be very helpful for traffic planning and management, as well for law enforcement. For instance, police can use such recognition systems to detect cars using dedicated bus lanes. In this work, we use a vehicle dataset that is available in [64] and which includes two vehicle types: buses and cars (figure 2.2). It consists of 1,868 images and the number of images for each category is 934. With the help of VGG16, 2,100 features were extracted and reduced to 46 using PCA. Sets of 20 images were selected as mini-batches for the online setting. According to table 2.3, the accuracy of BV-HDP-SSD and OV-HDP-SSD models are 80.6% and 81.75%, respectively. Also using the HPYP framework models, BV-HPYP-SSD and OV-HPYP-SSD, the accuracy increased to 86% and 87.5 % respectively. These amounts are higher than the other models we evaluated and therefore, we can see the effectiveness of the SSD in comparison with the SDMM, DMM, and GMM, especially when applying HPYP framework.

Table 2.3: Vehicle detection results

Method	***Accuracy***	***Precision***	***Recall***	***F*1*-score***
BV-HDP-SSD	80.6	80.80	80.50	80.65
BV-HDP-SDMM	68	68.8	66.3	67.52
BV-HDP-DMM	64	64.22	63.59	63.9
BV-HPYP-SSD	86	82.1	74	77.83
OV-HDP-SSD	81.75	86.6	81.75	84.1
OV-HPYP-SSD	87.5	90	87.5	87.29
GMM	80.5	85.33	80.49	82.83

Figure 2.2: Samples of vehicle detection dataset.

2.6.4 Texture Clustering

In this part, we test our model using a challenging application; texture clustering. Textures are used to find a specific object, pattern, or region in images. We used the "Describable Textures Dataset" (DTD) [65], which has 5,640 images in 47 different classes. The dataset was categorized by humans and each class has 120 images of size 300×300 to 640×640. We selected 3 categories from this dataset (360 images) to demonstrate the effectiveness of our model. Samples from these classes are shown in figure 2.3. Using SIFT and PCA, we obtained a 200-dimensional vector for each image. For the online variational setting, we used mini-batches with a size of 5 images. According to table 2.4 that contains the detailed results, we obtained 71.6% accuracy for BV-HDP-SSD, 74.88% for OV-HDP-SSD, 73% for BV-HPYP-SSD, and 83% for OV-HPYP-SSD which in comparison to GMM with 68% accuracy, our models have better results. Also it shows that our models have outperformed both BV-HDP-SDMM and BV-HDP-DMM. As such, it indicates that choosing SSD as the parent distribution helped our model to

better fit the data than the SD or Dirichlet. Also, if we want to be more specific, the F1-scores of our models are higher than other tested models. Therefore, they seem to be a better choice for clustering texture images.

Figure 2.3: Samples of DTD dataset.

Table 2.4: Texture clustering results

Method	***Accuracy***	***Precision***	***Recall***	***F*1*-score***
BV-HDP-SSD	71.6	73.26	71.28	72.25
BV-HDP-SDMM	68.99	71.1	68.65	69.85
BV-HDP-DMM	61.49	64.61	61.90	63.22
BV-HPYP-SSD	73	72	72	72
OV-HDP-SSD	74.88	76.22	73.83	75
OV-HPYP-SSD	83	76.2	71.8	73.93
GMM	68	71.36	67.61	69.43

Chapter 3

Shifted-Scaled-Dirichlet Based Hierarchical Dirichlet Process Hidden Markov Models with Variational Inference Learning

3.1 Hidden Markov Models

Hidden Markov Models are a common technique for modelling time series data. They have been used in speech recognition systems [66], text clustering [67], and pattern recognition applications [37] for decades. We can consider HMMs as a generalized version of mixture models. That is, the probability density functions produced by an HMM across all observable states can be seen as a mixture of densities formed by each state [68, 69]. The hidden Markov model is defined by two basic features; first, it presupposes that an observation at time t is the result of a process in state h_t which is hidden from the observer. Second, the present state h_t, given the value of $h_t - 1$ is independent of all previous states of time $t - 1$. The second feature is called the Markov property. To develop our HMM model, we introduce some notation. $\mathcal{X} = \{X_1, \ldots, X_T\}$ is the generated sequence of observations by hidden states $S = \{s_1, \ldots, s_t, \ldots, s_T\}$ and $s_t \in [1, N]$ where N is the number of states. $A = \{A_{ii'} = P(s_t = i' \mid s_{t-1} = i)\}$ is transition probabilities matrix that presents the probabilities of transition between the

states. $C = \{C_{ij} = P(m_t = j \mid s_t = i)\}$ is the emission probabilities matrix for continuous case (CHMM) where $i \in [1, N]$, and $j \in [1, K]$. The number of mixture components in set $L = \{m_1, \ldots, m_t, \ldots, m_K\}$ is denoted by K which is assumed to be uniform for all the states. The initial probabilities vector, π_i, represents the probability of starting the observation sequence from state i. In summary, using all the above-mentioned notation, an HMM is defined as $\lambda = \{A, C, \pi, \Theta\}$ where Θ is the set of mixture model parameters [29].

3.2 Variational Learning

In this section, we derive the equations of the variational learning approach to update our models parameters. As we mentioned in before, emission probability distributions for continuous observations are frequently assumed to have a Gaussian distribution [70, 71]. The ability of the Dirichlet and scaled Dirichlet (SD) mixture models to fit proportional data motivated us to use a more general form of them called the shifted-scaled Dirichlet mixture model as the emission probability for HMM [17–25, 72]. First, we derive the variational learning equations for SSD-based HMM, and then, we derive the equations of the SSD-based HDP HMM model.

3.2.1 Shifted-Scaled Dirichlet Based Hidden Markov Model

In varitional learning, all of the parameters including HMM parameters (A, C, and π) and emission distribution parameters $\theta = \{\alpha_{ijl}, \beta_{ijl}, \tau_{ij}\}$ are treated as random variables. We consider $\mathcal{X} = (\overrightarrow{X}_1, \ldots, \overrightarrow{X}_t, \ldots, \overrightarrow{X}_T)$ as a set of T independent identically distributed observations in which, each $\overrightarrow{X}_t$ is a D-dimensional positive vector which is generated from a mixture of SSD distributions. The finite SSD mixture model which is a linear combination of K components is expressed in (2.9). Also, the likelihood function of the SSD mixture model is written in (2.10). $p(\overrightarrow{X}_t \mid \overrightarrow{\theta}_j)$ is a mixture component with parameter θ_j that in our model is an SSD distribution. We expressed this distribution in (2.8), but by considering HMM notations, we can rewrite

it as follows [72]:

$$p(\overrightarrow{X_t} \mid \overrightarrow{\theta_j}) = \frac{\Gamma(\alpha_{ij}+)}{\prod_{l=1}^{D}\Gamma(\alpha_{ijl})}\frac{1}{\tau_{ij}^{D-1}}\frac{\prod_{l=1}^{D}\beta_{ijl}^{-\frac{\alpha_{ijl}}{\tau_{ij}}}X_{tl}^{(\frac{\alpha_{ijl}}{\tau_{ij}}-1)}}{\left(\sum_{l=1}^{D}(\frac{X_{tl}}{\beta_{ijl}})^{\frac{1}{\tau_{ij}}}\right)^{\alpha_{ij}+}} \tag{3.1}$$

where $\alpha = \{\alpha_{ijl}\}_{i,j,l}^{N,K,D}$, $\beta = \{\beta_{ijl}\}_{i,j,l}^{N,K,D}$, and $\tau = \{\tau_{ij}\}_{i,j}^{N,K}$ are positive SSD parameters. Also $X_t > 0$, $\sum_{l=1}^{D} X_{tl} = 1$, and $\alpha_{ij}+ = \sum_{l=1}^{D}\alpha_{ijl}$. $\Gamma(.)$ indicates the Gamma function which is $\Gamma(t) = \int_0^\infty x^{t-1}e^{-x}dx$.
For each X_{vt} (where v is the vth observed vector), we introduce a latent variable Z_{vij} that shows which cluster and state are assigned to X_{vt}. In other words, $Z_{vij} = 1$ if the X_{vt} belongs to state i and cluster j, otherwise $Z_{vij} = 0$. Also, Z_{vij} must satisfy this condition $\sum_{j=1}^{K} Z_{vij} = 1$.

In HMMs, the probability of the complete data can be stated as follows for given model parameters:

$$p(X, S, L \mid A, C, \pi, \theta) = \pi_{s_1}\left[\prod_{t=2}^{T} A_{s_{t-1},s_t}\right]\left[\prod_{t=1}^{T} C_{s_t,m_l} p\left(X_t \mid \theta_{s_t,m_l}\right)\right] \tag{3.2}$$

where X is a sequence of T observations, S is the set of hidden states, and L stands for the set of mixture components. It is worth mentioning that for the sake of simplification, the model is derived for a single observation series. In order to incorporate more observation sequences (which is recommended to prevent overfitting), the related equations need to be updated to include a summation of these sequences. Therefore the likelihood of X given model parameters is expressed as follows:

$$p(X \mid A, C, \pi, \theta) = \sum_{S}\sum_{L}\pi_{s_1}\left[\prod_{t=2}^{T} A_{s_{t-1},s_t}\right]\left[\prod_{t=1}^{T} C_{s_t,m_l} p\left(X_t \mid \theta_{s_t,m_l}\right)\right] \tag{3.3}$$

A precise computation of this equation is impossible because it requires the summation of all possible combinations of mixture components and states.

The most common approach for solving it is to use the Baum–Welch algorithm to maximize the data likelihood with regard to the model parameters [29]. However, there are several flaws with this strategy such as the potential for overfitting and the lack of a convergence guarantee. Another tested solution to compute intractable equations (3.3) is variational learning. This approach calculates the marginal likelihood of data using an approximate distribution Q of the true posterior p. In the SSD-HMM model, data marginal likelihood is expressed as:

$$p(X) = \int dAdCd\pi d\alpha d\beta d\tau \sum_{S,L} p(A, C, \pi, \alpha, \beta, \tau) \times p(X, S, L \mid A, C, \pi, \alpha, \beta, \tau) \tag{3.4}$$

The variational learning is based on (2.12) where $\mathcal{L}(Q)$ is the variational lower bound for $\ln p(X)$ defined in (2.13). Kullback-Leibler divergence between the approximation Q and the posterior p is represented by (2.14) where $\Theta = \{A, C, \pi, \alpha, \beta, \tau, S, L\}$. Minimizing KL allows the best approximation of the true posterior p and due to the fact that $KL \geq 0$, this can be accomplished by maximizing $\mathcal{L}(Q)$. Having Q and (2.13), we can take the lower bound as follows [39]:

$$\begin{aligned}
\ln(p(X)) = \ln\{ & \int dAdCd\pi d\alpha d\beta d\tau \sum_{S,L} p(A, C, \pi, \alpha, \beta, \tau) \\
& \times p(X, S, L \mid A, C, \pi, \alpha, \beta, \tau)\} \\
& \geq \int dAdCd\pi d\alpha d\beta d\tau \sum_{S,L} Q(A, C, \pi, \alpha, \beta, \tau, S, L) \\
& \times \ln \left\{ \frac{p(A, C, \pi, \alpha, \beta, \tau) p(X, S, L \mid A, C, \pi, \alpha, \beta, \tau)}{Q(A, C, \pi, \alpha, \beta, \tau, S, L)} \right\}
\end{aligned} \tag{3.5}$$

Now, using the mean-field assumption [73], we take a restricted family of distributions to be able to calculate $Q(\Theta)$. Therefore, we have factorized $Q(\Theta)$:

$$Q(A, C, \pi, \alpha, \beta, \tau, S, L) = Q(A)Q(C)Q(\pi)Q(\alpha)Q(\beta)Q(\tau)Q(S, L) \tag{3.6}$$

Using (3.5) and (3.6), the lower bound can be written as follows:

$$
\begin{aligned}
&\ln p(X) \geq \sum_{S,L} \int dAdCd\pi d\alpha d\beta d\tau Q(\pi)Q(A)Q(C)Q(\alpha)Q(\beta)Q(\tau)Q(S,L) \\
&\times \{\ln(p(\pi)) + \ln(p(A)) + \ln(p(C)) + \ln(p(\alpha)) + \ln(p(\beta)) + \ln(p(\tau)) \\
&+ \ln\left(\pi_{s_1}\right) + \sum_{t=2}^{T} \ln\left(A_{s_{t-1},s_t}\right) + \sum_{t=1}^{T} \ln\left(C_{s_t,m_t}\right) + \sum_{t=1}^{T} \ln\left(f\left(X_t \mid \theta_{s_t,m_t}\right)\right) \\
&- \ln(Q(S,L)) - \ln(Q(A)) - \ln(Q(\pi)) - \ln(Q(C)) - \ln(Q(\alpha)) \\
&- \ln(Q(\beta)) - \ln(Q(\tau))\} = F(Q(\pi)) + F(Q(C)) + F(Q(A)) \\
&+ F(Q(S,L)) + F(Q(\alpha)) + F(Q(\beta)) + F(Q(\tau))
\end{aligned} \tag{3.7}
$$

The above lower bound, in general, has several maxima; hence, it is not convex. As a result, the solution depends on initialization. We are now going to define prior distributions for model parameters to be able to evaluate (3.7). The priors for the parameters A, C, and π are selected as Dirichlet distributions $\mathcal{D}$ since their coefficients are positive and less than one:

$$p(\pi) = \mathcal{D}(\pi_1, \ldots, \pi_N \mid \phi_1^{\pi}, \ldots, \phi_N^{\pi}) \tag{3.8}$$

$$p(A) = \prod_{i=1}^{N} \mathcal{D}(A_{i_1}, \ldots, A_{i_N} \mid \phi_{i_1}^{A}, \ldots, \phi_{i_N}^{A}) \tag{3.9}$$

$$p(C) = \prod_{i=1}^{N} \mathcal{D}(C_{i_1}, \ldots, C_{i_K} \mid \phi_{i_1}^{C}, \ldots, \phi_{i_K}^{C}) \tag{3.10}$$

The conjugate priors for SSD parameters is chosen as follows [74]:

$$p(\alpha_{ijl}) = \mathcal{G}(\alpha_{ijl} \mid u_{ijl}, \nu_{ijl}) = \frac{\nu_{ijl}^{u_{ijl}}}{\Gamma(u_{ijl})} \alpha_{ijl}^{u_{ijl}-1} e^{-\nu_{ijl}\alpha_{ijl}} \tag{3.11}$$

$$p(\beta_{ijl}) = \mathcal{D}(\beta_{ijl} \mid \overrightarrow{h_{ij}}) = \frac{\Gamma(\sum_{l=1}^{D} h_{ijl})}{\prod_{l=1}^{D} \Gamma(h_{ijl})} \prod_{l=1}^{D} \beta_{ijl}^{h_{ijl}-1} \tag{3.12}$$

$$p(\tau_{ij}) = \mathcal{G}(\tau_{ij} \mid q_{ij}, s_{ij}) = \frac{q_{ij}^{s_{ij}}}{\Gamma(q_{ij})} \tau_{ij}^{q_{ij}-1} e^{-s_{ij}\tau_{ij}} \tag{3.13}$$

where u_{ijl}, ν_{ijl}, h_{ijl}, q_{ij} and s_{ij} are positive SSD hyper parameters and $\mathcal{G}$ is the Gamma distribution. Since the variables are considered statistically independent, the prior distributions for SSD parameters are:

$$p(\overrightarrow{\alpha}) = \prod_{i=1}^{N} \prod_{j=1}^{K} \prod_{l=1}^{D} p(\alpha_{ijl}) \tag{3.14}$$

$$p(\overrightarrow{\beta}) = \prod_{i=1}^{N} \prod_{j=1}^{K} \prod_{l=1}^{D} p(\beta_{ijl}) \tag{3.15}$$

$$p(\overrightarrow{\tau}) = \prod_{i=1}^{N} \prod_{j=1}^{K} p(\tau_{ij}) \tag{3.16}$$

Now we can optimize each factor $F(Q(.))$ in (3.7) by maximizing the lower bound with respect to that factor because they are independent of each other. First we will optimize $Q(\pi)$, $Q(A)$, and $Q(C)$ which are independent of the SSD parameters and have already been studied [39,75,76]. Therefore, according to the previous works, we can optimize $Q(A)$ using the Gibbs inequality in the following procedure:

$$F(Q(A)) = \int dA Q(A) \ln \left[\frac{\prod_{i=1}^{N} \prod_{i'=1}^{N} A_{ii'}^{w_{ii'}^{A}-1}}{Q(A)} \right] \tag{3.17}$$

$$Q(A) = \prod_{i=1}^{N} \mathcal{D}\left(A_{i1}, \ldots, A_{iN} \mid w_{i1}^{A}, \ldots, w_{iN}^{A}\right) \tag{3.18}$$

where

$$w_{ij}^{A} = \sum_{t=2}^{T} \gamma_{ijt}^{A} + \phi_{ij}^{A} \tag{3.19}$$

$$\gamma_{ijt}^{A} \triangleq Q\left(s_{t-1} = i, s_t = j\right) \tag{3.20}$$

Similarly for the $Q(\pi)$ and $Q(C)$, we have:

$$Q(\pi) = \mathcal{D}\left(\pi_1, \ldots, \pi_N \mid w_1^{\pi}, \ldots, w_N^{\pi}\right) \tag{3.21}$$

$$w_i^{\pi} = \gamma_i^{\pi} + \phi_i^{\pi} \tag{3.22}$$

$$\gamma_i^{\pi} \triangleq Q\left(s_1 = i\right) \tag{3.23}$$

and

$$Q(C) = \prod_{i=1}^{N} \mathcal{D}\left(C_{i1}, \ldots, C_{iK} \mid w_{i1}^{C}, \ldots, w_{iK}^{C}\right) \tag{3.24}$$

$$w_{ij}^{C} = \sum_{t=1}^{T} \gamma_{ijt}^{C} + \phi_{ij}^{C} \tag{3.25}$$

$$\gamma_{ijt}^{C} \triangleq Q\left(s_t = i, m_t = j\right) \tag{3.26}$$

The forward-backward procedure may then be used to derive the values of responsibilities γ_{ijt}^{A}, γ_i^{π}, and γ_{ijt}^{C} [77].

The next step is optimizing $Q(\alpha)$. Using (3.14) and (3.7) we have:

$$F(Q(\alpha)) = \int d\alpha Q(\alpha) \ln\{\frac{p(\overrightarrow{\alpha}) \prod_{t=1}^{T} p(X_t \mid \alpha_{ijl})^{\gamma^{c}{}_{ijt}}}{Q(\alpha)}\} \tag{3.27}$$

similarly for $Q(\beta)$ and $Q(\tau)$, we can obtain:

$$F(Q(\beta)) = \int d\beta Q(\beta) \ln\{\frac{p(\overrightarrow{\beta}) \prod_{t=1}^{T} p(X_t \mid \beta_{ijl})^{\gamma^c{}_{ijt}}}{Q(\beta)}\} \tag{3.28}$$

and

$$F(Q(\tau)) = \int d\tau Q(\tau) \ln\{\frac{p(\overrightarrow{\tau}) \prod_{t=1}^{T} p(X_t \mid \tau_{ij})^{\gamma^c{}_{ijt}}}{Q(\tau)}\} \tag{3.29}$$

The optimal values for $Q(\alpha)$, $Q(\beta)$, and $Q(\tau)$ can be calculated by:

$$Q\left(\alpha_{ijl}\right) = \prod_{i=1}^{N}\prod_{j=1}^{K}\prod_{l=1}^{D} \mathcal{G}(\alpha_{ijl} \mid u^{\star}_{ijl}, \nu^{\star}_{ijl}) \tag{3.30}$$

$$Q\left(\beta_{ijl}\right) = \prod_{i=1}^{N}\prod_{j=1}^{K}\prod_{l=1}^{D} \mathcal{D}(\beta_{ijl} \mid \overrightarrow{h}^{\star}_{ijl}) \tag{3.31}$$

$$Q\left(\tau_{ij}\right) = \prod_{i=1}^{N}\prod_{j=1}^{K} \mathcal{G}(\tau_{ij} \mid q^{\star}_{ij}, s^{\star}_{ij}) \tag{3.32}$$

The $\star$ superscript illustrates the optimized value of these parameters. We may calculate the above hyper parameters using following formulas:

$$\begin{aligned} u^{\star}_{ijl} = u_{ijl} + \sum_{v=1}^{V} \mathbb{E}_Q Z_{vij} \times \overline{\alpha}_{ijl}[\Psi(\sum_{s=1}^{D} \overline{\alpha}_{ijs}) - \Psi\big(\overline{\alpha}_{ijl}\big) + \sum_{s\neq d}^{D} \Psi'(\sum_{s=1}^{D} \overline{\alpha}_{ijs}) \\ \times \overline{\alpha}_{ijs}(\mathbb{E}_Q \ln \alpha_{ijs} - \ln \overline{\alpha}_{ijs})] \end{aligned} \tag{3.33}$$

$$\nu^{\star}_{ijl} = \nu_{ijl} - \sum_{v=1}^{V} \mathbb{E}_Q Z_{vij} \times [\frac{1}{\tau_{ij}} \ln \frac{\beta_{ijl}}{x_{vl}} + \ln (\sum_{s=1}^{D} (\frac{x_{vs}}{\beta_{ijs}})^{\frac{1}{\tau_{ij}}})] \tag{3.34}$$

$$h^{\star}_{ijl} = h_{ijl} + \sum_{v=1}^{V} \mathbb{E}_Q Z_{vij} \times [\frac{-\bar{\alpha}_{ijl}}{\bar{\tau_{ij}}} + \frac{\bar{\alpha_{ijl}}}{\bar{\tau_{ij}}} \times (\frac{x_{vl}}{\bar{\beta_{ijl}}})^{\frac{1}{\bar{\tau_{ij}}}} \times \frac{1}{\sum_{s=1}^{D} (\frac{x_{vs}}{\bar{\beta_{ijs}}})^{\bar{\tau_{ij}^{-1}}}}] \tag{3.35}$$

$$q^{\star}_{ij} = q_{ij} + \sum_{v=1}^{V} \mathbb{E}_Q Z_{vij} \times [1 - D + \frac{(\alpha_{ij}+)}{\tau_{ij}} \frac{\sum_{l=1}^{D} (\frac{x_{vl}}{\beta_{ijl}})^{\tau_{ij}^{-1}} \ln (\frac{x_{vl}}{\beta_{ijl}})}{\sum_{l=1}^{D} (\frac{x_{vl}}{\beta_{ijl}})^{\tau_{ij}^{-1}}}] \tag{3.36}$$

$$s^{\star}_{ij} = s_{ij} - \sum_{v=1}^{V} \mathbb{E}_Q Z_{vij} \times [\sum_{l=1}^{D} \frac{\alpha_{ijl}}{{\tau_{ij}}^2} \ln (\frac{x_{vl}}{\beta_{ijl}})] \tag{3.37}$$

Also, the expected values are given by:

$$\begin{aligned}
&\bar{\alpha}_{ijl} = \mathbb{E}_Q \alpha_{ijl} = \frac{u^{\star}_{ijl}}{v^{\star}_{ijl}}, \quad \overline{\beta}_{ijl} = \mathbb{E}_Q \beta_{ijl} = \frac{h^{\star}_{ijl}}{\sum_{l=1}^{D} h^{\star}_{ijl}} \\
&\bar{\tau}_{ij} = \mathbb{E}_Q \tau_{ijl} = \frac{q^{\star}_{ij}}{s^{\star}_{ij}}, \quad \mathbb{E}_Q Z_{vij} = \sum_{t=1}^{T} \gamma^{C}_{vijt} = p(s = i, m = j \mid X) \\
&\mathbb{E}_Q \ln \alpha_{ijl} = \Psi(u^{\star}_{ijl}) - \ln v^{\star}_{ijl}
\end{aligned} \tag{3.38}$$

Finally, we optimize the value of $Q(S, L)$. We can write $F(Q(S, L))$ as follows [39]:

$$F(Q(S, L)) = \sum_{S,L} Q(S, L) \times \ln \left(\frac{\pi^{\star}_{s_1} \prod_{t=2}^{T} A^{\star}_{s_{t-1}, s_t} \prod_{t=1}^{T} C^{\star}_{s_t, m_t} p^{\star} (X_t \mid \theta_{s_t, m_t})}{Q(S, L)} \right) \tag{3.39}$$

The optimized $Q(S, L)$ is written as:

$$Q(S, L) = \frac{1}{\Omega} \pi^{\star}_{s_1} \prod_{t=2}^{T} A^{\star}_{s_{t-1}, s_t} \prod_{t=1}^{T} C^{\star}_{s_t, l_t} p^{\star} (X_t \mid \theta_{s_t, l_t}) \tag{3.40}$$

where

$$\Omega = \sum_{S,L} \pi^{\star}_{s_1} \prod_{t=2}^{T} A^{\star}_{s_{t-1},s_t} \prod_{t=1}^{T} C^{\star}_{s_t,l_t} p^{\star}\left(X_t \mid \theta_{s_t,l_t}\right) \tag{3.41}$$

and

$$\begin{aligned} \pi^{\star}_i &\triangleq \exp\left[\mathbb{E}_Q \ln\left(\pi_i\right)_{Q(\pi)}\right] \\ \pi^{\star}_i &= \exp\left[\Psi\left(w^{\pi}_i\right) - \Psi\left(\sum_i w^{\pi}_i\right)\right] \end{aligned} \tag{3.42}$$

$$\begin{aligned} A^{\star}_{jj'} &\triangleq \exp\left[\mathbb{E}_Q \ln\left(A_{jj'}\right)_{Q(A)}\right] \\ A^{\star}_{jj'} &= \exp\left[\Psi\left(w^{A}_{jj'}\right) - \Psi\left(\sum_{j'} w^{A}_{jj'}\right)\right] \end{aligned} \tag{3.43}$$

$$\begin{aligned} C^{\star}_{ij} &\triangleq \exp\left[\mathbb{E}_Q \ln\left(C_{ij}\right)_{Q(C)}\right] \\ C^{\star}_{ij} &= \exp\left[\Psi\left(w^{C}_{ij}\right) - \Psi\left(\sum_{j} w^{C}_{ij}\right)\right] \end{aligned} \tag{3.44}$$

also,

$$\ln p^{\star}\left(X_t \mid \theta_{s_t,l_t}\right) = \int Q(\theta) \ln(p\left(X_t \mid \theta_{s_t,l_t}\right)) d\theta \tag{3.45}$$

In this work, $p\left(X_t \mid \theta_{s_t,l_t}\right) = \left[SSD(\alpha,\beta,\tau)\right]^{\gamma^C_{ijt}}$. SSD is defined in (3.1). Therefore, we have:

$$\begin{aligned} \ln p^{\star}\left(X_t \mid \theta_{s_t,l_t}\right) =& \gamma^C_{ijt} \int Q(\theta) \ln\left(\frac{\Gamma(\alpha_{ij}+)}{\prod_{l=1}^{D}\Gamma(\alpha_{ijl})}\right) d\alpha \quad + \\ & \gamma^C_{ijt} \int Q(\theta) \ln\left(\frac{1}{\tau_{ij}^{D-1}} \frac{\prod_{l=1}^{D} \beta_{ijl}^{-\frac{\alpha_{ijl}}{\tau_{ij}}} X_{tl}^{(\frac{\alpha_{ijl}}{\tau_{ij}}-1)}}{\left(\sum_{l=1}^{D}(\frac{X_{tl}}{\beta_{ijl}})^{\frac{1}{\tau_{ij}}}\right)^{\alpha_{ij}+}}\right) \end{aligned} \tag{3.46}$$

The expected value of the first part of this equation is analytically intractable. So, we use the lower bound introduced in [43] as an approximation for it:

$$\mathbb{E}_Q \ln\left(\frac{\Gamma(\alpha_{ij}+)}{\prod_{l=1}^{D}\Gamma(\alpha_{ijl})}\right) \geq \bar{\alpha}_{ijl}\ln\left(\alpha_{ijl}\right)\left\{\Psi\left(\sum_{l=1}^{D}\bar{\alpha}_{ijl}\right) - \Psi\left(\bar{\alpha}_{ijl}\right) + \sum_{d=1,d\neq l}^{D}\bar{\alpha}_{ijd}\Psi'\left(\sum_{l=1}^{D}\bar{\alpha}_{ijl}\right)\left(\mathbb{E}_Q\ln\left(\alpha_{ijd}\right) - \ln\left(\bar{\alpha}_{ijd}\right)\right)\right\} \tag{3.47}$$

The second integral of (3.46) can be rewritten as follows:

$$\mathbb{E}_Q \ln\left(\frac{1}{\tau_{ij}^{D-1}}\frac{\prod_{l=1}^{D}\beta_{ijl}^{-\frac{\alpha_{ijl}}{\tau_{ij}}}X_{tl}^{(\frac{\alpha_{ijl}}{\tau_{ij}}-1)}}{\left(\sum_{l=1}^{D}(\frac{X_{tl}}{\beta_{ijl}})^{\frac{1}{\tau_{ij}}}\right)^{\alpha_{ij}+}}\right) = -(D-1)\bar{\tau}_{ij}+ \tag{3.48}$$

$$\sum_{l=1}^{D}\{(-\frac{\bar{\alpha}_{ijl}}{\bar{\tau}_{ij}})\ln(\beta_{ijl}) + (\frac{\bar{\alpha}_{ijl}}{\bar{\tau}_{ij}}-1)\ln(X_{tl})\}-$$

$$(\alpha_{ij}+)\ln(\sum_{l=1}^{D}(\frac{X_{tl}}{\beta_{ijl}})^{\frac{1}{\tau_{ij}}})$$

The last term in the above equation is again analytically intractable, following [74] and we take the lower bound described below as its approximation:

$$\ln(\sum_{l=1}^{D}(\frac{X_{tl}}{\beta_{ijl}})^{\frac{1}{\tau_{ij}}}) \geq \frac{-\ln\tau_{ij}}{\overline{\tau}_{ij}}\frac{\sum_{l=1}^{D}(\frac{x_{tl}}{\overline{\beta}_{ijl}})^{\frac{1}{\overline{\tau}_{ij}}}\ln(\frac{x_{tl}}{\overline{\beta}_{ijl}})}{\sum_{l=1}^{D}(\frac{x_{tl}}{\overline{\beta}_{ijl}})^{\frac{1}{\overline{\tau}_{ij}}}} + const. \tag{3.49}$$

It is worth mentioning that by comparing (3.41) and (3.3), we can see that this equation represents the estimated likelihood of the improved model, which the forward-backward method can efficiently compute. Because $F(Q(.))$ reflects the model's logmarginal likelihood, the number of states N and the number of mixture components K may be calculated using it as a model selection criterion in the proposed model. Consequently, we may run our variational learning algorithm with various N and K values and choose ones that result in the greatest $F(Q)$. Algorithm 5 presents our suggested algorithm for learning the SSD-HMM model using the variational inference approach.

Algorithm 5 Variational learning of SSD-HMM

1. Initialize ϕ^A, ϕ^C, and ϕ^π.
2. Initialize u_{ijl}, ν_{ijl}, h_{ijl}, q_{ij} and s_{ij}.
3. Draw the initial responsibilities γ^A, γ^C, and γ^π from prior distributions using (3.8), (3.9), and (3.10).
4. Calculate w^A, w^C, and w^π with (3.19), (3.25) and (3.22).
5. Initialize A, C, and π using (3.42), (3.43), and (3.44).
6. **repeat**
7. Calculate data likelihood using X, u_{ijl}, ν_{ijl}, h_{ijl}, q_{ij} and s_{ij} using (3.1).
8. Calculate responsibilities γ^A, γ^C and γ^π using forward-backward procedure with (3.20) , (3.26), and (3.23).
9. Update hyper parameters using (3.33) to (3.37).
10. Update w^A, w^C, and w^π using γ^A, γ^C, and γ^π with (3.19), (3.25) and (3.22).
11. Update A, C, and π using w^A, w^C, w^π with (3.42), (3.43), and (3.44).
12. **until convergence criterion is reached**.

3.2.2 Shifted-Scaled Dirichlet Based Hierarchical Dirichlet Process Hidden Markov Model

The Hierarchical Dirichlet Process is built on a hierarchical framework that employs several Dirichlet Processes (DP) [26, 27, 45]. Our proposed model, SSD-HDP-HMM uses the hierarchical Dirichlet process over hidden Markov models. This structure comprises at least two layers, with the base measure of the DP dispersed by other DPs at each level. For the purpose of simplicity, we will use a two-level HDP model in this work following previously suggested HDP-HMM models [78, 79]. As the first layer, we take G_0 to be a top-level (global level) Dirichlet process. G_0 has H as its base distribution and γ as the concentration parameter. Therefore, we can write: $G_0 \sim \mathrm{DP}(\gamma, H)$. Moreover, G_0 is the base distribution of an unlimited number of second-level (local level) Dirichlet processes in the HDP. Thus, G_0 is shared between all the i states. A grouped dataset $\mathcal{X}$ with N sets exists at the second-level. Each set has a G_i with $i \in \{1, \ldots, N\}$ where $G_i \sim \mathrm{DP}(\lambda, G_0)$. In our SSD-HDP-HMM model, G_i is the transition distribution for state i where N is the number of states.

Stick-breaking construction will be used in the creation of our model since it is a very straightforward method to HDP model implementation [26, 46]. Since we have a two-level HDP, we use two stick-breaking structures in this work, one for the global level and the other one for the local level. Therefore, applying stick-breaking construction, the global level distribution G_0 would be expressed by:

$$G_0 = \sum_{i=1}^{\infty} \psi_i \delta_{\theta_i}, \quad \theta_i \sim H, \quad \sum_{i=1}^{\infty} \psi_i = 1 \tag{3.50}$$
$$\psi_i = \psi'_i \prod_{s=1}^{i-1} (1 - \psi'_s), \quad \psi'_i \sim \mathrm{Beta}(1, \gamma)$$

θ_i is a set of independent random variables derived from the base distribution H which in our model is an SSD distribution. δ_{θ_i} represents an atom at θ_i which is accessible for all G_i. Using the same manner, we can obtain the local level for the infinite number of DP's G_i using stick-breaking construction as

follows:

$$G_i = \sum_{j=1}^{\infty} \varepsilon_{ij} \delta_{\varpi_{ij}}, \quad \varpi_{ij} \sim G_0, \quad \sum_{j=1}^{\infty} \varepsilon_{ij} = 1 \tag{3.51}$$
$$\varepsilon_{ij} = \varepsilon'_{ij} \prod_{s=1}^{j-1} (1 - \varepsilon'_{is}), \quad \varepsilon'_{ij} \sim \text{Beta}(1, \lambda)$$

Following the previous work in [47], in order to map two HDP levels together, we take a binary latent variable $W_{ij,i'}$ which is equal to 1 if ϖ_{ij} is associated with $\theta_{i'}$, otherwise, it is equal to 0. Also, in order to produce a sequence of data $\mathcal{X} = \{X_1, \ldots, X_T\}$ for our HMM framework, we consider first $\theta' = \{\theta'_1, \ldots, \theta'_T\}$ as a sequence of parameters. Then, we draw θ'_1 from G_0 and the rest of the parameters from G_{Z_t}, $\theta'_{i+1} \sim G_{Z_t}$, where Z_t is a state index that equals i if $\theta'_t = \theta_i$. Finally, the sequence of data $\mathcal{X}$ is generated using these θ' parameters, $X_t \sim P(X \mid \theta'_t)$. Because each sequence is modeled independently, the joint likelihood of the SSD-HDP-HMM can be written as [78]:

$$\begin{aligned} p(X, \theta, \psi', \varepsilon', W, Z) &= p(\theta) p(\psi') p(\varepsilon') p(W \mid \psi') \\ &\quad \times \prod_v p\left(Z_v \mid \varepsilon', W\right) \prod_t p\left(X_{vt} \mid \theta, Z_{vt}\right) \end{aligned} \tag{3.52}$$

where v is the number of the observed sequence. As we mentioned in the previous section, in the variational learning approach we are trying to find an estimation $Q(\Theta)$ for the true posterior $p(\Theta \mid \mathcal{X})$ since it is intractable. By applying mean-field theory, we can rewrite $Q(\Theta)$ as disjoint tractable components:

$$\begin{aligned} Q(\theta, \psi', \varepsilon', W, Z) &= Q(\theta) Q(\psi') Q(\varepsilon') Q(W) Q(Z) \\ &= \prod_i Q\left(\theta_i\right) Q\left(\psi'_i\right) \prod_{i,j} Q\left(\varepsilon'_{ij}\right) Q\left(W_{ij}\right) \prod_v Q\left(Z_v\right) \end{aligned} \tag{3.53}$$

where $\Theta = \{\theta, \psi', \varepsilon', W, Z\}$. Then, using $Q(\Theta)$, we maximize the lower bound $\mathcal{L}(Q)$ introduced in (2.13) with respect to each Q factor to minimize the KL-divergence between Q and the posterior P. In SSD-HDP-HMM model,

we need the transition matrix $A_{ii'}$ to be able to calculate $Q(\theta)$ and $Q(Z)$. However, we have a challenge for this term:

$$\mathbb{E}_Q \ln A_{ii'} = \mathbb{E}_Q \ln \sum_j W_{iji'} \varepsilon_{ij} \tag{3.54}$$

Because we need to sum all of the sticks allocated to atom θ_i' (for atom θ_i stick breaking construction), this expectation is not tractable. Therefore, according to what they did in [78], we derive a lower bound for solving this issue as follows:

$$\begin{aligned} \mathbb{E}_Q \ln \sum_j W_{iji'} \varepsilon_{ij} &\geq \mathbb{E}_Q \ln \sum_j W_{iji'} e^{\mathbb{E}_Q \ln \varepsilon_{ij}} \\ &\approx \ln \sum_j \mathbb{E}_Q \left[W_{iji'} \right] e^{\mathbb{E}_Q \ln \varepsilon_{ij}} \end{aligned} \tag{3.55}$$

therefore,

$$\begin{aligned} A_{ii'}^{\star} &= \exp \left\{ \mathbb{E}_Q \ln A_{ii'} \right\} \\ &\approx \exp \left\{ \ln \sum_j \mathbb{E}_Q \left[W_{iji'} \right] e^{\mathbb{E}_Q \ln \varepsilon_{ij}} \right\} \end{aligned} \tag{3.56}$$

The other solution is that we could take the variational distribution $Q(W)$ and $Q(\varepsilon)$ instead of the above approximation to obtain $\mathbb{E}_Q \ln A_{ii'}$. The performance would stay almost the same but the algorithm would be more time consuming in this case [78].

Update $Q\left(Z_v\right)$ and $Q\left(\theta_i\right)$:

We have

$$\tilde{p}\left(X_{vt} \mid \theta_i\right) = \exp \left\{ \mathbb{E}_Q \ln p^{\star}\left(X_{vt} \mid \theta_i\right) \right\} \tag{3.57}$$

and we already obtained $\ln p^{\star}\left(X_{vt} \mid \theta_i\right)$ in (3.46) to (3.49). Also, the forward algorithm can be written as follows:

$$\alpha_{vt}(i) = \tilde{p}\left(X_{vt} \mid \theta_i\right) \sum_{s=1}^{\infty} \alpha_{v,t-1}(s) \tilde{A}_{si} \tag{3.58}$$

which is the variational joint probability of $Z_{vt} = i$ and the sequence X_v until step t [73, 78]. Also, the backward algorithm is:

$$\beta_{vt}(i) = \sum_{s=1}^{\infty} \tilde{A}_{is} \tilde{p}\left(X_{v,t+1} \mid \theta_s\right) \beta_{v,t+1}(s) \tag{3.59}$$

which is the variational probability of the sequence X_v after t given $Z_{vt} = i$ [73, 78]. Then, we can calculate $\gamma_{vt}(i)$ which is the marginal of Z_{vt} for $Q(Z_v)$:

$$\gamma_{vt}(i) = \frac{\alpha_{vt}(i)\beta_{vt}(i)}{\sum_s \alpha_{vt}(s)\beta_{vt}(s)} \tag{3.60}$$

For our SSD-HDP-HMM model, $\theta = \{\alpha, \beta, \tau\}$; therefore:

$$Q\left(\alpha_{ijl}\right) = \prod_{i=1}^{N} \prod_{j=1}^{K} \prod_{l=1}^{D} \mathcal{G}(\alpha_{ijl} \mid u^{\star}_{ijl}, \nu^{\star}_{ijl}) \tag{3.61}$$

$$Q\left(\beta_{ijl}\right) = \prod_{i=1}^{N} \prod_{j=1}^{K} \prod_{l=1}^{D} \mathcal{D}(\beta_{ijl} \mid \overrightarrow{h}^{\star}_{ijl}) \tag{3.62}$$

$$Q\left(\tau_{ij}\right) = \prod_{i=1}^{N} \prod_{j=1}^{K} \mathcal{G}(\tau_{ij} \mid q^{\star}_{ij}, s^{\star}_{ij}) \tag{3.63}$$

where the hyper parameters can be updated using following equations:

$$u^{\star}_{ijl} = u_{ijl} + \sum_{i'=1}^{N} \varphi_{ij,i'} \sum_{t=1}^{T} \sum_{v=1}^{V} \xi_{vt}(i,j) \times \overline{\alpha}_{ijl} \times \tag{3.64}$$
$$[\psi(\sum_{s=1}^{D} \overline{\alpha}_{ijs}) - \psi\big(\overline{\alpha}_{ijl}\big) + \sum_{s \neq d}^{D} \psi'(\sum_{s=1}^{D} \overline{\alpha}_{ijs}) \times \overline{\alpha}_{ijs}(\mathbb{E}_Q \ln \alpha_{ijs} - \ln \overline{\alpha}_{ijs})]$$

$$\nu^{\star}_{ijl} = \nu_{ijl} - \sum_{i'=1}^{N} \varphi_{ij,i'} \sum_{t=1}^{T} \sum_{v=1}^{V} \xi_{vt}(i,j)[\frac{1}{\tau_{ij}} \ln \frac{\beta_{ijl}}{X_{tl}} + \ln (\sum_{s=1}^{D} (\frac{X_{ts}}{\beta_{ijs}})^{\frac{1}{\tau_{ij}}})] \quad (3.65)$$

$$h^{\star}_{ijl} = h_{ijl} + \sum_{i'=1}^{N} \varphi_{ij,i'} \sum_{t=1}^{T} \sum_{v=1}^{V} \xi_{vt}(i,j)[\frac{-\bar{\alpha}_{ijl}}{\bar{\tau_{ij}}} + \frac{\bar{\alpha_{ijl}}}{\bar{\tau_{ij}}} \times (\frac{X_{tl}}{\bar{\beta_{ijl}}})^{\frac{1}{\bar{\tau_{ij}}}} \times \frac{1}{\sum_{s=1}^{D} (\frac{X_{ts}}{\bar{\beta_{ijs}}})^{\bar{\tau_{ij}^{-1}}}}] \quad (3.66)$$

$$q^{\star}_{ij} = q_{ij} + \sum_{i'=1}^{N} \varphi_{ij,i'} \sum_{t=1}^{T} \sum_{v=1}^{V} \xi_{vt}(i,j)[1 - D + \frac{(\alpha_{ij}+)}{\tau_{ij}} \frac{\sum_{l=1}^{D} (\frac{X_{tl}}{\beta_{ijl}})^{\tau_{ij}^{-1}} \ln (\frac{X_{tl}}{\beta_{ijl}})}{\sum_{l=1}^{D} (\frac{X_{tl}}{\beta_{ijl}})^{\tau_{ij}^{-1}}}] \quad (3.67)$$

$$s^{\star}_{ij} = s_{ij} - \sum_{i'=1}^{N} \varphi_{ij,i'} \sum_{t=1}^{T} \sum_{v=1}^{V} \xi_{vt}(i,j)[\sum_{l=1}^{D} \frac{\alpha_{ijl}}{{\tau_{ij}}^2} \ln (\frac{X_{tl}}{\beta_{ijl}})] \quad (3.68)$$

$\varphi_{ij,i'}$ and $\xi_{vt}(i,j)$ values are obtained in the following section.

Update $Q\,(W)$, $Q\,(\psi')$, and $Q\,(\varepsilon')$:

In the previous section, we used the lower bound $\mathcal{L}(Q)$ to update $Q(Z)$ and $Q(\theta)$ but for updating $Q\,(W)$, $Q\,(\psi')$, and $Q\,(\varepsilon')$ the case is different. Since our approximation of the expected log of A in (3.55) does not produce tractable variational parameter updates for $Q\,(W)$, $Q\,(\psi')$, and $Q\,(\varepsilon')$, we employ a latent variable S_v and a variational distribution $Q(S_v \mid Z_v)$ for the vth observed sequence to lower bound $\mathcal{L}(Q)$ locally [78]. S_v interacts

with Z_v and $W_{iji'}$ in the following way: the tuple $(Z_{v,t-1} = i, S_{vt} = j)$ denotes that the next state Z_{vt} may be discovered by selecting the jth stick of the ith DP and setting $Z_{vt} = i'$ if $W_{ij,i'} = 1$. In the paper [78], instead of using $Q(Z_{v,t-1}, Z_{vt})$ to obtain the state transition, they introduced a triple $Q(Z_{vt}, S_{v,t+1}, W_{Z_{vt},S_{v,t+1}})$. Therefore, they came up with this local lower bound:

$$\begin{aligned} &\mathbb{E}_Q 1\left(Z_{v,t-1} = i, Z_{vt} = i'\right) \ln \sum_j W_{ij,i'} \varepsilon_{ij} \\ &\geq \mathbb{E}_Q \sum_j W_{ij,i'} 1\left(Z_{v,t-1} = i, S_{vt} = j\right) \ln \varepsilon_{ij} \end{aligned} \tag{3.69}$$

Also, the joint variational distribution of S_v and Z_v is:

$$Q\left(S_v, Z_v\right) = Q\left(S_v \mid Z_v\right) Q\left(Z_v\right) = Q\left(Z_v\right) \prod_t Q\left(S_{vt} \mid Z_v\right) \tag{3.70}$$

In the forward-backward algorithm, we already updated $Q(Z)$. The expectation value of $Q\left(Z_{v,t-1} = i, S_{vt} = j\right)$ can be updated by:

$$\begin{aligned} Q\left(Z_{v,t-1} = i, S_{vt} = j\right) &= \mathbb{E}_Q 1\left(Z_{v,t-1} = i, S_{vt} = j\right) \\ &\equiv \xi_{vt}(i,j) \end{aligned} \tag{3.71}$$

where $\xi_{vt}(i,j)$ is equivalent to γ_{ijt}^C in (3.26) for the vth observed sequence and can be obtained by:

$$\xi_{vt}(i,j) \propto \alpha_{v,t-1}(i) \exp\left\{\mathbb{E}_Q \ln \varepsilon_{ij}\right\} \times \prod_{i'} \left[\exp\left\{\mathbb{E}_Q \left[\ln \theta_{i',X_{vt}}\right]\right\} \beta_{vt}\left(i'\right)\right]^{\varphi_{ij,i'}} \tag{3.72}$$

with

$$\mathbb{E}_Q \ln \varepsilon_{ij} = \mathbb{E}_Q \ln \varepsilon'_{ij} + \sum_s \mathbb{E}_Q \ln\left(1 - \varepsilon'_{is}\right) \tag{3.73}$$

Here, α and β are the forward and backward algorithms, respectively. We recall that W_{ij} is the atom associated with the jth stick in the ith state.
Also, we have:

$$Q\left(W_{ij,i'} = 1\right) \equiv \varphi_{ij,i'} \tag{3.74}$$

where

$$\varphi_{ij,i'} \propto \exp\left\{\mathbb{E}_Q \ln \psi_{i'} + \sum_{v,t} \xi_{vt}(i,j)\mathbb{E}_Q \ln \theta_{i',X_{vt}}\right\} \tag{3.75}$$

with

$$\mathbb{E}_Q \ln \psi_{i'} = \mathbb{E}_Q \ln \psi'_{i'} + \sum_{s<i'} \mathbb{E}_Q \ln (1-\psi'_s) \tag{3.76}$$

Using variational distributions $Q(\psi'_i) = \text{Beta}(c_i, d_i)$ and $Q(\varepsilon'_{ij}) = \text{Beta}(a_{ij}, b_{ij})$, we have:

$$\mathbb{E}_Q \ln \psi'_i = \Psi(c_i) - \Psi(c_i + d_i) \tag{3.77}$$

$$\mathbb{E}_Q \ln (1-\psi'_i) = \Psi(d_i) - \Psi(c_i + d_i) \tag{3.78}$$

$$\mathbb{E}_Q \ln \varepsilon'_{ij} = \Psi(a_{ij}) - \Psi(a_{ij} + b_{ij}) \tag{3.79}$$

$$\mathbb{E}_Q \ln \left(1-\varepsilon'_{ij}\right) = \Psi(b_{ij}) - \Psi(a_{ij} + b_{ij}) \tag{3.80}$$

where

$$a_{ij} = 1 + \sum_{v,t} \xi_{vt}(i,j) \tag{3.81}$$

$$b_{ij} = \lambda + \sum_{v,t} \sum_{j'>j} \xi_{vt}(i,j') \tag{3.82}$$

$$c_i = 1 + \sum_{i',j} \mathbb{E}_Q W_{i'j,i} \tag{3.83}$$

$$d_i = \gamma + \sum_{i',j} \sum_{s>i} \mathbb{E}_Q W_{i'j,s} \tag{3.84}$$

Our suggested algorithm for variational learning of the SSD-HDP-HMM is described in Algorithm 6.

Algorithm 6 Variational learning of SSD-HDP-HMM.

1. Initialize λ, γ, u_{ijl}, ν_{ijl}, h_{ijl}, q_{ij} and s_{ij}.
2. **repeat**
3. Update $Q(Z_v)$ using the forward and backward algorithms in (3.58) to (3.60).
4. Update $Q(\theta)$ using (3.61) to (3.68).
5. Update $Q\left(S_{vt} \mid Z_v\right)$ using (3.72).
6. Update $Q\left(W\right)$ using (3.74).
7. Update $Q\left(\psi'\right)$ and $Q\left(\varepsilon'\right)$ using (3.77) to (3.84).
8. **until convergence criterion is reached**.

3.3 Experimental Results

We tested our proposed models on two real-world applications: activity recognition [80] and texture clustering [81]. Then, to assess how successful they are, we compare them to three other models: HDP-HMM, hidden Markov model with Dirichlet mixture model emissions (DMM-HMM), and hidden Markov model with Gaussian mixture model emissions (GMM-HMM). To evaluate our model's performance, we use both real and anticipated cluster labels in four metrics stated in (2.100) to (2.103).

We present the average results after testing our model ten times with each dataset in order to deliver a more accurate result. Since our model is based on mixtures of shifted-scaled-Dirichlet distributions, the data must be proportional as we discussed in section 2. Therefore, we first normalize each record (row of the dataset) making it between 0 and 1 and then divide it by its summation of dimensions to make it proportional. Then, we use principal component analysis (PCA) to filter the most significant features of data [82].

3.3.1 Activity Recognition

The introduction of smartphones has significantly affected human lives. The existence of advanced features in these devices resulted in their continuous presence in our lives. Consequently, this presence provides a chance to keep track of our activities using a variety of sensors [83, 84]. Data generated by such devices could be used in many applications such as healthcare monitoring [85], life and fitness tracking [86], and transportation planning [87].

In our work, we applied an activity recognition (AR) dataset in which data were collected via accelerometer [88] and gyroscope sensors in a Samsung Galaxy S II. To gather the information, behaviour of 30 participants aged between 19-48 [84] were monitored while wearing smartphones on the waist. This method of collecting data is one of the easiest ways because we do not need any additional equipment [84]. This is in contrast to other existing techniques for collecting AR data, which rely on special devices. Data are collected during six activities: walking, walking upstairs, walking downstairs, sitting, standing, and lying down. This dataset is randomly divided into two partitions: 70% of data are considered as a training dataset and 30% as a test dataset and we have 10,299 data points in total. According to figure 3.1, nearly identical numbers of data points were received from all of the participants. Also, by looking at figure 3.2, we can see that the share of information associated with various activities are relatively similar. The lowest number of recorded activities belongs to the walking downstairs with 1,406 instances. Therefore, to have a balanced dataset (consist of the same number of samples in all output classes), we randomly select the same number of data points from other categories before feeding our model (overall 8,436 observations).

By looking at the probability density function (PDF) distribution in figure 3.3, we can see that motionless activities have a totally different distribution than moving activities. Besides, we can understand that the moving activity distributions are similar to each other which is the same case for motionless activities. To have a better understanding of the data, we provide a 2D scatter plot of our data using different colours for each cluster in figure 3.4. As we can see in this plot, almost all of the features can be separated into different regions except standing and sitting; although there is considerable overlap with data points from other clusters especially near the

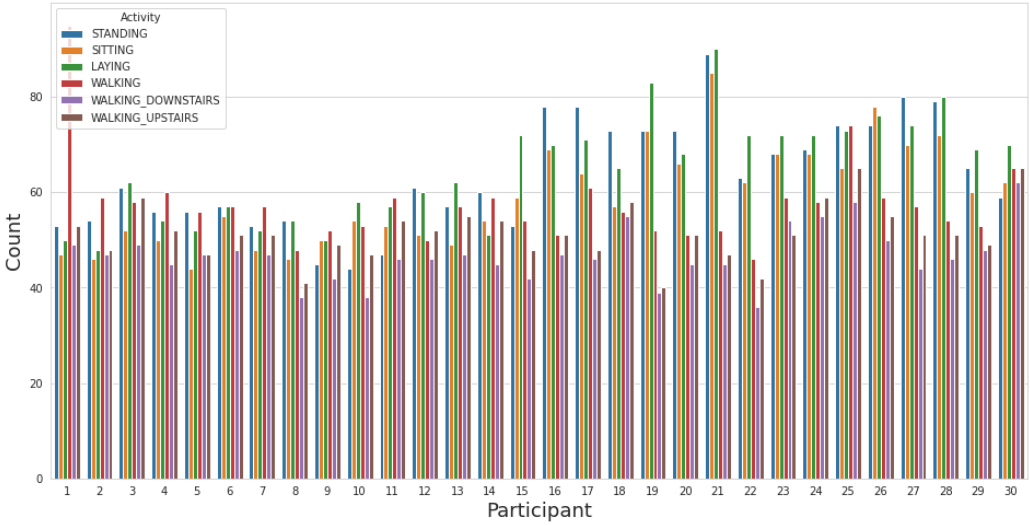

Figure 3.1: Activity count per participant

boundaries. Therefore, we expect that dividing standing and sitting into two separate clusters may be a challenging task for a machine learning algorithm. We should mention that by using PCA, we reduced the number of features from 562 to 122 in order to filter the most important features. The result of testing our models is presented in table 3.1. According to this table, the SSD-HDP-HMM model has a 97.66 percent accuracy, which is a better result than the previously studied model, HDP-HMM. Also, SSD-HMM model obtained higher accuracy than DMM-HMM and GMM-HMM models. As a result, using SSD as the parent distribution helped our models to fit the data better than either Dirichlet or Gaussian mixture models. In addition, by comparing the SSD-HDP-HMM model to the SSD-HMM model, we can find that hierarchical models are better in learning complicated data than non-hierarchical models.

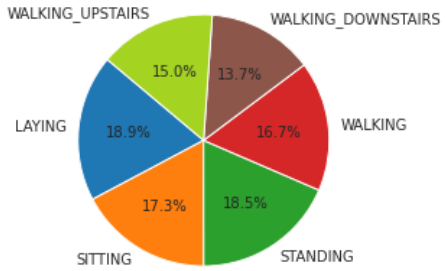

Figure 3.2: Percentage of each category.

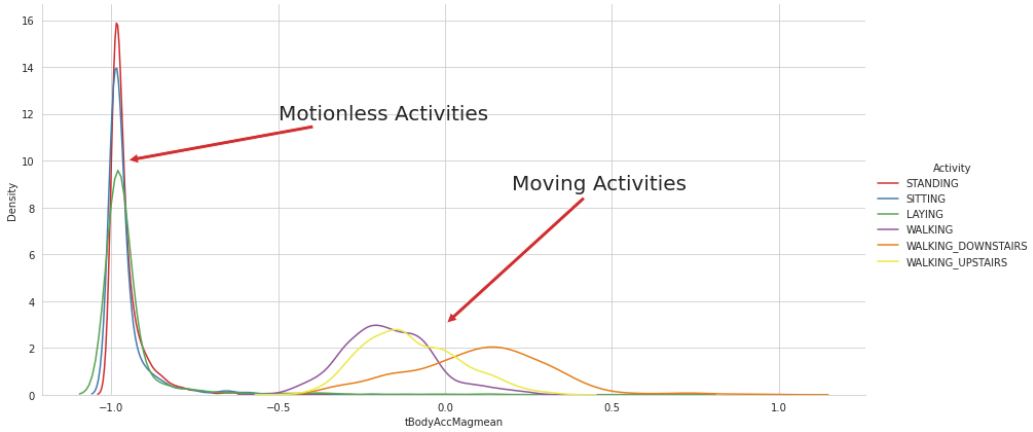

Figure 3.3: Probability density function of the AR dataset.

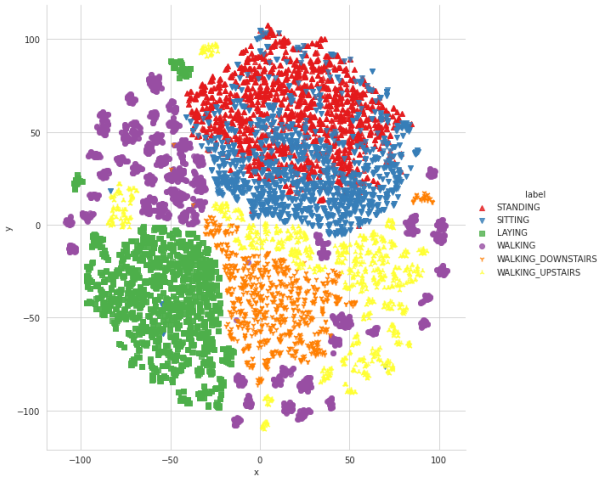

Figure 3.4: 2D scatter plot of 2 features of the AR dataset.

Table 3.1: Activity recognition dataset results

Method	***Accuracy***	***Precision***	***Recall***	***F*1*-score***
SSD-HDP-HMM	97.66	97.66	97.63	97.64
HDP-HMM	95.29	95.38	95.36	95.36
SSD-HMM	93.88	94.03	94.06	94
DMM-HMM	86.65	86.12	86.36	86.11
GMM-HMM	93.59	93.11	93.13	93.11

3.3.2 Texture Clustering

As the second application, we chose to use a challenging dataset called the University of Illinois Urbana Champaign (UIUC) texture dataset. This dataset has 25 classes, each with 40 images of size 480x640. Figure 3.5 shows some sample images of UIUC dataset. The diversity of 2D and 3D transformations, as well as lighting fluctuations in this dataset made it a difficult application for machine learning algorithms. Therefore, we applied VGG16 to extract features of images. VGG16 is a popular strong deep learning model which has already shown its capabilities in feature extraction. We also ac-

knowledge that this model is not designed for feature selection and we used an arbitrary intermediate layer to extract features of images. Then using PCA, we reduced the number of features extracted by VGG16 from 4,096 to 1,251. Figure 3.6 shows the PDF distribution of one of the most important features. As we can see in this figure, the distribution of classes is quite similar to one another, and we verified that this is true for other features as well. Therefore, this makes the clustering task hard. This conclusion may also be obtained by looking at figure 3.7, which is a 2D scatter plot. In this figure, some of the clusters are in totally separated regions and most of them have overlap with each other. As a result, we chose to test our model with the almost-separated classes 3, 4, and 7 (see figure 3.5). As we mentioned before, each of the classes has 40 images, so our dataset is already balanced. Table 3.2 displays the results of our model testing. Our proposed hierarchical model, SSD-HDP-HMM, has the greatest accuracy of the examined models, with 81.8 percent. This demonstrates that, owing to the SSD distribution, this model fits the data better than the classical HDP-HMM model. Furthermore, the SSD-HMM model outperforms the basic Dirichlet and Gaussian HMM mixture models.

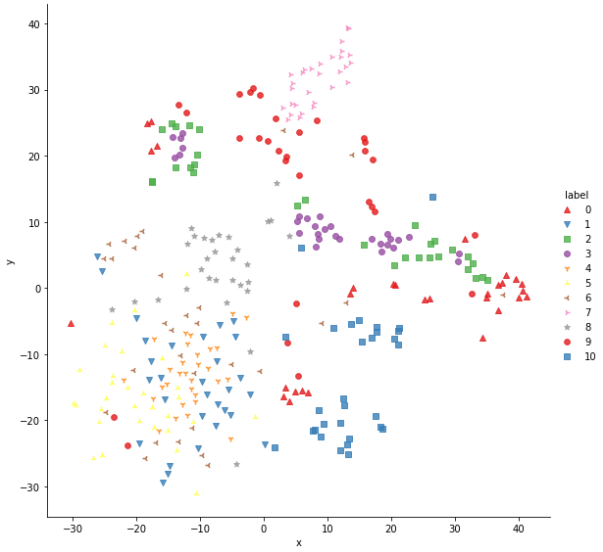

Figure 3.7: 2D scatter plot of 2 features of the UIUC dataset.

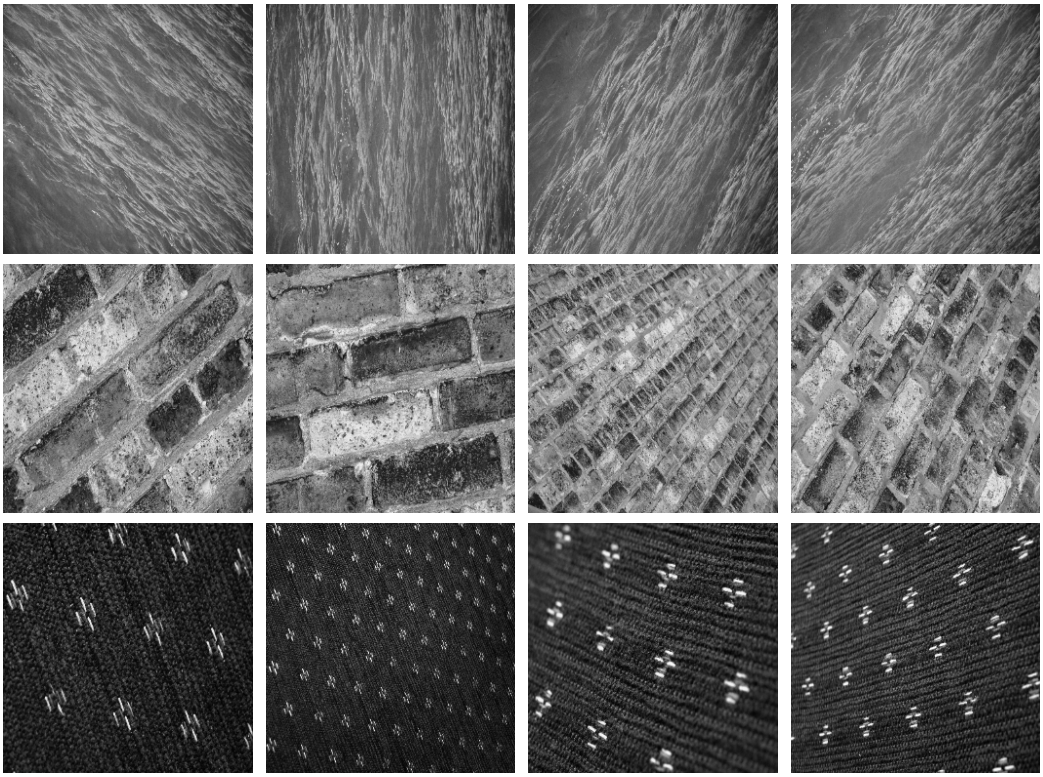

Figure 3.5: Samples of UIUC texture dataset.

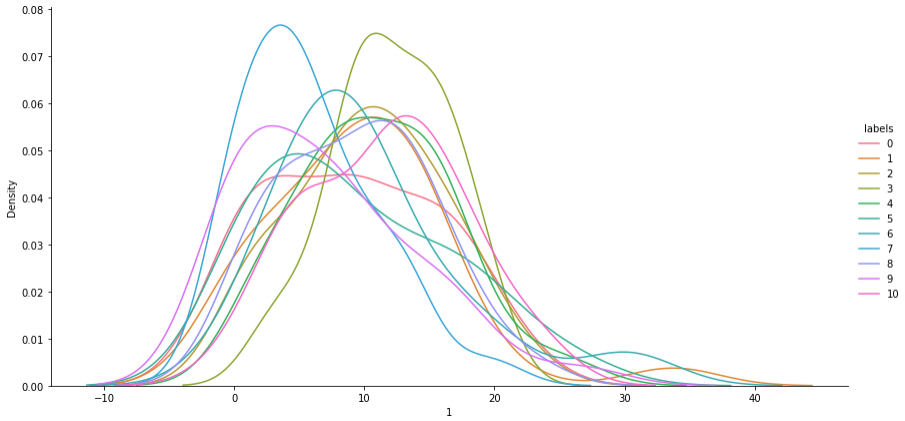

Figure 3.6: Probability density function of UIUC texture dataset.

Table 3.2: UIUC texture dataset results

Method	***Accuracy***	***Precision***	***Recall***	***F*1*-score***
SSD-HDP-HMM	81.8	80.7	69.7	80.6
HDP-HMM	80.2	77.2	75.3	75.2
SSD-HMM	77.2	81.1	80.1	73
DMM-HMM	75.7	71.9	69.7	69.7
GMM-HMM	56	53.8	54.1	48

Chapter 4

Conclusion

This thesis is based on shifted-scaled Dirichlet distributions (SSD). We show that for fitting proportional data, SSD mixture models outperform Dirichlet and Gaussian mixture models. In this thesis, we developed six innovative SSD-based models.

In summary, the following are the major contributions of this work:
First, in Chapter 2, we developed a hierarchical Dirichlet process based on SSD mixture models. Then we extend this model to the hierarchical Pitman-Yor process. We used two different variational settings to train these models: batch and online. The online setting provides us with the ability of learning model parameters from streaming data; therefore, these models can be used in many real-world applications.
Second, we commenced with a finite SSD-based hidden Markov model in Chapter 3. We then improved our model to an HDP model and introduced another unique approach called SSD-based HDP HMM to take advantage of non-parametric models. For learning our HMM models, we employ variational Bayes frameworks.

With the proposed non-parametric approaches, simultaneous parameter estimation and model complexity assessment are achievable owing to the hierarchical Bayesian frameworks adopted.

Traceable learning calculations, exact approximations, and assured convergence are some of the benefits of using the variational learning approach in

this thesis over other learning techniques (such as maximum likelihood and Markov Chain Monte Carlo). Finding appropriate initial values via variational inference, on the other hand, might take a long time.

Finally, in comparison to other alternative models, the experimental results on real-world applications have demonstrated the effectiveness of our models.

Although the data features are considered to have the same importance in this thesis, we are investigating several future works such as integrating feature selection to our proposed model to improve its generalization capabilities. This will allow us to work on more complex applications with data features that aren't equally weighted. Therefore, this can lead to higher performance for our proposed models.

List of References

[1] Mayra Z Rodriguez, Cesar H Comin, Dalcimar Casanova, Odemir M Bruno, Diego R Amancio, Luciano da F Costa, and Francisco A Rodrigues. Clustering algorithms: A comparative approach. *PloS one*, 14(1):e0210236, 2019.

[2] Caglar Aytekin, Xingyang Ni, Francesco Cricri, and Emre Aksu. Clustering and unsupervised anomaly detection with l 2 normalized deep auto-encoder representations. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2018.

[3] Max Landauer, Markus Wurzenberger, Florian Skopik, Giuseppe Settanni, and Peter Filzmoser. Dynamic log file analysis: An unsupervised cluster evolution approach for anomaly detection. *computers & security*, 79:94–116, 2018.

[4] Muyeed Ahmed, Mir Tahsin Imtiaz, and Raiyan Khan. Movie recommendation system using clustering and pattern recognition network. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 143–147. IEEE, 2018.

[5] Pan Li, Hoang Dau, Gregory Puleo, and Olgica Milenkovic. Motif clustering and overlapping clustering for social network analysis. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2017.

[6] Igor Andreevich Rytsarev, Dmitriy Victorovich Kirsh, and Alexander Victorovich Kupriyanov. Clustering of media content from social

networks using bigdata technology. *Computer Optics*, 42(5):921–927, 2018.

[7] Jorge Rodríguez, Ivana Semanjski, Sidharta Gautama, Nico Van de Weghe, and Daniel Ochoa. Unsupervised hierarchical clustering approach for tourism market segmentation based on crowdsourced mobile phone data. *Sensors*, 18(9):2972, 2018.

[8] Deepali Kamthania, Ashish Pawa, and Srijit S Madhavan. Market segmentation analysis and visualization using k-mode clustering algorithm for e-commerce business. *Journal of computing and information technology*, 26(1):57–68, 2018.

[9] Mohammed H Almannaa, Mohammed Elhenawy, and Hesham A Rakha. A novel supervised clustering algorithm for transportation system applications. *IEEE transactions on intelligent transportation systems*, 21(1):222–232, 2019.

[10] Zihan Hong, Ying Chen, and Hani S Mahmassani. Recognizing network trip patterns using a spatio-temporal vehicle trajectory clustering algorithm. *IEEE Transactions on Intelligent Transportation Systems*, 19(8):2548–2557, 2017.

[11] Geoffrey J McLachlan, Sharon X Lee, and Suren I Rathnayake. Finite mixture models. *Annual review of statistics and its application*, 6:355–378, 2019.

[12] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[13] Ines Channoufi, Sami Bourouis, Nizar Bouguila, and Kamel Hamrouni. Image and video denoising by combining unsupervised bounded generalized gaussian mixture modeling and spatial information. *Multimedia Tools and Applications*, 77(19):25591–25606, 2018.

[14] Zexuan Ji, Yong Xia, Quansen Sun, Qiang Chen, and Dagan Feng. Adaptive scale fuzzy local gaussian mixture model for brain mr image segmentation. *Neurocomputing*, 134:60–69, 2014.

[15] Wentao Fan, Hassen Sallay, and Nizar Bouguila. Online learning of hierarchical pitman–yor process mixture of generalized dirichlet distributions with feature selection. *IEEE Transactions on Neural Networks and Learning Systems*, 28(9):2048–2061, 2017.

[16] Jai Puneet Singh and Nizar Bouguila. Proportional data clustering using k-means algorithm: A comparison of different distances. In *2017 IEEE International Conference on Industrial Technology (ICIT)*, pages 1048–1052, 2017.

[17] Zhanyu Ma, Yuping Lai, W. Bastiaan Kleijn, Yi-Zhe Song, Liang Wang, and Jun Guo. Variational bayesian learning for dirichlet process mixture of inverted dirichlet distributions in non-gaussian image feature modeling. *IEEE Transactions on Neural Networks and Learning Systems*, 30(2):449–463, 2019.

[18] Sylvia Frühwirth-Schnatter and Gertraud Malsiner-Walli. From here to infinity: sparse finite versus dirichlet process mixtures in model-based clustering. *Advances in data analysis and classification*, 13(1):33–64, 2019.

[19] Junyang Chen, Zhiguo Gong, and Weiwen Liu. A dirichlet process biterm-based mixture model for short text stream clustering. *Applied Intelligence*, 50(5):1609–1619, 2020.

[20] Mario Beraha, Alessandra Guglielmi, and Fernando A Quintana. The semi-hierarchical dirichlet process and its application to clustering homogeneous distributions. *arXiv preprint arXiv:2005.10287*, 2020.

[21] Khadidja Meguelati, Benedicte Fontez, Nadine Hilgert, and Florent Masseglia. High dimensional data clustering by means of distributed dirichlet process mixture models. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 890–899. IEEE, 2019.

[22] Zhanyu Ma and Arne Leijon. Super-dirichlet mixture models using differential line spectral frequencies for text-independent speaker identification. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[23] Narges Manouchehri, Oumayma Dalhoumi, Manar Amayri, and Nizar Bouguila. Variational learning of a shifted scaled dirichlet model with

component splitting approach. In *2020 Third International Conference on Artificial Intelligence for Industries (AI4I)*, pages 75–78. IEEE, 2020.

[24] Narges Manouchehri, Hieu Nguyen, Pantea Koochemeshkian, Nizar Bouguila, and Wentao Fan. Online variational learning of dirichlet process mixtures of scaled dirichlet distributions. *Information Systems Frontiers*, 22(5):1085–1093, 2020.

[25] Hieu Nguyen, Maryam Rahmanpour, Narges Manouchehri, Kamal Maanicshah, Manar Amayri, and Nizar Bouguila. A statistical approach for unsupervised occupancy detection and estimation in smart buildings. In *2019 IEEE International Smart Cities Conference (ISC2)*, pages 414–419. IEEE, 2019.

[26] Dingcheng Li, Siamak Zamani, Jingyuan Zhang, and Ping Li. Integration of knowledge graph embedding into topic modeling with hierarchical dirichlet process. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 940–950, 2019.

[27] Yee Whye Teh and Michael I Jordan. Hierarchical bayesian nonparametric models with applications. *Bayesian nonparametrics*, 1:158–207, 2010.

[28] Jim Pitman and Marc Yor. The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, pages 855–900, 1997.

[29] Lawrence Rabiner and Biinghwang Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986.

[30] Takashi Fuse and Keita Kamiya. Statistical anomaly detection in human dynamics monitoring using a hierarchical dirichlet process hidden markov model. *IEEE Transactions on Intelligent Transportation Systems*, 18(11):3083–3092, 2017.

[31] Mayur Rahul, Narendra Kohli, Rashi Agarwal, and Sanju Mishra. Facial expression recognition using geometric features and modified hidden markov model. *International Journal of Grid and Utility Computing*, 10(5):488–496, 2019.

[32] Mohammed Kyari Mustafa, Tony Allen, and Kofi Appiah. A comparative review of dynamic neural networks and hidden markov model methods for mobile on-device speech recognition. *Neural Computing and applications*, 31(2):891–899, 2019.

[33] Weiyue Wang, Derui Zhu, Tamer Alkhouli, Zixuan Gan, and Hermann Ney. Neural hidden markov model for machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 377–382, 2018.

[34] Xi Zhang, Yixuan Li, Senzhang Wang, Binxing Fang, and S Yu Philip. Enhancing stock market prediction with extended coupled hidden markov model over multi-sourced data. *Knowledge and Information Systems*, 61(2):1071–1090, 2019.

[35] Gunasekaran Manogaran, V Vijayakumar, R Varatharajan, Priyan Malarvizhi Kumar, Revathi Sundarasekar, and Ching-Hsien Hsu. Machine learning based big data processing framework for cancer diagnosis using hidden markov model and gm clustering. *Wireless personal communications*, 102(3):2099–2116, 2018.

[36] Md Zia Uddin. Human activity recognition using segmented body part and body joint features with hidden markov models. *Multimedia Tools and Applications*, 76(11):13585–13614, 2017.

[37] K Martin Sagayam and D Jude Hemanth. A probabilistic model for state sequence analysis in hidden markov model for hand gesture recognition. *Computational Intelligence*, 35(1):59–81, 2019.

[38] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[39] Elise Epaillard and Nizar Bouguila. Variational bayesian learning of generalized dirichlet-based hidden markov models applied to unusual events detection. *IEEE transactions on neural networks and learning systems*, 30(4):1034–1047, 2018.

[40] Matthew D Hoffman. Learning deep latent gaussian models with markov chain monte carlo. In *International conference on machine learning*, pages 1510–1519. PMLR, 2017.

[41] Masa-Aki Sato. Online model selection based on the variational bayes. *Neural computation*, 13(7):1649–1681, 2001.

[42] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[43] Wentao Fan, Nizar Bouguila, and Djemel Ziou. Variational learning for finite dirichlet mixture models and applications. *IEEE transactions on neural networks and learning systems*, 23(5):762–774, 2012.

[44] Ali Baghdadi, Narges Manouchehri, Zachary Patterson, and Nizar Bouguila. *Shifted-Scaled-Dirichlet Based Hierachical Dirichlet Process Hidden Markov Models with Variational Inference Learning*. in Hidden Markov Models and Applications, N. Bouguila, W. Fan and M. Amayri, editors, Springer, 2021.

[45] Wentao Fan, Hassen Sallay, Nizar Bouguila, and Sami Bourouis. Variational learning of hierarchical infinite generalized dirichlet mixture models and applications. *Soft Computing*, 20(3):979–990, 2016.

[46] Haibin Zhang, Shang Huating, and Xianyi Wu. Topic model for graph mining based on hierarchical dirichlet process. *Statistical Theory and Related Fields*, 4(1):66–77, 2020.

[47] Chong Wang, John Paisley, and David Blei. Online variational inference for the hierarchical dirichlet process. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 752–760. JMLR Workshop and Conference Proceedings, 2011.

[48] Mahieddine M Ichir and Ali Mohammad-Djafari. A mean field approximation approach to blind source separation with l p priors. In *2005 13th European Signal Processing Conference*, pages 1–4. IEEE, 2005.

[49] David M Blei, Michael I Jordan, et al. Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

[50] Wentao Fan and Nizar Bouguila. Dynamic textures clustering using a hierarchical pitman-yor process mixture of dirichlet distributions. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 296–300. IEEE, 2015.

[51] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[52] Ashkan Banan, Amin Nasiri, and Amin Taheri-Garavand. Deep learning-based appearance features extraction for automated carp species identification. *Aquacultural Engineering*, 89:102053, 2020.

[53] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999.

[54] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 76, pages 1–2. Prague, 2004.

[55] RAZA Mansoor, Nathali Dilshani Jayasinghe, and Muhana Magboul Ali Muslam. A comprehensive review on email spam classification using machine learning algorithms. In *2021 International Conference on Information Networking (ICOIN)*, pages 327–332. IEEE, 2021.

[56] Ammara Zamir, Hikmat Ullah Khan, Waqar Mehmood, Tassawar Iqbal, and Abubakker Usman Akram. A feature-centric spam email detection model using diverse supervised machine learning algorithms. *The Electronic Library*, 2020.

[57] Nadjate Saidani, Kamel Adi, and Mohand Said Allili. A semantic-based classification approach for an enhanced spam detection. *Computers & Security*, 94:101716, 2020.

[58] Bilge Kagan Dedeturk and Bahriye Akay. Spam filtering using a logistic regression model trained by an artificial bee colony algorithm. *Applied Soft Computing*, 91:106229, 2020.

[59] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[60] Jianming Zhang, Manting Huang, Xiaokang Jin, and Xudong Li. A real-time chinese traffic sign detection algorithm based on modified yolov2. *Algorithms*, 10(4), 2017.

[61] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks*, number 1288, 2013.

[62] Amara Dinesh Kumar. Novel deep learning model for traffic sign detection using capsule networks. *arXiv preprint arXiv:1805.04424*, 2018.

[63] Yong Tang, Congzhe Zhang, Renshu Gu, Peng Li, and Bin Yang. Vehicle detection and recognition for intelligent traffic surveillance system. *Multimedia tools and applications*, 76(4):5817–5832, 2017.

[64] Prashant Chaturvedi. Object classification, 2019. `https://www.kaggle.com/positivepc/object-detection`. Accessed May 28, 2021.

[65] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[66] Biing Hwang Juang and Laurence R Rabiner. Hidden markov models for speech recognition. *Technometrics*, 33(3):251–272, 1991.

[67] Mangi Kang, Jaelim Ahn, and Kichun Lee. Opinion mining using ensemble text hidden markov models for text classification. *Expert Systems with Applications*, 94:218–227, 2018.

[68] Zoubin Ghahramani and Michael I Jordan. Factorial hidden markov models. *Machine learning*, 29(2):245–273, 1997.

[69] Rim Nasfi, Manar Amayri, and Nizar Bouguila. A novel approach for modeling positive vectors with inverted dirichlet-based hidden markov models. *Knowledge-Based Systems*, 192:105335, 2020.

[70] Manuele Bicego, Umberto Castellani, and Vittorio Murino. A hidden markov model approach for appearance-based 3d object recognition. *Pattern Recognition Letters*, 26(16):2588–2599, 2005.

[71] Ernesto L Andrade, Scott Blunsden, and Robert B Fisher. Hidden markov models for optical flow analysis in crowds. In *18th international conference on pattern recognition (ICPR'06)*, volume 1, pages 460–463. IEEE, 2006.

[72] Gianna Serafina Monti, Glòria Mateu i Figueras, Vera Pawlowsky-Glahn, Juan José Egozcue, et al. The shifted-scaled dirichlet distribution in the simplex. 2011.

[73] Matthew James Beal. *Variational algorithms for approximate Bayesian inference*. University of London, University College London (United Kingdom), 2003.

[74] Zeinab Arjmandiasl, Narges Manouchehri, Nizar Bouguila, and Jamal Bentahar. Variational learning of finite shifted scaled dirichlet mixture models. In *Learning Control*, pages 175–204. Elsevier, 2021.

[75] Shihao Ji, Balaji Krishnapuram, and Lawrence Carin. Variational bayes for continuous hidden markov models and its application to active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):522–532, 2006.

[76] Sotirios P Chatzis and Dimitrios I Kosmopoulos. A variational bayesian methodology for hidden markov models utilizing student's-t mixtures. *Pattern Recognition*, 44(2):295–306, 2011.

[77] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[78] Aonan Zhang, San Gultekin, and John Paisley. Stochastic variational inference for the hdp-hmm. In *Artificial Intelligence and Statistics*, pages 800–808. PMLR, 2016.

[79] Emily B Fox, Erik B Sudderth, Michael I Jordan, and Alan S Willsky. A sticky hdp-hmm with application to speaker diarization. *The Annals of Applied Statistics*, pages 1020–1056, 2011.

[80] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, Jorge Luis Reyes-Ortiz, et al. A public domain dataset for human activity recognition using smartphones. In *Esann*, volume 3, page 3, 2013.

[81] http://www-cvr.ai.uiuc.edu/.

[82] Arnaz Malhi and Robert X Gao. Pca-based feature selection scheme for machine defect classification. *IEEE transactions on instrumentation and measurement*, 53(6):1517–1525, 2004.

[83] Andrey Ignatov. Real-time human activity recognition from accelerometer data using convolutional neural networks. *Applied Soft Computing*, 62:915–922, 2018.

[84] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *International workshop on ambient assisted living*, pages 216–223. Springer, 2012.

[85] Akin Avci, Stephan Bosch, Mihai Marin-Perianu, Raluca Marin-Perianu, and Paul Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *23th International conference on architecture of computing systems 2010*, pages 1–10. VDE, 2010.

[86] Amir Nadeem, Ahmad Jalal, and Kibum Kim. Accurate physical activity recognition using multidimensional features and markov model for smart health fitness. *Symmetry*, 12(11):1766, 2020.

[87] Roya Alizadeh, Yvon Savaria, and Chahé Nerguizian. Human activity recognition and people count for a smart public transportation system. In *2021 IEEE 4th 5G World Forum (5GWF)*, pages 182–187. IEEE, 2021.

[88] Felicity R Allen, Eliathamby Ambikairajah, Nigel H Lovell, and Branko G Celler. Classification of a known sequence of motions and postures from accelerometry data using adapted gaussian mixture models. *Physiological measurement*, 27(10):935, 2006.