Action recognition in depth videos using nonparametric probabilistic graphical models

Action Recognition in Depth Videos using Nonparametric Probabilistic Graphical Models

Natraj Raman

Submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science and Information Systems

Birkbeck, University of London

October, 2016

Abstract

Action recognition involves automatically labelling videos that contain human motion with action classes. It has applications in diverse areas such as smart surveillance, human computer interaction and content retrieval. The recent advent of depth sensing technology that produces depth image sequences has offered opportunities to solve the challenging action recognition problem. The depth images facilitate robust estimation of a human skeleton's 3D joint positions and a high level action can be inferred from a sequence of these joint positions.

A natural way to model a sequence of joint positions is to use a graphical model that describes probabilistic dependencies between the observed joint positions and some hidden state variables. A problem with these models is that the number of hidden states must be fixed a priori even though for many applications this number is not known in advance. This thesis proposes nonparametric variants of graphical models with the number of hidden states automatically inferred from data. The inference is performed in a full Bayesian setting by using the Dirichlet Process as a prior over the model's infinite dimensional parameter space.

This thesis describes three original constructions of nonparametric graphical models that are applied in the classification of actions in depth videos. Firstly, the action classes are represented by a Hidden Markov Model (HMM) with an unbounded number of hidden states. The formulation enables information sharing and discriminative learning of parameters. Secondly, a hierarchical HMM with an unbounded number of actions and poses is used to represent activities. The construction produces a simplified model for activity classification by using logistic regression to capture the relationship between action states and activity labels. Finally, the action classes are modelled by a Hidden Conditional Random Field (HCRF) with the number of intermediate hidden states learned from data. Tractable inference procedures based on Markov Chain Monte Carlo (MCMC) techniques are derived for all these constructions. Experiments with multiple benchmark datasets confirm the efficacy of the proposed approaches for action recognition.

Declaration

This thesis is the result of my own work, except where explicitly acknowledged in the text.

Publications

Portions of this thesis have been published.

[1] Raman, Natraj, Stephen J. Maybank, and Dell Zhang. "Action classification using a discriminative non-parametric Hidden Markov Model." *In International Conference on Machine Vision (ICMV)*, London, UK, vol. 9067, p. 10, December 2013.

[2] Raman, Natraj, and Stephen J. Maybank. "Action classification using a discriminative multilevel HDP-HMM." *Neurocomputing (Journal)*, vol. 154, pp. 149-161, 2015.

[3] Raman, Natraj, and Stephen J. Maybank. "Activity recognition using a supervised non-parametric hierarchical HMM." *Neurocomputing (Journal)*, vol. 199, pp. 163-177, 2016.

[4] Raman, Natraj, and Stephen J. Maybank. "Non-parametric Hidden Conditional Random Fields for Action Classification." *In IEEE International Joint Conference on Neural Networks (IJCNN)*, Vancouver, Canada, July 2016 (preprint).

Acknowledgements

I would like to express my sincere thanks to Prof. Steve Maybank for his insight, feedback and support. Steve's breadth and depth of knowledge is a huge inspiration and the opportunity to study under his supervision was one of the main reasons I pursued a PhD course in Computer Vision. Steve – the rigour and patience with which you review the works is unparalleled and I am grateful for all your comments.

Thanks are also due to my second supervisor Dr. Dell Zhang for his comments and participation in the review meetings. I thank Prof. Mark Nixon and Prof. Shaogang Gong for their suggestions. Finally, I would like to thank my fellow PhD colleagues in the Birkbeck computer science department who were a constant source of motivation.

Contents

List of Figures

List of Tables

Notational Conventions

General	
$\mathbb{R}$	The set of real numbers
$\mathbb{Z}^+$	The set of positive integers
$\mathbb{I}(a = b)$	An indicator function that evaluates to 1 if $a = b$, 0 otherwise
x^n	The n^{th} training example sequence
y^n	The class that the n^{th} training example belongs to
x_t	Observation at time instant t
z_t	Hidden state at time instant t
θ	Set of all model parameters
HDP-HMM	
β_k	Probability of transitioning to state k
π_{jk}	Probability of transitioning to state k given state j
γ	Dirichlet Process hyper parameter for β
α	Dirichlet Process hyper parameter for π
μ_k, Σ_k	Mean and covariance of Gaussian distribution corresponding to component k
μ_0, Σ_0	Gaussian distribution hyper parameters for μ
ν_0, Δ_0	Inverse Wishart distribution hyper parameters for Σ
Chapter 4	
φ_{jk}^c	Probability of transitioning to state k given state j for class c
λ	Dirichlet Process hyper parameter for φ
ρ_k^c	Parameter for shifting mean μ_k for class c
Λ_k^c	Parameter for scaling covariance Σ_k for class c
ω_{jk}^c	Parameter used for scaling φ_{jk}^c for class c
Ω_0	Hyper parameter for ρ
ϑ_0, σ_0	Hyper parameters for Λ
ε_0	Hyper parameter for ω
θ^c	Set of model parameters for class c
$\theta^{\setminus c}$	Set of model parameters excluding class c
θ^s	Set of model parameters shared for all the classes
L	Upper bound on the number of HMM states
ξ_0	Prior controlling importance of discriminative term
ζ_0	Prior controlling the distance between distributions
Chapter 5	
a_t	Action state at time instant t
$\overline{a}$	Empirical frequencies of the action states
$\boldsymbol{a}$	Action state sequence from a sampling iteration
ρ_k^a	Probability of transitioning to pose state k given action a
φ_{jk}^a	Probability of transitioning to pose state k given state j, action a
f_t	Binary variable indicating whether a sequence of actions is complete
ψ	Probabilities of completion for action state
ϱ	Dirichlet Process hyper parameter for ρ
τ	Dirichlet Process hyper parameter for φ
$\kappa a, \kappa b$	Hyper parameters for ψ
η	Regression coefficients
λ	Hyper parameter for the regression coefficients
K^a	Upper bound on the number of action states
K^s	Upper bound on the number of skeleton states

K^o	Upper bound on the number of object states
$V(.)$	Forward message value
$m(.)$	Backward message value
Chapter 6	
Z	Normalization constant
ψ	Potential function
φ^{LBL}	Feature function for dependency between a hidden state and a label
φ^{TRN}	Feature function for dependency between two hidden states and a label
φ^{OBS}	Feature function for observations dependency
θ^{LBL}	Parameter group corresponding to φ^{LBL}
θ^{TRN}	Parameter group corresponding to φ^{TRN}
θ^{OBS}	Parameter group corresponding to φ^{OBS}
σ^2	Global scale
ϕ	The scale variable with exponential distribution prior
$\boldsymbol{\eta}$	The set of scale variables with the HDP prior
η^{LBL}	The scale variables corresponding to θ^{LBL}
η^{TRN}	The scale variables corresponding to θ^{TRN}
η^{OBS}	The scale variables corresponding to θ^{OBS}
κ	Dirichlet Process hyper parameter for η

1. Introduction

The topic of this thesis is introduced in this chapter. It begins with the motivation for action recognition in Section 1.1. This is followed with a discussion on the use of depth images and graphical models in Section 1.2. The specific problems that this thesis investigates are described in Section 1.3. The main contributions of this thesis are listed in Section 1.4 and finally the thesis structure is outlined in Section 1.5.

1.1 Motivation

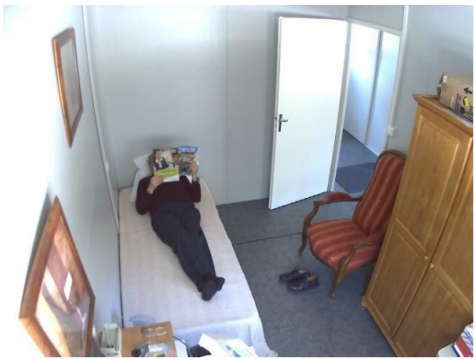
Videos provide visualization of complex and dynamic situations in an intuitive manner. They are a popular medium to convey information. The rate at which video data are generated has increased very rapidly of late due to the ubiquitous availability of devices that record videos. There are an estimated 50 million hours of footage generated every day by the surveillance cameras in the U.K. [57] and about 400 hours of video is uploaded every minute into the popular YouTube website [56]. With the advent of future developments in wearable devices, the amount of video content will increase even further. It is difficult to interact with such enormous amounts of video data without efficient tools that automatically describe, organize and manage them.

In order to effectively describe the content in a video, the objects and events occurring in the image sequences that comprise the video must be detected and recognized. State-of-the-art tools in computer vision provide the ability to detect and recognize the objects and their properties in images [59, 60]. However, robust and accurate recognition of events that occur in image sequences is still a problem. This is unsurprising since the cognitive underpinnings for understanding events are much more complicated. It requires application of complex spatiotemporal concepts. The research here addresses this challenging computer vision problem and provides mechanisms to recognize events involving humans in videos.

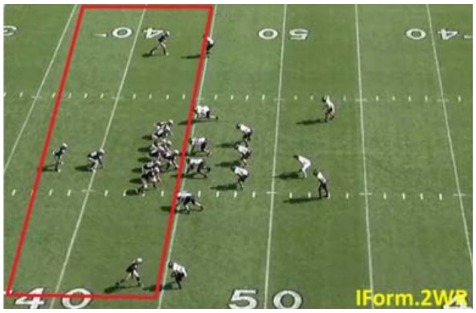
Automatic human event recognition has many applications across various domains (Figure 1.1). For example, in the security domain, there is an ever increasing need to monitor video feeds for interesting events. These video feeds may originate from CCTV cameras or from other sophisticated platforms used by the military such as unmanned aerial and ground vehicles. The current monitoring solution involves dedicated human operators actively watching live video streams. This is often undesirable since the human operators are expensive resources and it is difficult for them to remain focused at all times. Instead, an automated system that can detect

and recognize interesting events and then alert the human operators is required. Such a system is cost-effective and eliminates potential security risks.

(a)



(b)



(c)

(d)

(e)

Figure 1.1: Applications of automatic event recognition. (a) A smart surveillance system that detects interesting events in live video feeds [60]. (b) Monitoring the daily living activities in a care centre [61]. (c) Analysing an American football sports video for offensive team formation [3]. (d) Natural user interaction with a games console for a better gaming experience [62]. (e) Touchless interaction for browsing and manipulating medical images during surgery in an operating theatre [2].

Smart surveillance systems that discard routine events and highlight only interesting events have applications in other domains such as healthcare. For example, in a care centre for the elderly, the automatic recognition of an inmate's irregular sleep patterns, changes in the

frequency of toilet use and difficulties in performing regular activities helps in assessing the cognitive and physical well-being of the person [63]. Health monitoring surveillance systems can reduce expenses and improve the quality of life for the elderly.

Multimedia information retrieval is another important area where automatic event recognition is essential. A content based search and retrieval system would enable the efficient explorations of large volumes of archived video data. As example use-cases, a user may wish to view all archived videos that contain a *wedding* event or a security professional may wish to review frames that contain an *explosion* event in surveillance footages. The current technique for searching the videos is limited to metadata queries and text search based on manual annotations. Instead, searching directly for user-defined events provides a comprehensive mechanism to interact with the video content. With automatic event recognition, the videos can be indexed analogously to text document indexing and abstracts such as key frames or highlights can be extracted to form condensed summaries of the videos. In effect, the videos can be managed as structured artefacts and analysis can be performed on their contents [1]. Content based search, retrieval and analysis of videos have applications in innumerable areas including sports, education and arts.

The pervasive use of computing has encouraged researchers to explore more natural and intuitive mechanisms to interact with computers. In addition to voice and hand gestures, the use of the entire human body to communicate with computers has gained traction of late. For example, the Microsoft Xbox game consoles allow players to interact through their full body without the need for a games controller [62]. The player can perform actions such as *kick* or *jump* to naturally convey their intended motion to the console. This provides an immersive gaming experience for the player. In order to respond to player movements the console must detect and recognize the various events that occur during the interaction. The applications for such natural ways of interacting are not restricted to entertainment platforms. They can also be used in many other scenarios such as medical surgery. A surgeon can control and manipulate equipment without explicit contact, thus maintaining the boundaries between sterile and non-sterile parts of the surgical environment [2].

The above wide range of applications in diverse areas such as smart surveillance, content retrieval and human computer interaction provides a motivation for addressing the human event recognition problem.

1.2 Research Focus

The term “event” can refer to a variety of concepts at different levels of abstraction and at different time scales, ranging from elementary movements of a body part by an individual to

complex interactions between persons that can last for hours. In order to distinguish between the different types of events, a standard terminology [4] is followed. An elementary motion such as *raising a leg* is referred to as a "gesture"**.** The composition of multiple elementary motions, carried out by a single person and organized temporally is referred to as an "action". *Walking* and *sitting down* are examples of actions. The term "pose" refers to a particular configuration of the human body that is encountered while performing an action. Hence gestures and actions can alternatively be described by sequences of poses. An "activity" is composed of a set of actions that occur over time. For example the activity *rinsing the mouth* may contain *drink* and *spit* actions. This thesis focuses exclusively on action recognition for videos that involve a single individual and last less than a minute.

1.2.1 Recognition from Joint Positions

The famous Johansson experiments [12], illustrated in Figure 1.2, demonstrate that motion can be perceived from sparse visual input. It was shown that moving light displays attached to a small number of landmark joints on the human body provide sufficient motion cues to infer actions such as *walking*, *running* etc. The visual system can detect motion patterns by integrating the movements of individual joints over space and time. The absence of shape, colour and texture information does not inhibit the recognition of the motion. The use of a handful of body joints to model articulated human motion produces a compact representation for the human actions. Hence determining the locations of the joints corresponding to the various body parts and modelling the spatiotemporal transitions of these joint positions provides the necessary information to characterize motion and infer actions and activities.

Figure 1.2: Biological motion perception. Point lights are placed on joint locations. When a sequence of these point lights is viewed, the actions *walk* and *run* are apparent even though the figure outline is omitted [12, 13].

Recovering the body joints from images is a very difficult problem because there is a fundamental loss of information when a 3D scene is projected into a 2D image. It is often not possible to robustly identify the body parts in an image. The pixels in an image typically encode

intensity variations as RGB colour values. Different lighting conditions induce variations in the recorded pixel values. Human body parts in an image might be partially occluded by other objects or by the parts themselves from time to time. The image may contain shadows. Further, background clutter may make it difficult to locate the objects of interest and perspective deformations can make it difficult to recognize the objects. Even though the RGB videos contain rich visual information, their sensitivity to lighting conditions and the difficulty in performing robust background subtraction in these videos pose significant challenges for estimating articulated human body motion [6, 142].

A depth image, which contains information relating the distance of an object in a scene to a camera, is less affected by the above image representation issues. The depth images are robust to colour and texture variability induced by clothing, hair and skin of a human body. It is much easier to detect the human body silhouette using depth information rather than RGB values. The 3D data that includes depth information simplifies background subtraction, resolves silhouette ambiguities and is largely invariant to lighting, colour and texture [7, 9].

The traditional way to obtain 3D data is stereo vision [8] in which the depth information is reconstructed by capturing 2D images from multiple viewpoints. Unfortunately, the inference of depth information involves complex stereo geometry calculations and is affected by reflections, depth discontinuities and sparse textures in the images. Stereo vision suffers from the same lighting and segmentation problems associated with colour images. The need for multiple synchronized cameras and the unreliable depth information produced by an expensive reconstruction process limits the applications of stereo vision [5]. An alternative is to use motion capture systems [86] in which special markers are attached to the body and the 3D joint positions are obtained by triangulation using multiple cameras. Even though this procedure provides accurate body motion, its intrusive nature is infeasible in real world scenarios and the high cost of the hardware restricts its application to niche areas.

Recent advances in depth sensing technology have provided cameras that produce synchronized colour and depth images. The Microsoft Kinect sensor [10] contains an infrared projector, an infrared camera and a colour camera. It produces reasonably accurate depth images in addition to the RGB images at high frame rates. The distance of the 3D points in the world from the image plane is recorded as pixel values in the depth image as shown in Figure 1.3. Note that the sensor can provide depth information only up to a limited distance and the depth estimates are sometimes inaccurate. Further, the captured structure is pseudo 3D because the points that are not in front of the sensor cannot be recorded. In spite of these limitations, the low-cost and relatively small footprint of these sensors make them a popular choice for recording depth images.

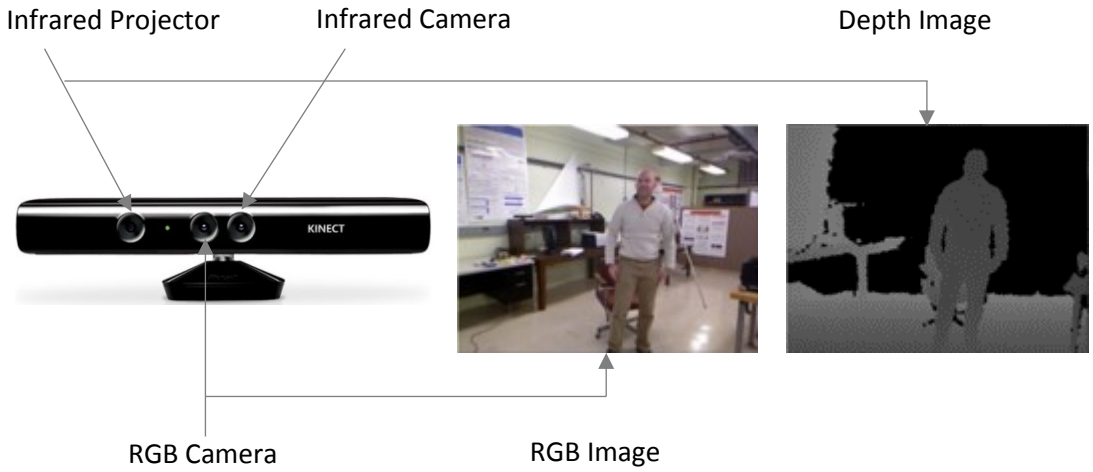


Figure 1.3: The Kinect sensor. The RGB image is produced by a RGB camera and the depth image is produced by an infrared projector and an infrared camera. The points close to the camera have darker pixel values. The black pixels indicate that depth values are not available for those pixels [10, 11].

The detection of joint positions is greatly simplified by the use of depth images. The pioneering work in [14] introduced a mechanism to robustly classify the depth image pixels associated with a human body, by assigning to them an appropriate body part label. The locations of the joints can then be estimated from these pixel labels. An overview of this approach is provided in Figure 1.4. The algorithm is computationally efficient and is built into the Kinect sensor so that the joint positions are provided in real-time.

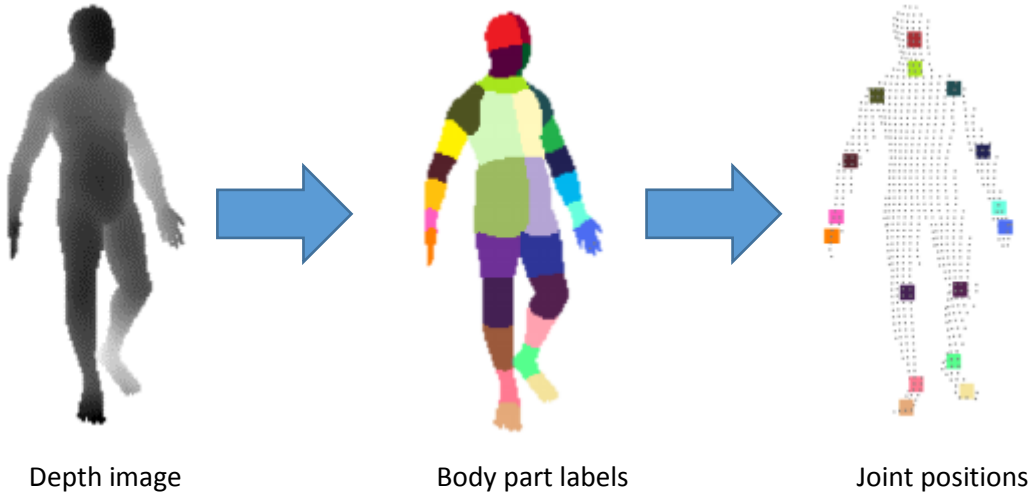


Figure 1.4: Joint positions estimation. An intermediate labelled image in which each pixel is classified into a body part is inferred from the depth image. The 3D joint positions are estimated from the labelled image [14, 9].

Inspired by the Johansson experiments and the recent breakthrough in depth sensing technology, the research in this thesis uses the locations of joints estimated from depth images to characterize the motion patterns. The action classes are modelled using sequences of these joint positions.

1.2.2 Challenges

Even with the availability of body joint positions, recognizing actions is not that simple. There exists similarities in different action classes and there are often differences within the same class of actions. For example, *walk* and *run* actions involve similar sets of joints. The movements for a *walk* action can differ in speed and style between individuals. As the number of action classes increase, the overlap between them will be higher, making it much harder to distinguish between actions of different classes. The actions are also of varying duration with sequences of different lengths. This makes them difficult to compare.

The joints information may be corrupted by noise due to inaccurate depth estimates. It may also be necessary to change the coordinate system of the positions to account for differences in recording environment and variations in size and shape between humans. In many cases, the joints space is of high dimension containing redundant information and it is important to find compressed representations to facilitate computationally inexpensive comparisons between the actions.

The need to generalize over large intra-class variations and maximize small inter-class distinctions, along with the need to handle temporal variations and noisy sequences make action recognition intrinsically challenging. Application of advanced statistical machine learning techniques is required to address this problem.

1.2.3 Graphical Models

Action recognition is usually regarded as a supervised classification problem [35]. Prototypical examples of videos and their corresponding action class labels are made available for training. The prediction of action class labels for new unseen videos is based on the information learned during training. What distinguishes action classification from traditional supervised classification is that an input observation is a sequence of data points that are strongly correlated over time. In effect, action classification is a sequence labelling problem in which each sequence of data is assigned a sequence of class labels.

A natural way to model the sequential data is to introduce a discrete valued state variable that compactly represents the observed data at a time instant. These state variables can then be reasoned about, as they evolve over time. The discrete valued state at a particular time is a snapshot of the relevant attributes of the observed data at that time [15]. As an example, in a

clap action, the various intermediate body poses such as *hands together*, *hands apart* etc. may correspond to different state variables and by examining the transitions between these state variables (i.e. body poses), an action is inferred. Since these states are not explicitly observed in the input data, they are often referred as hidden states or latent states.

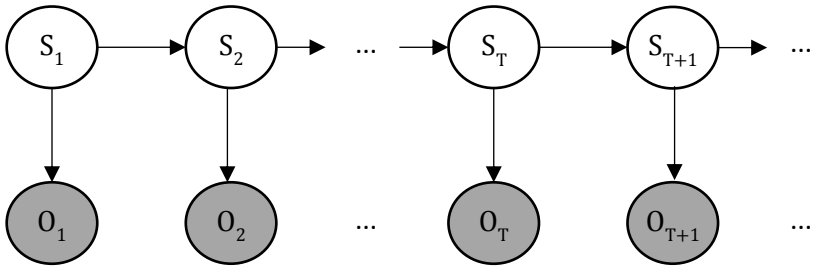


Figure 1.5: Sequential data in a graphical model. The state variables S describe the observations O_t at various time instants 1, 2, … , T, T+1 etc. The dependency relations between the variables are expressed in a graph structure. The states are conditioned only on the previous state and not on the entire history.

It is essential to perform a probabilistic reasoning over these state variables to account for uncertainties in the outcomes. For a probabilistic formulation, a joint distribution over the space of possible states must be constructed. It is daunting to represent these distributions over many variables naively. A diagrammatic representation provides mechanisms to visualize the structure in these complex distributions and exploit them. Probabilistic graphical models use a graph based representation to simplify dependencies over many variables to a smaller subset of variables. The nodes in the graph correspond to the variables and the graph edges express the dependency relationship between these variables.

It is impractical to assume that the future states depend on all previous states. Such an assumption leads to an intractable model that grows with the number of observations. A reasonable approximation would be to consider that the past is independent of the future given the present. This *Markov assumption* shown in Figure 1.5, together with the assumption that all the data are generated from the same distribution, allows the modelling of sequential data in a compact form [16].

The Hidden Markov Model (HMM) [32] is a well-known graphical model that is used to represent sequential data. An HMM uses a set of discrete states and a state is conditioned only on the state at a previous time instant. The Conditional Random Field (CRF) [33] is another probabilistic

model that uses a graph based representation to encode relations between states at different time instants. While the HMMs use directed graphs, the CRFs use undirected graphs.

The research in this thesis uses discrete state-space graphical models such as HMM and CRF to deal with the dynamics regulating the temporal evolution of the body joints. The graph based declarative structure provides a flexible framework for encoding complex interactions between many variables. Further, it also enables the development of a generic solution with the representation and inference procedures applicable to problems in many other domains.

1.3 Problem Definition

A general problem with the graphical models that use discrete state variables is that the number of hidden states must be fixed a priori. This number is not known in advance for most applications. Let us the take the example of the action class models described above in which the state variables represent the various body poses. A prior knowledge of the exact number of intermediate poses that are involved when performing an action is not available. The motion patterns and body positions may vary subtly between two subjects who perform the same action and consequently the number of poses may depend on the number of subjects. Further, these numbers must be specified separately for every action since almost certainly the number of poses will differ between actions depending on their complexity. If a large number of states is specified, it may result in a complex model that over fits the data and fails to fit new observations. A small number of states may not be adequate to capture the variations in the data.

The classical solution for this problem is to perform model selection – several models are fit to the data and then one of the models is selected using a model comparison metric. In the above problem, typically several models are trained with different numbers of states and a procedure such as cross-validation or regularization is used to choose a model with the correct number of states. In cross-validation, the model is evaluated on small subsets of the training data to see how well it generalizes and in regularization a penalty term that favours a simpler model is incorporated during training [17].

Unfortunately such procedures do not adapt well to changes in data complexity. Instead of these ad-hoc procedures that compare multiple models which vary in complexity, it is preferable to fit a single model that estimates the number of states automatically from data. Such a mechanism avoids any misfit between the number of states and the amount of training data. The model complexity, as measured by the number of states, increases as the amount of data increases. However, the formulation of a model with an unbounded complexity is nontrivial. The set of all possible solutions must be considered and the parameter space is now infinite dimensional.

A model over an infinite dimensional parameter space can be defined using Bayesian nonparametric methods [18, 73]. These methods employ an unbounded number of parameters but only a small subset of these parameters are actually used. Appropriate prior distributions control the number of parameters required to model the data. Small datasets produce simple models while complex datasets induce rich models, thereby adapting the effective model complexity to the data. The lack of an upper bound on the number of parameters mitigates under-fitting while the computation of a posterior distribution of the parameters in a Bayesian approach reduces the chance of over-fitting.

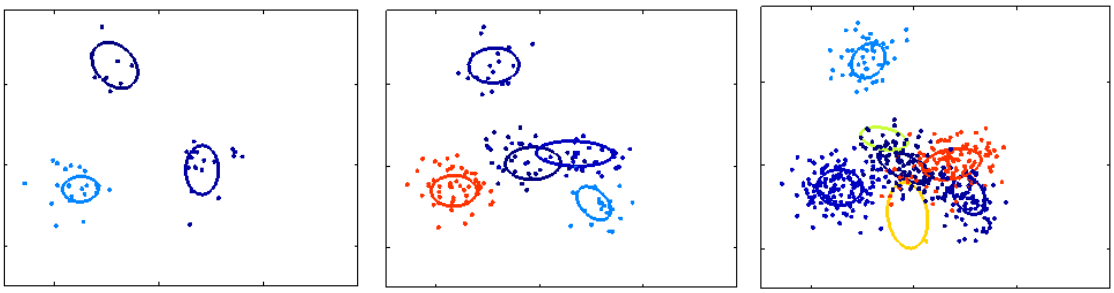


Figure 1.6: Clustering and Dirichlet processes. The data points are generated from a mixture of 2D Gaussians with 50 data points in the left, 150 data points in the middle and 500 data points in the right. The clusters learned through Dirichlet Process are shown as ellipses. The number of clusters increase with the number of data points.

The Dirichlet process [19] is one of the most popular priors employed in Bayesian nonparametric methods. It is a distribution over distributions i.e. a sample drawn randomly from a Dirichlet process is itself a probability distribution. A common application of Dirichlet process is as a prior distribution in mixture models used for clustering data. In mixture models, each data point is assumed to belong to a cluster, with the data points inside a cluster distributed randomly within that cluster. The number of clusters must be specified a priori in classical clustering techniques. The use of a Dirichlet process prior instead provides a mechanism that estimates both the number of clusters and the parameters of the distributions characterizing the clusters simultaneously from data. An unbounded number of clusters is available, but only a small number of them are used to model a given set of data points. Large clusters grow larger, faster. When the number of data points increases, new clusters may emerge as illustrated in Figure 1.6. This nonparametric solution is evidently better at dealing with the combinatorial challenge associated with model selection procedures.

Although the use of Dirichlet Process as a nonparametric prior for graphical models was explored before [40, 41, 44] these techniques by themselves are unsuitable for a supervised classification

problem. A straight forward application of these techniques would use a separate model to represent each action class and define a joint distribution over the input observations and the class label. Such generative models *describe* the input while in a classification problem the objective is to *discriminate* between the inputs. Formulating models such that they provide the best decision boundaries to distinguish the classes is necessary. Furthermore, the use of separate models prohibits the sharing of valuable information across the different action classes. Information exchange between classes is essential to facilitate effective learning with a small number of training examples. It is important to consider a nonparametric prior that is suitable for classification tasks.

The central computation problem in Bayesian nonparametric methods is posterior inference – i.e. estimating the posterior distribution of the model parameters given the observed data. The posterior distribution often has a highly complex form. Except in the simplest cases, there are no closed form expressions readily available to evaluate the posteriors analytically. The use of sequential data compounds the problem. When deriving inference algorithms, it is important to consider multiple variables together and make large moves in the probability space for computational efficiency.

The research presented in this thesis deals with the important problem of choosing models at an appropriate level of complexity and ensuring that these models are suitable for supervised classification. It investigates the following research questions in the context of action recognition:

Question 1. How to represent actions and activities using graphical models?

Question 2. How to learn the number of states in the graphical models from data rather than using model selection procedures?

Question 3. How to share information between the action classes?

Question 4. How to ensure that the models are discriminative in nature so that the best decision boundaries to distinguish the actions can be found?

Question 5. How to perform efficient posterior inference over the model parameters?

1.4 Thesis Contributions

Motivated by the lack of existing methods to address the above questions, this thesis proposes three different and original constructions of nonparametric graphical models that are suitable for action classification.

The actions are represented using the Hidden Markov Model (HMM) [32] and Hidden Conditional Random Field (HCRF) [95], two well-studied discrete state-space graphical models used widely in sequential pattern recognition. The activities contain an inherent hierarchical structure and they are represented using a Hierarchical Hidden Markov Model (H-HMM) [81]. All the three models use 3D joint positions obtained from depth video to define the features. In the attempt to answer Question 2, nonparametric variants of the canonical HMM, H-HMM and HCRF are developed. This avoids ad-hoc model selection procedures and flexibly adapts the state cardinality to changes in data. Further, the model parameters are formulated in terms of distributions that are common across the classes to facilitate information sharing. To address the fourth question, the models are constructed in such a way that they are suitable for supervised classification problems. Finally, posterior inference procedures that are efficient for sequential data are derived for all the models based on simulation [36] techniques. The main contributions are summarized as follows:

A discriminative nonparametric HMM for action classification

The classical HMM is extended with a nonparametric prior and augmented with a discriminative term. The resulting model infers the number of hidden states automatically, with the model parameters learnt in a manner that is suitable for classification tasks. The model formulation promotes effective transfer of information between action classes. The model is evaluated for action classification on benchmark depth video datasets containing locations of joints.

A supervised nonparametric H-HMM for activity classification

A hierarchical extension to the HMM with an unbounded number of action and pose states is developed. The formulation uses multinomial logistic regression to distinguish between the activity classes based on action states, thereby simplifying the model structure. The model efficacy is demonstrated for activity classification with joint positions and depth information used to characterize activities.

A nonparametric HCRF for action classification

A nonparametric extension to the HCRF that precludes the need to specify the number of intermediate hidden states is proposed. The discriminative HCRF models the classification rules directly. The Bayesian treatment of the training procedure provides realistic characterization of uncertainty in the parameters. Good classification results are achieved in two different depth video datasets containing human actions.

The proposed models are applicable to a wide variety of sequence labelling problems, besides action sequences. The investigations in this thesis have been published in [192, 193, 194, 195].

1.5 Thesis Structure

These contributions are discussed in greater detail in the subsequent chapters of this thesis. A brief description of the remaining chapters is as follows:

Chapter 2 reviews a broad range of works that are related to this thesis. The approaches used for vision based human action recognition in the literature are surveyed. The various features extracted from the depth images are discussed in detail. The different classification techniques are outlined. A review of the nonparametric solutions used in the literature and how they compare with the work in this thesis is also included.

Chapter 3 provides the technical background necessary to describe the models used in this thesis. The HMM and CRF models are introduced. The Dirichlet process, which is extensively used as a nonparametric prior in subsequent chapters, is described. Further background information including the techniques used to construct depth images and the statistical framework upon which the action class models are built is provided in the Appendix.

Chapter 4 presents an action classification technique using a discriminative nonparametric HMM. The action classes are represented by a multi-level Hierarchical Dirichlet Process (HDP) HMM. The model parameters are formulated as transformations from a base distribution and are learnt in a discriminative manner. The chapter begins with the motivation for this approach, presents the model and derives the posterior inference mechanism. Finally the experiments section discusses the results obtained on two different datasets.

Chapter 5 develops a two level hierarchical HMM to perform activity classification. The bottom level states characterize granular poses while the top level states characterize the coarser actions associated with activities. In order to perform classification, the relationship between the actions and activities are captured using multinomial logistic regression. The chapter begins with an overview of the approach, provides the activity model structure and explains the inference mechanism. The evaluations conducted on two different datasets are also discussed.

Chapter 6 proposes the use of a HCRF for classifying actions. The classical HCRF is extended with a nonparametric structure and the number of hidden states is automatically inferred. The training and inference procedures are fully Bayesian. The construction is based on scale mixtures of Gaussians as priors over the HCRF parameters and uses the slice sampling technique during inference. The model representation and the mechanism to perform Bayesian inference are presented along with the experiments.

Chapter 7 concludes the thesis by summarizing the main contributions. Several future directions and perspectives of the proposed techniques are presented.

2. Related Work

The aim in vision based action recognition is to determine the action type of a previously unseen video. It is an active research area and the vast amount of papers published in the literature every year related to this topic is a testimony to both its importance and the challenges involved. This chapter reviews the existing literature on action recognition. Over the years many techniques have been proposed. The focus in the review here is mainly on the approaches to recognition based on depth images and a graphical model based representation.

The approaches differ mainly in the features and the classification algorithms that are used. The various feature descriptors extracted from the image sequences are discussed in Section 2.2. The different classification techniques, ranging from those that explicitly model the temporal dynamics of the motion to those that do not, are covered in Section 2.3. The Bayesian nonparametric framework is used in the recognition procedure presented in this thesis. Section 2.4 surveys the various nonparametric approaches. A final summary is provided in Section 2.5. This chapter provides insight into how the thesis differs from the other related work.

2.1 Overview

The research efforts in vision based action recognition date back as far as the early 1990s when Yamato et al. [104] used Hidden Markov Models to classify tennis strokes. Some of the early methods used for motion analysis are reviewed in [105]. A variety of approaches have been proposed since then and there are several surveys in the literature that provide an overview of these methods. Some of the surveys are discussed below.

The techniques used for tracking, pose estimation and recognition are surveyed in [106]. The review presented in [107] expands the recognition scope to include methods used for interpreting cognitively higher level activities. The survey in [55] covers the various features that are extracted from the image sequences for action classification. In [4], a comprehensive summary of the approaches used for activity analysis is presented using a tree structured taxonomy. Yet another survey [108] lists the methods used for representing, segmenting and learning actions. The recent survey in [110] discusses the state-of-the-art research using the taxonomy defined in [4]. A survey of the datasets available for human action recognition is presented in [109].

The above surveys deal mainly with action recognition using visible light colour images. With the widespread availability of low-cost depth sensors, there has been lot of research interest of late in using the depth image sequences for human motion analysis. There are a few survey papers that review approaches based on 3D data in the context of action recognition. The surveys in [6] and [111] discuss depth data acquisition and the pre-processing steps involved. In addition, they review the algorithms used for action analysis. The other surveys that focus on human action recognition with 3D data include [5, 112, 113] and the very recent [114].

Most action recognition methods assume that some examples of videos and their corresponding action class labels are available. A typical system first defines an abstract and compact representation of the patterns in a video, commonly referred as features. A model is then learned for the action classes during a training process using the features extracted from the example videos. Given a video whose action label is not known, this video is matched against the learned model in order to classify it. The variations in recognition methods are mainly based on the features and the classification algorithms used for matching the features.

2.2 Features

This section discusses the methods used to determine an image sequence representation that is suitable for robust classification of the actions. It is important to choose informative and discriminative features. This process, known as feature extraction, is treated as the core problem in many action recognition works. Off-the-shelf classifiers are often used for matching the features once they are obtained.

It is crucial to capture the temporal correlations between the images in the video for successful recognition. Some methods extract the features frame by frame and convert the video into a sequence of feature vectors. The matching algorithm used during classification analyses this sequence to deduce the action. In other methods, the features explicitly include temporal information.

While the range of features used for action recognition can seem overwhelming, the majority of them can be divided broadly into two categories: image based and skeleton based. In the latter, an explicit model of the human body is defined and pose estimation is performed on the images to determine the configuration of the body. This provides the skeleton – a schematic representation of the locations of the body parts. The features are then chosen from the positions of the joints that are part of the skeleton. In contrast, image based methods avoid reconstructing the human form and rely on extracting features directly from the images in a video. It is not usual to define an intermediate body model or explicitly identify the body parts.

The two categories are discussed in detail below with an emphasis on the features used for action recognition in depth images. Figure 2.1 lists the feature types discussed in this review.

2.2.1 Image Based Features

A diverse palette of low-level visual features has been proposed for action recognition. The image based features fall under two types – those in which the features are encoded from the human as a whole and those that use a collection of local descriptors obtained from several image patches. Some methods use both types of feature. A pose estimation procedure is typically not performed when computing image based features. These features can be extracted even from images in which the resolution is low.

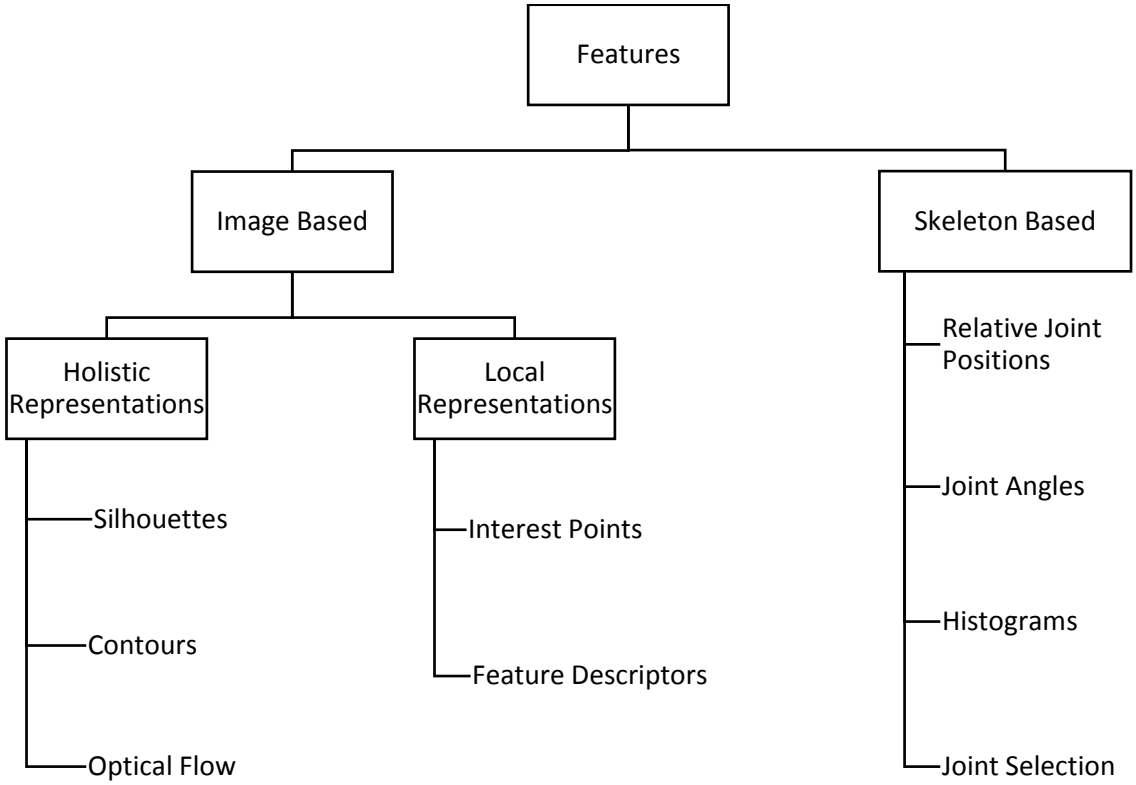


Figure 2.1: Features types. The various features used for action recognition are shown in a schematic representation. See text for more details.

Holistic Representations

The holistic representations consider the image region of interest in full. They often follow a top down approach, first detecting and extracting the human being before computing the features. The actions are characterized using the appearance and motion information obtained from the localized human. These methods are generally sensitive to noise and are affected by variations in viewpoint and occlusion [55]. However, they have been used successfully in many action recognition works both for colour and depth videos.

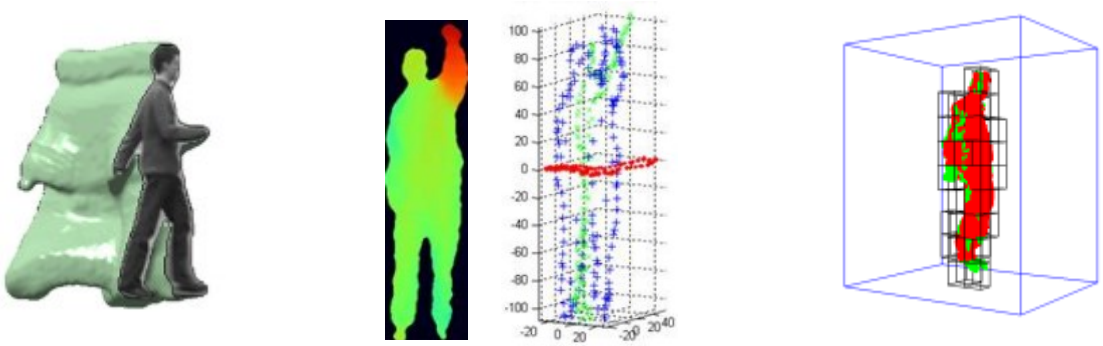
The human silhouette, in effect the foreground of a person in an image, provides a simple representation that carries useful shape information about the body pose. The evolution of the silhouettes over time can be used to recognize the actions. Instead of taking into account all the pixels within a silhouette, sometimes only the boundary pixels are used. These boundary points, which contain no information about the internal structure of the image, have also been used to approximate the body poses.

An early work using silhouettes is [115], where the differences between binary silhouettes are accumulated to construct a Motion Energy Image (MEI) and a Motion History Image (MHI). The former indicates where motion has occurred while the latter indicates how the motion evolves in the temporal domain. The MEI and MHI together define an action template and recognition is performed by matching these templates based on a statistical model of the moments. The work in [116] employed an extended Radon transform on the binary silhouette to define features that are invariant to geometrical transformations such as scaling and translation. The actions are regarded as 3D shapes induced by stacking the 2D silhouettes in the space-time volume in [117]. The space-time shapes (Figure 2.2 (a)) encode both the spatial information of the body and the global body motion. The extremities of a human body such as head, hands and feet are used in a representation of the body posture in [118]. These extremities are detected from a body contour. In [119], the contours of the MEI are used to obtain a contour coded MEI that is invariant to scale changes and translations.

It is not always easy to obtain stable shape information from colour images. The robustness of the extracted silhouettes and contours relies heavily on how accurate the background subtraction is. When compared with the colour images, it is much easier to perform background subtraction in depth images. Hence the silhouettes extracted from depth images are usually noise free. The above silhouette based features have been extended successfully from the 2D colour images to the 3D depth images for action recognition.

In [120], the MHI is extended to include the depth information. The resulting three dimensional motion history image (3D-MHI) augments the conventional MHI with additional channels that encode the motion history in the depth changing directions. The pixel values in the 3D-MHI include a history of the increase and decrease in depth values. An activity recognition system for smart homes is developed in [121] using depth silhouettes. The extended Radon transform employed for the binary silhouette in [116] is extended here to the depth silhouettes. The ambiguity for different poses is more pronounced among the binary silhouettes, while the depth silhouettes, with a richer set of intensity values, provide a better mechanism to differentiate between the poses.

In [74], a small set of representative 3D points (Figure 2.2 (b)) sampled from the depth silhouette is used to characterize the shape of salient postures. The idea here is that the points inside the silhouette carry redundant information and the body shape can be described sufficiently by a small number of extreme points of the contour. The depth map is projected on to the three orthogonal Cartesian planes XY, YZ and XZ and points are sampled at equal distance along the contours of the projection. The temporal dynamics of these sampled points are used to infer the actions. A similar planar projection method is used in [122]. The depth maps are projected on to the three orthogonal Cartesian planes and the motion energy obtained from the projected maps are stacked together to form Depth Motion Maps (DMM). The DMM representation encodes information about the body shape and motion in three projected planes and provides strong discriminative clues about the actions.



(a) (b) (c)

Figure 2.2: Holistic representations. (a) Space time shapes used in [117], containing both the spatial information as well as the motion information of the silhouette. (b) Representative 3D points sampled from the depth silhouette to characterize the shape of a posture in [74]. (c) Depth sequence are represented in a 4D space-time grid with the occupancy value of the grid cells used as features [123].

The approach in [117], where the 2D silhouettes are stacked to create a 3D space-time volume, has been extended to depth sequences as well. In [123], the space and time axes are divided into multiple cells to define a 4D space-time grid for a depth image sequence as shown in Figure 2.2 (c). A saturation scheme is used to enhance the role of the cells and make them suitable for recognition. The obtained feature vectors, called Space-Time Occupancy Pattern (STOP), uses the spatial and temporal contextual information while allowing intra-action variations. In [124], the depth sequence is described using a histogram of oriented 4D surface normal (HON4D). The features capture the distribution of the surface normal direction in the 4D space of spatial, depth and time axes. The 4D space is divided using a 4D extension to a 2D polygon when constructing

the features. It is argued that the distribution of the normal vectors for each cell in the 4D space contains more information than the occupancy patterns.

In addition to the shape based features, optical flow based features have also been used. The pixel wise oriented differences between frames are captured and used to estimate the optical flow in the image regions undergoing change. Optical flow based features are particularly applicable in the cases where background subtraction is difficult and the image resolution is poor. However they may fail when there are sudden changes in motion.

In [125], actions are recognized based on optical flow measurements obtained from sports footage in a setting where the image of a whole person may only be 30 pixels are so tall. The pixel-wise optical flow captures motion independent of appearance. Since the optical flow computation is inaccurate in noisy data, the optical flow vectors are treated as a spatial pattern of noisy measurements. The optical flow is used to extract person-centric motion features in [126] for recognizing actions such as biking, diving etc. in colour videos. In order to allow for the noise in the optical flow, a windowing scheme is used here.

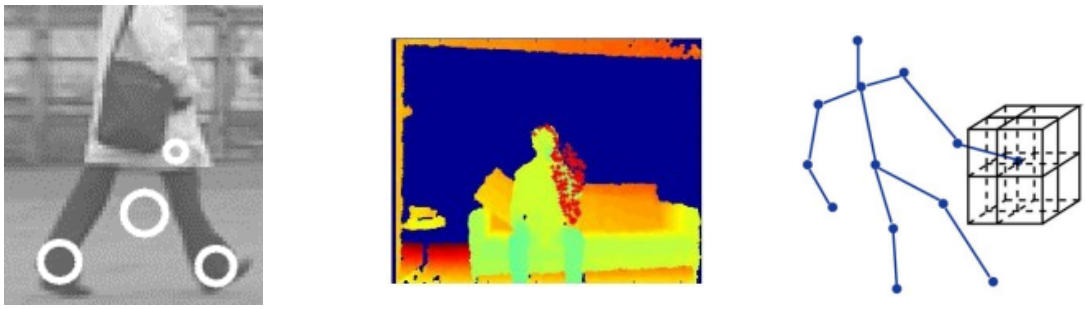
The application of optical flow to depth images for action recognition was explored in [127]. The optical flow is computed as an extension to the third dimension of the traditional 2D optical flow. However, the computation is restricted to some portions of the 3D scene. A grid based descriptor is used for representing the flow information extracted from the point cloud within a temporal sequence. The extraction of optical flow from depth data has been limited. The main challenge is that the computation of optical flow on all the 3D points in a scene is prohibitively expensive.

Local Representations

A collection of features extracted from independent image patches are used in the local representations. These local features effectively capture the shape and motion information in the video. These methods follow a bottom-up approach. First a set of interest points are identified and then the features are extracted from local patches around these interest points. The features from multiple patches are combined together to obtain a final representation.

Unlike the holistic representations, detecting the humans and performing background subtraction may not be necessary with this approach. Hence these methods are suitable even in the situations where action recognition must be performed in unconstrained poor quality videos. The methods are generally less sensitive to noise than holistic representations and may be invariant to rotation and scale. However, it is often computationally expensive to construct the features based on local representations.

The interest points such as corners and edges contain significant local variations of the image intensities and carry information that is stable under small perturbations. They are well studied in the spatial domain and have been applied to many object recognition tasks in static images. The notion of spatial interest point is extended to the temporal domain in [128] by requiring the image values in the spatiotemporal volumes to have significant variations along both the spatial and temporal directions. These spatiotemporal interest points (STIPs), shown in Figure 2.3 (a), correspond to image points that have large image intensity variations and non-constant motion. The features obtained by generalizing the Harris corner detector to the spatiotemporal domain are used in [128] to identify interesting events in image sequences.



(a) (b) (c)

Figure 2.3: Local representations. (a) Spatiotemporal interest points detected in [128] during a walking action. (b) Interest points detected from a depth image sequence for the drink action in [137]. (c) The numbers of points that fall into the cells of a localized spatial grid are used in [88].

There are other extensions of the 2D spatial interest point detection mechanisms to the 3D space-time axes for action recognition. In [129], the image sequences are represented using a collection of points that are salient both in space and time. The 2D saliency metric is based on measuring the changes in the information content of a circular image region over a set of different scales. This is extended to the temporal domain by considering cylindrical neighbourhoods at different scales and temporal depths. The obtained points using the 3D saliency detector correspond to activity variation peaks. A 3D Discrete Wavelet Transform (DWT) is used in [130] to detect spatiotemporal salient regions. The image sequences are represented in a 3D Euclidean space with time as the third dimension. A multiscale 3D DWT is applied to decompose the 3D volume and the resulting coefficients are used to compute saliency. The actions are represented using simple features of the salient regions. The interest point detector in [135] uses a Gabor filter on the temporal domain. At each interest point, a cuboid that contains the spatiotemporally windowed pixel values is extracted to determine the feature vectors. The detector errs on the side of detecting too many interest points rather than

too few. This is motivated by the observation that irrelevant features generated by scene clutter are handled well in object recognition tasks.

While the interest point detector selects locations and scales, the feature descriptors capture shape and motion information in the neighbourhoods of selected points using image gradients. They encode statistics of the pixel distributions. Similar to the extension of interest point detectors from the spatial domain to the spatiotemporal domain, the feature descriptors have also been extended to the spatiotemporal domain and have been applied to action recognition. The well-known Histogram of Oriented Gradients (HOG) descriptor [77] used for detecting humans in static images is generalized in [131] to the 3D spatiotemporal domain. The orientation of the spatiotemporal gradients is quantized using a polyhedron and the gradient histograms of all the 3D cells are concatenated and normalized. The resulting HOG3D descriptor is used to recognize actions. A similar extension is proposed in [132] to compute histogram descriptors of space-time volumes in the neighbourhood of interest points. The resulting descriptor is used to recognize human actions that occur in movie videos.

The local representations based on the spatiotemporal interest points and feature descriptors originally developed for colour images have been extended to depth videos. In [136], a 4-dimensional local spatiotemporal feature that combines both colour and depth information is used for activity recognition. This work is inspired by the local features developed for colour videos in [135]. It uses separate response functions along the spatial and temporal dimensions to detect the interest points. The features are obtained by computing the colour and depth gradients from a 4D hyper cuboid centred at the interest point. The work in [137] detects interest points in the depth image (Figure 2.3 (b)) using the same technique proposed in [135] for visible light images. An additional function is employed for correcting the noise encountered in the depth maps, for example holes and value jumps. A 3D cuboid which contains the spatiotemporally windowed pixel values around the interest points is used to define a descriptor. The various interest point detectors and feature descriptors used for depth images are evaluated in [134]. They include the Harris3D [128] and Cuboid [135] interest point detectors extended for the depth images. The feature descriptors include HOG3D [131], HOG/HOF [132] and HOG [77].

New types of feature descriptors that are motivated directly by action recognition in depth images have also been explored. In [138], a descriptor called Histogram of Oriented Principal Components (HOPC) is proposed to capture the local geometric characteristics around each point within a sequence of 3D point clouds. In order to obtain the descriptor at a point, first Principal Component Analysis (PCA) is performed on a spatiotemporal volume around the point. The resulting Eigenvectors are projected onto a number of directions corresponding to the

vertices of a polyhedron and are scaled by the Eigenvalues. The descriptor formed by concatenating these projected Eigenvectors is used when performing action recognition. The HOPC descriptor is claimed to be invariant to changes in viewpoints. In [139], a descriptor called Local Depth Pattern (LDP) is obtained by computing the average depth values in a spatial cell that is constructed from the interest points identified in a colour image. The Comparative Coding Descriptor (CCD) used for action representation in [140] encodes the structural relations of points in space and time. The video is treated as a spatiotemporal volume of depth values and a set of small atomic cuboids extracted from this volume is used to construct a sequence of codes that define the descriptor. The CCD has some invariance to perspective variations and sufficiently depicts the depth information necessary for action recognition.

The occupancy patterns of the 3D spatial point cloud used in the holistic representations are also applicable as local representations. The numbers of points that fall into the cells of a localized spatial grid (Figure 2.3 (c)) are used as features in [88]. These Local Occupancy Pattern (LOP) features describe the appearance in a sub region of the depth image and are useful in characterizing the interactions with objects when an action is performed. A set of features called Random Occupancy Pattern (ROP) is proposed in [141] for recognizing actions. The depth sequence is considered as a 4D spatiotemporal volume in which the pixel values are binary. The ROP features are defined by the sum of the pixel values in a sub-volume. There are a number of sub-volumes with different sizes and at different locations. Since the possible set of sub-volumes is prohibitively large, a random sampling approach is used to efficiently explore the sub-volumes.

The methods that rely exclusively on image based features for action recognition in depth videos are becoming less popular. The estimation of human body poses in real time has become possible with the use of depth images as demonstrated in [14]. The low-level visual features are less important when pose information is available. The locations of the various body joints provide essential information to discriminate between the actions. However, using image based features in conjunction with the pose information may be effective in some recognition scenarios.

The action recognition methods in this thesis are based on the pose information. Hence the above techniques where the features are extracted directly from the depth images are not applicable on this work. A notable exception is in Chapter 5 where a hybrid of the pose information and depth channel is used. The information in the depth image patches is used to characterize the objects a person performing an activity interacts with and some of the feature descriptors discussed above are employed in that chapter.

2.2.2 Skeleton Based Features

A number of approaches for action recognition, including those in this thesis, are motivated by the seminal study [12] of motion perception by Johansson, in which it was demonstrated that actions can be understood just from a small number of landmark joints. A hierarchy of joints connected by bones forms a skeleton and different joint configurations yield different poses. The actions can now be described using a sequence of positions of these joints in the skeleton rather than by the pixel values in an image.

The human body is capable of a wide range of motions and estimating the configuration of the human body from a sequence of monocular images is non-trivial. It is difficult to compensate the loss of depth information that results from the formation of a 2D image. As alluded to in Chapter 1, the variations in appearance, colour, texture and lighting further compound the problem. Despite several years of research [55, 58], pose estimation from visible light images remains largely unsolved [142].

The introduction of depth sensors provided a realistic opportunity to infer the body poses. In particular, it was demonstrated in [14] that pose estimation can be performed in real time if depth images are used. The algorithm proposed in [14] powered the commercially available Kinect sensor, which produces estimates of a skeleton structure that is composed of 20 joints. This algorithm is discussed in Appendix B, but in a nutshell, first a depth image is segmented probabilistically into body parts and then proposals of 3D body joint positions are generated from this intermediate segmented image.

The availability of 3D joint positions aroused considerable interest in the action recognition community and several works were published using the skeleton information. However, recognition is still a challenge even when using body joint positions. The variations within the same action class, similarities in motion patterns between the action classes and noisy skeletons due to sensor errors, occlusions etc. make it difficult to distinguish between the actions robustly. This necessitates further processing of the joint positions to derive alternative feature representations [114].

A simple feature for representing human motion is the pairwise relative position where the difference between the 3D positions of any two joints is used. The intuition behind this feature is that an action can be described in terms of the relations between any two body parts. For example, a “wave” action can be described as “hands” above “shoulder” and “wrists” to the left or right of “elbow”. In [88] the feature for a joint is determined by taking the difference between the position of a joint and all the other joints. The overall feature is determined by enumerating

all the pairwise joints. A similar mechanism is employed in [89] to determine a dynamic skeleton (DS) feature using relative joint positions.

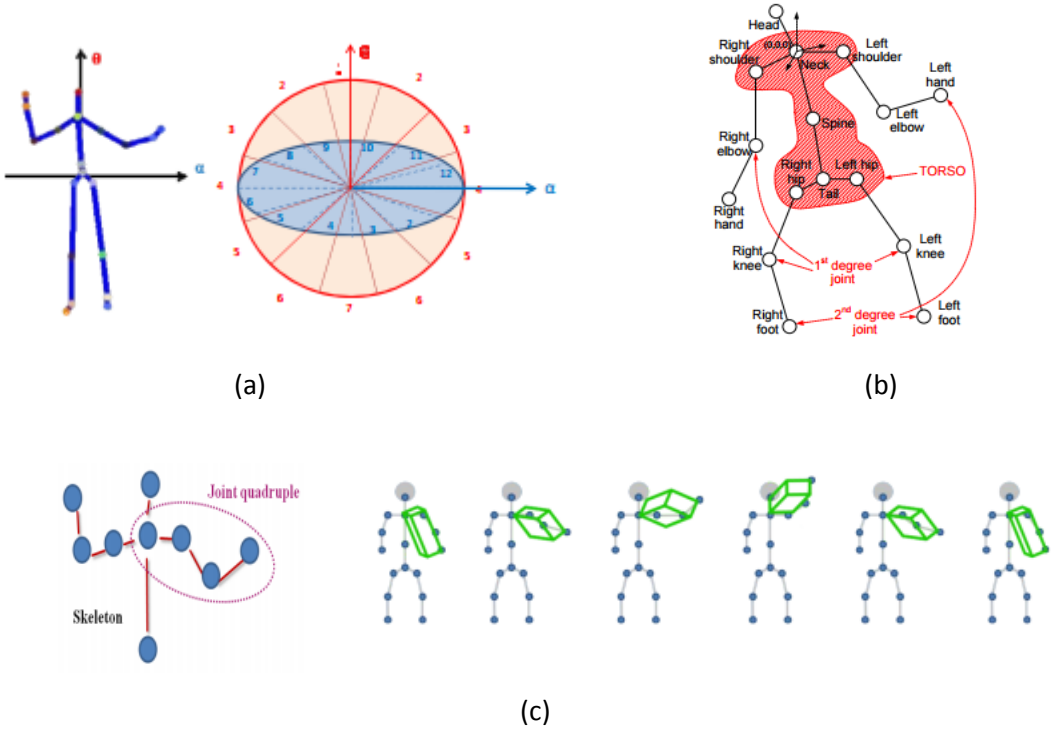


Figure 2.4: Skeleton data features. (a) Spherical coordinate system in [11] that uses the hip centre joints for aligning the coordinates with a person's direction. The angles are divided into equal sized bins to derive a histogram based representation. (b) Joint angles representation used in [149]. (c) A local skeleton descriptor that encodes the relative positions of joint quadruples is used in [156].

In [147], the relative joint positions computed from several video frames are used as features. Apart from the differences between the joints in the current frame, the pairwise differences are computed between the current frame and a preceding frame to capture the motion properties. The pairwise differences are also computed between the current frame and an initial frame that approximates the neutral posture. The combination of all these differences forms a feature representation. Instead of using the difference between two joints, the distance between two joints is used in [148]. The Euclidean distances between every pair of points in the current frame and previous frames are used in the feature representation. The Euclidean distances between all pairs of joints in the current and adjacent frames are also used in [159] to determine the features. This work additionally includes as features the velocity of a joint along the direction defined by two other joints and the velocity of a joint in the direction of the normal vector of the plane spanned by three other joints.

Instead of using the 3D joint positions, some action recognition methods use the joint angles as features. In a kinematic tree representation of the human body [142], a particular joint is selected as the root and the remaining joints are connected to the root in a hierarchical manner. A set of relative joint angles that represent the orientation of the body parts with respect to the parent in the hierarchy provides an alternative representation of the 3D locations of the joints. The 3D Cartesian coordinates representing the joint positions are transformed into 2D spherical angles representing the directions of the body parts. The radial distance is omitted in the representation thus excluding the length of the body parts. This angular skeleton representation provides some invariance to the size of the human and the orientation of the depth sensor.

In [80], the relative azimuth and elevation angles of each joint with respect to its parent in the skeleton hierarchy are used to compute the features. For example, in order to calculate the feature at the left elbow joint, first the sensor coordinate system at this joint is translated such that the origin is at the left shoulder. Then a local spherical coordinate system is constructed in terms of an elevation angle from the XY plane and an azimuth angle from the positive X axis. A spherical coordinate system is also used in [11] to derive view invariant features. The hip centre joint is defined as the centre of the spherical coordinates and the spherical coordinates are aligned with the direction of a person (Figure 2.4 (a)). The angles between the limbs and the angles between limbs and planes spanned by the body parts are used in [151]. The works in [152, 153, 154] also employ joint angles as features for action recognition with [154] using quaternions for representing rotations.

A similar joint angle representation is used in [149] with each joint position represented using a pair of azimuth and elevation angles that specify the joints in a locally defined spherical coordinate system. However, the angles are computed a little differently. The positions corresponding to the joints at neck, shoulder, spine and hips are considered as points of a torso that is a vertically elongated rigid body as shown in Figure 2.4 (b). An orthonormal basis is first obtained from these points and the other joints are represented relative to this basis. The joints adjacent to the torso such as elbows, knees and head are called first-degree joints and are represented relative to the adjacent joint in the torso in a spherical coordinate system derived from the torso frame. The same torso frame is used as a reference to convert the second-degree joints such as the hands and feet at the extremities. A problem with this method is that it may produce inconsistent angles and non-local descriptions for the second-degree joints. This method is improved in [150] by considering rotations of the torso orthonormal basis when constructing the angles for the second-degree joints.

Instead of using the joint positions or the joint angles, some methods propose representations that explicitly model the geometric relationships among the body parts. In the recent work [155],

the 3D geometric relationships between various body parts are described using rigid body transformations. A family of relative 3D geometry-based skeletal representations, referred as R3DG features, is introduced. In [156], a skeleton descriptor that encodes the relative positions of a set of four joints (Figure 2.4 (c)) is proposed. Given a quadruple of nearby joints, a coordinate system such that one of the joint position is the origin and one of them is mapped to $[1,1,1]$ is considered. A similarity transformation is applied on the remaining two joints with the quadruple encoded by six parameters that are well distributed in a 6D space.

Instead of directly using the joint positions or the joint angles as features, some methods apply further processing on these to derive sophisticated feature descriptors. For example, in [11] the azimuth and elevation angles of the hip centre joint are divided into equal sized bins as shown in Figure 2.4 (a) and the angles corresponding to the other joints are probabilistically assigned to the bins. The final descriptor called Histogram of Oriented Joints 3D (HOJ3D) is computed from the histogram bins. In [157], a histogram of the directions between joints in the current frame and adjacent frames is used. The resulting descriptor called Histogram of Oriented Displacements (HOD) represents the motion of an object based on the distance it moves. In [160], the spherical coordinates of the joint positions are quantized into a histogram with an action modelled as a set of histograms. The number of bins is different for the azimuth and elevation angles. The covariance matrix of the joint positions is used to derive a Covariance of 3D Joints (Cov3DJ) descriptor in [158].

Some methods hypothesize that not all the joints contain useful information for action recognition and a feature selection step is introduced to identify a subset of joints that are more helpful in discriminating the actions. This may be done manually using some a-priori knowledge on the data. For example in [152], a specific set of 8 joints are identified to recognize activities related to falling event. The joints on the limbs are excluded since they are perceived to introduce more noise than useful information required to decide whether a person has fallen. Similarly in [11], 12 joints are pre-selected manually before constructing the features. The excluded joints either contain redundant information or do not contribute to distinguishing the motions.

Instead of manual selection, in some methods the joints are selected automatically when constructing the features. For example, in [153] the most informative joints in a time window are identified based on the relative informativeness of all the joints in that time window. The joints that have high variance of their angular changes are defined as the most informative joints. In [160], a pose feature is defined as a weighted sum of all the joint features with the weights learned using a Partial Least Squares (PLS) method. A Support Vector Machine (SVM) model is trained in [88] to determine how discriminative the features extracted from a joint are.

This information is used to find the features for a subset of the joints. Even evolutionary computation methods such as Genetic Algorithm (GA) have been used to identify a subset of joints in the skeleton hierarchy that provides a good representation of the motion patterns [161].

Skeleton based features are used in the action recognition methods in this thesis. The simple pairwise relative positions of the joints are mainly used as features. More sophisticated descriptors are avoided. The temporal dynamics of the joints, as modelled by the classification algorithm, are relied upon to distinguish the actions. The classification algorithms proposed in this thesis are generally agnostic to the features and are designed to benefit from other types of sequential data.

2.3 Classification

While the previous section discussed the methods used to extract features from the video, this section describes the methods used to match the features. Once the features are available, the action recognition problem becomes a supervised classification problem. A variety of algorithms in statistical machine learning literature can be used to match the features. The classification algorithms used for action recognition can be divided broadly into two types – static and dynamic. In the static classifiers, the temporal domain is not considered while in the dynamic classifiers the variations of the features in time are explicitly modelled. Figure 2.5 lists the classification algorithm types discussed in this review.

2.3.1 Dimension Reduction

Before using the classification algorithms, many action recognition methods apply a dimension reduction technique. The features extracted from the video frequently contain redundant information, may be sparse vectors and are sometimes noisy. They are often in a very high dimensional space, for which a large number of training examples is required. Using a compressed form of the features hugely benefits the classification algorithm. Hence the features are subjected to a dimension reduction technique to obtain a robust and compact representation.

The Principal Component Analysis (PCA) [16] is a commonly used linear dimension reduction method which projects high dimensional features to a lower dimensional feature space. It is employed in various works such as [123, 136, 147] to reduce the number of features. Linear Discriminant Analysis (LDA) [35], which preserves the class discriminatory information while reducing the dimensions, is used in [11] and [121]. In [88], a short Fourier transform is applied to the feature vector at a time instant and the low frequency Fourier coefficients are used as

features. By discarding the high frequency Fourier coefficients the features are made robust to noise.

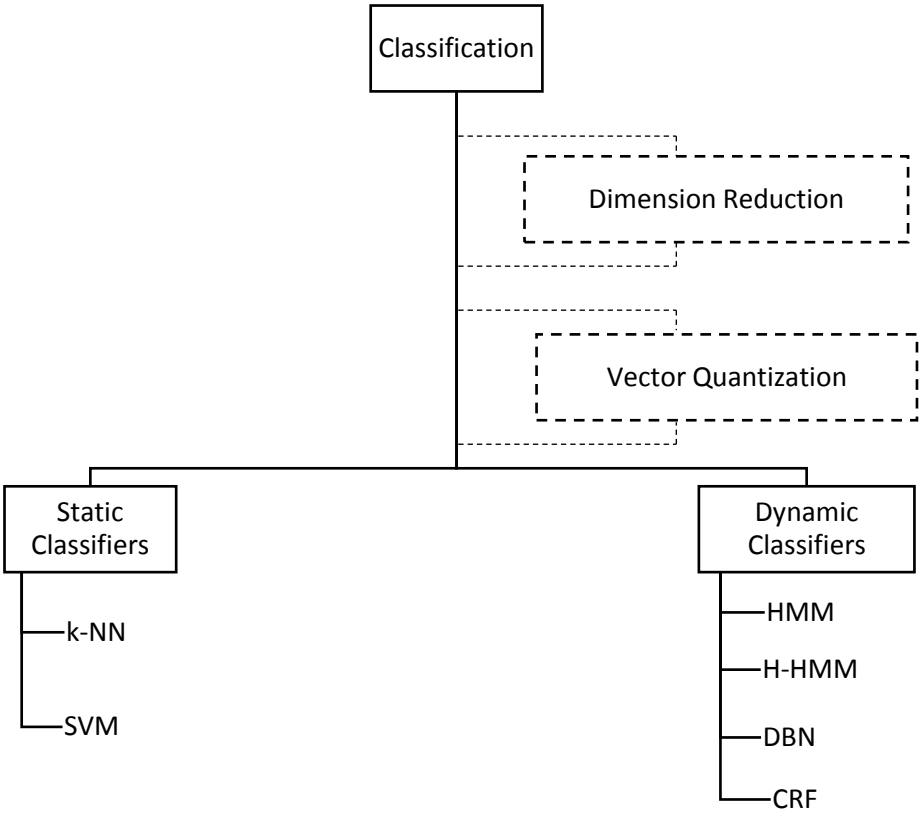


Figure 2.5: Classification algorithm types. The classification algorithms used for action recognition are shown in a schematic representation. Some methods optionally include a dimension reduction step and use code words. See text for more details.

Non-linear dimension reduction methods have also been explored for action recognition. These techniques, known as manifold learning, identify the underlying low dimensional manifold in which the high dimensional features are embedded in such a way that the properties of the original feature space are preserved. The assumption here is that by the nature of the human movements, the actions do not span the entire feature space and hence they must lie on a low dimensional manifold. The features obtained as a result of manifold learning are used by the classifier.

In [162], a low dimensional embedding of the actions is learnt from the high dimensional trajectories of the joints using a manifold functional variant of PCA. In [79], the trajectories described by the 3D joint positions are embedded in a Riemannian manifold. This formulation takes advantage of the Riemannian geometry in the resulting shape space when comparing the similarities between the shapes of different trajectories. The intuition behind this approach is that the feature descriptors used in vision applications typically lie on a curved space due to the

geometric nature of their definitions. Another recent work [163] uses an autoregressive moving average (ARMA) model, which is parameterized using an observability matrix, to represent the trajectory of the joint positions. The subspace spanned by the columns of the observability matrix corresponds to a point on a Grassmann manifold.

2.3.2 Static Classifiers

When static classifiers are used for action recognition, it is assumed that the feature representation already captures the information in the temporal dimension. Typically, the entire video is summarized by a single feature vector. This may result in feature vectors of different sizes because the number of frames in a video may vary between the actions.

Many methods such as [131, 134, 152] etc. use a bag-of-features or bag-of-words model in which the features are represented using a fixed size histogram. Typically a clustering algorithm such as K-means is applied to the feature vectors to learn a set of centroid vectors called the code words. Each feature vector is mapped to a code word by the index of its closest centroid. A set of feature vectors can now be represented by the histogram of the code words. This method of quantizing the feature vector is often employed to produce a global feature representation of the entire action sequence when static classifiers are used. The loss of temporal structure with this quantization does not matter since the static classifiers do not model the time dimension anyway.

The k-Nearest Neighbour (k-NN) classifier compares the distance between the feature vector of a test video and the feature vectors of the videos in the training examples to determine the class label. The label most common among the k closest training examples in the feature space is chosen. The k-NN classifier has been used for classifying actions in many methods such as [79, 115, 117, 119, 125, 127, 129, 147]. Different distance measures have been used. For example, the Euclidean distance is used in [117], Mahalanobis distance is used in [115], the Chamfer distance is used in [129], the geodesic distance is used in [79] and a video to class distance based on naïve Bayes is used in [147]. It is also possible to use a distance measure in k-NN that compares two feature sequences, possibly of different lengths. For example, in [154], the Dynamic Time Warping (DTW) algorithm is used as a distance measure.

The k-NN classifier scales well with the number of classes and also avoids the over fitting problem. It also does not generally need a training procedure. However, a stored database of previously seen actions is necessary with this classifier. If there are a number of training examples, comparisons become computationally expensive. Hence an adequately representative set of training examples must be identified with this classification method.

One of the most popular static classifiers for action recognition is the Support Vector Machine (SVM) [65]. This discriminative classifier, which learns a hyperplane in the feature space that separates the classes, has been used by several methods such as [122, 124, 131, 132, 134, 137, 139, 140, 141, 152 157]. Both linear SVMs (e.g. [157]) and non-linear SVMs, (e.g. [134]) in which the inputs are mapped to a high dimensional feature space using kernel functions such as chi-squared kernel, have been used. The probabilistic variant of the SVM, the Relevant Vector Machine (RVM) has also been used for action recognition in [129].

The use of SVM as off-the-shelf classifier by many methods is unsurprising since it has produced stellar results for many other computer vision problems such as object recognition and human detection [77]. However, since SVMs cannot model temporal data the classifier performance depends on how well the features capture the time dimension.

2.3.3 Dynamic Classifiers

Unlike static classifiers that consider a single data point, the dynamic classifiers analyse a sequence of data points. The temporal dynamics are explicitly modelled in the dynamic classifier and the order of the features are considered when matching them. Sequential patterns of data are observed in many other fields such as speech recognition (e.g. phoneme sequences), genomics (e.g. DNA sequences) and natural language processing (e.g. sentences). There is a rich body of literature on sequential pattern recognition. This survey focuses on those methods that use a state-space graphical model for action recognition. See [4, 107, 114] for other approaches.

In most state-space graphical models, discrete valued state variables are encoded as graph nodes. The edges between the states and the observed features characterize the model. The most well-known model in this family is the Hidden Markov Model (HMM) [32]. The sequence of states in an HMM follow a Markov assumption i.e. each state is conditioned only on the previous state and not on the entire previous history. Together with the additional assumption that the observations are independent when conditioned on the current state, the HMM becomes a tractable model.

The HMMs are very popular in the speech recognition [51] literature and their use in action recognition can be dated as far back as 1992 when Yamato et al. [104] used them to classify actions in tennis such as ‘smash’, ‘serve’, ‘backhand stroke’ etc. Each action is represented by a separate HMM and the transition and observation parameters are learned during training. The classification of an unseen action is performed by comparing the observation likelihood of all the trained HMMs. The HMMs have been used in several action recognition works such as [11] and [121] since their introduction in [104].

The features in an HMM are considered frame by frame and hence sequences can be of any length. Unlike the static classifiers, a vector quantization step that produces fixed size features is not necessary. The HMMs also allow the observed features to be discrete or continuous. Using a Gaussian mixture for the continuous densities is common while a quantized set of symbols using a discrete distribution is also possible.

The HMMs are generative models in which a joint distribution over the features and class labels is modelled. The HMM parameters learned during training are intended to explain the examples corresponding to the appropriate class label. Such a training procedure does not necessarily guarantee good results in classification problems. Many works [164, 165, 166] pursue alternative training criteria to ensure that the learned HMM parameters produce good classification results. For example, instead of the traditional Maximum Likelihood Estimation (MLE) method where the training criterion is based on the likelihood of observing the examples, a Minimum Classification Error (MCE) method that minimizes the empirical classification error rate on the training examples or the Maximum Mutual Information Estimate (MMIE) has been used [172]. In a recent work in [175], Fisher kernels are employed to discriminatively learn the generative HMM parameters. The class similarity distances between the likelihood gradients for same classes are minimized while those for other classes are maximized.

The HMMs are part of a larger class of models called Dynamic Bayesian Networks (DBNs). There are other models that generalize the HMM at increased costs for inference and learning. For example, the Hierarchical HMM (H-HMM) [81] extends the canonical HMM by introducing a hierarchy of states. Each state in the H-HMM can emit another sub-HMM. In [167] the H-HMM is applied to recognize activities that are four levels deep in the hierarchy. When modelling interactions between two persons, it may be necessary to express the temporal evolution of the states corresponding to these persons individually and yet also tie the states together. The coupled HMMs provide such a construct. They are used in [168] for activity recognition. The method in [168] relaxes the Markov assumption and introduces explicit state duration models producing a coupled semi-Markov model. An event driven multi-level DBN is proposed in [169] in order to model the interactions between groups of people. The scenario is a group level meeting in which there are top level events such as ‘presentation’, ‘discussion’ and ‘break’ and sub-events such as ‘lecturing’ and ‘Q&A’ corresponding to the ‘presentation’ event.

Probabilistic topic models such as Latent Dirichlet Allocation (LDA) can be used to automatically discover the dominant themes in data. By including temporal information, the sequential nature of the activity patterns can be discovered in a better manner, as in [146]. The HMMs have been used together with LDA. In [145], the HMM is combined with LDA to produce a hierarchical

model called Markov Clustering Topic Model that allows simple actions to be combined into complex global behaviours.

Conditional Random Field (CRF) [33] based models have also been explored for action recognition. While the HMMs, H-HMMs and the DBNs are directed graphical models, the CRF is an undirected graphical model that is discriminative by nature. It models the classification rules directly and is a popular method for classifying sequential data.

In [53], the spatiotemporal relations between human poses and objects are modelled using a CRF in order to detect past activities and predict future activities. Since there is an inherent ambiguity in the temporal segmentation of the sub-activities that constitute an activity, a range of possible graph structures are investigated using dynamic programming techniques. In [170], the Hidden CRF (HCRF) [95] is applied to recognize gestures. It is not necessary to segment the gesture substructures because of the use of hidden states. In [171], a modified HCRF is used to categorize actions. The initial parameters for an HCRF must often be carefully selected. To overcome this problem, the hidden states are learnt using an HMM.

The classification algorithms proposed in this thesis consider sequences of features and represent actions by graphical models composed of a set of states. Hence they are closely related to the dynamic classifiers discussed in this section. However, there is a key difference from the state-space models used in the above works for action classification. The HMM, H-HMM and the HCRF models above assume that the number of states is fixed in advance. This constraint is relaxed in the models proposed in this thesis, by using a nonparametric extension that allows the number of states to be learned automatically from the data.

2.4 Bayesian Nonparametric methods

Many methods in machine learning build a model with a fixed number of parameters where the parameters can be thought of as a convenient summary of the training data. Consider as an example a solution to the clustering problem which uses a mixture of Gaussians to define a density function over the data. The parameters are the mean and covariance of a Gaussian for each of the mixture component. In this parametric model, the number of mixture components (i.e. the clusters) is assumed to be known in advance and hence there is a fixed finite set of parameters.

The nonparametric models allow the number of parameters to grow with the data. In the above clustering scenario, a nonparametric solution does not need the number of clusters to be specified a-priori. It is assumed that there is an unbounded number of mixture components (clusters), with only a finite number of them actually used to model the data.

Bayesian methods represent uncertainty in the model parameters in terms of probability distributions. The applicability of Bayesian data analysis increased widely with the availability of posterior inference procedures based on simulations [36, 38]. The Bayesian nonparametric methods extend the methodology of prior and posterior distributions to a model with an unbounded number of parameters. In order to produce a tractable model, these methods use appropriate priors to limit the number of parameters required to model the data.

The probability distributions on an infinite dimensional space are called stochastic processes. Gaussian Process, Dirichlet Process, Beta Process and Pitman-Yor process are some examples of such stochastic processes. The most popular one by far in the machine learning literature is the Dirichlet Process [19]. It has been used for a wide variety of problems including clustering, regression, density estimation, latent feature modelling, sequential pattern recognition and modelling random effects distributions [45], to name a few. Of particular interest is the Hierarchical Dirichlet Process (HDP) [44] model which couples multiple Dirichlet Processes within a hierarchical framework. It models data which comes in multiple groups and captures both the similarities and differences across the data points within these groups. The classification algorithms discussed in this thesis uses the HDP as priors to construct nonparametric models and hence the review here focuses on HDP. The survey in [173] reviews nonparametric Bayesian inference and the fairly recent survey in [174] discusses other priors used to induce dependency between random measures.

In [68], the HDP was applied to an object recognition problem. A family of hierarchical models is defined based on the HDP for a visual scene, with a scene being made up of objects and the objects comprised of parts. The parts are shared between the different object categories. The number of parts underlying the object categories and the number of objects in a scene are both learnt automatically from the data. The HDP is augmented with transformation variables that describe the locations of the objects in an image. In the recent work in [177], the HDP is used to learn admixture models of image patches similar to the topic models used for text documents. It explores the co-occurrence of image features at different hierarchical levels. The HDP model captures the similarities within image patches using image specific mixture component distributions. This adapts the topic proportions to each image with smooth patches favoured for some images and textured patches for others. The use of HDP prior allows learning the number of topics from data. The learning algorithm in [177] uses variational inference [16] rather than the traditional simulation based inference. In [133], the Dirichlet Process is used to automatically discover recurrent temporal patterns in time series. Activity patterns such as *car passing*, *pedestrian crossing* are identified in an unsupervised manner.

The HDP can be used as a prior over the HMM discussed in the previous section to derive the HDP-HMM, which is applicable for modelling sequential data. The number of states in an HDP-HMM is unbounded with new states being instantiated when the data is not adequately explained by the current set of states. Thus the cardinality of the states adapts to the data. In [76] the HDP-HMM was used to segment an audio recording of a meeting into different temporal segments corresponding to individual speakers. Prior assumptions on the number of speakers in the meeting are avoided by using a nonparametric model. The HDP-HMM in this work introduces a new variable to encourage slower transition dynamics between the states. The bias towards smoothly varying state dynamics provides better segmentation for speech data. In [178], the HDP-HMM is used to detect activities that occur rarely and have not been anticipated. An ensemble approach is used here in which first a set of HDP-HMM based classifiers is used to learn a decision boundary around the normal data in the feature space. This boundary is used to classify activities as normal or abnormal via one-class SVM. A learning approach for jointly segmenting and recognizing sequential data is proposed in [179]. Unlike many methods in which the entire data set is available during training, the model handles streaming data by receiving them in mini batches and segmenting and recognizing them on the fly. The sticky HDP-HMM proposed in [76] is used in this method as well. The nonparametric nature of the model allows an unbounded number of classes. The spatiotemporal dependencies in complex dynamic scenes is automatically learnt using a HDP-HMM in [143]. The model captures the state of the scene as a whole and explains how the state changes over time and how likely the changes are.

The HDP has been used as a prior over other HMM variants as well. The Switching Linear Dynamical Systems (SLDS) can be viewed as an extension of HMMs in which each HMM state is associated with a linear dynamical process. They capture complex temporal dependencies that exhibit structural changes over time. The HDP prior to the SLDS in [41] produces a model in which the number of dynamical modes is not fixed in advance while allowing for returns to previously exhibited dynamical behaviours. A mixture of SLDS is used in [180] to discover actions that describe low-level motion dynamics and behaviours that are composed from actions to capture high-level temporal dynamics. By using the HDP prior over SLDS, the number of actions and the number of behaviours are learnt from data. This unsupervised method segments tracks into sequences of common actions and clusters the actions into behaviour patterns of people.

The advantages of semi-Markovian models and nonparametric models are combined in [92]. The generative process of the HMM is augmented with random duration times and each state's duration is given an explicit distribution. The HDP prior is applied to this Hidden Semi Markov Model (HSMM) to produce a model in which the strict Markovian constraints of the HMM is relaxed and the number of hidden states is inferred from data. The HDP-HSMM structure is

applied to an unsupervised power signal disaggregation problem. The idea of explicitly parameterizing and controlling the dwell-time for the HMM states is also explored in [176] in a nonparametric setting. In some applications, HMMs may have restricted topologies such as precluding all states that are already visited. The infinite structured hidden semi-Markov model (ISHSMM) in this work allows building nonparametric models for the HMMs in which the states are never re-visited and where each state is imbued with an explicit duration distribution.

The infinite factorial HMM (IFHMM) in [181] introduces a probability distribution over a potentially infinite number of binary Markov chains. The hidden states are represented in a factored form that allows information from the past to be propagated in a distributed manner through a set of parallel Markov chains. The distribution over the Markov chains is defined using the nonparametric Bayesian factor model called Indian Buffet Process (IBP). In [182] this IFHMM is extended to allow for an unbounded number of states in addition to the unbounded number of non-binary Markov chains. This model is applied to the Multiple-Input Multiple-Output (MIMO) communication systems to infer both the number of transmitters and the number of transmitted symbols based on the data. A nonparametric generalization of the hierarchical HMM [81] is presented in [84]. It allows an unbounded number of hierarchical levels instead of requiring the specification of the fixed hierarchy depth. The dependency structure between the state variables is much simplified when compared with the canonical hierarchical HMM for tractability. Additionally, cardinality of the state variables is fixed in this model.

Unlike the above works which use nonparametric HMMs in an unsupervised setting, the HDP based solutions proposed in this thesis are intended for supervised classification. When using HDP-HMM for classification, the traditional approach is to train a classifier for each class and use the class conditional distributions to determine the classification decision boundaries. As mentioned in Section 2.3.3, this training procedure does not provide the best decision boundaries in terms of minimizing the classification error rates. The HDP-HMM proposed in Chapter 4 learns the model parameters in a discriminative manner. Further, it allows sharing information across the action classes and considers both positive and negative examples during training. It thereby combines the advantages of a generative model and discriminative classification.

The nonparametric model proposed in Chapter 5 enables supervised classification by using logistic regression on the states learnt using a hierarchical HMM with HDP priors. This idea of using a linear model with a generative process to capture the relationship between groups of observations and their associated labels has been explored in the natural language processing literature. Relevant examples are the supervised Latent Dirichlet Allocation [183] and its nonparametric extension the supervised HDP [184]. These techniques were mainly used in topic

models for labelling text documents. The work in this thesis is different, in that it includes an H-HMM, which considers a factored hierarchical nature of observations, and an application to a vision problem involving sequence classification. In particular, the inference procedure in action recognition considers the correlation between the observations in time while in document labelling tasks this is not usually needed.

A nonparametric HCRF with HDP prior is used in Chapter 6. While there have been many works in the literature using HDP priors for directed graphical models, the nonparametric extensions for undirected graphical models is a new research area. The works in [185] and [186] propose HCRFs with an unbounded number of states. The simulation based inference method in [185] is not applicable for continuous observation features and the variational inference method in [186] has non-negative constraints on the observation features. In contrast, the model in Chapter 6 is well suited for continuous observations and does not enforce any constraints on the features or HRCF parameter weights. Perhaps the most important difference is that a fully Bayesian treatment of the model is made. The posterior distribution for the HCRF parameters is estimated.

2.5 Summary

The action recognition pipeline contains the following stages – image acquisition, feature extraction and classification. The various image based features such as silhouettes, contours, interest points and feature descriptors were discussed. With the availability of skeleton joints obtained from depth images, recent works use joint information rather than low-level image features. The skeleton joints offer a convenient and a fairly reliable mechanism to characterize actions. The methods in this thesis mainly use simple pairwise relative joint positions as features unlike the sophisticated descriptors used in many works.

The static classifiers rely on the feature descriptors to capture temporal correlations of the features, in addition to the frame specific features for classification. While static classifiers such as SVM have produced good results with image classification, the dynamic methods that model temporal evolution are intuitive and compelling for sequential data. When dealing with higher order event structures such as complex activities, the use of dynamic classifiers is inevitable. The state-space graphical models are a natural choice when using dynamic classifiers, with the well-studied HMM and its variants having been used in many action recognition problems.

The Bayesian nonparametric extensions to HMM and its variants preclude the need to fix the number of states. Most of these models use the HDP as a nonparametric prior. While this HDP prior is used in the nonparametric models defined in in this thesis, the key difference from other works is the applicability of these models for supervised classification problems. The

discriminative manner in which the HDP-HMM parameters are learnt and the integration of a linear model with hierarchical HMM makes the models in this thesis suitable for classification. The fully Bayesian treatment of the nonparametric HCRF also distinguishes the work in this thesis from others.

The survey in this chapter is by no means exhaustive. In addition to the popular methods discussed here, there are Deep Learning approaches based on Convolutional Neural Networks (CNNs) [187] that avoid explicitly engineering the features and learn complex features automatically from data. There are also methods that focus on the recognition speed rather than the recognition accuracy in order to scale up to large size problems. However, the survey does cover important methods that are related to this thesis and highlights key differences.

3. Background

This chapter provides the technical background essential to describe the thesis contributions. The two discrete state Markov models used in this thesis, namely the Hidden Markov Model and the Conditional Random Field, are introduced in Section 3.1 and Section 3.2 respectively. Background material necessary to develop nonparametric models is provided in Section 3.3. This includes the stochastic Dirichlet process and its hierarchical extension.

Further technical background can be found in the Appendix. The techniques used to construct depth images using active 3D sensing is discussed in Appendix A. A brief overview of the mechanism to estimate 3D joint positions from a depth image is provided in Appendix B. Appendix C provides an introduction to Bayesian analysis and Appendix D provides an overview of the graphical models. The approximate inference techniques that are necessary to compute posterior distributions are reviewed in Appendix E.

3.1 Hidden Markov Model

The Hidden Markov Model (HMM) [32] is a popular model for representing sequential data. It is a directed graphical model and a special case of a Bayesian network. HMMs are widely used in many fields such as speech recognition [51], biological sequence analysis [48], econometrics [50] and natural language processing [49].

In order to abstract the input characteristics and produce a rich set of models, the HMM includes a discrete variable Z_t corresponding to each input value x_t. This variable concisely summarizes the attributes of an input observation at time t. The variable is usually latent and is referred as a *hidden state* variable. Let z_t denote the value assigned to Z_t and $\boldsymbol{z} = \{z_t\}_{t=1}^{T}$. There is a finite number K of hidden states and $z_t \in \{1, 2, \dots K\}$. The Markov assumption is now applied to hidden states instead of the input observations.

$$z_{t+1} \perp z_{1:t-1} \mid z_t \tag{3.1}$$

Further, the model assumes that conditioned on the hidden states the observations, which may be discrete or continuous, are independent. Let $\backslash t$ denote all variables except the variable at time instant t. Then,

$$x_t \perp x_{\backslash t}, z_{\backslash t} \mid z_t \tag{3.2}$$

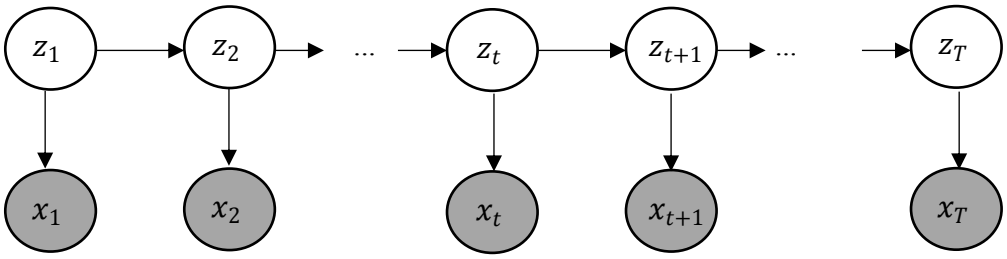


Figure 3.1: HMM representation. The hidden states $\boldsymbol{z}$ have the Markov property. The observations $\boldsymbol{x}$ are independent, conditioned on the hidden states.

In order to express the relationship between the latent variables $\boldsymbol{Z}$ and the input variables $\boldsymbol{X}$, distributions over the combined set of variables $\boldsymbol{X} \cup \boldsymbol{Z}$ must now be built. Following the HMM representation shown in Figure 3.1, the joint density function of the combined set of variables factorizes as[1]:

$$p(x_{1:T}, z_{1:T}) = \prod_{t=1}^{T} p(z_t|z_{t-1})p(x_t|z_t) \tag{3.3}$$

An HMM is defined by the state transition distribution $p(z_t|z_{t-1})$ and the observation distribution $p(x_t|z_t)$. The state transition distribution is specified by the $(K+1) \times K$ matrix π defined by:

$$p(z_t = k \mid z_{t-1} = j) = \pi_{j,k} \qquad \begin{matrix} j = 0 \ldots . K \\ k = 1 \ldots . K \end{matrix} \tag{3.4}$$

The π_0 row contains the initial probability of being in a state k with $p(z_1 = k) = \pi_{0,k}$. The matrix π is called the transition matrix. It follows from the definition of (3.4) of π that $0 \leq \pi_{0,k} \leq 1$ and $\sum_{k=1}^{K} \pi_{j,k}$ for $j = 0 \ldots K$. The conditional distribution of the input value x_t is defined by:

$$p(x_t|z_t = k) \sim F(\theta_k) \tag{3.5}$$

Here θ_k are the parameters of the family F of distributions. The model is tractable for a wide range of distribution families. It is common to use a member of the exponential family for F. Let $\theta = (\theta_1 \ldots \theta_K)$. The set of parameters that govern the HMM model is given by:

$$\lambda = \{\pi, \theta\} \tag{3.6}$$

[1] When $t = 1$, $p(z_t)$ is simply $p(z_1)$

3.1.1 Inference

The marginal distributions $p(z_t|x_{1:t})$ and $p(z_t|x_{1:T}), T > t$ are of particular interest when performing inference in an HMM. The former, referred as filtering, is used to estimate the state z_t given all observations at times up to and including t. The latter, referred as smoothing, is used to estimate the present state conditioned on the past and future observations.

The message passing technique outlined in Appendix D.3 can be used to compute the marginal distribution z_t in both cases. Note that in the case of an HMM, there are additional observation nodes that are conditioned upon during inference. The messages in equations (D.17) and (D.18) are updated to include the conditional distribution of the observations and are now written as:

$$m_{t-1,t}(z_t) = \sum_{z_{t-1}} p(x_{t-1}|z_{t-1})p(z_t|z_{t-1})m_{t-2,t-1}(z_{t-1}) \tag{3.7}$$

$$m_{t+1,t}(z_t) = \sum_{z_{t+1}} p(x_{t+1}|z_{t+1})p(z_{t+1}|z_t)m_{t+2,t+1}(z_{t+1}) \tag{3.8}$$

The marginals can then be computed from the messages as follows:

$$p(z_t|x_{1:t}) \propto p(x_t|z_t)m_{t-1,t}(z_t) \tag{3.9}$$

$$p(z_t|x_{1:T}) \propto p(x_t|z_t)m_{t-1,t}(z_t)m_{t+1,t}(z_t) \tag{3.10}$$

The above message based representation of the marginals can also be derived using the forward-backward algorithm [32]. This classical algorithm defines two terms $\alpha_t(z_t)$ and $\beta_t(z_t)$. The former term is a *forward message* that represents the joint probability of a state z_t and the observations up to time t. The latter term is a *backward message* that represents the conditional probability of all future observations from time $t+1$ to T given the state as z_t. These terms relate to the messages in (3.7) and (3.8) as follows:

$$\alpha_t(z_t) = p(x_1, \dots, x_t, z_t) = p(x_t|z_t)m_{t-1,t}(z_t) \tag{3.11}$$

$$\beta_t(z_t) = p(x_{t+1}, \dots, x_T|z_t) = m_{t+1,t}(z_t) \tag{3.12}$$

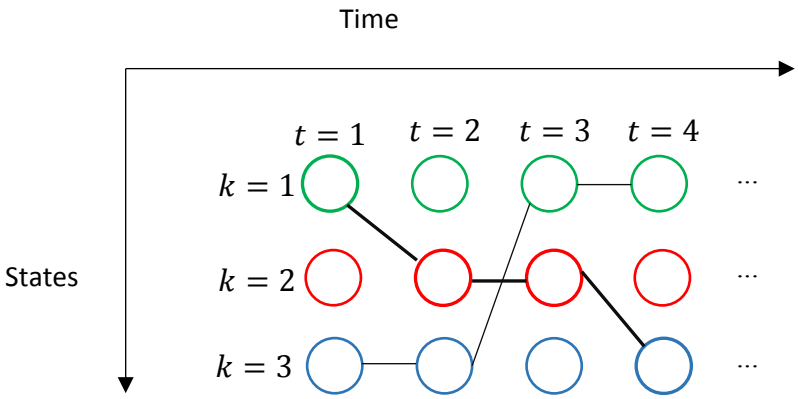


Figure 3.2: Viterbi decoding. Each circle represents one of the possible states at a time instant and the lines denote a possible path for the state sequence. The Viterbi algorithm determines the most probable sequence of states (1, 2, 2, 3 here).

In some inference situations, it is useful to find the most probable sequence of hidden states for a given observation sequence (see Figure 3.2). Maximizing the marginals $p(z_t|x_{1:T})$ individually at each node may not yield the most likely state sequence and may even produce an infeasible sequence. In detail, we wish to find:

$$\hat{z} = \underset{z_1..z_T}{\operatorname{argmax}} \prod_{t=1}^{T} p(z_t|z_{t-1})p(x_t|z_t) \tag{3.13}$$

The same technique used to distribute the summations efficiently for computing the marginals can be applied here, with the summations now replaced by maximization. Let us define a new term $\delta_t(z_t)$ such that

$$\delta_t(z_t) = \max_{z_1..z_t} p(x_t|z_t)m'_{t-1,t}(z_t) \tag{3.14}$$

$$m'_{t-1,t}(z_t) = \max_{z_1..z_{t-1}} p(x_{t-1}|z_{t-1})p(z_t|z_{t-1})m'_{t-2,t-1}(z_{t-1}) \tag{3.15}$$

The maximizing assignment can then be found from the δ terms. This yields the Viterbi algorithm that efficiently finds the most likely state sequence with a time complexity that grows linearly with the number of observations.

3.2 Conditional Random Fields

The Conditional Random Field (CRF) [33] is a probabilistic method that combines the advantages of discriminative classification techniques with graphical modelling. It is widely used for labelling sequential data. It is an undirected graphical model belonging to the family of Markov networks.

CRFs have been applied successfully in computational biology [52], computer vision [53] and natural language processing [54].

In a CRF, there is a discrete random variable Y_t at each time instant t. Let y_t be the value assigned to Y_t. Given an input sequence $\boldsymbol{x}$, the output sequence $\boldsymbol{y} = \{y_t\}_{t=1}^{T}$ is predicted. The outputs are usually class labels. In a classic example, $\boldsymbol{x}$ is a sequence of words and $\boldsymbol{y}$ is the sequence of part-of-speech labels for each word. There is a finite number K of the labels and $y_t \in \{1, 2, \dots K\}$. The Markov assumption is applied to the outputs in order to simplify the graph structure. The resulting model is referred as a Linear Chain CRF.

In a linear chain CRF, the relationship between the combined set of input and output variables $\boldsymbol{X} \cup \boldsymbol{Y}$ is given by a conditional distribution. Let ψ denote a potential function with values in $\mathbb{R}$. The conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$ factorizes in linear chain CRF as follows:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{\mathcal{Z}(\boldsymbol{x})} \prod_{t=1}^{T} \psi_t(y_t, y_{t-1}, t, \boldsymbol{x}) \tag{3.16}$$

Here $\mathcal{Z}(\boldsymbol{x})$ is normalization factor over all output sequences given an input sequence $\boldsymbol{x}$.

$$\mathcal{Z}(\boldsymbol{x}) = \sum_{\boldsymbol{y}} \prod_{t=1}^{T} \psi_t(y_t, y_{t-1}, t, \boldsymbol{x}) \tag{3.17}$$

The conditional dependency of each output on the inputs is defined in CRFs through a set of real valued functions called the *feature functions*. A feature function φ is defined as follows:

$$\varphi: \boldsymbol{y}, \boldsymbol{x}, t \rightarrow \mathbb{R} \tag{3.18}$$

It can be understood as a feature on the input sequence that determines the likelihood of an output value at a time instant. It is not required to have a probabilistic interpretation for the range of a feature function. It is common to have a number of feature functions at each time instant and they may be nonzero only for a particular output. Consider for example that the input is a sequence of words and the outputs are the category of each word such as Name, Location etc. A feature function may be defined to have a value 1 if and only if the output at y_t is a Location and the input at x_t appear in a list of country names. The exact form of the feature functions is problem specific.

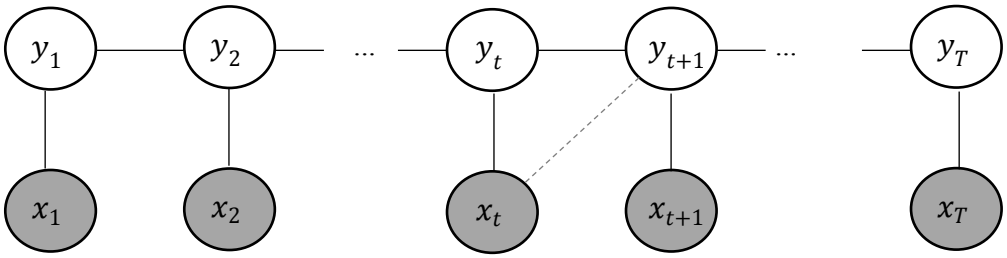


Figure 3.3: Linear Chain CRF. The outputs follow a Markov assumption. The dashed line edge between nodes y_{t+1} and x_t illustrates an output y_{t+1} depending on the input observations x_t and x_{t+1}.

The CRF model is parameterized by a set of real-valued weights, one for each feature function. Let θ_l be the weight corresponding to a feature function φ_l. The potential function ψ_t at time instant t is defined as follows in a linear chain CRF:

$$\psi_t(y_t, y_{t-1}, t, \boldsymbol{x}) = \exp\left\{\sum_{l=1}^{L} \theta_l \varphi_l(y_t, y_{t-1}, t, \boldsymbol{x})\right\} \tag{3.19}$$

The main advantage of the CRF is that it does not make any conditional independence assumptions among the input observations and it can model interdependent features. Hence it is better suited to cases in which the features overlap. This is evident in (3.19), where the feature function accepts the entire set of input variables $\boldsymbol{x}$ and has the flexibility to examine all these input variables. Figure 3.3 depicts the graphical model of a linear chain CRF.

The message passing technique derived for the HMM can be used unchanged for the linear chain CRF. The only difference is in the interpretation. Instead of the conditional distributions used in the HMM, the potential function as defined in (3.19) is used in the linear chain CRF. The messages in (3.7) and (3.8) are now written as:

$$m_{t-1,t}(y_t) = \sum_{y_{t-1}} \psi_t(y_t, y_{t-1}, t, \boldsymbol{x}) m_{t-2,t-1}(y_{t-1}), \tag{3.20}$$

$$m_{t+1,t}(y_t) = \sum_{y_{t+1}} \psi_t(y_{t+1}, y_t, t, \boldsymbol{x}) m_{t+2,t+1}(y_{t+1}). \tag{3.21}$$

3.3 Nonparametric Models

The statistical models discussed above use a fixed number of parameters and the parameter space has a finite dimension. In contrast to these parametric models, a *nonparametric model* uses an unbounded number of parameters and the parameter space is infinite dimensional.

Even though there are an unbounded number of parameters, only a finite subset of these parameters are used to explain a given dataset. The number of parameters may grow (or shrink) depending on the data.

Several well-known problems benefit from using a nonparametric method. Consider the traditional finite mixture modelling approach to clustering in which the number of clusters (i.e. mixtures) are specified in advance. In a nonparametric model, the number of clusters needed to model the data is estimated from the observed data and new clusters are instantiated as new data points are observed. If the complexity of the model is measured by the number of clusters used, it is evident that in a nonparametric model the effective complexity adapts to the data. The nonparametric models have been applied to a wide range of machine learning problems including clustering, classification, regression and sequence learning [18].

3.3.1 Dirichlet Process

In a Bayesian approach the parameters are treated as random variables and are assigned prior distributions. Bayesian nonparametric methods define a prior over an infinite dimensional parameter space in such a way that the number of parameters used vary with the data complexity. The Dirichlet Process (DP) [42, 19] is a commonly used nonparametric prior over the infinite dimensional space of distributions. It is a distribution over probability distributions. Each sample drawn from a DP is a discrete distribution. Tractable posterior inference procedures can be developed when employing a DP prior, making it practically useful.

There are several perspectives on the Dirichlet Process. The DP can be constructed from finite-dimensional Dirichlet distributions. It can also be defined implicitly by an underlying process that generates a sequence of random variables. Yet another perspective is to describe the random draw from a Dirichlet Process explicitly using a so-called stick-breaking construction. Finally, the DP can be viewed as the infinite limit of finite mixture models. These perspectives are mentioned below.

Let H be a distribution over a probability space Θ and γ be a positive real number. Let $(A_1, \dots A_K)$ be a partition over Θ such that $\bigcup_{k=1}^{K} A_k = \Theta$ and $A_k \cap A_j = \emptyset, \forall\, k \neq j$. A random probability distribution G_0 is Dirichlet Process distributed if for every partition of Θ, the joint distribution of random probabilities is Dirichlet distributed as follows:

$$\big(G_0(A_1), \dots, G_0(A_K)\big) \sim Dir(\gamma H(A_1), \dots, \gamma H(A_K)) \tag{3.22}$$

The draw from a Dirichlet process is denoted as $G_0 \sim DP(\gamma, H)$, where the base distribution H is the mean of the DP and the concentration parameter γ can be interpreted as an inverse

variance that controls the variability around H. A larger value for γ will result in the DP concentrating its mass around the mean.

It is possible to draw samples from G_0 because G_0 is a random distribution. Let $\phi_1, \dots, \phi_i, \dots, \phi_n$ be a sequence of independent samples drawn from G_0 with each ϕ_i taking values in Θ. The posterior distribution of the DP is given as follows:

$$\big(G_0(A_1), \dots, G_0(A_K)\big) \,|\phi_1, \dots \phi_n \sim Dir(\gamma H(A_1) + N_1, \dots, \gamma H(A_K) + N_K) \tag{3.23}$$

Here $N_k = \#\{i: \phi_i \in A_k\}$ denotes the number of samples in A_k. The posterior distribution of the DP is also a DP. Thus the DP provides a conjugate prior over distributions, a desirable property that simplifies posterior computation.

The values drawn from G_0 are repeated because it is a discrete distribution. Let $\theta_1, \dots, \theta_k, \dots \theta_K$ be the unique values among $\phi_1, \dots \phi_n$ and N_k be the number of times θ_k is observed. Marginalizing out G_0, a new value ϕ_{n+1} is sampled as follows:

$$\phi_{n+1} |\, \phi_1 \dots \phi_n \sim \sum_{k=1}^{K} \frac{N_k}{\gamma + n} \delta_{\theta_k} + \frac{\gamma}{\gamma + n} H \tag{3.24}$$

where δ_θ is an atom located at θ. The probability that θ_k will be repeated depends on the number of times it has already been observed.

The unique θ_k values induce a random partition of the set $\{1, \dots, n\}$ into clusters. Consider the set of ϕ_i's with identical θ_k values as belonging to a cluster k. The above sampling scheme assigns an observation into an existing cluster k with a probability that depends on the number N_k of observations already assigned to the cluster and creates a new cluster with probability $\frac{\gamma}{\gamma+n}$. The larger N_k is, the higher the probability that cluster k will be assigned more observations. Larger clusters grow larger, faster. This implies a rich-gets-richer phenomenon that is desirable in clustering.

The above distribution over the partitions is understood intuitively by the *Chinese Restaurant Process* metaphor shown in Figure 3.4. There is a restaurant with an infinite number of tables (clusters). The first customer is seated at the first table. A second customer sits either at the first table or in a new table. In general, the $(n+1)^{th}$ customer joins an already occupied table with probability proportional to the number N_k of customers already seated at table k. The customer can also choose a new table with probability proportional to γ.

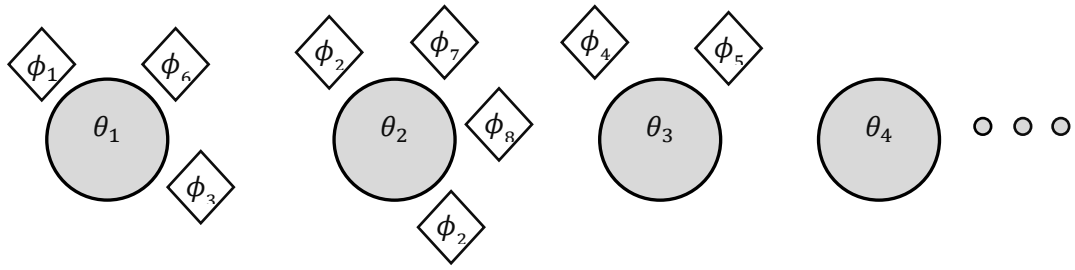


Figure 3.4: Chinese Restaurant Process. The circles denote the tables and θ_k is the unique value associated with table k. The diamonds represent the customers with ϕ_n being the n^{th} customer. A new customer selects an existing table with a probability proportional to the number of customers already seated at a table. A new table may also be selected.

The stick-breaking representation [43] shows explicitly the discreteness of the random distributions drawn from a DP. The random distribution G_0 can be determined by drawing an unbounded number of samples from a Beta distribution and the base distribution H. The generation process is as follows:

$$\beta'_k | \gamma \sim Beta(1, \gamma) \qquad \beta_k = \beta'_k \prod_{l<k} (1 - \beta'_l) \qquad k = 1,2 \ldots \tag{3.25}$$

$$\theta_k \mid H \sim H \qquad k = 1,2 \ldots \tag{3.26}$$

$$G_0 = \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k} \tag{3.27}$$

The values θ_k are drawn independently from the base distribution H and β_k is a probability associated with the atom δ_{θ_k} and $\sum_{k=1}^{\infty} \beta_k = 1$. The construction of β_k can be understood metaphorically by the division of a stick into an infinite number of segments as depicted in Figure 3.5. We start with an unit length stick, choose β'_1 according to (3.25) and break the stick at β'_1. For the remaining segment, we choose β'_2 and break off the β'_2 proportion of the remainder of the stick and so on. This provides a distribution on the strictly positive integers. It is common to write the weights $\beta = \{\beta_k\}_{k=1}^{\infty}$ obtained using (3.25) as $\beta \sim GEM(\gamma)$, named after Griffiths, Engen, and McCloskey.

The above representation is useful in the interpretation of the DP as the infinite dimensional generalization of a finite parametric model. Consider a finite mixture with K components. The density for this mixture model over observations x is represented as follows:

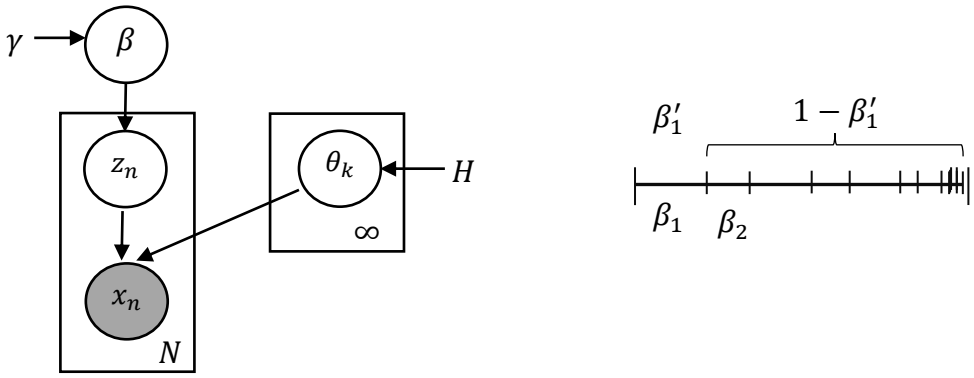


Figure 3.5: Dirichlet Process. *Left*: A representation of the Dirichlet Process mixture in a graphical format. The variables are represented as nodes in the graph. The latent variable z_n indicates the mixture component an observation x_n belongs to. The replicated variables are compactly represented using the plate notation [29]. The rectangles denote replication with the number of replicates given in the bottom right corner. *Right*: The stick breaking construction. The stick is broken at β'_k and subsequent weights β_{k+1} are obtained as random proportions of this segment.

$$p(x) = \sum_{k=1}^{K} w_k \, p(x|\theta_k) \tag{3.28}$$

Here w_k is the weight for the k^{th} component of the mixture and θ_k is a set of parameters associated with the k^{th} component. This density can also be written as

$$p(x) = \int p(x|\theta) G_0(\theta) d\theta \tag{3.29}$$

where G_0 is a mixing discrete distribution and is defined as follows:

$$G_0 = \sum_{k=1}^{K} w_k \delta_{\theta_k} \tag{3.30}$$

In the nonparametric extension to the finite mixtures, the mixing distribution in equation (3.27) is used instead of (3.30). This gives rise to a mixture model with an unbounded number of mixture components. The DP is used as a prior over G_0 and the resulting mixture model is called the DP mixture [42]. The β_k terms in (3.27), which defines a probability distribution on the set $\mathbb{Z}^+$, is interpreted as the mixture weights. This probability distribution is chosen at random via the stick breaking process.

The process by which observations are generated from a Dirichlet process mixture with concentration parameter γ and base distribution H is described as follows:

(i) Generate the stick breaking weights β as $\beta \mid \gamma \sim GEM(\gamma)$

(ii) Draw independent samples of the parameters from the base distribution as $\theta_k \mid H \sim H, k = 1, 2 \ldots$.

(iii) Draw a latent variable z_n that indicates the mixture component that the n^{th} observation belongs to. This is done using the stick breaking weights and is denoted as $z_n \mid \beta \sim \beta, n = 1 \ldots N$.

(iv) Draw an observation x_n given the latent variables and the parameters as $x_n \mid z_n, \{\theta_k\}_{k=1}^{\infty} \sim F(\theta_{z_n}), k = 1,2, \ldots \quad n = 1, \ldots N$. Here F denotes the distribution family of the mixture component using θ as its parameter.

Figure 3.5 illustrates this process. In the Dirichlet process mixture, the mixture component k is described by the distribution $F(\theta_k)$. Note that the ϕ_n discussed in the Chinese restaurant process is simply θ_{z_n}. The probability that an observation is assigned to the k^{th} component is β_k. Note that the number of components $K \rightarrow \infty$. Thus the DP can be used to model a mixture with no upper bound on the number of components.

3.3.2 Hierarchical Dirichlet Process

In many situations, we encounter data organized into distinct groups. There is a need to capture both the similarities and differences across the individuals within these groups. As an example, consider the problem of modelling the topics [183] embedded in a corpus of documents. A document consists of a number of words which arise from a set of underlying semantic themes referred as topics. We want to describe the way in which the topics are shared across the documents and yet capture the document specific properties of a topic – i.e. we wish to share a common set of clusters (topics) among several related groups (documents). The Dirichlet Process as such cannot be used to model grouped data. The need to share clusters among groups motivates the use of a hierarchical model.

The Hierarchical Dirichlet Process (HDP) [44] is an extension of the DP. It is used to model groups of data. Each group has a separate DP prior but all these DPs are linked through a global DP. This provides a mechanism for inferring group specific probability masses while at the same time sharing parameters across the groups.

As before, let G_0 denote a draw from a Dirichlet Process with concentration parameter γ and a base distribution H. The HDP defines a set of random distributions $\{G_j\}_{j=1}^{J}$ over J groups of

data. Given the global distribution G_0, the set of distributions over the J groups are conditionally independent.

$$G_0 \mid \gamma, H \sim DP(\gamma, H) \tag{3.31}$$

$$G_j \mid \alpha, G_0 \sim DP(\alpha, G_0) \qquad j = 1, \ldots J \tag{3.32}$$

The j^{th} group's distribution G_j contains values drawn from G_0 with $\alpha \in \mathbb{R}^+$ controlling the variability around G_0. The distribution G_0 can be interpreted as the mean distribution across all the groups.

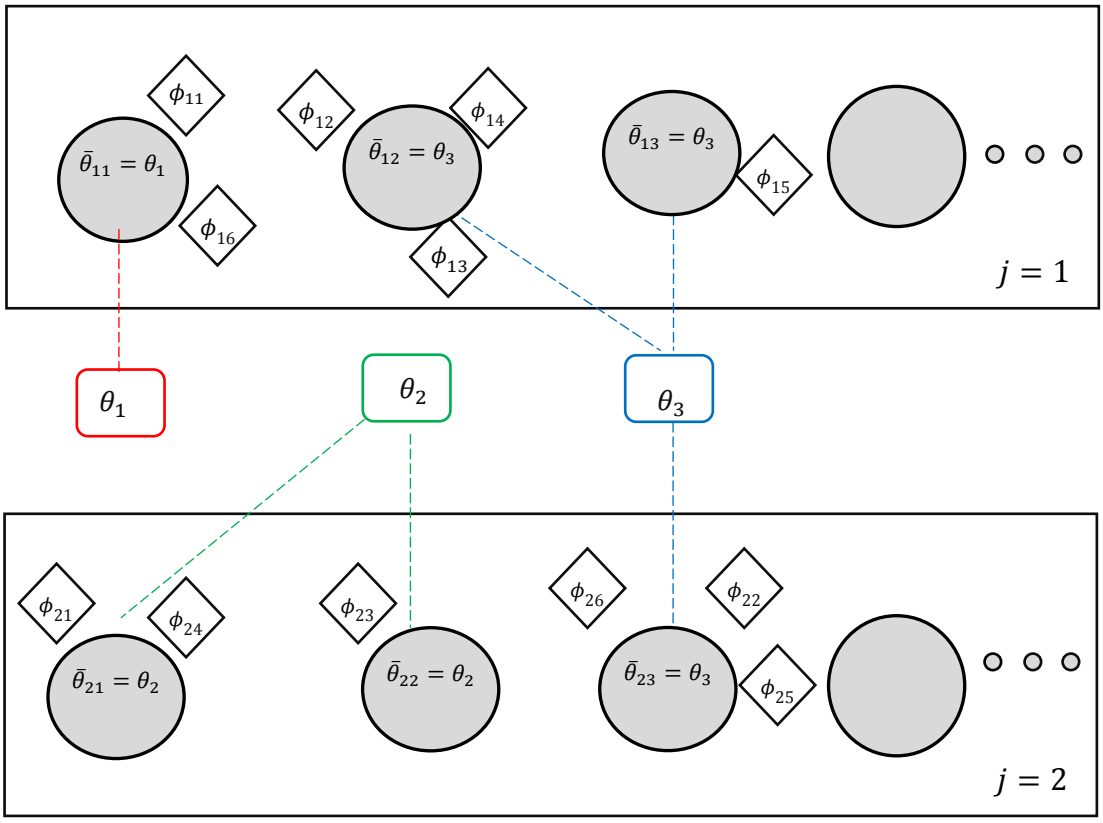


Figure 3.6: Chinese Restaurant Franchise. The diamonds represent the customers with ϕ_{jn} being the n^{th} customer at restaurant j. The circles denote the tables. The dishes are shared across the restaurants with the dish (parameter) θ_3 being re-used. The tables and dishes are selected based on the proportion of them being used. A new table or a new dish may also be selected.

The HDP can also be described using a metaphor, now called the *Chinese Restaurant Franchise* illustrated in Figure 3.6. There are now J restaurants (each corresponding to an HDP group) with all the restaurants sharing a single menu of an infinite number of dishes (parameters). Let ϕ_{jn} denote the n^{th} customer from the j^{th} restaurant. Let $\theta_1, \ldots, \theta_k, \ldots \theta_K$ be the dishes served

across all the restaurants and let $\bar{\theta}_{jt}$ denote the dish served at table t in restaurant j. Note that each ϕ_{jn} is associated with one $\bar{\theta}_{jt}$ while each $\bar{\theta}_{jt}$ is associated with one θ_k. Let the number of tables in restaurant j be $M_{j.}$ A new customer ϕ_{jn+1} selects an existing table t proportional to the number of customers N_{jt} already seated at the table. A new table may also be selected.

$$\phi_{jn+1} | \, \phi_{j1} \dots \phi_{jn} \sim \sum_{t=1}^{M_{j.}} \frac{N_{jt}}{\alpha + n} \delta_{\bar{\theta}_{jt}} + \frac{\alpha}{\alpha + n} G_0 \tag{3.33}$$

A dish k is selected at a table based on the number $M_{.k}$ of tables across the restaurants serving the dish. A new dish may also be selected with a probability $\frac{\gamma}{\gamma + M_{..}}$ where $M_{..}$ is the total number of occupied tables.

$$\bar{\theta}_{jt} | \, \bar{\theta}_{11}, \bar{\theta}_{12}, \dots, \bar{\theta}_{21}, \dots \bar{\theta}_{j1}, \bar{\theta}_{jt-1} \sim \sum_{k=1}^{K} \frac{M_{.k}}{\gamma + M_{..}} \delta_{\theta_k} + \frac{\gamma}{\gamma + M_{..}} H \tag{3.34}$$

Similar to the stick breaking representation of the global distribution G_0 in equations (3.25) to (3.27), the group specific G_j distributions can be explicitly written as follows:

$$G_j = \sum_{k=1}^{\infty} \pi_{jk} \delta_{\theta_k} \tag{3.35}$$

The δ_{θ_k} atoms are shared across all the groups but each group j has a different set of weights $\{\pi_{jk}\}_{k=1}^{\infty}$. Let $\pi_j = \{\pi_{jk}\}_{k=1}^{\infty}$ and $\sum_{k=1}^{\infty} \pi_{jk} = 1$. The stick breaking construction for π_j is based on independent sequences of the $\{\pi'_{jk}\}_{k=1}^{\infty}$ random variables drawn from a Beta distribution. It is written as follows:

$$\pi'_{jk} \mid \alpha, \beta \sim Beta\left(\alpha\beta_k, \alpha(1 - \sum_{l<k} \beta_l)\right) \qquad k = 1,2 \dots \tag{3.36}$$

$$\pi_{jk} = \pi'_{jk} \prod_{l<k} (1 - \pi'_{jl}) \qquad k = 1,2 \dots \tag{3.37}$$

The weights π_j are independent given β and each π_j is independently distributed according to $DP(\alpha, \beta)$. Both the β and π_j terms are interpreted as a random probability distribution on the set $\mathbb{Z}^+$. The average weight of the clusters is determined by β, while α controls the variability of the weights across groups.

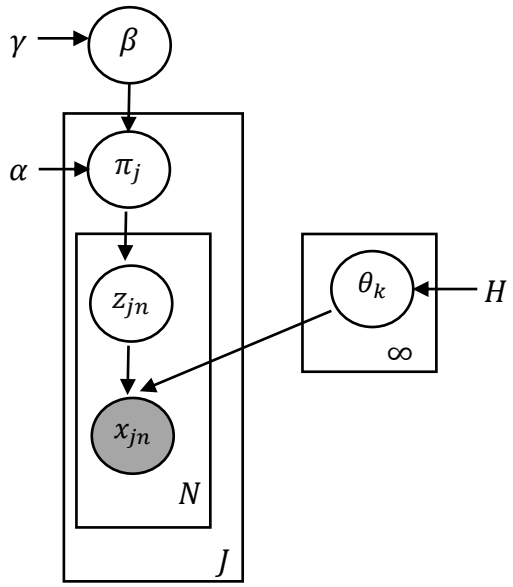


Figure 3.7: Hierarchical Dirichlet Process. A representation of the HDP used as a nonparametric prior for clustering grouped data. The variables are represented as nodes in the graph. The concentration parameters γ, α and the prior H are shown as text. The latent variable z_{jn} indicates the mixture component an observation x_{jn} belongs to. The parameter θ_k is shared by all groups while the mixture proportion π_j may vary among the groups. The plate notation is used.

While the DP is used as a prior for a mixture model for clustering data, the HDP is used as a nonparametric prior distribution for a set of mixture models that are used for clustering grouped data. The process by which observations are generated based on the HDP is given as follows:

$$\begin{aligned}
\beta \mid \gamma &\sim GEM(\gamma) && \\
\pi_j \mid \alpha, \beta &\sim DP(\alpha, \beta) && j = 1, \dots J \\
\theta_k \mid H &\sim H && k = 1,2, \dots \\
z_{jn} \mid \pi_j &\sim \pi_j && j = 1, \dots J, n = 1 \dots N \\
x_{jn} \mid z_{jn}, \{\theta_k\}_{k=1}^{\infty} &\sim F(\theta_{z_{jn}}) && j = 1, \dots J, n = 1 \dots N
\end{aligned} \tag{3.38}$$

Here z_{jn} is a latent variable that indicates the mixture component that the j^{th} group's n^{th} observation x_{jn} belongs to. The ϕ_{jn} discussed in the Chinese restaurant Franchise is $\theta_{z_{jn}}$. For a given component k, all the J groups share the same set of parameters θ_k but the j^{th} group uses π_{jk} proportion. This process is shown in Figure 3.7.

This thesis uses HDP extensively to enable construction of nonparametric models.

4. Discriminative nonparametric HMM

In this chapter, a nonparametric HMM based on the Hierarchical Dirichlet Process (HDP) is proposed for classifying human actions. The proposed model addresses an important limitation of the classical HMM, namely the need to fix the number of hidden states a-priori. The novel construction provided here produces a flexible model that is better suited for classification tasks. The formulation enables information sharing and allows the use of unlabelled examples.

The chapter begins with an overview of the proposed approach in Section 4.1 and introduces the HDP-HMM in Section 4.2. Instead of using separate models for each action class, a single HDP-HMM is used to model all the actions. In order to distinguish between the actions, the HDP is extended by an additional level and class specific transformations are introduced for the distributions of HDP parameters. Section 4.3 elaborates on this model structure. During training, the parameters are learnt in a discriminative manner. This process is discussed in Section 4.4 and is followed in Section 4.5 by the derivation of the posterior inference procedure. Experiments are conducted on two different publicly available datasets that contain depth image sequences. The information in the skeletal joint positions is used to classify the actions using the proposed model. The results are presented in Section 4.6. The chapter ends with some concluding remarks in Section 4.7. Portions of this chapter have been published [192, 193].

4.1 Overview

Depth sensors such as Kinect, with inbuilt human motion capturing techniques, provide estimates of a human skeleton's 3D joint positions over time [14]. High level actions can be inferred from these joint positions. However, robust and accurate inference is still a problem.

Given a sequence of 3D joint positions, a state space model such as a Hidden Markov Model (HMM) is a natural way to represent an action class. The HMMs are proven models for sequential pattern recognition [48, 49, 50]. Recall that in an HMM, a sequence of discrete state variables are linked in a Markov chain by a state transition matrix. Each observation is drawn independently from a distribution conditioned on the appropriate state [32]. If each action class is represented using an HMM, the model parameters corresponding to a given class, namely the state transition matrix and the state specific observation distributions, can be learnt from

examples belonging to that class. The prediction of a new input's class is obtained from the class conditional posterior densities.

In classical parametric HMMs, the number of states must be specified a-priori. In many applications this number is not known in advance. For instance, there is no a-priori knowledge about the number of intermediate poses that comprises an action. This number will vary depending on the complexity of the action and the number of subjects in the data set. A typical solution to this problem is to carry out training using different choices for the number of states and then apply a model selection criterion to find the best result. There is little understanding of the strengths and weaknesses of this procedure and often complex application specific tuning is involved.

Instead of this ad hoc model selection, it is preferable to estimate the correct number of states automatically from data. This allows the model complexity to adapt to the size of the data set. The nonparametric methods [18, 19] provide the necessary statistical framework to model the data with an unbounded number of parameters. The number of parameters grows with the sample size. In [44], a nonparametric Bayesian method, the Hierarchical Dirichlet Process (HDP), is defined. The HDP is used to construct an HMM with an unbounded set of states. The prior distribution on the HMM transition matrix is over an infinite state space but for a given set of observations, only a finite number of the states is used to explain the data.

It would be straight forward to use separate HDP-HMMs for each action class and train them individually. However, this would prohibit the sharing of training examples across the action classes. To see the merit of sharing examples, consider that an action is a sequence of poses. It is quite likely that two or more actions share many similar poses with possibly a few poses unique to particular actions. In fact, two actions such as 'stand-up' and 'sit-down', may have the same set of poses with only the temporal order of pose sequences differing. What necessarily differentiates one action from another are the transition probabilities of the poses. If a particular pose is absent from an action class then there is a low probability of transition to the state for that pose. In this work, a single HDP-HMM is used to model all the action classes. The canonical HDP-HMM is extended with an additional class specific hierarchical level that accounts for differences in the state transition probabilities among the action classes.

In the canonical construction of the HDP-HMM, the mixture components are shared across the hierarchical levels. It would be more flexible to allow the mixture components of an action class to vary *slightly* from the other classes; i.e. we seek a class specific transformation of the shared mixture component parameters so that the classes can be discriminated in a better manner. Note that this is different from using individually trained HDP-HMM models where the

component parameters are not shared among the classes. In this work, the mixture components are assumed to have Gaussian distributions, and class specific affine transformations of the Gaussian distribution parameters (mean and covariance) are used. An overview of the approach is provided in Figure 4.1.

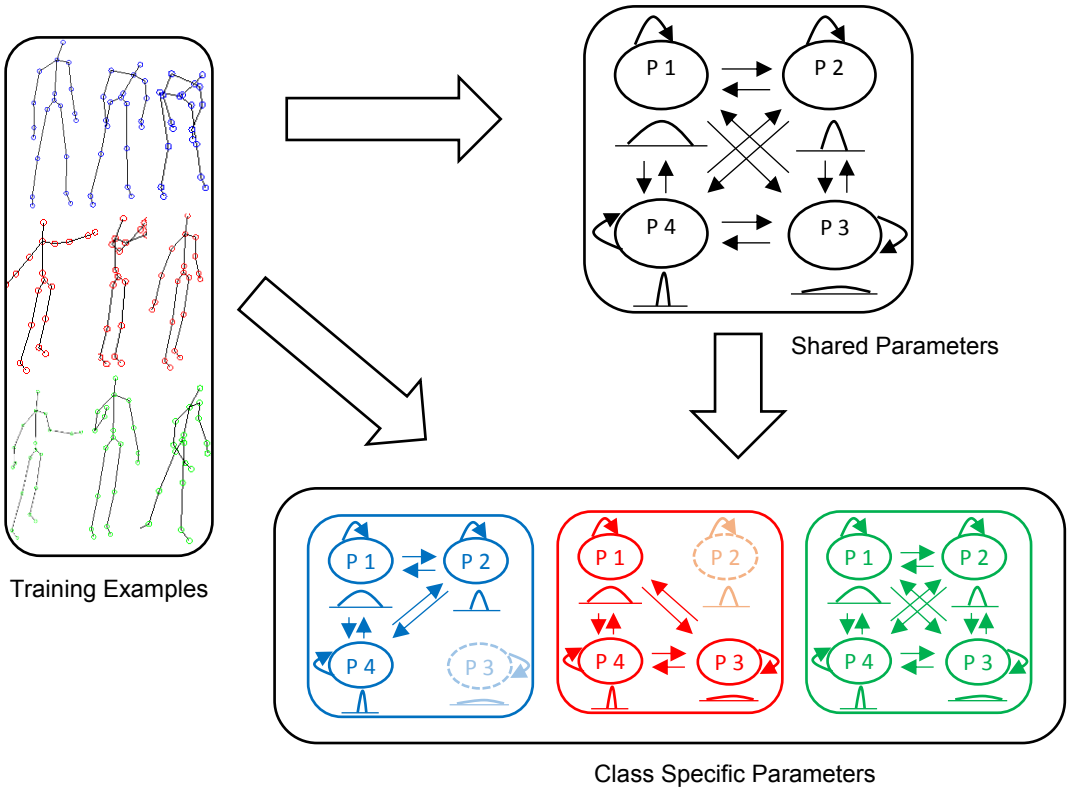


Figure 4.1: Discriminative HDP-HMM overview. *Training examples* contain joint position sequences from different action classes. The examples from all these action classes are combined in order to infer the shared pose transitions and pose definitions. P1, P2, P3 and P4 in the *Shared Parameters* group represent the various poses (states). Each pose is defined by a distribution. The action *class specific* transitions and distributions are inferred as *transformations* of this shared representation. Pose P3 may be absent in the first action class and hence there is a low probability of transition to it (shown with an absence of arrow to this state). The action class labels in the training examples and the learned shared parameters are used to infer the class specific parameters.

The HDP-HMM based classification approach described above defines a joint distribution of the input data and class labels to train the classifier. This *generative* model allows the augmentation of the labelled training examples with unlabelled examples and thus provides a framework for semi-supervised learning. In contrast, a *discriminative* model uses the conditional distribution of the class labels given the input data to train the classifier. This approach often produces good

classification results [64]. For example, Support Vector Machines (SVMs) [65] use a margin based likelihood that maximizes the distances of the feature vectors to the classification boundary while minimizing the empirical error rate on the training set. Inspired by this, a margin based term is incorporated in the likelihood function used in HDP-HMM training. The inclusion of this discriminative term in the otherwise generative model, compensates for potential model misspecification and leads to better classification results.

Incorporation of a discriminative term into the HDP-HMM model makes the posterior sampling less straight-forward. The HDP model as such has no provision for including an additional term for the mixing proportions. To address this, a normalized gamma process formulation [66] of the HDP is used. This allows a scaling of the mixing proportions of a DP through a discriminative weighting term. For the mixture components with Gaussian distribution parameters, the prior is no longer of the same form as the likelihood and hence is not conjugate. Slice sampling [39] based techniques allow sampling from any likelihood function, even if the normalization is unknown. A Gaussian prior is placed on the parameters and Elliptical Slice Sampling [67] is used to sample the posterior efficiently.

Contributions

The main contributions in this chapter are the construction of a discriminative nonparametric HMM and the derivation of a tractable inference mechanism. The proposed model has the following advantages:

(a) The nonparametric formulation allows the number of states to be inferred automatically.
(b) The use of a single HDP-HMM promotes information sharing.
(c) The discriminative terms ensure that the HDP-HMM is suitable for classification tasks.
(d) The model can be used for semi-supervised learning.
(e) The model is generic, in that it is applicable to other sequence classification problems.

4.2 HDP-HMM

Recall the HMM in Section 3.1. The HMM is parameterized by the transition matrix π where the j^{th} row of the matrix defines the probabilities of the transitions from the state j. The hidden states $z_{1:T}$ have the Markov property. The probability of transitioning to a state z_t at a time instant t from a previous state z_{t-1} is specified by the transition matrix. Additionally, there are state specific observation density parameters $\{\theta_k\}_1^K$ where K is the number of hidden states.

For the Bayesian version of the classical HMM, it is necessary to introduce priors. Note that the rows of π cannot have independent priors because the transitions out of the different states

must be coupled. Let the priors be $\beta \sim Dir(\frac{\gamma}{K})$ and $\pi_j \sim Dir(\alpha\beta_1 \dots \alpha\beta_K)$ where Dir is the Dirichlet distribution and γ, α are some positive real numbers. The observation density parameters are assigned a prior H. With this definition, an observation is generated in the Bayesian HMM as follows:

$$\beta \mid \gamma \sim Dir\left(\frac{\gamma}{K}\right) \tag{4.1}$$

$$\pi_j \mid \alpha, \beta \sim Dir(\alpha\beta_1 \dots \alpha\beta_K) \qquad j = 1, \dots K \tag{4.2}$$

$$\theta_k \mid H \sim H \qquad k = 1, \dots K \tag{4.3}$$

$$z_t \mid \pi, z_{t-1} \sim \pi_{z_{t-1}} \qquad t = 1, \dots . T \tag{4.4}$$

$$x_t \mid z_t, \{\theta_k\}_{k=1}^{\infty} \sim F(\theta_{z_t}) \qquad t = 1, \dots T \tag{4.5}$$

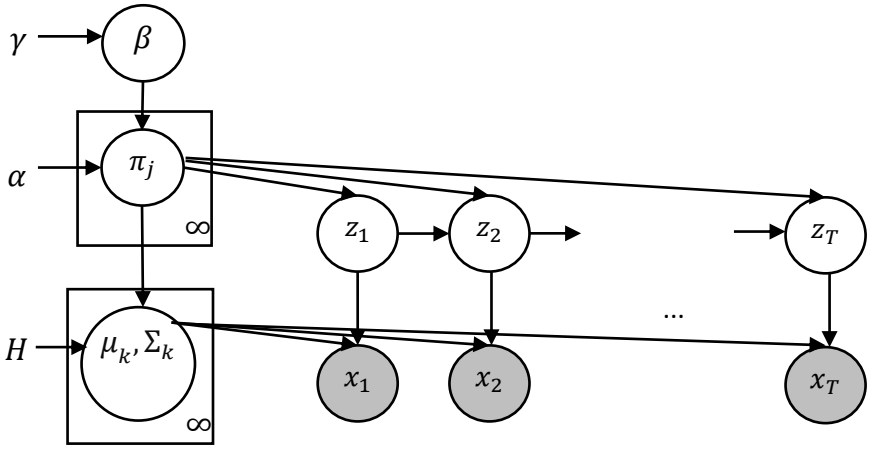


Figure 4.2: Graphical representation of HDP-HMM. The states $z_{1:T}$ have the Markov property. An observation x_t is conditioned on the state z_t. The states are generated from the transition matrix π_j and the observations are generated from the mixture component parameters. The number of states and the number of mixture components are unbounded.

This generation process is remarkably similar to the process by which the observations are generated using HDP shown in Equation (3.38), Section 3.3. A group specific π_j distribution in the HDP is a state specific distribution in the HMM with the *groups* in the HDP formulation corresponding to the *states* in the HMM. If an HDP prior is assigned over the state transition matrix, the matrix will have an infinite number of rows and columns with the HDP semantics ensuring that only a finite subset of these states are actually instantiated. Thus the HMM is now nonparametric. The HDP-HMM can also be interpreted as an infinite extension of a dynamic

mixture model. The mixture weight of an observation depends on the previous observations and there are an unbounded number of mixtures.

To complete the definition of the HDP-HMM, let F be the Gaussian density. The density parameters for the observation density associated with state k are now the mean μ_k and covariance Σ_k. It is convenient to write $\theta_k = (\mu_k, \Sigma_k)$. Let the mixture mean have a normal prior $\mu \sim \mathcal{N}(\mu_0, \Sigma_0)$ and let the covariance have an Inverse-Wishart prior $\Sigma \sim IW(\nu_0, \Delta_0)$. An observation is generated in the nonparametric HMM as follows.

$$\beta \mid \gamma \sim GEM(\gamma) \tag{4.6}$$

$$\pi_j \mid \alpha, \beta \sim DP(\alpha, \beta) \qquad j = 1,2, \dots \tag{4.7}$$

$$\mu_k \mid \mu_0, \Sigma_0 \sim \mathcal{N}(\mu_0, \Sigma_0) \qquad k = 1,2, \dots \tag{4.8}$$

$$\Sigma_k \mid \nu_0, \Delta_0 \sim IW(\nu_0, \Delta_0) \qquad k = 1,2, \dots \tag{4.9}$$

$$z_t \mid \pi, z_{t-1} \sim \pi_{z_{t-1}} \qquad t = 1, \dots . T \tag{4.10}$$

$$x_t \mid z_t, \{\mu_k, \Sigma_k\}_{k=1}^{\infty} \sim \mathcal{N}(\mu_{z_t}, \Sigma_{z_t}) \qquad t = 1, \dots T \tag{4.11}$$

Figure 4.2 provides a graphical representation of this HDP-HMM.

4.3 Model

Let a training dataset comprising N observations $X = \{x^n\}_{n=1}^N$ together with labels $Y = \{y^n\}_{n=1}^N$ be given. Here $x^n = x_1^n, \dots, x_t^n, \dots, x_T^n$ is an input sequence and $y^n \in \{1 \dots c \dots C\}$ is the class label corresponding to the sequence x^n. For example, in action classification, x^n is an input image sequence and y^n is an action class label. The observations and their labels are drawn independently from the same fixed distribution. Each $x_t^n \in \mathbb{R}^d$ corresponds to the features extracted at time step t from the input. Further discussion of the features is deferred to Section 4.6. Let the set of all model parameters be θ. The objective is classification, where given a new test observation sequence $\hat{x}$, the corresponding action class $\hat{c}$ must be predicted. A suitable prediction is

$$\hat{c} = \underset{c}{\operatorname{argmax}}\, p(c \mid \hat{x}, X, Y) \tag{4.12}$$

The distribution $p(c \mid \hat{x}, X, Y)$ can be written in the form

$$p(c \mid \hat{x}, X, Y) = \int p(c \mid \hat{x}, \theta)\, p(\theta \mid X, Y)\, d\theta \tag{4.13}$$

The model proposed in this chapter differs from the canonical HDP-HMM in two key aspects. First, an extra level in the HDP is introduced to model class specific mixture proportions as discussed in Section 4.3.1. The second difference is the extension of the HDP parameter space with class specific distributions for the mean and covariance parameters. This novel formulation is presented in Section 4.3.2.

4.3.1 Two level HDP

If each action class is represented by a separate HDP-HMM, then $\theta^c = \{\beta^c, \pi^c, \mu^c_{1..\infty}, \Sigma^c_{1..\infty}\}$ are the parameters for class c, $\theta = \{\theta^c\}_{c=1}^{C}$ is the set of all parameters for the different classes and $\gamma, \alpha, \mu_0, \Sigma_0, \nu_0, \Delta_0$ are the hyper parameters. It would be straight forward to estimate the posterior density of parameters $p(\theta \mid X, Y)$ if each HDP-HMM model were to be trained separately i.e. a class conditional density $p(x \mid c)$ can be defined for each class and the posterior can be estimated from

$$p(\theta^c \mid X, Y) = p(\theta^c) \prod_{n:y^n=c} p(x^n \mid \theta^c) p(c) \tag{4.14}$$

However, in this approach the training examples from other classes are not used when learning the parameters of a class. As noted in Section 4.1, many actions contain similar poses and it is useful to incorporate pose information from other classes during training. Specifically, the inclusion of additional observations for a similar pose benefits estimation of the Gaussian mixture parameters. The state transition parameters must continue to be different for each action class since it is these parameters that necessarily distinguish the actions.

Instead of separate HDP-HMMs, a single HDP-HMM is defined for all the action classes albeit with an extra level that is class specific i.e. in addition to the global distribution G_0 and the state specific distributions G_j in the canonical HDP, there are now class specific distributions G_j^c for every state. The two-level HDP-HMM is defined as follows.

$$G_0 \mid \gamma, H \sim DP(\gamma, H) \tag{4.15}$$

$$G_j \mid \alpha, G_0 \sim DP(\alpha, G_0) \qquad j = 1,2, \ldots \tag{4.16}$$

$$G_j^c \mid \lambda, G_j \sim DP(\lambda, G_j) \qquad c = 1, \ldots C \tag{4.17}$$

Just as the G_js are conditionally independent given G_0, the G_j^cs are conditionally independent given G_j. All the classes for a given state share the same subset of mixture parameters but the proportions of these mixtures will differ for each class as determined by the positive real valued

concentration parameter λ. The varying mixture proportions induce differences in the state transition probabilities between the action classes and ensure that classification can be performed.

Recall the stick breaking construction of the canonical HDP presented in Equations (3.35) to (3.37). The extension of this construction to the additional class specific measure is straightforward. In addition to the stick breaking weights β and π, a new weight term φ is now introduced to represent the second level. Independent sequences of the $\{\varphi_{jk}^{c'}\}_{k=1}^{\infty}$ random variables are drawn from the Beta distribution (Appendix C.3). The formulation is as follows:

$$\begin{aligned} \varphi_{jk}^{c'} \mid \lambda, \pi_j &\sim Beta\left(\lambda\pi_{jk}, \lambda(1-\sum_{l<k}\pi_{jl})\right) && k = 1,2 \dots \\ \varphi_{jk}^{c} &= \varphi_{jk}^{c'}\prod_{l<k}\left(1-\varphi_{jl}^{c'}\right) && k = 1,2 \dots \\ G_j^c &= \sum_{k=1}^{\infty}\varphi_{jk}^{c}\delta_{\theta_k} \end{aligned} \tag{4.18}$$

Similar to β and π_j, $\varphi_j^c = \{\varphi_{jk}^c\}_{k=1}^{\infty}$ can be interpreted as a random probability distribution on the set $\mathbb{Z}^+$. Assuming the variables π_j, μ_k and Σ_k are defined as in equations (4.7) to (4.9), the generative story for an observation x_t^n belonging to class c, sampled at time t from the two level HDP-HMM is written as

$$\begin{aligned} \varphi_j^c \mid \lambda, \pi_j &\sim DP(\lambda, \pi_j) \\ z_t^n \mid z_{t-1}^n, y^n = c, \{\varphi_j^c\}_{j=1,c=1}^{\infty,C} &\sim \varphi_{z_{t-1}^n}^c \\ x_t^n \mid z_t^n, \{\mu_k, \Sigma_k\}_{k=1}^{\infty} &\sim \mathcal{N}(\mu_{z_t^n}, \Sigma_{z_t^n}) \end{aligned} \tag{4.19}$$

Consequently, for the two level HDP-HMM, the set of all model parameters is $\theta = \{\beta, \pi, \varphi^{1..C}, \mu_{1..\infty}, \Sigma_{1..\infty}\}$ with $\gamma, \alpha, \mu_0, \Sigma_0, \nu_0, \Delta_0, \lambda$ as the hyper parameters.

4.3.2 Transformed HDP Parameters

In the HDP, the same mixture component parameters are used by the different groups i.e. the parameters θ_k remain the same in all G_j (and G_j^c in case of an additional level). This is less flexible than allowing the parameters to vary across the groups. As an example, the position and orientation of the joints in a squat pose might mostly look the same across action classes such as sit-up, sit-down and pick-up while it may slightly differ for pick-up class. In this case, it would be useful to capture the deviation from the standard squat pose for the pick-up action class –

i.e. we wish to introduce a transformation of the parameters, one for each action class, from its canonical form θ_k.

The affine transformation of the Gaussian distribution parameters mean μ and covariance Σ is considered here [68]. Let ρ be a vector and let Λ be an invertible matrix. The transformation of the Gaussian distribution defined by ρ, Λ is as follows

$$\mathcal{N}(\mu, \Sigma) \longmapsto \mathcal{N}(\Lambda\mu + \rho, \Lambda\Sigma\Lambda^T) \tag{4.20}$$

It is usual to restrict Λ in order to ensure computational tractability. A useful simplification is to set Λ equal to the identity matrix. This is equivalent to restricting the transformations to a translation of the Gaussian mean by ρ. Other restrictions include requiring Λ to be diagonal, to account for scaling.

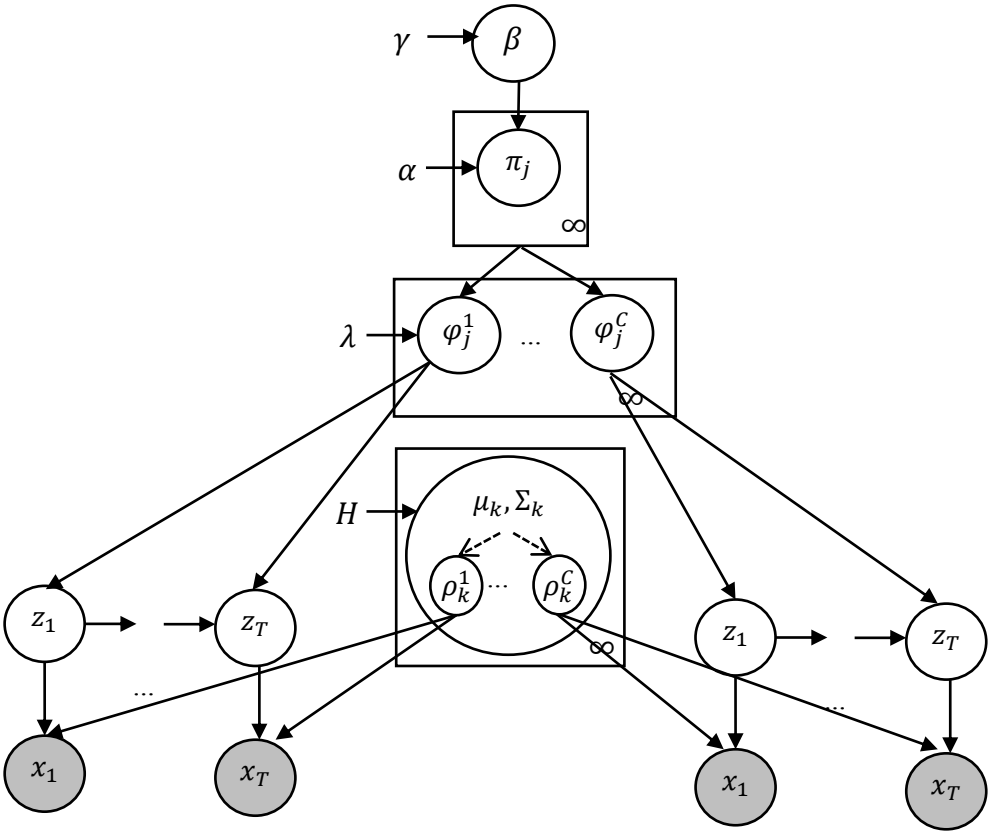


Figure 4.3: Graphical representation of the two level HDP-HMM. The HDP-HMM is extended with the class specific mixture weights φ_j^c and class specific transformation parameters ρ_k^c. The observations on the left side are generated by the parameters for class $c = 1$ while those on the right side by the parameters for class $c = C$.

The class specific transformation based on (4.20) is introduced here to the Gaussian mixture parameters. Let the variable responsible for shifting the mean have a zero mean normal prior

i.e. $\rho \sim \mathcal{N}(0, \Omega_0)$. The focus here is only on scale transformations. Thus Λ is assumed to be diagonal. In effect, the scale transform variable is now a vector and independent log normal priors can be assigned for each element i.e. $log(\Lambda_j) \sim \mathcal{N}(\vartheta_0, \sigma_0)$. An observation x_t^n belonging to class c sampled at time t from the two level HDP-HMM that uses Gaussian mixtures with transformed parameters is generated now as follows.

$$\begin{aligned}
\beta \mid \gamma \sim GEM(\gamma) \qquad \pi_j \mid \alpha, \beta \sim DP(\alpha, \beta) \qquad \varphi_j^c \mid \lambda, \pi_j \sim DP(\lambda, \pi_j) \\
\mu_k \mid \mu_0, \Sigma_0 \sim \mathcal{N}(\mu_0, \Sigma_0) \qquad \Sigma_k \mid \nu_0, \Delta_0 \sim IW(\nu_0, \Delta_0) \\
\rho_k^c \mid \Omega_0 \sim \mathcal{N}(0, \Omega_0) \qquad log(\Lambda_{jk}^c) \mid \vartheta_0, \sigma_0 \sim \mathcal{N}(\vartheta_0, \sigma_0) \\
z_t^n \mid z_{t-1}^n, y^n = c, \{\varphi_j^c\}_{j=1,c=1}^{\infty,C} \sim \varphi_{z_{t-1}^n}^c \\
x_t^n \mid z_t^n, y^n = c, \{\mu_k, \Sigma_k\}_{k=1}^{\infty}, \{\rho_k^c, \Lambda_k^c\}_{k=1,c=1}^{\infty,C} \sim \mathcal{N}(\Lambda_{z_t^n}^c \mu_{z_t^n} + \rho_{z_t^n}^c, \Lambda_{z_t^n}^c \Sigma_{z_t^n} {\Lambda_{z_t^n}^c}^T)
\end{aligned} \tag{4.21}$$

Inclusion of the class specific transforms can be interpreted as an extension of the parameter space. The global distribution is now being drawn from $G_0 \sim DP(\gamma, H_s \times H_1 \ldots \times \ldots H_C)$, where H_s is a base distribution for parameters that are shared across the classes while $H_1, \ldots, H_C$ are class specific. During inference, the posterior distributions for the shared parameters do not depend upon the class labels unlike the class specific parameters. With the augmentation of transform variables, the set of all model parameters is $\theta = \{\beta, \pi, \varphi^{1..C}, \mu_{1..\infty}, \Sigma_{1..\infty}, \rho_{1..\infty}^{1..C}, \Lambda_{1..\infty}^{1..C}\}$ and $\gamma, \alpha, \mu_0, \Sigma_0, \nu_0, \Delta_0, \lambda, \Omega_0, \vartheta_0, \sigma_0$ are the hyper parameters. A graphical representation of the full model is shown in Figure 4.3.

4.3.3 Chinese Restaurant Process Metaphor

The mixture components generated by the extended HDP model can be understood using the Chinese Restaurant Process metaphor discussed in Section 3.3.1. Recall that in the HDP analogue, multiple restaurants share a single menu of dishes across the tables in the restaurants.

In the HDP extended to a second level, each restaurant in the franchise has sections namely family, kids and adults section. There is still a single menu across the sections and the restaurants. Given the customer's preferred section, the customer entering a given restaurant selects a table in proportion to the number of customers already seated in the tables of that section of the restaurant. The customer can also select a new table in that section. Each table is now assigned a dish in proportion to the number of tables across the sections, across the franchise serving that dish. In this two-level HDP metaphor, the sections correspond to the action classes.

In the case of two-level HDP-HMM with transformed parameters, each dish now contains a base part and a flavouring part. A dish contains flavours for every section viz. spicy flavour for family, bland for kids and hot for adults. A dish served at a table in a given section (of any restaurant in the franchise) has its base part seasoned according to that section's flavour. In this metaphor, the flavours correspond to the class specific transformation parameters while the base part correspond to the parameters shared across the classes.

4.4 Discriminative Learning

In the two level HDP-HMM with transformed parameters described above, let the model parameters specific to a class c be $\theta^c = \{\varphi^c, \rho^c_{1..\infty}, \Lambda^c_{1..\infty}\}$ and let the shared parameters across the classes be $\theta^s = \{\beta, \pi, \mu_{1..\infty}, \Sigma_{1..\infty}\}$. Note that $\theta = \theta^s \cup \{\theta^c\}_{c=1}^C$. The posterior distribution for the class specific parameters is very similar to the form of (4.14), but with an additional conditioning on the shared parameters.

$$p(\theta^c \mid X, Y, \theta^s) \propto p(\theta^c) \prod_{n:y^n=c} p(x^n \mid \theta^c, \theta^s) \tag{4.22}$$

The joint distribution over the inputs and labels $p(x, c \mid \theta^c)$ is used in this formulation. This type of learning is intended to best *explain* the training examples belonging to a class. In the asymptotic limit, as the number of training examples is increased, the distribution specified by the model converges to the true distribution of data. This generative model is a very effective way of learning. However, in practice, the specified model is often inaccurate because of a shortage of training data. In addition it may be necessary to compensate for model misspecification [64].

In contrast, the large margin based training used in discriminative learning methods often produces good classification results. The empirical error rate on the training data is balanced against the error rate arising from the generalization of the test data. The tolerance to mismatch between training and test data is due to a wide separation between the classifier decision boundary and the classes – i.e. the decision boundary has a large margin between it and the training examples. Since the class conditional data likelihood is used during prediction in the generative model above, the classifier margin is a function of the model parameters. Adjusting the parameters alters the margins.

There is an implicit assumption in (4.22) that the parameters of a class are (conditionally) independent of the parameters of other classes i.e. $\theta^c \perp \theta^{\setminus c} \mid \theta^s$. Let us relax this assumption and consider a slightly different formulation.

$$p(\theta^c \mid X, Y, \theta^s, \theta^{\setminus c}) \propto p(\theta^c) * p\left(\theta^{\setminus c} \middle| \theta^c, X, Y, \theta^s\right) * \prod_{n:y^n=c} p(x^n \mid \theta^c, \theta^s) \tag{4.23}$$

Here the Bayes theorem product rule for $p(\theta^c \mid X, \theta^{\setminus c})$ is used. The introduction of the second term $p\left(\theta^{\setminus c} \mid \theta^c, X\right)$, referred henceforth as the discriminative term, offers more flexibility. For example, this term can be used during inference to minimize classification error on the training set and introduce margin constraints. This discriminative term compensates for the model misspecification and improves classification results.

4.4.1 Scaled HDP and Normalized Gamma Process

The HDP with its stick breaking construction does not provide any mechanism for influencing the per-group component proportions through additional factors. This makes incorporation of the discriminative term during inference for φ^c tricky. An alternative construction for the last level in the two-level HDP in (4.18) is

$$\begin{aligned} \varphi_{jk}^c \mid \lambda, \pi_j &\sim Gamma\left(\lambda\pi_{jk}, 1\right) \\ G_j^c &= \sum_{k=1}^{\infty} \frac{\varphi_{jk}^c}{\sum_{k'=1}^{\infty} \varphi_{jk'}^c} \delta_{\theta_k} \end{aligned} \tag{4.24}$$

A Dirichlet distributed vector can be generated by independently drawing from a gamma distribution and normalizing the values. Its nonparametric extension relates to the above normalized gamma process construction of G_j^c. The representation in (4.24) as such does not allow using an additional factor. Let each component be associated with a latent location and let the group specific distribution of the HDP be formed by scaling the probabilities of an intermediate distribution. More specifically, let us modify the last level in the two-level HDP described in (4.17) as

$$\begin{aligned} G_j^{c\prime} \mid \lambda, G_j &\sim DP(\lambda, G_j) \\ G_j^c \mid G_j^{c\prime}, \omega_j^c &\propto G_j^{c\prime} * e^{\omega_j^c} \end{aligned} \tag{4.25}$$

Here $G_j^{c\prime}$ is an intermediate distribution for the existing parameters and ω_j^c is a scaling factor that depends on the latent location. Based on this *scaled* HDP structure, the second variable of the gamma distribution can be used to draw the class specific component proportions as

$$\varphi_{jk}^c \mid \lambda, \pi_j, \omega_j^c \sim Gamma\left(\lambda\pi_{jk}, e^{-\omega_{jk}^c}\right) \tag{4.26}$$

The derivation of (4.26) follows from the property that if $y \sim Gamma(a, 1)$ and is scaled by $b > 0$ to produce $z = by$, then $z \sim Gamma(a, b^{-1})$ [66]. This additional scaling factor allows the incorporation of the discriminative term. During inference, ω_{jk}^{c} is drawn in such a way that the posterior φ_{jk}^{c} is primed for classification.

4.4.2 Elliptical Slice Sampling

Conjugate priors cannot be used for the transform parameters $\rho_{1..\infty}^{c}, \Lambda_{1..\infty}^{c}$ because of the presence of the discriminative term. Hence there is no closed form solution for posterior inference of these parameters. Slice sampling [39] provides a way to sample from a density function without having to find a good proposal distribution. As discussed in Appendix E.3, the challenge in slice sampling is to define an appropriate horizontal slice, which encloses the current sample value, from which a new value will be drawn. This is especially difficult if the target variable takes values in a high dimensional space, as is the case here.

If the density function is a product of a likelihood function and a zero mean Gaussian prior, then Elliptical Slice sampling [67, 103] provides a better sampling mechanism. The idea in this algorithm is to define an ellipse that passes through the current sample value and use a likelihood threshold similar to slice sampling for determining a slice. It is much easier though to define a sampling interval with an elliptical slice unlike slice sampling.

Let $L(\phi)$ be a likelihood function and let the prior for the target variable ϕ be a zero mean Gaussian distribution $\mathcal{N}(0, \Sigma)$. Let u be an auxiliary variable drawn such that $u \sim \mathbb{U}[0, L(\phi^{i})]$ where $\mathbb{U}$ is the uniform distribution. Let an ellipse at the current value ϕ^{i} be defined as

$$\phi^{*}(\psi) = \phi^{i} \cos\psi + \vartheta \, sin\psi \tag{4.27}$$

where $\vartheta \sim \mathcal{N}(0, \Sigma)$ and ψ is a parameter denoting the angle with $\psi \in [0, 2\pi]$. This ellipse goes through both the current value ϕ^{i} and an auxiliary ϑ drawn from the Gaussian prior. The algorithm proposes angles from a bracket $[\psi_{min}, \psi_{max}]$ which is shrunk repeatedly in an exponential manner until an acceptable value is found. Similar to slice sampling a new value $\phi^{*}(\psi)$ is accepted if $L(\phi^{*}(\psi)) > u$.

The values considered for an update lie in a two dimensional plane. The elliptical slice captures the structural properties of the Gaussian prior in a better manner than the horizontal slice used in slice sampling. This algorithm provides an efficient mechanism for sampling even high dimensional variables. Elliptical slice sampling is used here for inferring the transform parameters $\rho_{1..\infty}^{c}, \Lambda_{1..\infty}^{c}$ from the density function defined in (4.23).

4.5 Posterior Inference

The central computation problem is posterior inference for the parameters. It is intractable to compute the exact posterior and the Markov Chain Monte Carlo (MCMC) technique is used to draw posterior samples from $p(\theta \mid X, Y)$. Recall that the shared parameters are $\theta^s = \{\beta, \pi, \mu_{1..\infty}, \Sigma_{1..\infty}\}$ and the class specific parameters are $\theta^c = \{\varphi^c, \rho^c_{1..\infty}, \Lambda^c_{1..\infty}\}$ with $\gamma, \alpha, \mu_0, \Sigma_0, \nu_0, \Delta_0, \lambda, \Omega_0, \vartheta_0, \sigma_0$ being the hyper parameters. Gibbs sampling, as discussed in Appendix E.2, is applied here. The shared parameters θ^s are sampled first and then given θ^s, the samples for each class are drawn one by one. The inference algorithm is outlined in Table 4.1.

4.5.1 Truncated Approximation

For sampling the HDP-HMM parameters, one option is to marginalize over the infinite state transition distributions π and component parameters (μ, Σ) and sequentially sample the hidden states z_t. Unfortunately this technique, referred as *direct assignment* or *collapsed* sampler, exhibits slow mixing rates because the HMM states are temporally coupled.

A better technique is to *block sample* the hidden state sequence z_t using the standard HMM forward-backward algorithm discussed in Section 3.1.1. In this sampler, the state transition distributions and component parameters are explicitly instantiated. Slice sampling techniques [69, 70] or truncated approximations [71] can be employed to take account of the fact that the number of states and parameters is unbounded. In almost sure truncations, for a given number L the stick breaking construction is discarded for $L+1, L+2 \ldots \infty$ by setting $\beta'_L = 1$ in equation (3.25). An alternative technique is to consider a *weak limit approximation* to DP and set

$$GEM(\gamma) \triangleq Dir(\frac{\gamma}{L}, \ldots \frac{\gamma}{L}) \tag{4.28}$$

Here L is an upper bound on the number of components and as $L \to \infty$, the marginal distribution of this finite model approaches the DP [76, 91]. This weak limit approximation is used for its computational efficiency in this work. With this approximation, (4.21) simplifies to

$$\begin{aligned} \beta \mid \gamma &\sim Dir(\frac{\gamma}{L}, \ldots \frac{\gamma}{L}) \\ \pi_j \mid \alpha, \beta &\sim Dir(\alpha\beta_1, \ldots \alpha\beta_L) \\ \varphi^c_j \mid \lambda, \pi_j &\sim Dir(\lambda\pi_{j1}, \ldots \lambda\pi_{jL}) \end{aligned} \tag{4.29}$$

The prior induced by HDP ensures that only a subset of L states are used. The L value is usually set to a large number. Given this truncated approximation, the standard forward-backward algorithm [32] is employed to sample the hidden state sequences.

4.5.2 Sampling State Transitions

The sampler is initialized by drawing the initial values of the parameters from their respective priors. For a training example x^n whose $y^n = c$, given the state transitions $\{\varphi^c\}_{j=0,k=1}^{L,L}$, the component means $\{\Lambda_k^c \mu_k + \rho_k^c\}_{k=1}^{L}$ and the covariances $\left\{\Lambda_k^c \Sigma_k {\Lambda_k^c}^T\right\}_{k=1}^{L}$, the hidden state sequence is sampled from

$$p(z_t^n = k) \propto \varphi_{z_{t-1}^n k}^c \; m_{t+1,t}(k) \; \mathcal{N}(x_t^n;\, \Lambda_k^c \mu_k + \rho_k^c, \Lambda_k^c \Sigma_k {\Lambda_k^c}^T) \tag{4.30}$$

Here $m_{t,t-1}(k)$ is the HMM backward message that is passed from z_t^n to z_{t-1}^n and is determined recursively as

$$m_{t,t-1}(k) = \sum_{j=1}^{L} \varphi_{kj}^c \; m_{t+1,t}(j) \; \mathcal{N}(x_t^n;\, \Lambda_j^c \mu_j + \rho_j^c, \Lambda_j^c \Sigma_j {\Lambda_j^c}^T) \tag{4.31}$$

Let $n^c \in \mathbb{Z}^{(L+1)\times L}$ be a matrix of counts computed from the sampled hidden state sequences with n_{jk}^c being the number of transitions from states j to k for class c. The notation n_{jk} Is used to denote the number of transitions from j to k for all the classes and $n_{\cdot k}$ to denote the number of transitions to k. The scaling factor ω_j^c in (4.26) is used as the discriminative term and is set as

$$\omega_{jk}^c = 1/\varepsilon_0 \left[\frac{n_{jk}^c - n_{jk} + D}{\sum_{k'} n_{jk'}^c - n_{jk'} + D}\right] \tag{4.32}$$

Intuitively, the weight for a state k will be higher if there are fewer transitions to this state from classes other than c. Here ε_0 is a prior that controls the importance of the scaling factor and D is a sufficiently large constant to ensure that the scaling factor is positive. The posteriors are then sampled as

$$\begin{aligned} \beta \mid \gamma, \bar{m} &\sim Dir(\frac{\gamma}{L} + \bar{m}_{\cdot 1}, \ldots \frac{\gamma}{L} + \bar{m}_{\cdot L}) \\ \pi_j \mid \alpha, \beta, \bar{n} &\sim Dir(\alpha\beta_1 + \bar{n}_{j1}, \ldots \alpha\beta_L + \bar{n}_{jL}) \\ {\varphi_{jk}^c}' \mid \lambda, \pi_j, \omega_j^c, n^c &\sim Gamma\left(\lambda\pi_{jk} + n_{jk}^c, e^{-\omega_{jk}^c}\right) \end{aligned} \tag{4.33}$$

$$\varphi_{jk}^{c} = \frac{\varphi_{jk}^{c}{}'}{\sum_{k'=1}^{L} \varphi_{jk'}^{c}{}'}$$

Here $\bar{m}, \bar{n}$ are auxiliary count matrices that are sampled from the class specific matrices n^c. In the Chinese restaurant metaphor, these matrices correspond to the number of tables across the franchise serving a dish and the number of tables across sections in a restaurant serving a dish. These auxiliary matrices and the hyper parameters γ, α, λ are sampled in the standard way as outlined in [44].

4.5.3 Sampling Component Parameters

The shared parameters are sampled first, followed by the class specific parameters. Further the posteriors are sampled one component at a time. Let the set of observations belonging to class c and assigned to hidden state k be $\mathcal{X}_k^c = \{x_t^n \in X : z_t^n = k \wedge y^n = c\}$ with $\mathcal{X}_k = \{\mathcal{X}_k^c\}_{c=1}^{C}$. The mean and covariance parameters that are shared across the classes use conjugate priors and the posteriors can be computed using the standard closed form updates discussed in Appendix C.3 as

$$\Sigma_k \mid \nu_0, \Delta_0, \mu_k, \mathcal{X}_k \sim IW(\bar{\nu}_k, \bar{\Delta}_k)$$

$$\mu_k \mid \mu_0, \Sigma_0 \Sigma_k, \mathcal{X}_k \sim \mathcal{N}(\bar{\mu}_k, \bar{\Sigma}_k)$$

where

$$\begin{aligned} \bar{\nu}_k &= \nu_0 + |\mathcal{X}_k| \\ \bar{\Delta}_k &= \Delta_0 + \sum_{x_n \in \mathcal{X}_k} (x_n - \mu_k)(x_n - \mu_k)^T \\ \bar{\Sigma}_k &= ({\Sigma_0}^{-1} + |\mathcal{X}_k| {\Sigma_k}^{-1})^{-1} \\ \bar{\mu}_k &= \bar{\Sigma}_k \left({\Sigma_0}^{-1} \mu_0 + \Sigma_k \sum_{x_n \in \mathcal{X}_k} x_n \right) \end{aligned} \tag{4.34}$$

For the transform parameters, the posterior must be sampled from (4.23) after defining the form of $p\left(\theta^{\setminus c} \mid \theta^c, X, \theta^s\right)$. There are several choices for the discriminative term. One option is to set it based on the distance between the distributions of component parameters. If the distribution distances are large, the parameters are well separated and this will result in a larger margin for the classifier decision boundary. For the state k of class c whose transform parameters need to be sampled, the density is set as

$$p\left(\theta^{\setminus c} \mid \theta^c, \theta^s\right) = \tag{4.35}$$

$$\prod_{c' \in \setminus c} \prod_{k'=1}^{L} \exp\left\{-\xi_0 \max\left(0,\ \zeta_0 - D\left(\mathcal{N}(\bar{\mu}_k^c, \bar{\Sigma}_k^c) \,||\, \mathcal{N}\left(\bar{\mu}_{k'}^{c'}, \bar{\Sigma}_{k'}^{c'}\right)\right)\right)\right\}$$

where

$$\bar{\mu}_k^c = \Lambda_k^c \mu_k + \rho_k^c$$
$$\bar{\Sigma}_k^c = \Lambda_k^c \Sigma_k {\Lambda_k^c}^T$$

Here $D(P||Q)$ measures the similarity between two distributions P and Q, ζ_0 is a prior that specifies the minimum separation distance and ξ_0 is a constant that controls the overall importance of the discriminative term. Since normal distributions are used, the Hellinger or Bhattacharya distance [72] can be used as a similarity measure. Intuitively, the distribution of a component k from class c that we wish to sample is compared with all the competing classes and their corresponding components. If the distance is less than a pre-specified minimum separation, then the pdf value will be lower and perhaps the sample is inappropriate. The discriminative term specified in (4.35) is computationally simple since it does not involve the training examples and instead uses the sufficient statistics.

Another option for the discriminative term is to use the likelihood of observations. The idea here is to ensure that the Gaussian pdf value of an observation from class c assigned to a component k is larger than the pdf value of competing classes and their corresponding components.

$$p\left(\theta^{\setminus c} \,\middle|\, \theta^c, \theta^s, X, Y\right) = \prod_{x_t^n \in X} \exp\left\{-\xi_0 \max\left(0,\ \zeta_0 - \left(\mathcal{N}\left(x_t^n;\ \bar{\mu}_{z_t^n}^{y_n}, \bar{\Sigma}_{z_t^n}^{y_n}\right) - \max_{\substack{c':y_n \neq c \\ k'=1:L}} \mathcal{N}\left(x_t^n;\ \bar{\mu}_{k'}^{c'}, \bar{\Sigma}_{k'}^{c'}\right)\right)\right)\right\} \tag{4.36}$$

where

$$\bar{\mu}_k^c = \Lambda_k^c \mu_k + \rho_k^c$$
$$\bar{\Sigma}_k^c = \Lambda_k^c \Sigma_k {\Lambda_k^c}^T$$

If the model is considered as a single component Gaussian instead of an HMM with Gaussian mixtures, then (4.36) tends to make the pdf value for the correct class greater than the pdf value of competing classes. The above discriminative term can be treated as an approximation to the empirical error rate and ζ_0 offers the flexibility for a soft margin.

By plugging in (4.35) or (4.36) into (4.23), the posterior distribution for the transform parameters is obtained. The term $\Lambda_k^c | \rho_k^c, \mu_k, \Sigma_k$ is sampled followed by $\rho_k^c | \Lambda_k^c, \mu_k, \Sigma_k$. Since the priors for both these variables are Gaussian distributions, elliptical slice sampling, as specified in Section 4.4.2, can be used to obtain the posterior updates. Note that for a Gaussian prior with non-zero mean, a shift must be performed to produce zero mean but this shift can be done trivially.

4.5.4 Prediction

The label for a test sequence $\hat{x}$ is determined during prediction. Given the parameters corresponding to a posterior sample, the class conditional likelihood of the observation is used to obtain the class label as shown in (4.37). The likelihood is obtained using the standard HMM forward-backward algorithm. This process is repeated for all the posterior samples and the final label is selected based on the mode.

$$\hat{c} = \underset{c}{\operatorname{argmax}}\, p(\hat{x}|c, \theta^s, \theta^c) \tag{4.37}$$

Table 4.1: Posterior Inference Algorithm

Input: Training observations with their corresponding class labels. **Output:** Samples of posterior parameters.
1. Sample the initial values $\beta, \pi, \mu_{1..L}, \Sigma_{1..L}, \varphi^{1..C}, \rho_{1..L}^{1..C}, \Lambda_{1..L}^{1..C}$ from their respective hyper parameters. 2. Sample hidden state sequences z_t^n using HMM forward backward algorithm as per (4.30). 3. For all classes, compute the matrix of counts n^c from the sampled hidden states. 4. For all classes and all states, determine the scaling factor ω_{jk}^c as per (4.32). 5. Sample the top level stick breaking weights β according to (4.33) using an auxiliary count matrix. 6. Sample the state specific stick breaking weights π for all states according to (4.33) using an auxiliary count matrix. 7. Sample the class specific stick breaking weight φ for all classes and all states according to (4.33). 8. For all components, sample the shared covariance Σ_k and then the mean μ_k as per (4.34). 9. For all classes and for all components, use (4.35) or (4.36) in (4.23) and sample the transform parameters ρ_k^c, Λ_k^c using elliptical slice sampling. 10. Sample the hyper parameters. 11. Repeat from step (2) to collect more samples.

4.6 Experiments

The experiments for action recognition are conducted on the publicly available UTKinect-Action [11] and MSR Action 3D [74] datasets. The datasets contain various actions performed by different human subjects. Each action involves only one individual and there are no objects involved when an action is performed. All these datasets use an infrared camera to capture the depth image sequences as outlined in Appendix A.1. The datasets also contain annotated 3D joint positions of the subjects. These joint positions were estimated from the depth image sequence as outlined in Appendix B. The estimated joint positions may contain errors and the experiments are conducted with these noisy joint positions.

4.6.1 UTKinect-Action dataset

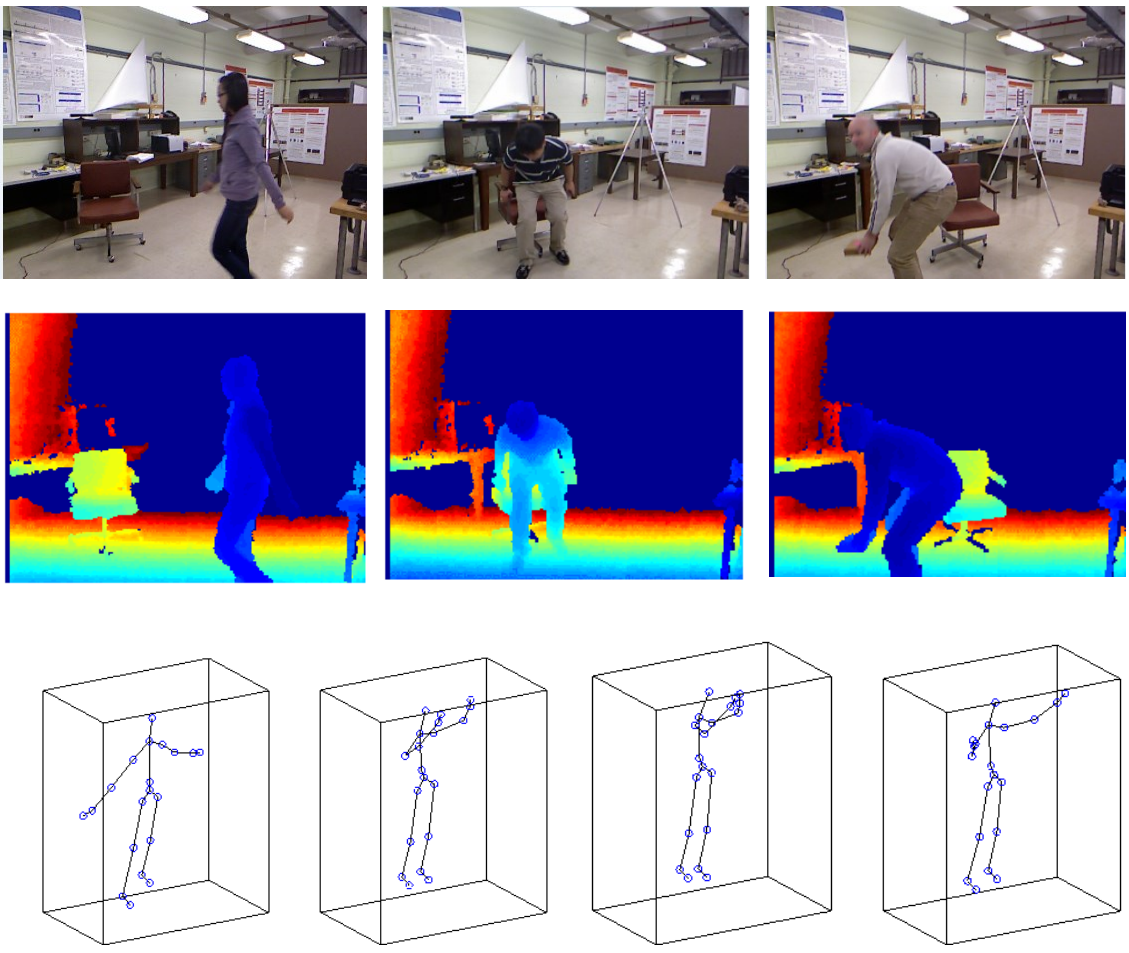


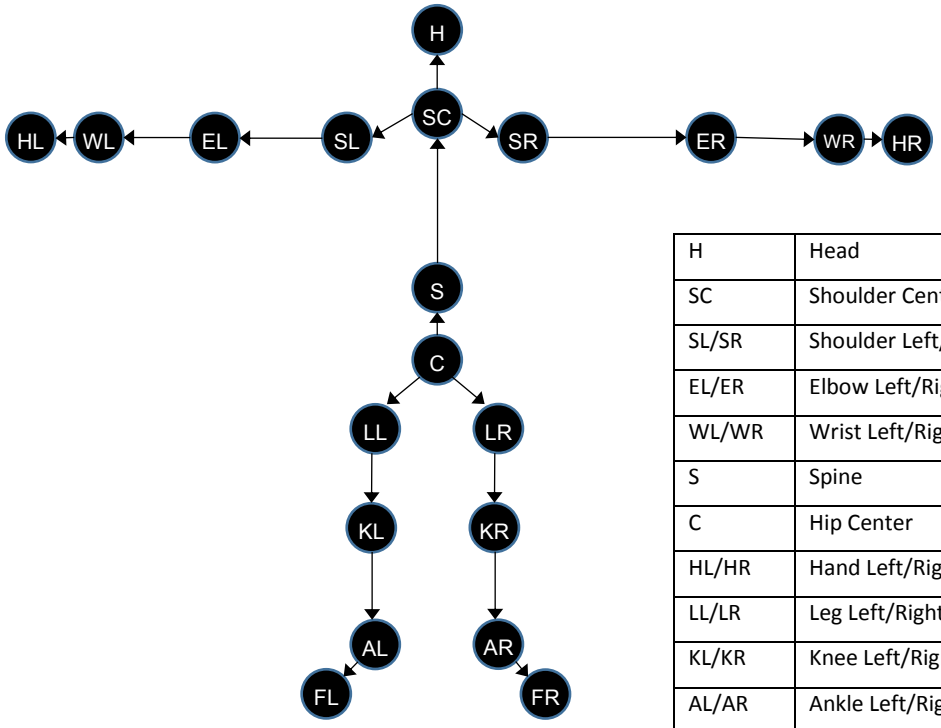
Figure 4.4: UTKinect-Action dataset samples. The top row shows the RGB image for the actions *walk, sit-down* and *pick-up* from the UTKinect-Action [11] dataset. The middle row shows the depth image corresponding to these RGB images for the same set of actions. The last row shows a sequence of 3D skeletal joint positions for the *wave* action. The information in the joint positions is used for classifying the actions.

The videos in the UTKinect-Action [11] dataset were captured using a single stationary Microsoft Kinect camera. The RGB, depth and the skeleton joint locations were all recorded in a synchronized manner. The final frame rate is about 15 frames per second. The resolution of the depth map is 320x240 and the depth range is 4 to 10 feet. Altogether the data set contains 6220 frames of 200 action sequences with an average frame length of 32 per sequence.

The dataset contains the actions *walk*, *sit-down*, *stand-up*, *pick-up*, *carry*, *throw*, *push*, *pull*, *wave* and *clap-hands*. Each action was performed by ten different subjects with one of the subject being a female. The actions were all performed indoor. The action sequences were taken from different views and there are significant variations in the realization of the same action. Further, occlusions and body parts out of view add to the difficulty of this dataset. A sample of actions from this dataset is shown in Figure 4.4.

Each depth image frame contains 20 joint positions with coordinates (x, y, z) in a world coordinate frame. The pairwise relative joint positions within a frame are used as features. The relative positions $P_i - P_j$ of 19 joint position pairs (P_i, P_j), where $P_i, P_j \in (x, y, z)$, are used. The skeleton hierarchy that determines these pairs is pre-defined according to Figure 4.5. Some examples of the joint position pairs are (Head, Shoulder Centre) and (Left ankle, Left feet). The total number of features is 57 per frame. By using relative joint positions, invariance to uniform translation of the body is ensured.

The experiments are conducted for the challenging setting in which the subject is seen for the first time during prediction. 60% of the subjects were used for training while the rest of the subjects were used for testing. Following Bayesian hierarchical modelling, the hyper parameters have weakly informative hyper priors. The concentration parameters were all given a vague gamma prior similar to [44, 76] ensuring that the initial choice of the concentration parameters is not important. In the first iteration during posterior inference, all the hyper parameters are initialized from their respective priors. All the other parameters are sampled from their respective prior distributions. The hyper parameters are re-sampled after each sampling iteration. The first 500 samples were discarded and a total of 100 samples were collected. When sampling the posterior for the Gaussian distribution parameters and the concentration parameters, a further burn-in period of 50 iterations was used. The posterior inference procedure uses the forward-backward algorithm which has a time complexity of $O(TK^2)$ where T is the length of the sequence and K is the number of states. To verify convergence, the change in the number of instantiated states and the difference in the Gaussian distribution parameter values between iterations were checked.



H	Head
SC	Shoulder Center
SL/SR	Shoulder Left/Right
EL/ER	Elbow Left/Right
WL/WR	Wrist Left/Right
S	Spine
C	Hip Center
HL/HR	Hand Left/Right
LL/LR	Leg Left/Right
KL/KR	Knee Left/Right
AL/AR	Ankle Left/Right
FL/FR	Feet Left/Right

Figure 4.5: Skeleton Hierarchy. The predefined skeleton structure [75] used for defining the joint position pairs is shown. The arrows indicate the parent-child relationship that determine the joint position pairs. For example, the Left Ankle and Left Feet structure define the joint position pair (AL, FL).

In order to verify the efficacy of the model, additional experiments are conducted with the parametric HMM and a nonparametric HMM. The multi-level nonparametric HMM is then evaluated and finally the results are presented for the full model in which the parameters are learnt in a discriminative manner. The results are reported using the standard performance measures for a classification problem namely precision, recall and accuracy. The precision for a class is the ratio of the correct predictions (true positives) to all the positive predictions (true positives and false positives), while recall is the ratio of the correct predictions to all the members of the class (true positives and false negatives). The accuracy of the classifier is the ratio of the correctly classified instances (true positives and true negatives) to the total number of instances.

Parametric HMM

A classifier is trained, independently for each class, based on the classical HMM. The standard Baum-Welch algorithm [32] is used for learning the HMM parameters. Since the number of states must be specified a-priori for parametric HMMs, different numbers of states for each class

are tried during training. In the absence of priors, an additional clustering step with K-Means is performed to estimate the initial values of the transition matrix and the mean and covariance parameters. During testing, a test example is evaluated against all the classes and the class with the largest likelihood is selected as the predicted class. The observed best classification accuracy was ***56.8%***. The summary of classification results for HMM is presented in Table 4.2.

Table 4.2: Classical Parametric HMM classification results

Number of States	Accuracy (%)	Precision (%) (Average across classes)	Recall (%) (Average across classes)
3	49.1	52.0	49.8
5	47.7	60.1	48.4
7	52.8	64.2	53.3
10	56.8	65.3	58.4
15	55.3	70.1	55.8

Nonparametric HMM

A HDP-HMM based classifier is also trained, independently for each class as before. The upper bound on the number of states is set to 20 with the weak limit approximation discussed in Section 4.5.1 being used. The number of states is automatically learnt from the data for HDP-HMM unlike the parametric HMM. Figure 4.6 shows the total number of states for the different action classes in a sample collected during training. In an equivalent parametric HMM, a tedious and ad hoc model selection step for each class must be run individually because the optimum number of states varies between classes. This advantage of automatic state inference with HDP-HMM is reflected in an improved classification accuracy of ***74.1%***. The results are shown in Table 4.3.

Table 4.3: HDP-HMM classification results

Action	Precision (%)	Recall (%)
walk	100	87.5
sit-down	50	50
stand-up	66.6	100
pick-up	100	50
carry	77.7	100
throw	100	50
push	71.4	62.5

pull	66.7	75
wave	85.7	75
clap-hands	50	75

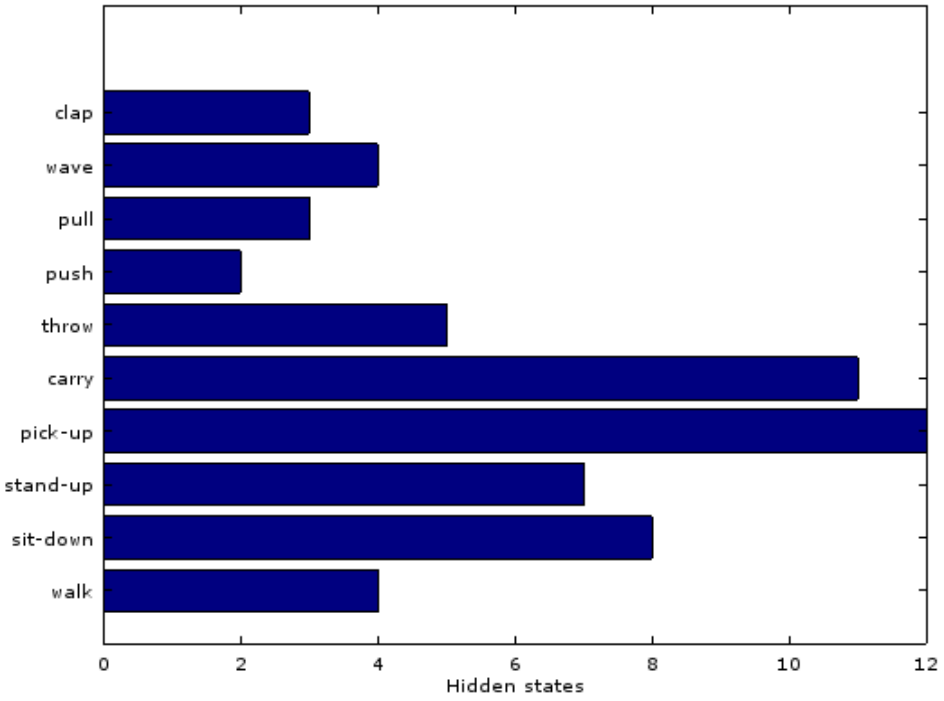


Figure 4.6: Hidden states plot to show the number of hidden states active for different action classes in a sample collected during training. An active state is one to which at least one observation is assigned.

Multi-level HDP-HMM

The results are evaluated on the two-level HDP-HMM excluding the discriminative criteria. In this method, examples from all the classes are used during parameter estimation. Thus it allows sharing of parameters across classes and enables semi-supervised learning. In order to exclude the discriminative conditions for the state transitions, the scaling factor ω_j^c is simply set to zero. This is equivalent to sampling φ_{jk}^c (probability of transitioning to state k given the current state is j for a class c) as per equation (4.18) instead of (4.34). Similarly for the class specific transformation parameters, $p(\theta^{\setminus c})$ is set to be a constant in equation (4.25) thereby excluding the discriminative conditions. The classification results are shown in Table 4.4. The accuracy is ***75.3%***. These results confirm that sharing parameters across classes doesn't make the classification any worse. The lack of a big increase in accuracy when compared with HDP-HMM is interpreted as an indication that there is a need for an additional discriminative condition. In addition, the smaller number of training examples in this dataset could be a factor. Nevertheless, this technique provides a way to learn parameters in situations where unlabelled examples can be incorporated.

Table 4.4: Multi-level HDP-HMM classification results

Action	Precision (%)	Recall (%)
walk	100	50
sit-down	63.6	87.5
stand-up	88.8	100
pick-up	80	50
carry	70	100
throw	100	50
push	63.6	87.5
pull	80	50
wave	75	75
clap-hands	58.3	87.5

Multi-level HDP-HMM with Discriminative Learning

Finally, the results are evaluated on the two-level HDP-HMM including the discriminative conditions. The upper bound on the number of states is set as 25 and the parameters are all initialized from their respective prior distributions as outlined above. The hyper parameters corresponding to the Gaussian distribution parameters were initialized from the empirical mean and the empirical covariance itself. The confusion matrix for the classification results is shown in Figure 4.7. The diagonal entries show the percentage of true positives and the off diagonal entries show the percentage of misclassified instances. The reported overall classification accuracy is ***82.7%***.

Discussion

It is evident from the confusion matrix that the recognition rate is good for most actions. However, there are a few misclassifications. For example, the classifier has incorrectly classified the action *pick-up* as *sit-down* in some instances. These misclassifications are attributed to the fact that these actions involve very similar motion patterns. Further, there are variations in the way the same action is performed by different subjects in this dataset. This also affects the classifier performance.

A summary of the classification results for this dataset is provided in Table 4.5. The HDP-HMM improves the classification accuracy when compared with a parametric HMM since the number of states are automatically determined. The multi-level HDP-HMM allows sharing the parameters across classes and it does not make the classification any worse. The introduction of

discriminative conditions on the multi-level HDP-HMM has improved the classification results. Some comparison with the other results in the literature is also provided. The accuracy is better than [78] while it is less than the results reported in [11] and [79]. In the evaluation method used here, 40% of the subjects are excluded and the instances of these subjects are unseen during prediction. This makes classification more difficult than in the alternative arrangement, in which training samples for all the subjects are included and only specific samples for each subject are excluded. In [11] and [79] a Leave-One-Out-Cross-Validation method is used. In a particular iteration, they use only one observation sequence for testing and the rest of the observation sequences are used for training. This procedure is repeated to include all the observation sequences for testing and finally the average accuracy across iterations is reported. In the experiments here, the training and test examples are completely separated and more challenging cross subject evaluation is performed. This tests the variations of actions performed by different subjects in a more realistic manner. In addition, the method proposed here compares favourably in complexity with the parametric equivalents. For example, in [11] a parametric HMM is trained for each action class. Even though both parametric and nonparametric HMMs have the same time complexity $O(TK^2)$ during inference, in the model proposed here we need not train separate HMMs for each action class. Further, it is also not necessary to train multiple HMMs for each action class in order to select an action class's state cardinality.

Table 4.5: Summary of classification results for the UTKinect-Action dataset.

	Method	**Accuracy %**
This work	Parametric HMM	56.8
	Nonparametric HMM	74.1
	Multi-level HDP-HMM (Generative Learning)	75.3
	Multi-level HDP-HMM (Discriminative Learning)	***82.7***
Previous Works	STIP [78]	81.0
	HOJ3D [11]	90.9
	Shape Analysis [79]	91.5

Figure 4.7: UTKinect-Action dataset results. The classification results for the dataset shown using a confusion matrix. The diagonal entries show the percentage of correct predictions for a particular class.

4.6.2 MSR Action 3D dataset

The videos in the MSR Action 3D [74] dataset were captured at about 15 frames per second by a depth camera that acquires depth through the structured infrared light principle discussed in Appendix A.1. The resolution of the depth map is 320x240 and there are 23797 frames of depth maps with 567 depth map sequences in total.

The dataset contains the actions *high-arm-wave, horizontal-arm-wave, hammer, hand-catch, forward-punch, high-throw, draw-x, draw-tick, draw-circle, hand-clap, two-hand-wave, side-boxing, bend, forward-kick, side-kick, jogging, tennis-swing, tennis-serve, golf-swing* and *pickup-throw*. These actions appear in the context of interactions with a game console and involve various movements of the arms, legs, torso and their combinations. The actions were performed by seven different subjects and were repeated two or three times. The subjects were facing the camera when the actions were recorded. Sample actions from this dataset are shown in Figure 4.8.

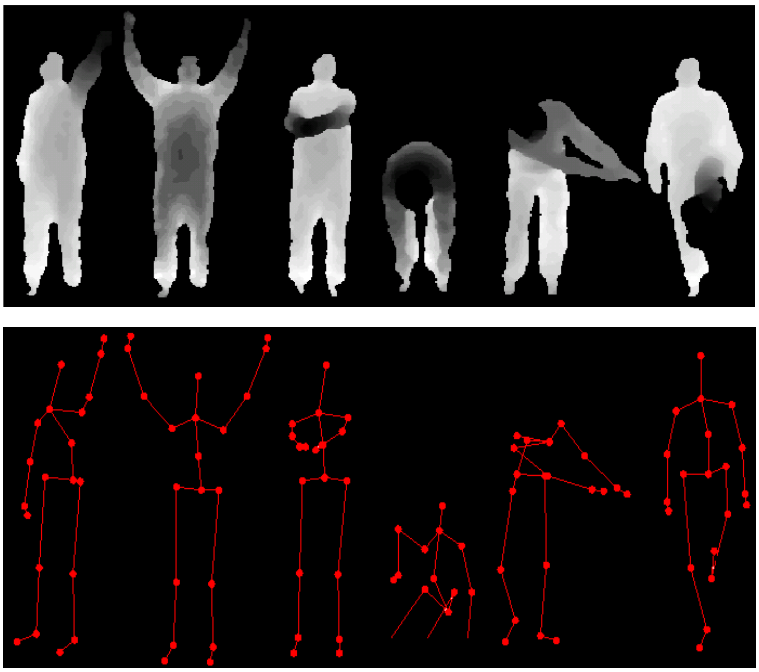


Figure 4.8: MSR Action3D dataset samples. Actions *high-arm-wave, two-hand-wave, hand-clap, bend, golf-swing* and *forward-kick* from the MSR Action 3D [74] dataset are shown. *Top*: The depth image representation of the poses encountered when performing these actions. *Bottom*: The annotated joint positions represented in a skeleton format. The circles denote the joint positions.

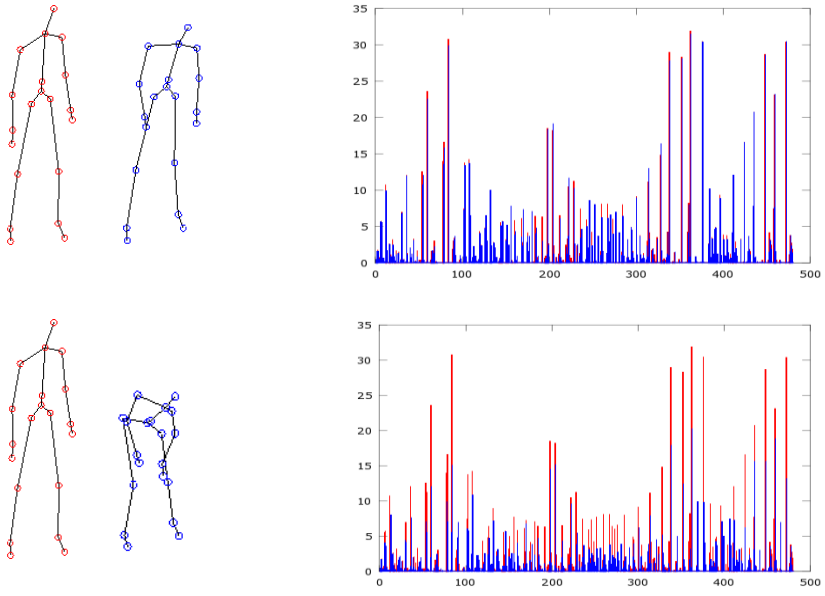


Figure 4.9: Feature descriptor visualization. *Top*: For two similar poses, the descriptor values appear overlapped. *Bottom*: For two different poses, the descriptor values appear less overlapped.

As before, the information in the sequence of joint positions is used to define the features. There are 20 joint positions in each depth frame. The relative position of a joint to all the other joints

in the current frame and previous frame is used in order to determine the features. Further, instead of a single 3D coordinate vector, three 2D values representing the projection of a relative position on the orthogonal (xy, yz, xz) Cartesian plane are used. Let the set $\{P_i\}_{i=1}^{20} \in (x, y, z)$ denote the set of joint positions. For a joint i the features are

$$\begin{aligned} f_i(x,y) &= \{(P_i^x - P_j^x, P_i^y - P_j^y) \quad \forall\, j \in P(t-1), P(t), P(t+1) \wedge i \neq j\} \\ f_i(y,z) &= \{(P_i^y - P_j^y, P_i^z - P_j^z) \quad \forall\, j \in P(t-1), P(t), P(t+1) \wedge i \neq j\} \\ f_i(x,z) &= \{(P_i^x - P_j^x, P_i^z - P_j^z) \quad \forall\, j \in P(t-1), P(t), P(t+1) \wedge i \neq j\} \end{aligned} \tag{4.38}$$

where $P_i^x(t)$ is the x co-ordinate of the i^{th} joint position at time t. The gradients $f_i(x, y)$ are then assigned to a histogram of 8 bins based on the direction, with the sums of gradient magnitudes as bin values. This technique is very similar to the Histogram of Oriented Gradients (HOG) [77]. Repeating the step for $f_i(y, z)$ and $f_i(x, z)$, there are now 24 bins for each joint. Concatenating the bins for all the joints yields a descriptor of length 480 for every frame. Figure 4.9 provides a visualization of the feature descriptors, highlighting that for dissimilar poses the descriptor values are different. Finally, Principle Component Analysis (PCA) [35] is used to project the descriptor to a lower dimensional vector space. The subset of vectors that captures at least 90% of the total variance is used.

When evaluating this dataset, the twenty actions are divided into three subsets each containing 8 actions as shown in Table 4.6. This is a common practice [11, 74] for this dataset since there are many overlapping actions. The experiments are conducted for the challenging setting in which the instance of a subject is not used during training and is encountered for the first time during prediction. 60% of the subjects were used for training while the rest of the subjects were used for testing. The parameters are initialized in the same manner described in Section 4.6.1.

Table 4.6: Actions organized into three different action sets in the MSR Action3D dataset.

Set 1	Set 2	Set 3
horizontal-arm-wave	*high-arm-wave*	*high-throw*
hammer	*hand-catch*	*forward-kick*
forward-punch	*draw-x*	*side-kick*
high-throw	*draw-tick*	*jogging*
hand-clap	*draw-circle*	*tennis-swing*
bend	*two-hand-wave*	*tennis-serve*
tennis-serve	*forward-kick*	*golf-swing*
pickup-throw	*side-boxing*	*pickup-throw*

The structure of the action states is explored in Figure 4.10. The action states sampled during an iteration for the action *horizontal-arm-wave* are shown under A, B and C. These correspond to different subjects for the same action. The action states for *bend* and *pickup-throw* are shown in D and E. It can be seen that some of the states are shared across the action classes. For instance, the state 23, which corresponds to the pose at the start and the end, is shared across all the three classes and the state 19, which corresponds to the pose while bending, is shared by the *bend* and *pickup-throw* actions. There are also some unique states (e.g. 9, 16 for A, 4 for B and 7 for C) that represent the poses that are specific to these actions. This behaviour validates the information sharing across the classes and the discriminative structure that is learnt.

The evaluation results for the MSR Action3D dataset under the full model is shown in Figure 4.11. The three confusion matrices in the figure correspond to the above three sets of actions. As before, the diagonal entries show the percentage of correctly predicted examples. The overall classification accuracy is 81.2% for the first set, 78.1% for the second set and 90.6% for the third set.

Discussion

As evident from the confusion matrices, the classifier has predicted the action class labels correctly for the most part. However, there are some misclassifications. As examples, some *bend* actions are classified as *pickup-throw*, *hand-clap* actions as *tennis-serve* and *hand-catch* actions as *arm-wave*. These actions involve very similar motion patterns. The *draw-x* and *draw-tick* actions are particularly challenging to classify and have less labelling accuracy compared to other actions.

Table 4.7 provides a summary of the classification results for this dataset. While the confusion matrix in Figure 4.11 showed the classification accuracy for the action sets A, B and C separately, the accuracy is pooled across all the three sets and shown in the fourth row of Table 4.7. The behaviour of the classical HMM, nonparametric HMM and the multi-level HDP-HMM is in line with expectations as discussed above for the UTKinect-Action dataset. The comparisons with the other work in the literature show that the results are better than those reported in [11] and [78]. The results are less accurate when compared with the state-of-the-art results reported in [80]. The work in [80] uses the depth information in addition to the skeleton information. The depth information is particularly useful in the MSR Action 3D dataset because there are many corrupted joint positions. The inclusion of depth channel may benefit action recognition in this dataset.

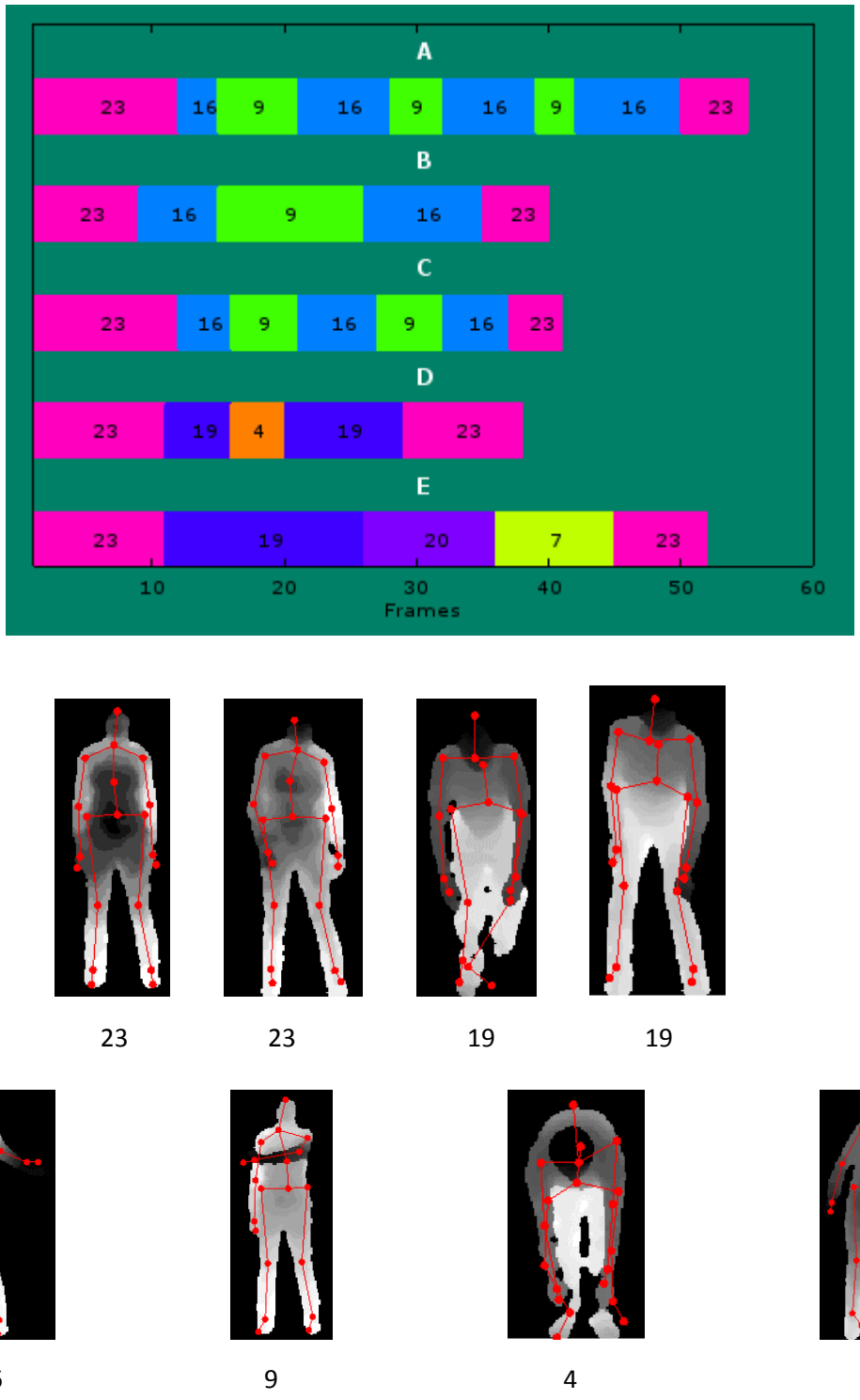
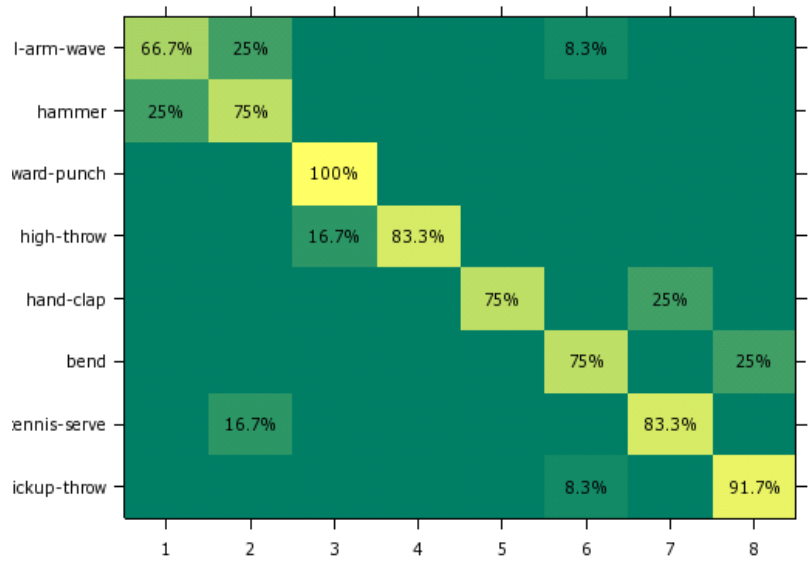
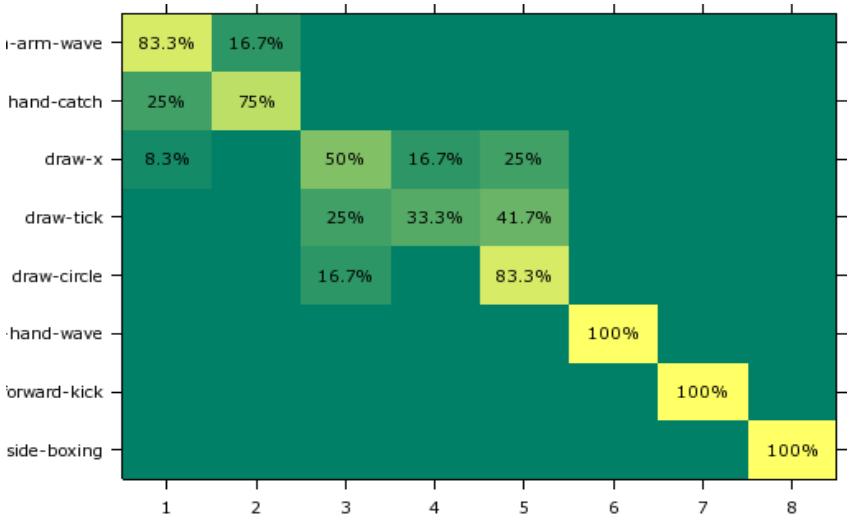


Figure 4.10: Action states. *Top*: The sampled action states for different instances are shown in a colour coded plot. A, B and C correspond to the action *horizontal-arm-wave* while D and E correspond to actions *bend* and *pickup-throw*, respectively. *Middle*: The poses corresponding to the action states shared between the classes. *Bottom*: The poses corresponding to the action states that are unique to the classes.

(a)


l-arm-wave
hammer
ward-punch
high-throw
hand-clap
bend
ennis-serve
ickup-throw
66.7%
25%
8.3%
25%
75%
100%
16.7%
83.3%
75%
25%
75%
25%
16.7%
83.3%
8.3%
91.7%
1
2
3
4
5
6
7
8

(b)


-arm-wave
hand-catch
draw-x
draw-tick
draw-circle
hand-wave
orward-kick
side-boxing
83.3%
16.7%
25%
75%
8.3%
50%
16.7%
25%
25%
33.3%
41.7%
16.7%
83.3%
100%
100%
100%
1
2
3
4
5
6
7
8

(c)

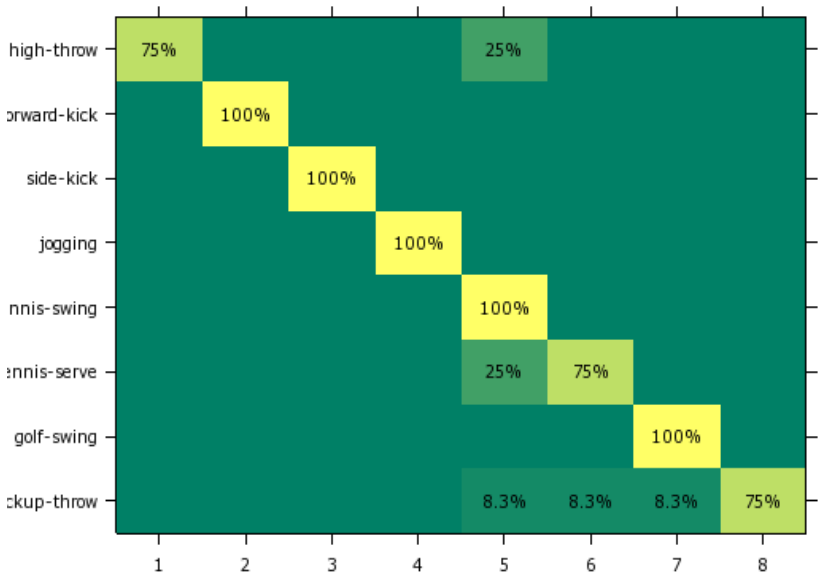

high-throw
orward-kick
side-kick
jogging
nnis-swing
ennis-serve
golf-swing
ckup-throw
75%
25%
100%
100%
100%
100%
25%
75%
100%
8.3%
8.3%
8.3%
75%
1
2
3
4
5
6
7
8

Figure 4.11: MSR Action 3D dataset results. The classification results for the dataset are shown using a confusion matrix for the three different sets of action groupings. (a), (b) and (c) correspond to Set 1, Set 2 and Set 3 respectively. The diagonal entries show the percentage of correct predictions for a particular class.

Table 4.7: Summary of classification results for the MSR Action 3D dataset.

	Method	Accuracy %
This work	Parametric HMM	48.7
	Nonparametric HMM	75.3
	Multi-level HDP-HMM (Generative Learning)	76.8
	Multi-level HDP-HMM (Discriminative Learning)	***83.3***
Previous Works	STIP [78]	80.8
	HOJ3D [11]	78.9
	JAS [80]	94.8

4.7 Conclusion

This chapter proposed a discriminative nonparametric HMM for classifying actions that occur in depth videos. It addresses an important drawback of classical HMMs, namely the need to specify the number of hidden states in advance. The proposed construction enables information sharing across the action classes with the use of a single HDP-HMM. However, this single HDP-HMM is not suitable for classification tasks. Hence the HDP-HMM is extended to include another level that captures variations specific to each class using additional parameters. The distributions for these parameters are formulated as transformations from a shared base distribution. A tractable inference procedure that learns the parameters in a discriminative manner is also provided. The construction facilitates the use of unlabelled examples and is applicable for the classification of a wide variety of sequential data.

The experiments conducted on benchmark video datasets demonstrate the usefulness of the proposed method in classifying human actions. The model provided here has produced good classification results. However, there are instances in which the classification accuracy is low. This is particularly evident for actions that involve very similar motion patterns. One mechanism to address this problem would be to include the objects that a subject interacts with when performing an action. This additional information would be helpful in distinguishing actions that have similar motion trajectories. Further, the inclusion of additional information from the RGB and depth channels may benefit the model.

A limitation of the HMM in general is that it cannot capture complex temporal dependencies that are usually encountered in real-world data. For example, it is difficult to represent sequential data that has an inherent hierarchical structure using the HMM. The model proposed in the next chapter addresses this problem by using an extension to the HMM. It is also possible to let each state depend on a number of previous states in an HMM. The use of these higher order Markov chains allows the capture of long term dependency in the data. However, inference may be more difficult in such models. The HMM uses a joint distribution over the class labels and the observations. Directly modelling the conditional distribution of the class labels given the observations may be more appropriate for classification problems, as discussed in Chapter 6. In spite of these limitations, the nonparametric HMM serves as a building block for constructing more sophisticated models.

5. Supervised nonparametric Hierarchical HMM

A nonparametric hierarchical HMM is proposed in this chapter for classifying human activities. An activity is typically composed of a set of actions with an action in turn being composed of a set of poses. The proposed model exploits this hierarchical structure inherent in the activities. Unlike classical hierarchical HMM, the nonparametric variant described here does not require the number of hidden states corresponding to the actions and poses to be fixed in advance. The generic formulation proposed here is applicable for the classification of any sequential data that exhibits a hierarchical structure.

The chapter begins with an overview of the proposed approach in Section 5.1 and is followed in Section 5.2 by a description of the nonparametric hierarchical HMM (H-HMM). The activity model uses a two level H-HMM with the bottom level characterizing the granular poses and the top level characterizing the coarser actions. The relationship between the actions and the activity labels are captured using multinomial logistic regression. Section 5.3 describes this model structure, which is suitable for supervised classification problems. The posterior inference procedure that ensures the alignment of actions from activities with similar labels is discussed in Section 5.4. Experiments are conducted on depth videos to evaluate the model. The skeletal joint positions extracted from a depth image and the information in the depth image patches around the joint positions are used to define the features of an activity. The results for activity classification are presented in Section 5.5. The chapter ends with some concluding remarks in Section 5.6. Portions of this chapter have been published [194].

5.1 Overview

The action classification solution presented in the previous chapter addresses the recognition of simple actions. However, in many situations there is a need to reason at a cognitively higher level based on these simple actions. An activity, which is a sequence of actions, characterizes this higher level construct. As an example, the *jump, walk* and *run* actions are part of an *exercise* activity. Similarly a *rinse-mouth* activity is composed of *drink* and *spit* actions. We now have an organization in which the activities are composed of actions and the actions are composed of poses.

An additional benefit of this organization is the opportunity to share information across the activity classes. For many actions and activities that are related, the same pose may be present in multiple actions and the same action may be present in multiple activities. Hence it is beneficial to share the action and pose information between the activity classes. Such information sharing facilitates effective learning from a limited set of examples. Furthermore the activities can now be classified just from the action representations without explicitly taking into account the pose representations. This results in a simplified model.

The organization of an activity as a sequence of actions and an action in turn as a sequence of poses indicates a natural hierarchical structure. The canonical Hidden Markov Model (HMM) [32] that is widely used for representing sequential data is not sufficient to capture hierarchical structures. There are now two levels – a top level for the coarse action sequence and a bottom level for the granular pose sequence. Intuitively, we want an action to emit a sequence of poses and when the action completes, it must transition to a different action which will emit a different pose sequence. Hence for a given action state at the top level, there is a sub-HMM conditioned on this state that emits a pose sequence. The *hierarchical* HMM (H-HMM) captures such a multi-level structure by making each hidden state an autonomous probabilistic model [81]. It generates sequences by recursively activating the sub-states of a state. In this context, when an action state is activated, this state will use its own probabilistic model to emit a sequence of pose states with a pose state emitting an observation. There are multiple such probabilistic models corresponding to each action state. Although the H-HMM can be transformed into the standard HMM by using a large number of states, the inference is easier with H-HMM for hierarchical data. Further, the H-HMM provides a multi-scale interpretation of the data [83].

The subject performing an action often interacts with the objects in a scene. The pose representation in the previous chapter used the information in the skeletal joint positions to detect the actions. It may be useful to include the information about the objects as part of the pose space. If we do so, then there are now two types of poses – the skeleton pose, which is a particular arrangement of the joint positions and the object pose, which is a specific representation of an object associated with the action. As an example, the *drink* action may involve skeleton poses corresponding to *lifting an arm* and the object pose may be the representation of a *mug*. In order to include the object information, the H-HMM is extended with two independent Markov chains at the bottom level. One of the Markov chain corresponds to the skeleton poses and the other to the object poses. This factorized representation, provides the flexibility to include additional data channel if available in the model. The joint positions extracted from a depth image can be used for representing the skeleton poses as discussed

earlier. The information in the depth image patches around the joint positions provides a representation for the object poses. Figure 5.1 provides an illustration.

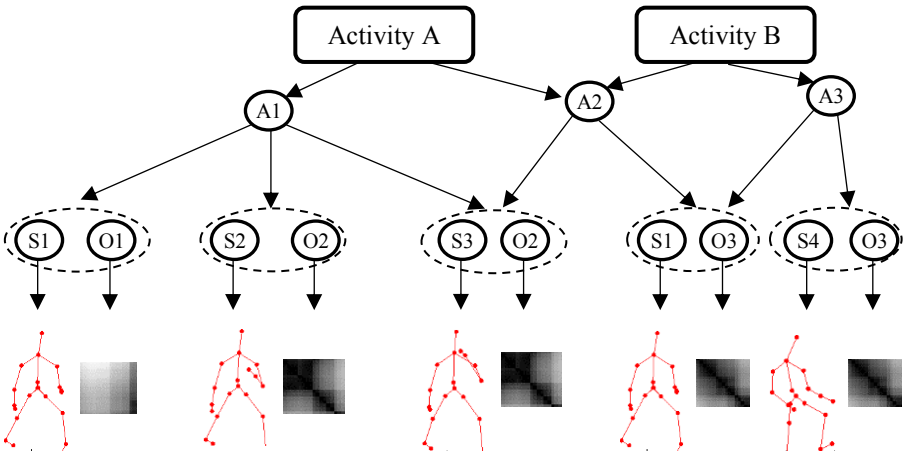


Figure 5.1: Activity Recognition Overview. Poses are learnt from observations and actions are learnt from poses. S1 to S4 represent the skeleton pose states, O1 to O3 the object pose states and A1 to A3 the action states. The skeleton poses are based on the 3D joint positions and the object poses are based on the depth image patch around the joint positions. The same pose can be present in multiple actions and different activities can contain the same action. The activities are classified only based on the action states.

In the standard H-HMM, the number of action states and pose states must be specified in advance. This is a problem in general with all variants of the classical parametric HMMs where the number of hidden states are fixed a-priori, even though for many applications this number is not known in advance. The usual technique of carrying out training using different choices for the number of states is even more difficult in the case of H-HMM since there are different states at multiple levels in the hierarchy. A better approach is to estimate the correct number of states automatically from the data. As discussed in the previous chapters, the Hierarchical Dirichlet Process (HDP) provides a nonparametric prior that allows an unbounded number of states to be used. This HDP prior is applied to the different levels in the H-HMM with the cardinality of the action and pose state space estimated from data.

In order to apply the nonparametric H-HMM for classification tasks, a separate model must be trained individually for each class with the prediction based on the learned class conditional density. However, it will not be possible to exploit the advantages of information sharing and model simplification discussed above using this approach. Hence a single H-HMM is used here with the actions and poses being shared across the activity classes. To enable classification,

multinomial logistic regression is used to capture the relationship between the activity labels and the actions. More specifically, the activity labels are regressed on the action states with the regression coefficients learned using a sparsity promoting Laplacian prior [82]. When sampling the action states during inference, the conditional likelihood of actions for a given activity label is incorporated. This ensures that the learnt actions not only explain the observations but also can predict the activity labels.

Contributions

The main contribution in this chapter is the definition of a novel factorized nonparametric H-HMM model integrated with multinomial logistic regression. A tractable inference procedure that is suitable for sequential data classification is also derived. The proposed model offers the following advantages:

(a) The hierarchical composition of actions and poses enables information sharing and model simplification.
(b) The nonparametric extension precludes the need for specifying a priori bounds on the number of states.
(c) The factorized state representation allows incorporation of multiple data channels.
(d) Unlabeled examples can be used thus promoting semi-supervised learning.
(e) The model is generic and can be applied for other hierarchical sequence classification problems.

5.2 Hierarchical HMM

In the HMM described in Section 3.1, x_t denotes an observation at a time instant t and z_t denotes the corresponding state. The HMM is parameterized by the state transition probability matrix π, where the probability of transitioning from state j to k is given by $P(z_t = k|z_{t-1} = j) = \pi_{jk}$, and the state specific observation density parameters $\{\theta_k\}_1^K$ with K being the number of hidden states.

In the hierarchical extension of the HMM [83], there are multiple levels $1 \ldots l \ldots L$ and there is a hidden state z_t^l corresponding to each level l at time t. A state at level l emits a sequence of states for level $l+1$ and when this state enters the end state, it activates the level above it to emit $l-1$ level's subsequence. Intuitively, each level has a sub-HMM conditioned on its state with each state being a self-contained probabilistic model that is itself a hierarchical HMM. Unlike the HMM in which the states only emit observations, the states in an H-HMM can also emit other state sequences.

In order to indicate whether a level has completed emitting its subsequence, a binary variable f_t^l is used. When $f_t^l = 0$, the states at all levels above l remains unchanged. The transition to a different state occurs horizontally within the same level. When $f_t^l = 1$, the state one level above the current level is activated. This can be interpreted as a vertical transition.

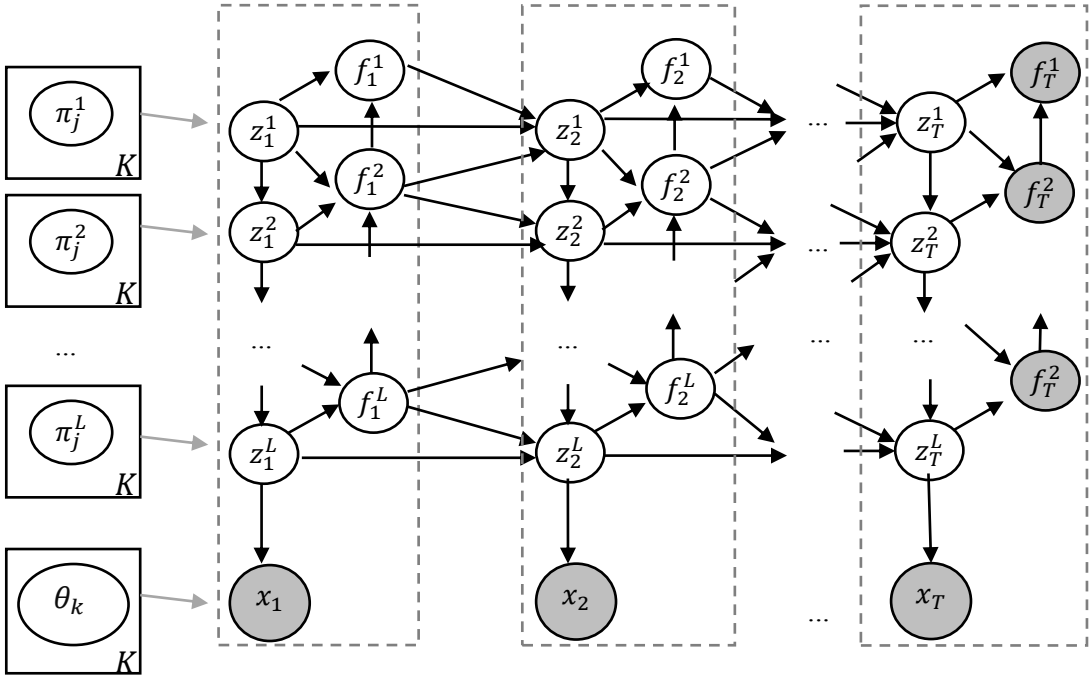


Figure 5.2: Graphical representation of a Hierarchical HMM [83]. The states at various levels, the observations and the parameters are shown. Each dotted rectangle groups the variables at a time instant. If the current level l has not finished ($f_{t-1}^l = 0$), then the state z_t^{l-1} is the same state z_{t-1}^{l-1} at the previous time $t-1$. Otherwise, its new value is determined from the state at previous levels $z_t^{1:l-1}$. The transition parameters $\pi^{1..L}$ and emission distribution parameters θ are shown on the left.

The state transition probabilities of the H-HMM need additional conditioning on the states in the level above them. Consequently, the transition probability matrix π must now include an additional dimension for the hierarchical levels. The probability of transitioning to state k from j for level l is now given by $P\left(z_t^l = k \mid z_{t-1}^l = j, z_t^{1..l-1} = i\right) = \pi_{j,k}^l$ where i is the current state corresponding to all the parent levels $z_t^{1..l-1}$. Let the conditional probability of the binary variables when the level below has completed be $P\left(f_t^l = 1 \middle| f_t^{l+1} = 1, z_t^l, z_t^{l-1}\right) = \psi^{l,z_t^l,z_t^{l-1}}$. Here each ψ is a parameter that controls the probability of a state transition at a particular level. Finally, it is common to assume that the states that emit the observations are at the bottom level and hence the state specific observation density parameters remain identical to the HMM. A graphical representation is provided in Figure 5.2.

Nonparametric variant

In order to construct the nonparametric variant of the H-HMM, it is useful to define the Bayesian extension first by introducing priors for the parameters. Similar to the Bayesian HMM described in Section 4.2, the rows of the transition matrix must be coupled. Let the priors be $\beta^l \sim Dir(\frac{\gamma}{K} \ldots \frac{\gamma}{K})$ and $\pi_j^l \sim Dir(\alpha\beta_1^l \ldots \alpha\beta_K^l)$ where Dir is the Dirichlet distribution, K is the number of states and γ, α are some positive real numbers. Note that the Dirichlet priors are now extended to multiple levels $\beta^{1:L}$ and $\pi_j^{1:L}$ when compared with HMM. The ψ parameters can be assigned a Beta prior $Beta(a, b)$ where $a, b \in \mathbb{R}^+$. Let us also assign the observation density parameters a prior H with the exact form of this prior depending on the parameters themselves. With this definition, an observation is generated in the Bayesian H-HMM as follows:

$$\beta^l \mid \gamma \sim Dir\left(\frac{\gamma}{K}\right) \qquad l = 1, \ldots L \tag{5.1}$$

$$\pi_j^l \mid \alpha, \beta^l \sim Dir(\alpha\beta_1^l \ldots \alpha\beta_K^l) \qquad \begin{matrix} j = 1, \ldots K \\ l = 1, \ldots L \end{matrix} \tag{5.2}$$

$$\psi^{l,k,k'} \mid a, b \sim Beta(a, b) \qquad \begin{matrix} l = 1, \ldots L \\ k, k' = 1..K \end{matrix} \tag{5.3}$$

$$\theta_k \mid H \sim H \qquad k = 1, \ldots K \tag{5.4}$$

$$z_t^l \mid z_{t-1}^{1:l}, f_{t-1}^{1:l}, \pi \sim \begin{cases} \pi_{z_{t-1}^l}^l & if\ f_{t-1}^l = 0 \\ \pi_{0,z_t^{l-1}}^{l-1} & otherwise \end{cases} \qquad t = 1, \ldots . T \tag{5.5}$$

$$x_t \mid z_t^L, \{\theta_k\}_{k=1}^{\infty} \sim F(\theta_{z_t^L}) \qquad t = 1, \ldots T \tag{5.6}$$

If each level of the H-HMM is considered separately, then the generation process in (5.1) and (5.2) is similar to the process by which the observations are generated using HDP and the nonparametric HMM. The π_j distributions in the HDP are similar to the π_j^l distributions in the H-HMM. By assigning independent HDP priors to the different levels of the H-HMM, a nonparametric variant of the H-HMM is obtained. The number of states in each level is unbounded and the equations (5.1) and (5.2) with the HDP prior are written as

$$\beta^l \mid \gamma \sim GEM(\gamma) \qquad l = 1, \ldots L \tag{5.7}$$

$$\pi_j^l \mid \alpha, \beta^l \sim DP(\alpha, \beta^l) \qquad \begin{matrix} j = 1,2, \ldots \\ l = 1, \ldots L \end{matrix} \tag{5.8}$$

The rest of the generative process in (5.3) to (5.6) remains the same except that K is unbounded. Note that in an alternative construction, the number of levels L in the H-HMM can be

unbounded [84]. This is not considered here since the number of hierarchical levels is known a-priori in this context.

5.3 Activity Model

Let i.i.d training data $X = \{x^n\}_{n=1}^N, Y = \{y^n\}_{n=1}^N$ be given. Here $x^n = x_1^n \dots x_T^n$ is an observation sequence and $y^n \in \{1 \dots c \dots C\}$ is the class label corresponding to the observation x^n. For example, in activity classification, x^n is the input image sequence and y^n is the activity class label. At a time instant t, the activity observation x_t contains a factored set of features. There are two types of features – the skeleton features $xs_t \in \mathbb{R}^{d_s}$ that represent the information about the joint positions and the object features $xo_t \in \mathbb{R}^{d_o}$ that represent an object. Let $x_t = (xs_t, xo_t)$. Further discussion of the features is deferred to Section 5.5. The objective is classification: given a new test activity sequence $\hat{x}$, the corresponding activity label $\hat{c}$ must be predicted using the information learned when training the model. A suitable prediction is

$$\hat{c} = \underset{c}{\operatorname{argmax}}\, p(c \mid \hat{x}, X, Y) \tag{5.9}$$

5.3.1 Structure

The H-HMM model structure proposed here differs from the canonical H-HMM in three key aspects. Firstly, the number of levels is restricted to two. This simplified model with only two levels ensures tractable inference, while retaining the benefits of a compact hierarchical structure. Secondly, a factored form of the states is presented, which provides flexibility to add new data channels if necessary. Finally, unlike H-HMM, the output label corresponding to the input activity is integrated directly into the model structure.

The top level contains the action states that emit the pose state sequence. Let a_t be the hidden action state at time t. The bottom level contains the pose states which in turn emit the observations. A pose state at time t is factored into two different discrete states – a skeleton state zs_t and an object state zo_t. The action, skeleton pose and object pose have separate HDP priors and there is no a priori upper bound on the number of states.

A single binary variable f_t is used to indicate whether an action state has completed or not. If the binary variable has a zero value, then the action states remain unchanged and a new pose state is determined based on the pose state at previous time instant. Otherwise, there is a transition to a new action state. The single indicator variable controls the transition of the bottom level states and the action states.

The pose states always emit a single observation. Each observation consists of two samples from independent Gaussian distributions - one for the skeleton pose and the other for the object pose. Given a skeleton pose state k, the skeleton observations are drawn from $\mathcal{N}(\mu s_k, \Sigma s_k)$. Similarly, the object observations are drawn from $\mathcal{N}(\mu o_k, \Sigma o_k)$ for an object pose state k. The observations are independent of the action state given the pose state. The density parameters are assigned appropriate priors. For example, the mean parameters have independent normal priors and the covariance parameters have independent Inverse-Wishart priors.

In order to perform classification, an activity label is modelled by multinomial logistic regression [16] conditioned on the assignment of action states. Let $\overline{a}$ be a vector that represents the number of transitions from one action state to another in an unknown activity. Let the set of all regression coefficients be $\boldsymbol{\eta} = [\eta^1 \ldots \eta^C]$ where η^c are the coefficients corresponding to the activity class c. The linear predictor for a label is then computed as $\eta^{c\top}\overline{a}$. Intuitively, the parameter space is now extended to include the linear predictor and during inference, as explained in Section 5.4, the regression coefficients influence the action states and vice-versa. This ensures that the activities can be discriminated from one another based on the assignment of action states. The labels do not depend directly on the skeleton and object pose states.

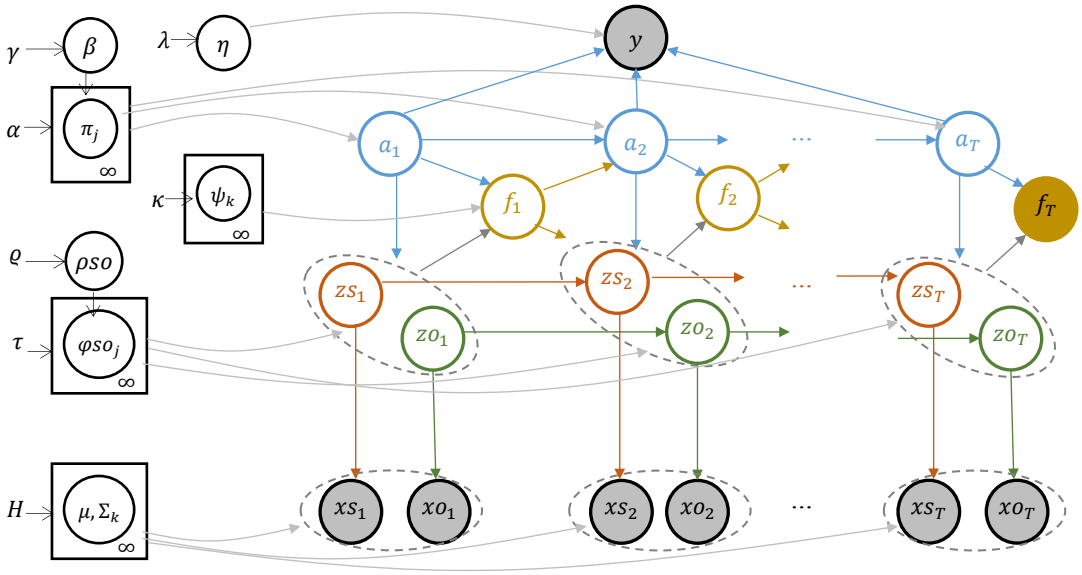


Figure 5.3: Graphical representation of the activity Model. The top level contains action states $a_{1:T}$ that emit the bottom level skeleton pose $zs_{1:T}$ and object pose $zo_{1:T}$ states which in turn emit the observations $x_{1:T}$. The binary variable f_t determines whether the action state remains unchanged or not. The action states determine the activity label y. The parameters and their priors are in the left side. This representation differs from the standard H-HMM in three ways: (i) the integration of label y, (ii) the factored form of x and z and (iii) the presence of the priors.

5.3.2 Generative Process

Following the above model, the generation process for an observation is summarized as follows. First the priors are drawn from their hyper priors in steps 1 to 5 and then the observations are drawn according to steps 6 to 12. Figure 5.3 provides an overview.

1) Draw the HDP priors for the top level action states from the hyper parameters. Let β be the overall distribution for the action states, and let π_j be the set of transition probabilities.

$$\begin{aligned} \beta \mid \gamma &\sim GEM(\gamma) \\ \pi_j \mid \alpha, \beta &\sim DP(\alpha, \beta) \qquad j = 1,2, \dots \end{aligned} \tag{5.10}$$

2) Draw the HDP priors for the bottom level states from the hyper parameters. This includes both the skeleton pose states and object pose states. Note that for each action state, there is a set of skeleton states and a set of object states. Given an action state a, let ρs^a be the overall distribution of skeleton states and let φs_j^a be the distribution for the j^{th} skeleton state. Similarly, let ρo^a be the overall distribution of object states and let φo_j^a be the distribution for the j^{th} object state.

$$\begin{aligned} \rho s^a \mid \varrho &\sim GEM(\varrho) && a = 1,2, \dots \\ \rho o^a \mid \varrho &\sim GEM(\varrho) && a = 1,2, \dots \\ \varphi s_j^a \mid \tau, a, \rho s^a &\sim DP(\tau, \rho s^a) && j = 1,2, \dots \\ \varphi o_j^a \mid \tau, a, \rho o^a &\sim DP(\tau, \rho o^a) && j = 1,2, \dots \end{aligned} \tag{5.11}$$

3) Draw the normal distribution mean and covariance parameters from the hyper parameters. The skeleton pose parameters are μs, Σs and object pose parameters are $\mu o, \Sigma$o respectively. The hyper-parameters μ_0, Σ_0 correspond to the mean and covariance of the normal prior and the hyper-parameters ν_0, Δ_0 correspond to the Inverse-Wishart prior.

$$\begin{aligned} \mu s_k \mid \mu_0^s, \Sigma_0^s &\sim \mathcal{N}(\mu_0^s, \Sigma_0^s) && k = 1,2, \dots \\ \Sigma s_k \mid \nu_0^s, \Delta_0^s &\sim IW(\nu_0^s, \Delta_0^s) && k = 1,2, \dots \\ \mu o_k \mid \mu_0^o, \Sigma_0^o &\sim \mathcal{N}(\mu_0^o, \Sigma_0^o) && k = 1,2, \dots \\ \Sigma o_k \mid \nu_0^o, \Delta_0^o &\sim IW(\nu_0^o, \Delta_0^o) && k = 1,2, \dots \end{aligned} \tag{5.12}$$

4) Draw the Bernoulli priors from which the completion indicator variables are drawn.

$$\psi^{a,ks,ko} \mid \kappa a, \kappa b \sim Beta(\kappa a, \kappa b) \qquad a, ks, ko = 1,2, \dots \tag{5.13}$$

5) Draw the regression coefficients for all the classes from a Laplacian prior. Let $||\boldsymbol{\eta}||_1$ denote the l_1 norm. The Laplacian prior is given as follows with λ being a hyper-parameter.

$$\boldsymbol{\eta} \mid \lambda \sim \exp(-\lambda||\boldsymbol{\eta}||_1) \tag{5.14}$$

6) Repeat steps 7 to 12 for each observation n and steps 7 to 11 for each time instant t of the observation.

7) Draw the action state from π if the binary variable at previous time instant is on. Otherwise, the action states remain unchanged.

$$a_t^n \mid a_{t-1}^n, f_{t-1}^n, \pi \begin{cases} \sim \pi_{a_{t-1}^n} & if\ f_{t-1}^n = 1 \\ = \delta(a_t^n, a_{t-1}^n) & otherwise \end{cases} \tag{5.15}$$

8) Draw the skeleton pose state from φs. If the binary variable at the previous time instant is on, which indicates that a new action state has begun, then the pose state is drawn from an initial distribution indicated with subscript 0.

$$zs_t^n \mid a_t^n, f_{t-1}^n, \varphi s \sim \begin{cases} \varphi s_0^{a_t^n} & if\ f_{t-1}^n = 1 \\ \varphi s_{zs_{t-1}^n}^{a_t^n} & otherwise \end{cases} \tag{5.16}$$

9) Draw the object pose state similarly from φo. As above, if the binary variable at the previous time instant is on, then the pose state is drawn from an initial distribution.

$$zo_t^n \mid a_t^n, f_{t-1}^n, \varphi o \sim \begin{cases} \varphi o_0^{a_t^n} & if\ f_{t-1}^n = 1 \\ \varphi o_{zo_{t-1}^n}^{a_t^n} & otherwise \end{cases} \tag{5.17}$$

10) Draw the binary variable from its prior shown in step 4.

$$f_t^n \mid a_t^n, zs_t^n, zo_t^n\ \psi \sim Ber(\psi^{a_t^n, zs_t^n, zo_t^n}) \tag{5.18}$$

11) Draw the skeleton and object observations from the normal distribution using the mean and covariance parameters drawn according to (5.12).

$$\begin{aligned} xs_t^n \mid zs_t^n, \mu s, \Sigma \mathrm{s} &\sim \mathcal{N}(\mu s_{zs_t^n}, \Sigma \mathrm{s}_{zs_t^n}) \\ xo_t^n \mid zo_t^n, \mu o, \Sigma \mathrm{o} &\sim \mathcal{N}(\mu o_{zo_t^n}, \Sigma \mathrm{o}_{zo_t^n}) \end{aligned} \tag{5.19}$$

12) Finally draw the activity label from a multinomial distribution based on the linear predictor. The linear predictor is computed from the action states drawn according to (5.15) and the regression coefficients are drawn according to (5.14).

$$y^n \mid a_{1..T}^n, \boldsymbol{\eta} \sim Mult\left(\frac{\exp\left(\eta^{1\top}\overline{a}\right)}{\sum_{c'=1}^{C}\exp\left(\eta^{c'\top}\overline{a}\right)} \cdots \frac{\exp\left(\eta^{C\top}\overline{a}\right)}{\sum_{c'=1}^{C}\exp\left(\eta^{c'\top}\overline{a}\right)}\right) \tag{5.20}$$

Due to the clustering nature of HDP, some observations, across activity labels, may be assigned the same pose states. For example, two different observations that involve *push* and *pull* actions may contain very similar alignment of skeletal joints during the course of the action. As a result, the set of pose states for these actions could be identical. This reduces the overall number of pose states necessary for describing all observations and promotes parameter sharing across the labels. By extension, two different activity labels may also share the same action states if they involve a similar sequence of pose states. Thus the model enables a reduction in the numbers of action and pose states. The sequence of action states assigned to the observations is sufficient for distinguishing the activities. The structure of the model is simplified by removing the direct dependency of the activities on the observations.

5.4 Posterior Inference

Exact posterior inference is intractable in DP based models and an approximate inference technique such as Markov Chain Monte Carlo (MCMC) must be used. The Gibbs sampling procedure discussed in Appendix E.2 is applied here to draw posterior samples for all the variables. The Gibbs sampler proceeds by sampling the hidden state sequence variables assuming that the parameter variables are given and then sampling the parameter variables assuming the hidden state sequences are available. This process is repeated for a number of iterations until convergence. Table 5.1 lists the inference steps.

The parameter variables are $\pi, \varphi s, \varphi o$ (the transition parameters for action, skeleton and object states), $\mu s, \Sigma s, \mu o, \Sigma o$ (the normal distribution parameters for skeleton and object), ψ (the Bernoulli parameter for finish variable) and $\boldsymbol{\eta}$ (the regression coefficients). The state sequences are $a_{1:T}$ (action), $zs_{1:T}$ (skeleton pose), $zo_{1:T}$ (object pose) and $f_{1:T}$ (action completion indicators).

5.4.1 Sampling Hidden State Sequence

Many DP based models sample one state at a time, given all other state assignments. This method marginalizes over the unbounded number of state transitions. Sequentially sampling a state conditioned on the states at other time instants can lead to slow mixing rates because of

the temporal correlations. Instead of this collapsed sampling procedure, it is better to explicitly instantiate all the parameters and block sample the state sequence based on the standard belief propagation techniques used in Bayesian networks.

The hidden states cannot be block sampled without resorting to some truncated approximation [71]. An effective way to approximate the countably infinite hidden states is to use the weak limit approximation to DP [76] discussed in Section 4.5.1. In this approximation, the number of expected components is set to a large value. As this value increases without limit, the model's marginal distribution approaches the DP. Specifically, the stick breaking weights are approximated as

$$GEM(\gamma) \triangleq Dir(\frac{\gamma}{K}, \dots \frac{\gamma}{K}) \tag{5.21}$$

where K is a number that exceeds the expected number of components. The DP prior ensures that only a small subset of the K components is used. This truncated approximation is computationally efficient and allows the use of existing well-studied Bayesian message passing techniques [76]. Consequent to this approximation, equations (5.10) and (5.11) become the following.

$$\begin{aligned}
\beta \mid \gamma &\sim Dir(\frac{\gamma}{K^a} \dots \frac{\gamma}{K^a}) && \\
\pi_j \mid \alpha, \beta &\sim Dir(\alpha\beta_1, \dots, \alpha\beta_{K^a}) && j = 1, \dots, K^a \\
\rho s^a \mid \varrho &\sim Dir(\frac{\varrho}{K^s} \dots \frac{\varrho}{K^s}) && a = 1, \dots, K^a \\
\varphi s_j^a \mid \tau, a, \rho s^a &\sim Dir(\tau\rho s_1^a, \dots, \tau\rho s_{K^s}^a) && \begin{matrix} j = 1, \dots, K^s \\ a = 1, \dots, K^a \end{matrix} \\
\rho o^a \mid \varrho &\sim Dir(\frac{\varrho}{K^o} \dots \frac{\varrho}{K^o}) && \\
\varphi o_j^a \mid \tau, a, \rho o^a &\sim Dir(\tau\rho o_1^a, \dots, \tau\rho o_{K^o}^a) && \begin{matrix} j = 1, \dots, K^o \\ a = 1, \dots, K^a \end{matrix}
\end{aligned} \tag{5.22}$$

Here K^a, K^s and K^o are the maximum number of states corresponding to action, skeleton and object respectively. Typically the maximum number of action states is set smaller than that of skeleton and object states. The joint distribution of an observation, its label and the state sequence, given the parameter variables are provided as follows.

$$
\begin{aligned}
& p(xs_{1:T}, xo_{1:T}, y, a_{1:T}, zs_{1:T}, zo_{1:T}, f_{1:T} |\, \pi, \varphi s, \varphi o, \mu s, \Sigma s, \mu o, \Sigma o, \psi, \boldsymbol{\eta}) \\
= \; & p(y|a_{1:T}) \prod_{t=1}^{T} p(a_t|a_{t-1}, f_{t-1})\; p(zs_t|zs_{t-1}, a_t, f_{t-1})\; p(zo_t|zo_{t-1}, a_t, f_{t-1}) \\
& \qquad p(f_t|a_t, zs_t, zo_t)\; p(xs_t|zs_t)\; p(xo_t|zo_t)
\end{aligned} \tag{5.23}
$$

A key challenge is to ensure that when sampling at time t, the action state a_t, skeleton state zs_t, object state zo_t and the binary variable f_t are block sampled together. It is essential to perform efficient message passing inference similar to the dynamic programming algorithm of Section 3.1.1 used in HMMs for convergence. In order to achieve this efficiency, the H-HMM is flattened and all possible combinations of these variables are considered. This flattening might not be practical if there are too many states at the various levels of the hierarchy. However, the number of possible states is bounded here by the use of a truncated DP. The flattening and the weak limit approximation in (5.22), allows the use of variations of the forward-backward algorithm to sample the state sequence.

There is no straight forward mechanism available to include the conditional likelihood term $p(y|a_{1:T})$ while block sampling the posterior states using the forward-backward algorithm. This likelihood is approximated by using the action state sequence sampled from a previous iteration. Let $\boldsymbol{a}$ be a sampled action state sequence corresponding to an observation sequence in a previous sampling iteration.

Let $V(.)$ be an intermediate term that represents the joint probability for being in a specific combination of states and emitting an observation given another set of states.

$$
\begin{aligned}
V(.) = \; & P(a_m = k^a, zs_m = k^s, zo_m = k^o, f_{m-1} = f', xs_m, xo_m, y| \qquad m, n \\
& a_n = k^{a'}, zs_n = k^{s'}, zo_n = k^{o'}, f_n = f'') \qquad \in \{1 \ldots T\}
\end{aligned} \tag{5.24}
$$

Using the factorization of the joint distribution in (5.23), and substituting the terms from equations (5.15) to (5.20), $V(.)$ can be written as

$$
\begin{aligned}
& V\left(k^a, k^s, k^o, f', xs_t, xo_t, y \mid k^{a'}, k^{s'}, k^{o'}, f'', \boldsymbol{\eta}, \boldsymbol{a}\right) = \\
& \left[\left[\delta\left(f'', 1\right) \pi_{k^{a'}, k^a}\; \varphi s_{0,k^s}^{k^a}\; \varphi o_{0,k^o}^{k^a}\right] + \left[\delta\left(f'', 0\right) \delta\left(k^a, k^{a'}\right) \varphi s_{k^{s'}, k^s}^{k^a}\; \varphi o_{k^{o'}, k^o}^{k^a}\right]\right] \\
& Ber\left(f'; \psi^{k^a, k^s, k^o}\right) \mathcal{N}(xs_t;\, \mu s_{k^s}, \Sigma s_{k^s})\; \mathcal{N}(xo_t;\, \mu o_{k^o}, \Sigma o_{k^o})\; \frac{\exp\left(\eta^{y\top} \overline{a}\right)}{\sum_{c'=1}^{C} \exp\left(\eta^{c'\top} \overline{a}\right)}
\end{aligned} \tag{5.25}
$$

In the above, δ is the Kronecker Delta function with $\delta(m, n) = 1$, if $m = n$. The second line represents the probability of transitioning from $k^{a'}$ to k^a, $k^{s'}$ to k^s and $k^{o'}$ to k^o, all together, for a binary variable with value f''. The third line represents the f' probability, observations

probability and the label probability for a particular state sequence. The term $\overline{a}$ is computed from $\boldsymbol{a}$.

In order to block sample the states, messages passed backwards in time from $t = T \dots 1$ and forward from $t = 1 \dots T$ are used. The forward and backward messages in a standard HMM are computed recursively from the probability of transitioning from one state to another and the probability of emitting an observation. Here there are three different state variables a_t, zs_t and zo_t and also a finish variable f_{t-1} on which the state transitions are conditioned upon. The observation likelihood now includes the xs_t and xo_t terms, in addition to the label likelihood. The $V(.)$ term represents these probabilities and it can be used to compute the forward and backward messages. The forward message $m_{t-1,t}$, which represents the joint probability of states at time t and the observations up to time t, is computed recursively as follows:

$$m_{t-1,t}(k^a, k^s, k^o, f'|xs_{1:T}, xo_{1:T}, y, \boldsymbol{\eta}, \boldsymbol{a}) = \sum_{k^{a'} \in K^a} \sum_{k^{s'} \in K^s} \sum_{k^{o'} \in K^o} \sum_{f'' \in \{0,1\}} m_{t-2,t-1}\left(k^{a'}, k^{s'}, k^{o'}, f''|xs_{1:T}, xo_{1:T}, y, \boldsymbol{\eta}, \boldsymbol{a}\right) V(k^a, k^s, k^o, f'|k^{a'}, k^{s'}, k^{o'}, f'', xs_t, xo_t, y, \boldsymbol{\eta}, \boldsymbol{a}) \quad (5.26)$$

The backward message $m_{t,t-1}$ is the message that is passed from time t to $t-1$ with $m_{T+1,T} = 1$. It represents the probability of observing states in the future from a current state. The backward message is also based on the probability of state transitions and the observation probability, which the V term represents. The backward message is determined recursively over all possible states as follows.

$$m_{t,t-1}(k^a, k^s, k^o, f'|xs_{1:T}, xo_{1:T}, y, \boldsymbol{\eta}, \boldsymbol{a}) = \sum_{k^{a'} \in K^a} \sum_{k^{s'} \in K^s} \sum_{k^{o'} \in K^o} \sum_{f'' \in \{0,1\}} m_{t+1,t}\left(k^{a'}, k^{s'}, k^{o'}, f''|xs_{1:T}, xo_{1:T}, y, \boldsymbol{\eta}, \boldsymbol{a}\right) V(k^{a'}, k^{s'}, k^{o'}, f''|k^a, k^s, k^o, f', xs_t, xo_t, y, \boldsymbol{\eta}, \boldsymbol{a}) \quad (5.27)$$

The conditional distribution of the states given the observations is a product of the forward and backward messages. The states are sampled from this conditional distribution. Let $\boldsymbol{a}^{-t}$ denote the terms excluding the t^{th} term with $\boldsymbol{a} = \boldsymbol{a}^{-t} + (a_t = k^a)$ for some action state k^a. Using the forward and backward messages, the hidden state at time t is sampled as below.

$$p(a_t = k^a, zs_t = k^s, zo_t = k^o, f_t = f' \,|xs_{1:T}, xo_{1:T}, y, \boldsymbol{\eta}, \boldsymbol{a}) \propto \\ m_{t+1,t}(k^a, k^s, k^o, f'|xs_{1:T}, xo_{1:T}, y, \boldsymbol{\eta}, \boldsymbol{a}) \;\times \\ m_{t-1,t}(k^a, k^s, k^o, f'|xs_{1:T}, xo_{1:T}, y, \boldsymbol{\eta}, \boldsymbol{a}^{-t} + k^a) \tag{5.28}$$

5.4.2 Sampling Parameters

The sampled state sequences are used to compute count matrices. Each element in the matrix records the number of transitions from one state to another. Let $A \in \mathbb{Z}^{(K^a+1)\times K^a}$ be a matrix of counts computed from the sampled action state sequences with A_{jk} being the number of transitions from action state j to k across the training set. The number of transitions for an initial action state $A_{0k}$is maintained in the $K^a + 1$ row. Similarly, let $S^a \in \mathbb{Z}^{(K^s+1)\times K^s}$ and $O^a \in \mathbb{Z}^{(K^o+1)\times K^o}$ be the count matrices corresponding to the skeleton and object states for an action a. The HDP transition parameters $\beta, \rho s, \varphi s, \rho o, \varphi o$ and their hyper parameters can be sampled from their posteriors, one at a time, through standard inference methods [44] using these count matrices. The notable exception is for action state transition parameter π for which it is preferable to disable self-transitions π_{jj} since $f_t = 0$ already implies that $a_t = a_{t+1}$. The data augmentation procedure in [92] is followed to resample π.

Let $F^{k^a,k^s,k^o} \in \mathbb{Z}$ be the number of times an action, skeleton and object state combination is in a completed state. This value can be computed from the sampled $f_{1:T}$ values. The Bernoulli variable ψ has a conjugate Beta prior and the posterior updates can be performed analytically using the computed F^{k^a,k^s,k^o} values as discussed in Appendix C.3.

In order to sample the Normal distribution parameters, the sampled state sequences are used again. Let the set of skeleton observations assigned to hidden state k^s be $\mathcal{X}S_{k^s} = \{xs_t^n \in X : zs_t^n = k^s\}$. The mean and covariance parameters have conjugate priors and hence a closed form posterior update for $\mu s_{k^s}, \Sigma s_{k^s}$ can be directly computed based on the set $\mathcal{X}S_{k^s}$ as seen in the previous chapters. A similar procedure is followed for the object pose mean and covariance parameters. The term $\overline{a}$ can be computed from the action state sequence $\boldsymbol{a}$ for (5.24) or from any $a_{1:T}$ in general. For example, if a_t is represented as a K length vector, with an element k being 1 when $a_t = k$ and 0 otherwise, then $\overline{a} = \frac{1}{T}\sum_t a_t$. Alternatively, $\overline{a}$ can represent the number of transitions from one action state to another.

This leaves only the estimation of regression coefficients $\boldsymbol{\eta}$. Using a Laplacian prior for $\boldsymbol{\eta}$ rather than a normal prior on the coefficients encourages coefficient values which are either relatively large or near to zero. The sparseness of this prior leads to structural simplification and often results in generalization and better classification results [93]. The sparse multinomial logistic

regression [82] provides a fast exact algorithm that scales favourably as the dimension of the features increases. The algorithm can be applied to large data sets in high dimensional feature spaces. This procedure is used here for estimating the regression weights .

5.4.3 Prediction

During prediction, a test activity sequence's label is determined. In order to do this, first the action state sequence is inferred using Viterbi decoding, discussed in Section 3.1.1, from the parameters corresponding to a posterior sample. The conditional likelihood term in (5.23) is not used in this process. The linear predictor is then determined from the regression coefficients η and $\overline{a}$, where $\overline{a}$ is computed from the inferred action state sequence. The label is obtained using (5.29).

$$\begin{aligned} \hat{c} &= \underset{c}{\operatorname{argmax}} \frac{\exp(\eta^{c^\top}\overline{a})}{\sum_{c'=1}^{C}\exp(\eta^{c'^\top}\overline{a})} \\ &= \underset{c}{\operatorname{argmax}} \exp(\eta^{c^\top}\overline{a}) \end{aligned} \tag{5.29}$$

Table 5.1: Posterior Inference Algorithm

Input: Training examples of activity sequences with their corresponding labels. **Output:** Samples of posterior parameters.
1. Sample initial values of $\beta, \pi, \rho s, \varphi s, \rho o, \varphi o, \psi, \mu s, \Sigma s, \mu o, \Sigma o$ from their respective distributions. 2. For each training example, sample hidden state sequences a_t, zs_t, zo_t, f_t using forward and backward messages as per (5.23) to (5.28). 3. Compute the count matrices from the sampled hidden state sequences as in Section 5.4.2. 4. Sample HDP parameters β, π for action states from the count matrix A. 5. Sample HDP parameters $\rho s, \varphi s$ for skeleton states from the count matrix S^a for all the actions. 6. Sample HDP parameters $\rho o, \varphi o$ for object states from the count matrix O^a for all the actions. 7. Sample Bernoulli parameter ψ from F^{k^a,k^s,k^o} for all the action, skeleton and object states. 8. Sample normal distribution parameter $\mu s, \Sigma s$ for all skeleton states using $\mathcal{X}S$. 9. Sample normal distribution parameter $\mu o, \Sigma o$ for all object states using $\mathcal{X}O$. 10. Compute $\overline{a}$ from the sampled action state sequence. 11. Estimate $\boldsymbol{\eta}$ using sparse multinomial logistic regression. 12. Sample the hyper parameters. 13. Repeat from step (2) to collect more samples.

5.5 Experiments

The activity model is evaluated against the publicly available Cornell Activity dataset [85]. The dataset contains depth image sequences annotated with the 3D joint positions of the human skeleton. The joint positions were estimated from the depth images. The activities are performed by different human subjects and each activity involves only one individual. Unlike the experiments discussed in the previous chapter, the information in the depth channel is included in the experiments. However, the RGB images are not used. In order to verify the generic applicability of the approach, the model is also evaluated against a motion capture based dataset [86]. The skeletal joint positions in this dataset were obtained through an optical marker based system.

The precise definition of the time scale for the actions and activities may depend on the task. In this work, evaluations are performed on fairly simple activities that span less than a minute. The experiments here focus on offline activity recognition. Hence in all the evaluations below, the observation sequence is treated as a whole unit for both training and testing. Further, the datasets used here pre-segment the activities into their corresponding classes.

5.5.1 Cornell Activity Dataset

The Cornell Activity [85] dataset contains image sequences captured using a Kinect sensor. The depth images and the RGB images are available in the dataset along with the estimated 3D joint positions. The sensor produces a depth image with a range of 1.2m to 3.5m and the image resolution in the dataset is 320x240. Each activity sequence spans about 45 seconds and on average the dataset provides 1400 frames of data per activity.

This dataset contains 12 activities – *rinsing mouth*, *brushing teeth*, *wearing contact lens*, *talking on phone*, *drinking water*, *opening pill container*, *cooking (chopping)*, *cooking (stirring), talking on couch*, *relaxing on couch*, *writing on whiteboard* and *working on computer*. The activities were performed by four different subjects. The subjects were not given any specific instructions on how to carry out the activity. The activities were recorded in different locations viz. *office*, *kitchen*, *bedroom*, *bathroom* and *living room* of a regular household in a realistic environment. Some examples are shown in Figure 5.4.

Each frame contains 15 3D joint positions with coordinates (x, y, z) in a world coordinate frame. Pairwise relative joint positions within a frame are used for skeleton based features. This ensures invariance to uniform translation of the body. All 105 joint pairs are considered, resulting in a 315 dimensional vector for the skeleton features. This high dimensional feature vector is projected to a lower dimensional vector space using Principal Component Analysis (PCA) [35].

The components of the projected vector correspond to the skeleton features. The first d^s components that capture at least 90% of the total variance are used.

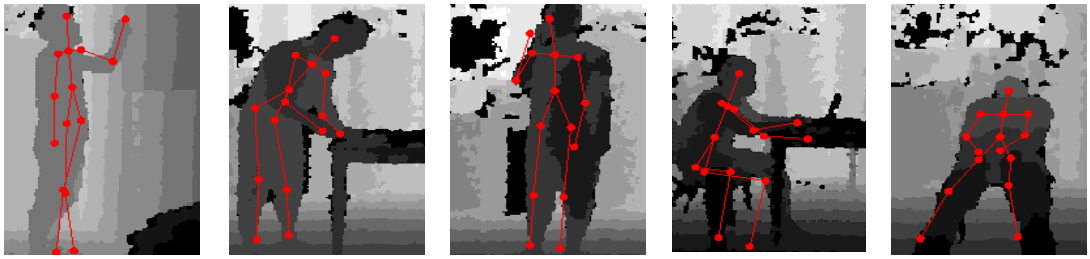


Figure 5.4: Cornell-Activity dataset samples. Some activities from the Cornell Activity [85] dataset is shown. The skeleton joint positions are overlaid on top of the depth images.

The local image patterns around the joint positions in a depth image frame are used as object based features. Note that no specific object detection algorithm is used. Specifically, features are extracted from a bounding box that is constructed around each joint position. The image inside this bounding box is treated as a greyscale image and a Histogram of Oriented Gradient (HOG) [77] descriptor is computed. Further, the bounding box 3D space is divided into cells and the scatter-ness, linear-ness and surface-ness of the point distribution inside each cell is used to compute a point cloud feature descriptor. Let $\lambda_1 > \lambda_2 > \lambda_3$ denote the eigenvalues of the covariance matrix of the 3D points inside a cell. Scatter-ness is λ_1, linear-ness is $\lambda_1 - \lambda_2$ and surface-ness is $\lambda_2 - \lambda_3$ [87]. The descriptors obtained in this way from all the joints are concatenated and the resultant 2400 dimensional vector is projected using PCA. The components of the projected vector correspond to the object features. The first d^o components that capture at least 85% of the total variance are used.

During evaluation, as outlined in [85], the activities are grouped based on their locations: office, kitchen, bedroom, bathroom and living room. Classification is performed in each location group. The experiments are conducted for the setting in which the instance of a subject has been seen before (S-Seen). It is also conducted for the challenging setting in which the subject is new and is being seen for the first time during prediction (S-New). For the S-New setting, the model was trained on three of the four subjects and tested on the fourth while for S-Seen setting the training included sequences from all the subjects.

Following Bayesian hierarchical modelling, the hyper parameters have weakly informative hyper priors. The concentration parameters $\gamma, \alpha, \varrho, \tau$ were all given a vague gamma prior similar to [44], namely $Gamma(1,0.01)$. The use of a gamma prior ensures that the initial choice of the concentration parameter is not important and it is the data that drives the sampled concentration parameter. The maximum number K^a of action states was set a value less than both K^s the maximum number of skeleton states and K^o the maximum number of object states.

The exact value for these parameters is immaterial since even with a very large value, the sparse nature of Dirichlet Process ensures that only a subset of these states are activated. However the number of action states should at least exceed the number of activities that are classified. Although large values were tried for K^a, K^s and K^o, it was found that the values $K^a = 20$ and $K^s, K^o = 30$ were sufficiently large. The hyper parameters μ_0^s, Σ_0^s that define the Gaussian prior for the mean value for the skeleton observations were set equal to the empirical mean and the empirical covariance respectively of the skeleton features. The hyper parameter ν_0^s was given a large value (1000) and the scale matrix Δ_0^s was set to the empirical covariance. A similar procedure was followed for the hyper parameters corresponding to object observations. The values of both $\kappa a, \kappa b$ were set to 0.5 to ensure a wide range of initial values.

In the first iteration of the posterior inference, all the hyper parameters are initialized as described above. All the other parameters are sampled from their respective prior distribution based on the hyper parameter values. When sampling the hidden state sequence, the conditional likelihood term is not used for the first few iterations. In all subsequent iterations, the regression coefficients are computed based on the sampled action state sequence and the conditional likelihood term computed from the regression coefficients is used during state sequence sampling. The sparse multinomial logistic regression algorithm was set a convergence tolerance of 0.001 and a maximum of 10000 iterations. Following standard practice [44], the hyper parameters are re-sampled after each sampling iteration. The first 300 samples are discarded and then every 3rd sample is recorded to collect a total of 100 samples. Further, a burn-in period of 50 iterations is used every time the posterior for the Gaussian distribution parameters and the Dirichlet process concentration parameters are sampled. In order to verify that an adequate number of sampling iterations are used, the change in the number of activated states and the difference in Gaussian distribution parameter values between iterations were checked. Further, an increase in the number of sampling iterations did not affect the classification results which appeared to indicate convergence. As expected, unlike a collapsed sampler that takes many iterations to converge, the use of block sampling resulted in fewer iterations for convergence. Each posterior sample contains the hyper parameter values, the parameter values and the regression coefficients. During prediction, the PCA embedding for a test sequence's skeleton and object features is computed based on the components extracted from training examples. For each sample collected during training, based on the sample parameters, the Viterbi decoding discussed in Section 3.1.1 is used to determine the action state sequence and in conjunction with the regression coefficients, the activity label is predicted. The mode of all these predicted labels is used to determine the test sequence's final label.

The block sampling procedure discussed in Section 5.4.1 flattens the H-HMM and considers all possible states when applying the forward–backward algorithm. Hence the computational cost is $O(TK^2)$ where T is the length of the sequence and K is the total number of states with $K = K^a * K^s * K^o$. The block sampler used here offers some advantages such as faster mixing rate [76]. The block sampling procedure needs $O(TK)$ space in order to store the messages.

Hierarchical Structure

The hierarchical nature of the model is examined. Figure 5.5 shows the sampled state sequences for the *rinsing mouth* activity from one of the samples collected during training activities in the *bathroom* location for the full model. The major motion sequence in this activity involves the subject bending down to the sink and drinking with raised arms. The subject is also in a stationary position (perhaps gargling). The subject performs these motions multiple times. The sampled bottom level sequence has 6 unique skeleton states and 15 unique object states with a total of 90 possible states. The number of action states sampled is 4 with three predominant states. This is much less than the number of possible states in the bottom level. As a consequence of the fewer number of states, the state segmentation in the top level is smoother than that of the bottom level. This behaviour is desirable as it ensures that a small number of states are sufficient to determine the activity, thereby simplifying the model. Note that the action states need not correspond exactly to the actions that make up an activity since the objective is merely to distinguish the activities.

The distribution of the sampled action states when training activities in the *kitchen* location is shown in Figure 5.6. The action state sequences were collected during a sampling iteration and correspond to the training observation sequences from four different activities. There are some action states that are shared between activities. For example, both the *drinking water* and *opening pill container* activities involve sequences where the subject holds an object and these frames are assigned the same action state. There are also some action states that are unique to an activity. It is preferable that the activities contain unique action states in order to distinguish them between the activities more effectively. If a subject performs an action differently from the other subjects in the training example, it may lead to multiple unique action states that correspond to the same underlying action.

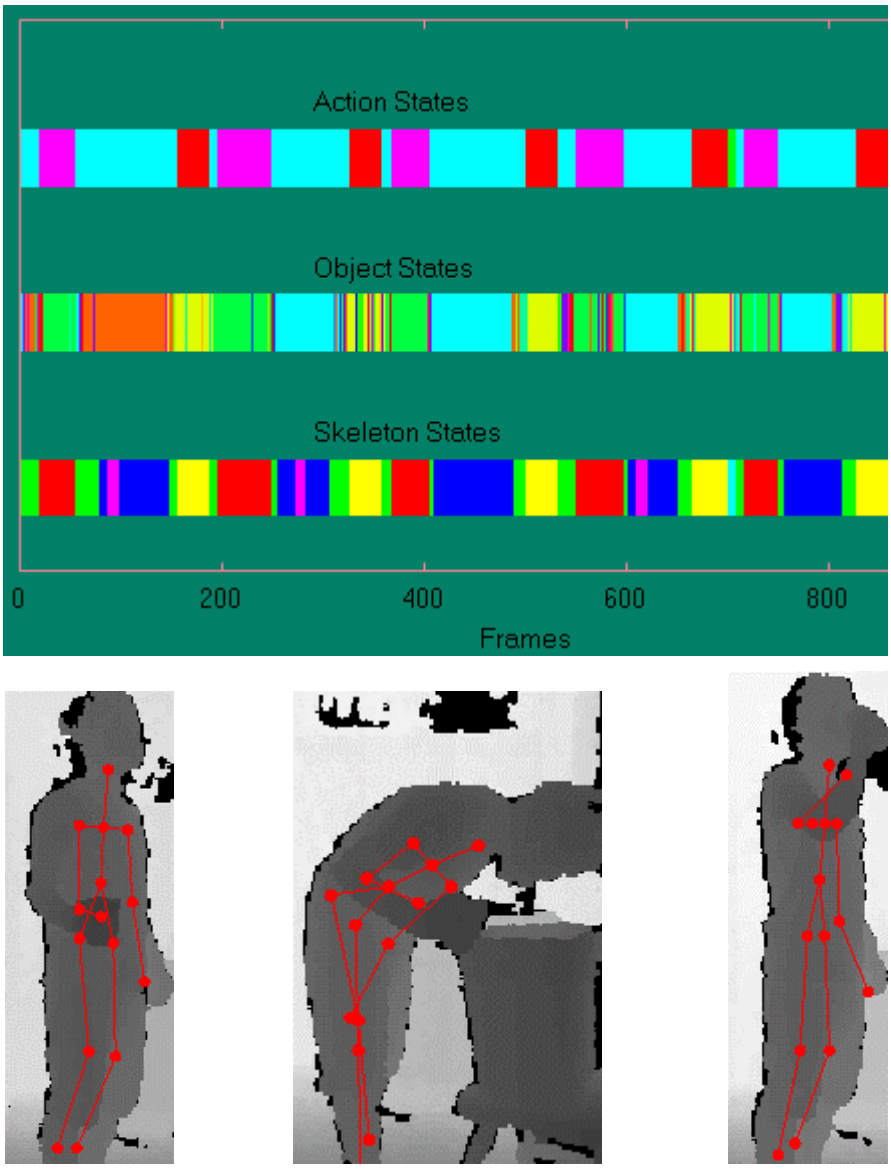


Figure 5.5: Learned hierarchical structure for the *rinsing mouth* activity. *Top:* The sampled skeleton, object and action states corresponding to an observation sequence are shown in a color coded format. The number of top level action states is fewer than the number of skeleton and object state combinations. The state segmentation for actions is smoother than that of skeleton and object. This validates the hierarchical nature of the model with *coarser* actions and *granular* poses. *Bottom*: The skeleton and depth frame corresponding to the three pre-dominant action states. The left frame corresponds to the action state in light blue, the middle frame corresponds to the action state in red and the right frame corresponds to the action state in pink.

During evaluation, it was observed that the total number of instantiated action states, involving multiple activities in a location group, was always less than 15 and had a mean value of 11. The average number of skeleton states and object states in a location group were 17 and 21 respectively. In general, the number of states depends on the complexity of the activities in the

dataset. The HDP prior used here has the advantage that the state cardinality can be estimated automatically using the data.

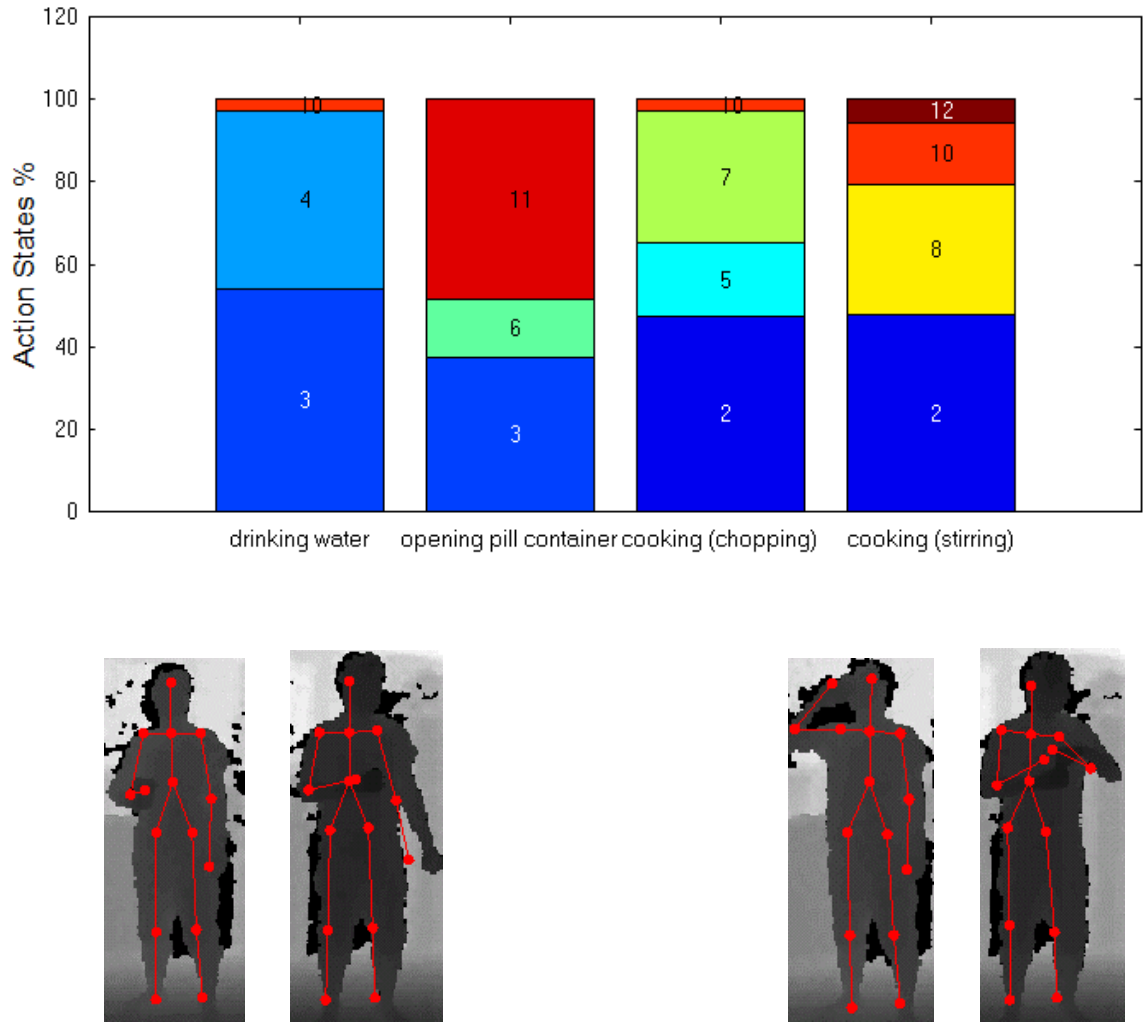


Figure 5.6: Action states for the activities involved in the *kitchen* location. *Top*: There are 10 instantiated action states (2, 3, 4, 5, 6, 7, 8, 10, 11 and 12) for four activities in this sampled action state sequence. The stacked bars indicate the percentage of observations that were assigned to a particular state. For example, in the *drinking water* activity, 54% of the observations across all the training sequences belonging to this activity were assigned to action state 3. There are some action states shared across activities (e.g. state 2 is shared between the two cooking activities) and some states unique to an activity (e.g. state 6 and 11). *Bottom left*: Frames that correspond to the shared action state for *drinking water* and *opening pill container* activity. *Bottom Right*: Frames that correspond to the unique action state for the same activities.

In order to verify the efficacy of the model, additional experiments are conducted with the standard parametric version of HMM and Hierarchical HMM. The model with regression excluded is then evaluated and finally the results for full model are presented.

Parametric HMM

The skeleton and object features at a time instant are concatenated to create a single observation vector. The Gaussian distribution mean and covariance parameters are assigned Normal and Inverse-Wishart priors respectively, as before. The state transitions are given an independent symmetrical Dirichlet distribution prior $Dir(1,..1)$. Multiple HMMs are then trained, one corresponding to each activity class label. When training an HMM, the standard forward-backward procedure discussed in Section 3.1.1 is used to sample the state sequences. About 100 posterior samples were collected for each HMM. Each sample contains values of the transition distribution and the observation distribution parameters. During prediction, given a test sequence, a set of class conditional likelihoods, one for each posterior sample, is computed using the HMM forward messages. The mean class conditional likelihood is then calculated from this set. This evaluation of mean class conditional likelihood is repeated, one for each possible class label and the class which has the maximum mean likelihood is selected as the label. Note that in a parametric HMM, the number of states must be specified in advance. Hence different numbers of states are tried for each class. The results are averaged over all the location groups and shown in Table 5.2. The observed classification accuracy for the S-Seen setting was ***62.5***% and for S-New setting was ***47.7***%.

Table 5.2: Cornell activity dataset - Classical Parametric HMM classification accuracy

Number of States	S-Seen Accuracy (%)	S-New Accuracy (%)
5	46.1	31.6
10	57.2	34.4
25	59.8	46.5
40	62.5	47.7
60	58.4	44.1

Parametric H-HMM

A classifier based on a two level parametric Hierarchical HMM [83] is trained. Similar to the parametric HMM, the skeleton and object features are concatenated to create a single observation vector and a separate classifier is trained for each activity label. The priors for the Gaussian distribution parameters and the transition matrices in both the levels are same as in the parametric HMM. The finish indicator variables are assigned a beta prior of $Beta(0.5,0.5)$. Here as well, different numbers of states are tried, but this time there are additional combinations corresponding to the top and bottom levels. The predicted class is selected by evaluating the class conditional likelihood of a test example similar to the parametric HMM. As before, the results are averaged over all location groups and are shown in Table 5.3. The

observed classification accuracy for the S-Seen setting was ***68.9***% and for S-New setting was ***56.4***%.

Table 5.3: Cornell activity dataset - Parametric H-HMM classification accuracy

Number of top level states	Number of bottom level States	S-Seen Accuracy (%)	S-New Accuracy (%)
5	5	42.1	31.4
5	10	53.6	34.7
5	30	68.9	56.4
5	60	57.5	43.9
10	15	63.3	49.2
10	20	57.8	47.0

Table 5.4: Cornell activity dataset - Classification accuracy (in %) for the model with regression excluded

Location	Activities	S-Seen	S-New
bathroom	*rinsing mouth. brushing teeth, wearing contact lens*	91.7	83.3
bedroom	*talking on phone, drinking water, opening pill container*	75.0	58.3
kitchen	*drinking water, opening pill container, cooking (chopping), cooking (stirring)*	75.0	62.5
living room	*talking on phone, drinking water, talking on couch, relaxing on couch*	81.2	75.0
office	*talking on phone, drinking water, writing on whiteboard, working on computer*	87.5	68.7

Unshared Parameters

The nonparametric H-HMM model in Section 5.3 excluding the regression aspect of the model is then evaluated. Unlike the full model, the examples and parameters are not shared across the activity labels, because a separate classifier needs to be trained for each activity label. However, the skeleton and object features are treated separately and the HDP prior is used to automatically infer the number of states. The predicted class is selected in the same way as for the parametric HMM. The classification accuracy is shown for each location group in Table 5.4 and an average of ***82.1***% and ***69.56***% was observed for the S-Seen and S-New setting respectively.

Full Model

Finally, the results are evaluated against the full nonparametric H-HMM model integrated with multinomial logistic regression. The examples and parameters are shared across the activity labels with the regression coefficients influencing the transition parameters with the use of the conditional distribution. The classification accuracy for the setting where an instance of the subject has been seen is consistently ***100%***. For the more challenging new subject settings, an accuracy of ***85.4%*** was observed. The results are provided in Table 5.5.

Table 5.5: Cornell activity dataset - Classification accuracy (in %) for the full model.

Location	Activities	S-Seen	S-New
bathroom	*rinsing mouth. brushing teeth*, *wearing contact lens*	100.0	100.0
bedroom	*talking on phone*, *drinking water*, *opening pill container*	100.0	83.3
kitchen	*drinking water*, *opening pill container*, *cooking (chopping)*, *cooking (stirring)*	100.0	75.0
living room	*talking on phone*, *drinking water*, *talking on couch*, *relaxing on couch*	100.0	87.5
office	*talking on phone*, *drinking water*, *writing on whiteboard*, *working on computer*	100.0	81.2

Discussion

The correct predictions for the S-Seen setting is unsurprising. The classifier has already seen the instance of the subject and it is not difficult to determine the class label here. The confusion matrix in Figure 5.7 provides the classification performance for each activity label with the S-New setting in which the subject is seen for the first time. As before, the diagonal entries show the percentage of true positives and the off diagonal entries how the percentage of incorrectly classified instances. It is evident from the confusion matrix that the classifier labels for the activities are accurate for most of the classes. However, there are instances that have been labelled incorrectly. For example, the two cooking activities are mislabelled in some instances. The mislabelling occurs when there are very similar motion patterns and the object information is not sufficient to distinguish between the activities. In some instances, the classification is also affected by the mechanism with which a subject performs an activity.

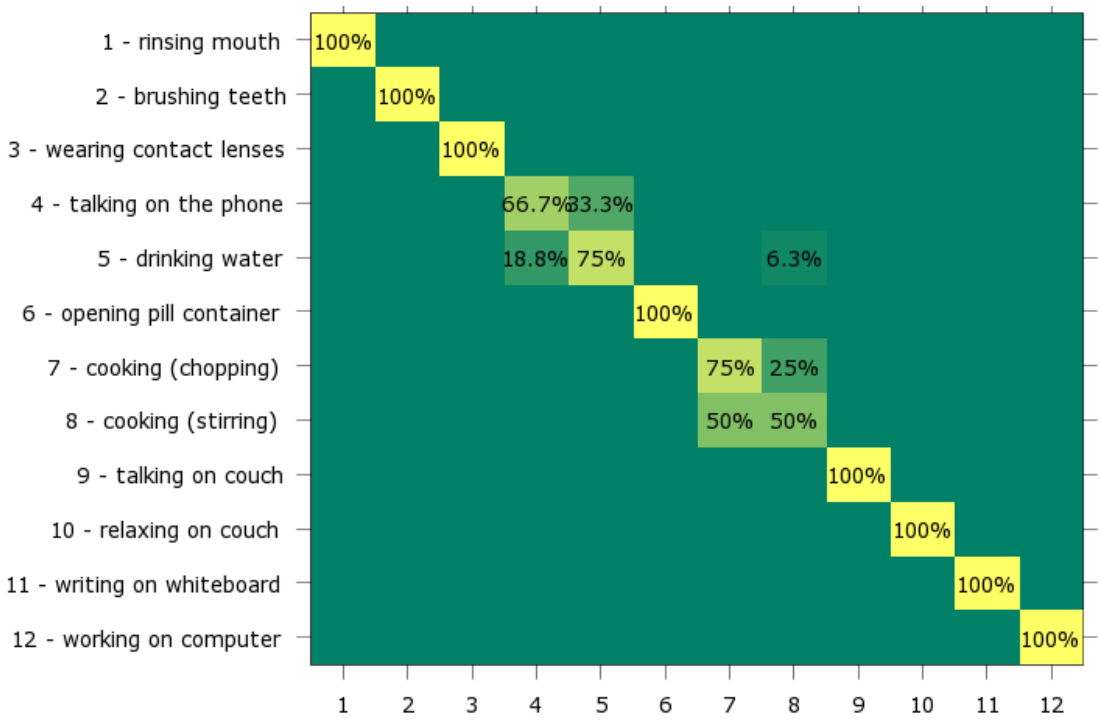


Figure 5.7: Cornell Activity dataset - Confusion matrix for the full model with the S-New setting.

Table 5.6: Summary of classification results for Cornell Activity dataset.

	Method	S-Seen Accuracy (%)	S-New Accuracy (%)
Previous Works	STIP [78]	N/A	62.5
	MEMM [85]	84.3	64.2
	Actionlet [88]	94.1	74.7
	Heterogeneous Features [89]	N/A	84.1
	Action Templates [90]	100	91.9
This work	Parametric HMM	62.5	47.7
	Parametric H-HMM	68.9	56.4
	Unshared Parameters	82.1	69.5
	Full Model	100	85.4

A summary of the classification results for this dataset is provided in Table 5.6. The under-performance of the parametric HMM and H-HMM is expected. The usefulness of the nonparametric prior and the factorized structure can be observed from the significantly improved accuracy for the Unshared Parameters case when compared with the outcomes of its parametric counter-part. In particular, the importance of using appropriate numbers of states is

evident. The use of a nonparametric prior clearly benefits the model. The full model evaluation that includes regression for classification with parameter sharing, outperforms the rest. When compared with previous works in the literature, the approach here is equivalent to the state-of-the-art for the setting where a subject has been seen. For the new subject setting, the method here outperforms all the other existing approaches except the action templates [90]. The work in [90] involves segmenting the actions and identifying key poses in advance before performing learning. There is a dependency on the data set being used for such an approach. Instead, the focus here is largely on a data/feature agnostic model. It is worth noting that without any explicit manual processing, results comparable to state-of-the-art are obtained. In addition, the method proposed here compares favourably in terms of complexity. For example, in [90] a parametric HMM must be trained for each action class and multiple times per class as part of state cardinality selection. In the SVMs used in [78], several parameters must be tuned in order to achieve optimal classifier performance. This would often involve training the SVMs multiple times with different values for these parameters. Such repeated training is not needed in the model proposed here.

5.5.2 HDM05 Dataset

The proposed model is also evaluated on a dataset for which the 3D body positions were obtained from a motion capture system. There are no depth images involved in this dataset. Motion capture data is widely used to analyse human motions in fields such as sports, biometrics and computer animation. The dataset uses an optical marker based technology. The actor is equipped with a set of 40 to 50 reflective markers attached to a suit. These markers are tracked by an array of high-resolution cameras (Figure 5.8). The 3D marker positions are reconstructed from the recorded 2D images of the marker positions.

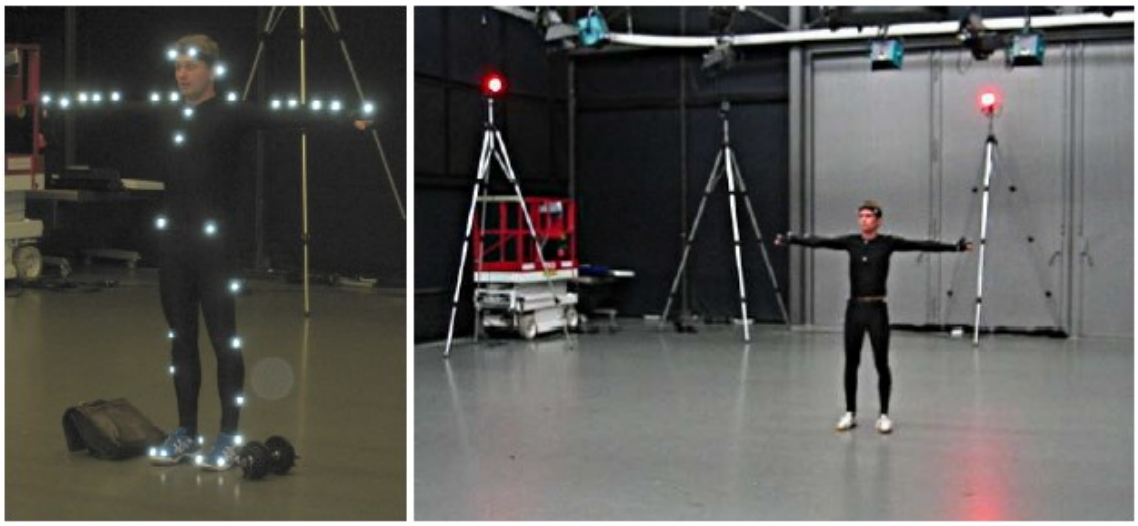


Figure 5.8: Motion capture system. *Left*: Several reflective markers are attached to a suit. *Right*: The markers are tracked by multiple cameras that are arranged in a circle [86].

The motion sequences in the dataset were performed by five different subjects. The subjects were given instructions on how to perform the motion and there are some free style sequences as well. The motion sequences are grouped into logical activities. The activities *badminton*, *dancing*, *grabbing/depositing* and *workout* are used to evaluate the model. Each activity contains sub-sequences of actions. For example, the *workout* activity involves jumping jacks, skiing exercise, elbow to knee exercise and squats. Figure 5.9 shows some example poses from this activity.

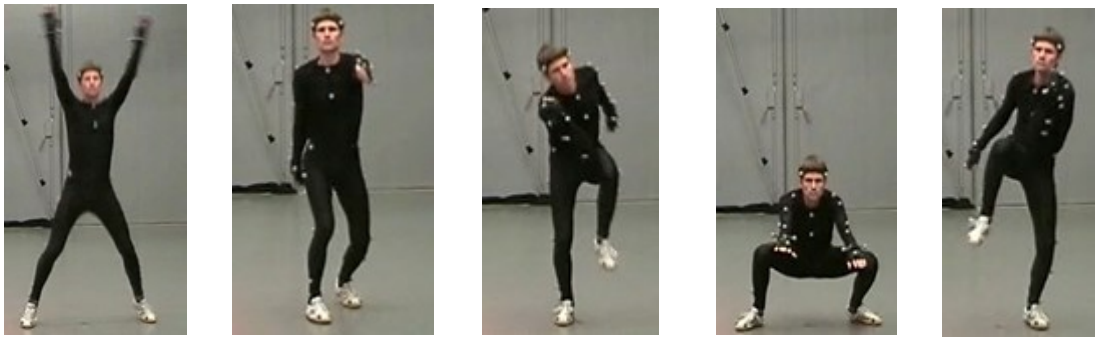


Figure 5.9: HDM05 dataset samples. Some example poses in the HDM05 [86] dataset for the *workout* activity are shown.

The C3D format data in which bone lengths are not normalized is used. The frames in the sequences contain 44 3D joint positions corresponding to the markers. Since there are no depth images, the object pose is assumed to be unavailable and the object related parameters are dropped. Only the skeleton features are used. In general, the joint positions available from the motion capture based dataset are of better quality with less errors when compared with the joint positions estimated from the depth images. The upper bounds on the number of actions and skeleton poses are set to large values as discussed above.

Table 5.7: Summary of classification results for HDM05 dataset.

Method	Accuracy (%)
Parametric HMM	70.9
Parametric H-HMM	74.2
Unshared Parameters	85.0
Full Model	92.5
SMIJ [153]	91.5
Cov3DJ [158]	95.4

Figure 5.10 shows the poses corresponding to the various states during a sampling iteration for the *badminton* activity. It can be seen that the poses appear similar for the states with the same colour indicating a clustering behaviour. This clustering behaviour is expected in an HMM based model. A summary of the classification results is presented in Table 5.7. As before, the activity classes are evaluated for parametric HMM and parametric H-HMM. In both these parametric settings, different numbers of states namely 5, 10, 25 and 50 were tried similar to the Cornell Activity dataset. In the Unshared Parameters setting, yet again a nonparametric prior was used with the number of states automatically inferred from data. However, the factorized setting is not applicable here because the object pose states are not used in this dataset. A classification accuracy of ***92.5%*** was observed for this dataset. This high accuracy confirms the applicability of the model for data that does not include depth images. A comparison of the accuracy with the works in [153] and [158] are also provided. It must be noted that both [153] and [158] do not consider complex activities such as *workout* and *dance* used in the experiments here. They focus rather on simple actions such as *throw*, *sit down* etc.

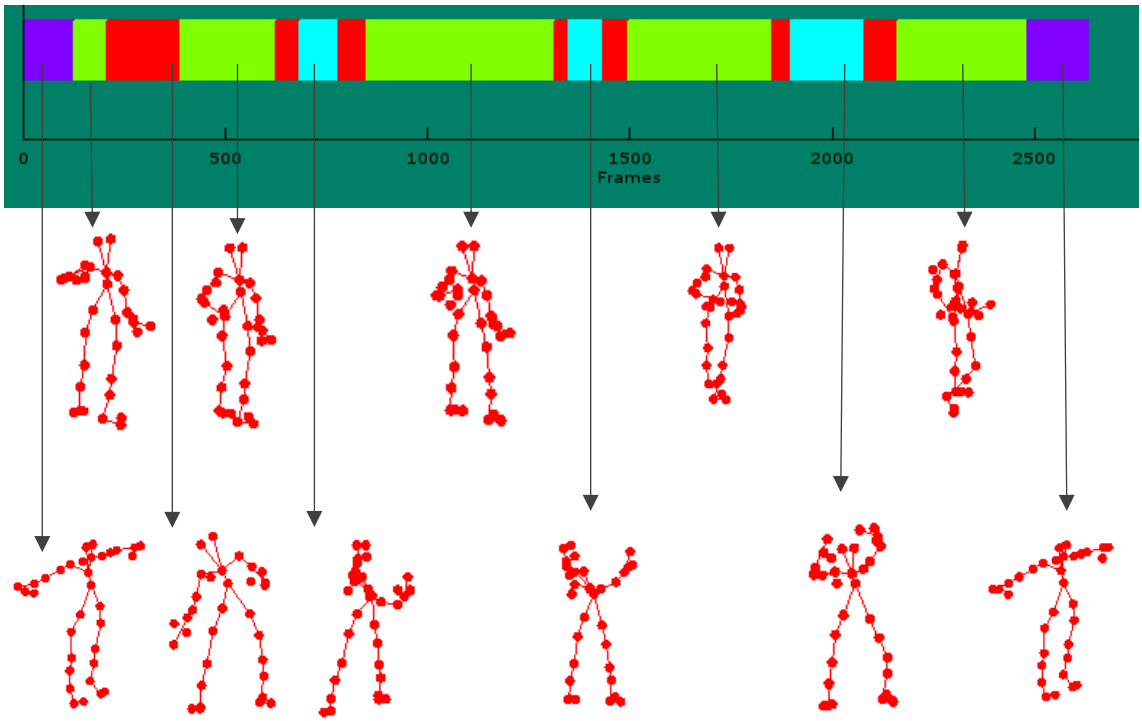


Figure 5.10: Pose clustering. The similarity in the 3D poses for the various states is shown. These states were sampled during an iteration for the *badminton* activity. The poses appear similar for the same state indicating a clustering effect that is expected with HMM based models. Not all the 44 joint positions are shown in the skeleton model. The joint positions corresponding to the markers behind the body are omitted.

5.6 Conclusion

This chapter has proposed a nonparametric hierarchical HMM that is suitable for classifying human activities. An activity was decomposed into a set of coarse actions and an action in turn into a set of granular poses. This decomposition enabled information sharing by reusing the actions and poses across the activity classes. The activity labels were regressed on the actions in order to perform classification. This produces a simplified model. The hierarchical structure of the data was captured using a hierarchical HMM in which the numbers of action and pose states were not fixed in advance. This nonparametric extension allows the model to adapt to the size of the data. The flexible factorized formulation allows additional data channels to be incorporated seamlessly. An inference procedure that uses efficient block sampling to infer the posterior parameters was also provided. The generic model formulation is applicable for the classification of a wide variety of sequential data that exhibits a hierarchical structure.

The experiments conducted on the activity datasets demonstrate the efficacy of the approach. The model produced good classification results both for the depth channel based videos and the motion capture based videos. However, the model did misclassify some activity labels. The activities that have very similar motion patterns and overlapping poses are affected. The model may benefit if the objects that a subject interacts with are detected and this contextual information is included. The occlusions of poses and poses out of view are not handled by the model. These cases can be addressed by improving the feature extraction procedure and introducing new variables at the cost of increased complexity.

The hierarchical HMM representation here has only two levels. In some applications, the hierarchical structure might span more than two levels. As an example, the activities themselves may be grouped into an even higher level construct such as events. The above model must then be extended to include these additional levels. The state sampling procedure that uses the forward-backward algorithm may become intractable for multiple hierarchical levels. Alternative approximation methods would then be required. The hierarchical HMM proposed in this chapter is sufficiently expressive for the two level representation of activities.

6. Nonparametric HCRF

Conditional Random Fields (CRF), a structured prediction method, combines probabilistic graphical models and discriminative classification techniques in order to predict class labels in sequence recognition problems. Its extension, the Hidden Conditional Random Fields (HCRF) method uses hidden state variables to capture intermediate structures. The number of hidden states in an HCRF must be specified a priori even though this number is not known in advance. This chapter proposes a nonparametric extension to the HCRF, in which the number of hidden states is automatically learned from data. The proposed model is applied to the classification of human actions in depth videos.

The chapter begins with the motivation for a discriminative learning method and an overview of the proposed model in Section 6.1. The parametric HCRF is then introduced in Section 6.2. The nonparametric extension to the HCRF makes use of the Hierarchical Dirichlet Process (HDP) to produce priors over the HCRF parameters. This formulation is explained in Section 6.3. The training and inference procedures in the proposed model are fully Bayesian, eliminating the over fitting problem associated with frequentist methods. The posterior HCRF parameters are obtained using elliptical slice sampling, which is a Markov Chain Monte Carlo (MCMC) method. Section 6.4 describes this MCMC inference procedure. The results for action classification experiments performed using the nonparametric HCRF model are furnished in Section 6.5. As in Chapters 4 and 5, the skeletal joint positions extracted from depth image sequences are used to characterize the human actions. Section 6.6 contains some concluding remarks. Portions of this chapter have been published [195].

6.1 Overview

The objective in classification problems is to predict a class label given an input vector of features. One approach to this problem is to describe the distribution of the input conditional on the label and during prediction select the label by evaluating these distributions. More accurately, a model of the joint probability distribution $p(x, y)$ over the input x and label y is learnt during training. When making predictions, the most likely label is selected by using Bayes theorem to calculate $p(y|x)$ as $p(x|y)p(y)$[2]. Indeed this was the approach used in Chapter 4 where the nonparametric HMM parameters corresponding to each action class were learnt

[2] $p(y|x) = \frac{p(x,y)}{p(x)} \propto p(x|y)p(y)$

during training and a test observation was labelled by evaluating a set of class conditional likelihoods, one for each possible label.

A drawback of this generative approach is the need to construct a probability distribution over the input x. The input features often have complex dependencies and modelling these dependencies is difficult. It is arguable that $p(y|x)$ is all that is needed for classification, so why solve a more general problem by modelling the intermediate $p(x|y)$. An alternate to the generative approach is the discriminative approach in which the conditional distribution $p(y|x)$ is modelled directly. An advantage of this conditional model is that the dependencies involving the input features are not modelled and the model is agnostic about the form of $p(x)$. Hence the conditional model can have a simpler structure when compared with a joint model. In classification problems, this discriminative approach often outperforms a generative approach [64, 94].

Conditional Random Field (CRF) [33] is a widely used discriminative model that is applicable to sequential data classification. It models the conditional distribution $p(y|x)$ directly where the input x and the output y are sequences. As seen in Chapter 3, the graphical models provide powerful representations that can capture the dependencies between large numbers of variables. The CRF uses undirected graphs to express the conditional independence relationships between the input and output variables. As it is a discriminative model, it can include a rich set of input features by allowing the output variables to depend on several local features. It can be viewed as a discriminative analogue of the Hidden Markov Models (HMMs).

A limitation of the CRF is that it cannot capture intermediate structures. For example in a sequence labelling problem such as human action classification, an action may be composed of intermediate poses that are not observed. It is useful to incorporate the pose structure in the model and study the pose dynamics. The Hidden Conditional Random Field (HCRF) [95] uses intermediate hidden state variables in order to model the latent structure of the input observations. The inclusion of these hidden state variables produces a model that is more expressive than the canonical CRF. In HCRF, a joint distribution over the class label and the hidden state variables conditioned on the input observations is used. The dependencies between the hidden variables are expressed by an undirected graph as in the CRF. Typically the graph is assumed to be a linear chain for tractable inference.

The cardinality of the hidden states in a HCRF is fixed in advance. As discussed in Chapter 4 and 5, this is a problem in general with all latent variable graphical models where the number of hidden states must be specified a priori, even though this number is not known. This problem is evident in action classification, where the number of intermediate poses depends on complexity

of the action. The exact number of poses is not available. The usual model selection techniques are avoidable if the number of hidden states is automatically inferred from the data. In this chapter, a nonparametric extension to the HCRF is proposed. The Hierarchical Dirichlet Process (HDP) [44] is used to construct a nonparametric model with an unbounded number of states.

The general training procedure for CRFs and HCRFs is to maximize the conditional (log) likelihood using iterative scaling or quasi-Newton gradient descent methods [96]. However these procedures are prone to over fitting especially if there are large numbers of correlated features [97]. In a high dimensional setting these point estimates often break down. In contrast, estimating the posterior distribution of the HCRF parameters provides a realistic characterization of uncertainty in the parameter estimates and addresses over fitting [98].

The nonparametric HCRF proposed here follows a fully Bayesian training and inference procedure. In particular, the HCRF parameters are assigned a normal scale mixture prior. This includes a global scale that is common to all parameters and local scales that allow deviations from the global scale for each parameter. One of the local scale parameters follows an exponential distribution, resulting in a sparsity inducing Laplacian prior for the parameters. Another local scale parameter is assigned a HDP prior which ensures that only a subset of the unbounded number of hidden states are actually used. Elliptical slice sampling [67], a Markov Chain Monte Carlo (MCMC) technique, is used to sample the posterior parameters. This hierarchical Bayesian model, with a HDP-Laplace prior for the HCRF parameters, produces optimal and sparse posterior estimates.

Contributions

The main contribution in this chapter is the definition of a fully Bayesian nonparametric HCRF model. A tractable inference procedure that produces sparse and optimal posterior samples is also derived. The proposed model has the following advantages:

(a) The discriminative approach in a CRF, which models a direct mapping from the inputs to the class labels, is suitable for classification tasks.
(b) The use of a nonparametric HDP prior precludes the need to fix in advance the number of hidden states in an HCRF.
(c) The estimation of a posterior distribution rather than point estimates for the HCRF parameters eliminates potential over fitting.
(d) The model is generic and is applicable to other sequence labelling problems.

6.2 Parametric HCRF

Recall the CRF defined in Section 3.2 that considers distributions over sequences of input and output variables. Let $\boldsymbol{X}$ denote a sequence of input variables and $\boldsymbol{Y}$ denote a sequence of discrete valued output variables. Both $\boldsymbol{X}$ and $\boldsymbol{Y}$ are of length T. An assignment to $\boldsymbol{X}$ is denoted by $\boldsymbol{x}$ and to $\boldsymbol{Y}$ by $\boldsymbol{y}$ with x_t and y_t denoting the value at time t, $\forall\, 1 \leq t \leq T$. In a linear chain CRF, the conditional distribution takes the form

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \prod_{t=1}^{T} \psi_t(y_t, y_{t-1}, t, \boldsymbol{x}) \tag{6.1}$$

where ψ is a function with values in $\mathbb{R}$, often referred as the potential function and $Z(\boldsymbol{x})$ is a normalization constant that sums over all possible values of $\boldsymbol{y}$.

$$Z(\boldsymbol{x}) = \sum_{\boldsymbol{y}} \prod_{t=1}^{T} \psi_t(y_t, y_{t-1}, t, \boldsymbol{x}) \tag{6.2}$$

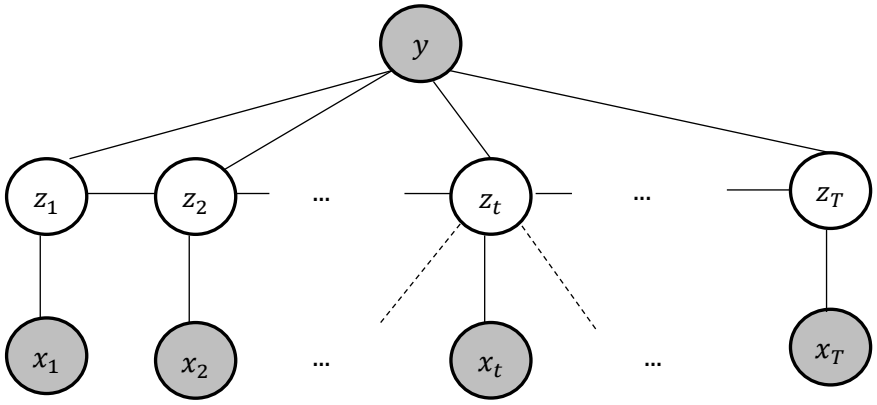


Figure 6.1: Graphical representation of a HCRF [95]. The input sequence x_1 to x_T has associated hidden states z_1 to z_T that follow a Markov assumption. There is a single output variable y. The dashed edges illustrate that the state at a time instant t may depend on input observations at time instants other than t.

In an HCRF, for an input sequence $\boldsymbol{x}$ there is an associated sequence $\boldsymbol{z}$**.** Each discrete valued $z_t, \forall\, 1 \leq t \leq T$ is a member of a finite set of states. The z_t values are not observed and represent one of the possible hidden states associated with an input observation. The outputs $\boldsymbol{y}$ are assumed to have identical values for the entire input sequence and the output now contains a single discrete valued y. The conditional distribution for the HCRF takes the form

$$p(y|\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(y, \boldsymbol{z}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \sum_{\boldsymbol{z}} \prod_{t=1}^{T} \psi_t(y, z_t, z_{t-1}, t, \boldsymbol{x}) \tag{6.3}$$

with the summation in the normalization constant $Z(\boldsymbol{x})$ now including the possible values of both the output y and the hidden states $\boldsymbol{z}$.

$$Z(\boldsymbol{x}) = \sum_{y,\boldsymbol{z}} \prod_{t=1}^{T} \psi_t(y, z_t, z_{t-1}, t, \boldsymbol{x}) \tag{6.4}$$

As in the CRF, the potential functions are log linear over a set of problem specific feature functions $\{\varphi_l\}_{l=1}^{L}$. The feature functions are now defined by the input observations, the labels and the state values at the current and previous time instants.

$$\psi_t(y, z_t, z_{t-1}, t, \boldsymbol{x}) = \exp\left\{\sum_{l=1}^{L} \theta_l \varphi_l(y, z_t, z_{t-1}, t, \boldsymbol{x})\right\} \tag{6.5}$$

The model parameters $\boldsymbol{\theta}$ are a set of real valued weights with $\boldsymbol{\theta} = \{\theta_l\}_{l=1}^{L}$. A graphical representation of the HCRF is provided in Figure 6.1.

6.3 Model

The model proposed here differs from the standard HCRF because of the Bayesian extension. In particular, instead of the usual isotropic Gaussian prior for the HCRF parameters, a scale mixture prior with a vector of local scales is used. This prior formulation for the parameters that produces a nonparametric HCRF is presented in Section 6.3.2 and Section 6.3.3.

Assume that a set of N training pairs $\{(\boldsymbol{x}^n, y^n)\}_{n=1}^{N}$ is given. As discussed above, $\boldsymbol{x} = \{x_t\}_{t=1}^{T}$ is an observation sequence, $y \in \{1 \dots c \dots C\}$ is its corresponding label and $\boldsymbol{z} = \{z_t\}_{t=1}^{T}$ is the hidden state sequence with $z_t \in \{1 \dots k \dots K\}$. Here K is the number of hidden states. In an action classification problem $\boldsymbol{x}$ is the action sequence, y is the action class label and $\boldsymbol{z}$ is the intermediate pose sequence.

6.3.1 Parameters

Before defining the nonparametric HCRF, it is useful to define the form of the feature functions φ and formulate the set of parameters. There are several possibilities. In the linear chain HCRF used in this work, the feature functions encode three different dependencies.

(a) The first dependency is between a label and a hidden state. It is denoted as φ^{LBL} and is given by

$$\varphi_{ck}^{LBL}(y, z_t, z_{t-1}, t, \boldsymbol{x}) = \mathbb{I}(y = c)\mathbb{I}(z_t = k) \qquad \begin{matrix} c = 1 \dots C \\ k = 1 \dots K \end{matrix} \tag{6.6}$$

where $\mathbb{I}(a = b)$ is an indicator function that evaluates to 1 if $a = b$, 0 otherwise.

(b) The second dependency is between a label, and the hidden states at current and previous time instants. It is denoted as φ^{TRN} and is given by

$$\varphi_{cjk}^{TRN}(y, z_t, z_{t-1}, t, \boldsymbol{x}) = \mathbb{I}(y = c)\mathbb{I}(z_{t-1} = j)\mathbb{I}(z_t = k) \qquad \begin{matrix} c = 1 \dots C \\ j, k = 1 \dots K \end{matrix} \tag{6.7}$$

(c) The third dependency is between a label, a hidden state and an observation. It is denoted as φ^{OBS}. Let the feature at a time instant be a D dimensional vector with $x_t = (x_{t,1}, \dots x_{t,d} \dots x_{t,D})$. This feature function is defined as

$$\varphi_{ckd}^{OBS}(y, z_t, z_{t-1}, t, \boldsymbol{x}) = \mathbb{I}(y = c)\mathbb{I}(z_t = k)x_{t,d} \qquad \begin{matrix} c = 1 \dots C \\ k = 1 \dots K \\ d = 1 \dots D \end{matrix} \tag{6.8}$$

The model is parameterized by three different parameter groups namely θ^{LBL}, θ^{TRN} and θ^{OBS} corresponding to φ^{LBL}, φ^{TRN} and φ^{OBS} respectively. With this definition, the potential function in (6.5) is written as

$$\psi_t(y, z_t, z_{t-1}, t, \boldsymbol{x}) = \exp\left\{\theta_{y,z_t}^{LBL} + \theta_{y,z_{t-1},z_t}^{TRN} + \sum_{d=1}^{D} \theta_{y,z_t,\boldsymbol{d}}^{OBS}\, x_{t,d}\right\} \tag{6.9}$$

Note that a parameter group contains a vector of parameters. Intuitively, a parameter in the θ^{LBL} group measures the compatibility between a hidden state and a class label. The number of parameters in θ^{LBL} is $C \times K$. The parameters in the θ^{TRN} group are analogous to the entries in an HMM transition matrix. However, there is a separate transition matrix for each class label and the number of parameters in θ^{TRN} is $C \times K \times K$. Finally, a parameter in the θ^{OBS} group measures the compatibility between a label, a hidden state and a feature. In the CRF based models, the feature vector at a time instant may depend on observations at any previous or future time instants. Hence the x_t term in (6.9) is problem specific. It is assumed here to be a vector of length D and θ^{OBS} contains $C \times D \times K$ parameters. The set of all model parameters is now $\boldsymbol{\theta} = \{\, \theta^{LBL} \cup \theta^{TRN} \cup \theta^{OBS}\} = \{\theta_l\}_{l=1}^{L}$.

Estimating the parameters $\boldsymbol{\theta}$ from the training data is the central computation problem in a HCRF model. The conventional approach is to choose a point estimate $\boldsymbol{\theta}^*$ of the parameters such that the training data has the highest probability under the model as in (6.11). It is also common to introduce a penalty on the parameters whose norm is large. The following regularized log likelihood function is often maximized in a HCRF.

$$L(\boldsymbol{\theta}) = \sum_{n=1}^{N} \log p(y^n \mid \boldsymbol{x}^n;\, \boldsymbol{\theta}) - \frac{1}{2\sigma^2}\|\boldsymbol{\theta}\|^2 \tag{6.10}$$

$$\boldsymbol{\theta}^* = \operatorname*{argmax}_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) \tag{6.11}$$

Here σ^2 controls the importance of the regularization term and the conditional likelihood $p(y \mid \boldsymbol{x})$ is as defined in (6.3). There is no closed form solution for maximizing this function and in practise numerical optimization methods such as gradient ascent are used to search for the optimal parameter values.

6.3.2 Bayesian Extension

As observed in Appendix C.2, the Bayesian approach provides a realistic characterization of the uncertainty in the parameter estimates by obtaining posterior distributions for the parameters rather than point estimates. In the Bayesian extension to HCRF, the parameters must be assigned a prior distribution. A straight-forward choice is a set of independent Gaussian priors over the parameters. In fact, the parameters obtained based on the regularized likelihood function in (6.10) can be understood as having a Gaussian prior with zero mean and σ^2 variance. However, as it will become evident below, it is essential to look beyond this Gaussian prior to construct a nonparametric model.

The number of parameters in a HCRF is typically large. For instance, the number of parameters L in the HCRF that has a potential function of the form in (6.9) is $(C \times K) + (C \times K \times K) + (C \times D \times K)$. In such high dimensional settings, it is preferable to encourage sparsity in the parameter estimates. A L_1 norm penalty for the parameters rather than the L_2 norm used in (6.10) often produces such sparse estimates in the learned parameters with many of the parameter values being close to zero. The L_1 norm penalty corresponds to a Laplacian prior. Hence the parameters are modelled as a Laplacian distribution with a scale parameter σ here.

$$\theta_l \sim DE(\sigma) \qquad l = 1 \ldots L \tag{6.12}$$

Here $DE(a)$ denotes a zero mean Laplace or Double Exponential distribution that has a density that is concentrated near zero and has heavy tails. The probability density function of a Laplace distributed variable χ with scale a is given by

$$p(\chi; a) = \frac{1}{2a} \exp\left\{\frac{-|\chi|}{a}\right\} \qquad \forall\, \chi \in \mathbb{R} \qquad (6.13)$$

The Laplace distributed variable in (6.12) can be written equivalently as a variable drawn from a global-local scale mixture of normal distribution [99, 100] as follows.

$$\theta_l \sim \mathcal{N}(0, \phi_l\, \sigma^2) \qquad l = 1 \ldots L \qquad (6.14)$$

$$\phi_l \sim Exp(1/2) \qquad l = 1 \ldots L \qquad (6.15)$$

The global scale σ^2 is common to all the parameters. The local scale ϕ_l is specific to each parameter and is drawn from the exponential distribution $Exp(a)$ that has a probability density function $p(\chi; a) = ae^{-a\chi}$ for any χ that is real valued and positive. This formulation as a scale mixture prior is useful when performing posterior inference.

6.3.3 Nonparametric HCRF

In order to construct a nonparametric HCRF, the global scale is first reformulated as a vector of scales. Let $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_l, \ldots \eta_L)$ denote a L dimensional vector. The single global scale σ^2 in (6.14) is replaced by the L length vector $(\eta_1\sigma^2, \ldots \eta_L\sigma^2)$. The parameters are now modelled as follows.

$$\theta_l \sim \mathcal{N}(0, \phi_l\eta_l\, \sigma^2) \qquad l = 1 \ldots L \qquad (6.16)$$

$$\phi_l \sim Exp(1/2) \qquad l = 1 \ldots L \qquad (6.17)$$

$$\boldsymbol{\eta} \sim F(\zeta) \qquad (6.18)$$

Here F denotes a distribution in which the draws from the distribution produce an L length vector and ζ is the distribution's parameters. In [98], F is the Dirichlet distribution and the above construct is used as a shrinkage prior for regression parameters. The introduction of the $\boldsymbol{\eta}$ variables offers an additional mechanism to control the parameter weights. These variables will be exploited below to derive the nonparametric extension.

In the nonparametric HCRF, the number of hidden states K is unbounded. Consequently, the total number of parameters L is unbounded as well. As discussed in Section 3.3.1, a draw from

a Dirichlet Process (DP) provides a probability distribution with infinitely many atoms. The probability distribution chosen at random via the stick breaking process can be used to produce a vector of unbounded length. However, a straight forward application of the DP prior to the $\boldsymbol{\eta}$ variable is not suitable. Recall that there are three different parameter groups namely θ^{LBL}, θ^{TRN} and θ^{OBS}. Each of these parameter groups are unbounded when there is no upper bound on K.

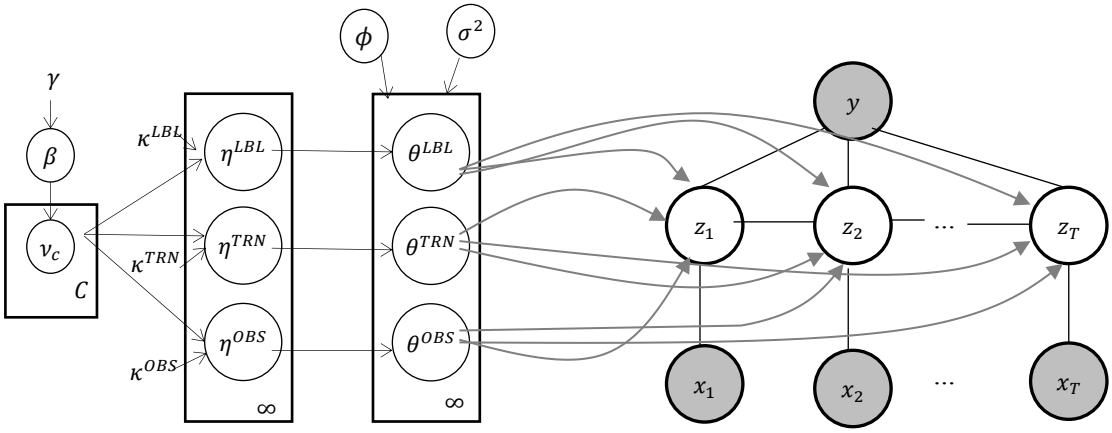


Figure 6.2: Graphical representation of a Bayesian nonparametric HCRF. A linear chain HCRF with the input sequence $\boldsymbol{x}$, the hidden states $\boldsymbol{z}$ and the output y are shown in the right side. The three parameter groups θ^{LBL}, θ^{TRN} and θ^{OBS} and their priors are shown on the left side. The number of hidden states and the number of parameters are unbounded. The priors on the left side makes the representation here different from the canonical HCRF.

The Hierarchical Dirichlet Process (HDP) models groups of data by assigning a separate DP prior to each group and linking all these DPs through a global DP. The HDP prior provides the necessary flexibility in this case. By assigning separate DP priors to each of the parameter groups and linking them through a global DP, the grouped structure in the parameters can be exploited. Intuitively, there is an overall probability for being in a hidden state k, but this probability may be different for classes c and c'. Further, within the same class label c, these probabilities may be different for the θ^{TRN} and the θ^{OBS} parameter groups.

Let $\boldsymbol{\eta}$ be decomposed into $\boldsymbol{\eta} = (\eta^{LBL}, \eta^{TRN}, \eta^{OBS})$ where η^{LBL}, η^{TRN} and η^{OBS} are vectors themselves and their lengths correspond to θ^{LBL}, θ^{TRN} and θ^{OBS} respectively. Let $\kappa^{LBL}, \kappa^{TRN}$ and κ^{OBS} be positive real numbers. The HDP prior is defined as follows.

$$\beta \mid \gamma \sim GEM(\gamma) \tag{6.19}$$

$$\nu_c \mid \beta \sim DP(\alpha, \beta) \qquad c = 1 \dots C \tag{6.20}$$

$$\eta_c^{LBL} \mid \nu_c \sim DP(\kappa^{LBL}, \nu_c) \qquad c = 1 \dots C \tag{6.21}$$

$$\eta_{c,k'}^{TRN} \mid \nu_c \sim DP(\kappa^{TRN}, \nu_c) \qquad \begin{matrix} c = 1 \dots C \\ k' = 1,2, \dots \end{matrix} \tag{6.22}$$

$$\eta_{c,d}^{OBS} \mid \nu_c \sim DP(\kappa^{OBS}, \nu_c) \qquad \begin{matrix} c = 1 \dots C \\ d = 1 \dots D \end{matrix} \tag{6.23}$$

The above equations define probability distributions on the set of positive integers based on HDPs. The global DP in (6.19) links together the class specific independent DPs in (6.20) which in turn link the DPs in (6.21) to (6.23) resulting in a two-level HDP. The overall probability for being in a state k is provided by β_k in (6.19) while the probability of state k for a class c is given by $\nu_{c,k}$ in (6.20). The probability of being in a state for the scale variables corresponding to LBL, TRN and OBS groups is given by the DPs in (6.21) to (6.23). For example, $\eta_{c,d,k}^{OBS}$ is the probability of observing state k for the d^{th} dimensional feature of class c. The hyper-parameters $\gamma, \alpha, \kappa^{LBL}, \kappa^{TRN}$ and κ^{OBS} are the DP concentration parameters. Let the $\boldsymbol{\eta}$ variables obtained using the above mechanism be written compactly as $HDP(\gamma, \alpha, \kappa^{LBL}, \kappa^{TRN}, \kappa^{OBS})$.

The nonparametric HCRF is defined by using the HDP-Laplace prior on the HCRF parameters. The use of the HDP prior allows an unbounded number of hidden states in the HCRF and the Laplacian prior produces sparse parameter estimates. A graphical representation of the full model is shown in Figure 6.2. The full Bayesian hierarchical model that defines the prior on the HCRF parameters is formulated as follows.

$$\theta_l \mid \phi_l, \eta_l, \sigma^2 \sim \mathcal{N}(0, \phi_l \eta_l \, \sigma^2) \qquad l = 1 \dots L \tag{6.24}$$

$$\phi_l \sim Exp(1/2) \qquad l = 1 \dots L \tag{6.25}$$

$$\boldsymbol{\eta} \sim HDP(\gamma, \alpha, \kappa^{LBL}, \kappa^{TRN}, \kappa^{OBS}) \tag{6.26}$$

$$\sigma^2 \sim Inverse\ Gamma(a, b) \tag{6.27}$$

6.4 Posterior Inference

It is not possible to compute the posterior distribution over the HCRF parameters analytically and an approximate inference technique such as Markov Chain Monte Carlo (MCMC) must be used. The Gibbs sampling procedure discussed in Appendix E.2 is applied here. The proposed sampler alternates between sampling the hidden state sequence given the parameters, sampling the parameters given the scale mixture variables and sampling the scale mixture variables given the parameters and hidden states.

6.4.1 Hidden state sequence sampling

The forward-backward algorithm [32] discussed in Section 3.1 provides an exact procedure to sample the hidden state sequence. However, it cannot be applied to the nonparametric extension with an unbounded number of hidden states unless a truncated approximation [71] is employed. The weak limit approximation to the Dirichlet Process [76], discussed in Chapters 4 and 5 is used here as well. The number of expected states is set to a large value and as this number tends to infinity, the approximation becomes more and more accurate. Specifically, the stick breaking weights obtained in (6.19) to (6.23) are approximated as follows.

$$\beta \mid \gamma \sim Dir\left(\frac{\gamma}{K}, \dots \frac{\gamma}{K}\right) \tag{6.28}$$

$$\nu_c \mid \beta \sim Dir(\alpha\beta_1, \dots \alpha\beta_K) \qquad c = 1 \dots C \tag{6.29}$$

$$\eta_c^{LBL} \mid \nu_c \sim Dir\left(\kappa^{LBL}\nu_{c,1} \dots \kappa^{LBL}\nu_{c,K}\right) \qquad c = 1 \dots C \tag{6.30}$$

$$\eta_{c,k'}^{TRN} \mid \nu_c \sim Dir\left(\kappa^{TRN}\nu_{c,k',1} \dots \kappa^{TRN}\nu_{c,k',K}\right) \qquad \begin{matrix} c = 1 \dots C \\ k' = 1,2, \dots \end{matrix} \tag{6.31}$$

$$\eta_{c,d}^{OBS} \mid \nu_c \sim Dir\left(\kappa^{OBS}\nu_{c,d,1} \dots \kappa^{OBS}\nu_{c,d,K}\right) \qquad \begin{matrix} c = 1 \dots C \\ d = 1 \dots D \end{matrix} \tag{6.32}$$

Here Dir is the Dirichlet distribution and K is an upper bound on the number of hidden states that is set to a large value. The prior induced by HDP ensures that only a subset of states from the K possible states are actually used. This approximation allows sampling from the entire model as though it were finite. As $K \to \infty$, the marginal distribution approaches the distribution obtained from the DP.

With this weak limit approximation, the hidden state sequence of the HCRF in (6.3) can be efficiently block sampled using the forward-backward procedure. Given the parameters $\boldsymbol{\theta}$, the potential function defined in (6.9) is used to obtain the forward and backward messages. The hidden state sequence $\boldsymbol{z}$ is then sampled using these messages. Based on $\boldsymbol{z}$, a set of count matrices N^{LBL}, N^{TRN} and N^{OBS} that maintain the number of hidden states visited for each parameter group is computed. For example, $N^{TRN} \in \mathbb{Z}^{C\times K\times K}$ records the number of transitions from hidden state k' to k for a class label c. These matrices are necessary for collecting the posterior samples of the $\boldsymbol{\eta}$ variables that have a HDP prior.

6.4.2 Sampling parameters

The parameters are sampled given the scale mixture variables ϕ, $\boldsymbol{\eta}$ and σ^2. There is no closed form solution for obtaining the posterior parameters $\boldsymbol{\theta}$ based on the conditional likelihood function in (6.3). The slice sampling [39] procedure discussed in Appendix E.3 provides a

mechanism to sample from the probability density function when the density function is known up to a scale factor. However, this method is difficult to employ in this case since the target variable $\boldsymbol{\theta}$ is in a high dimensional space.

The Elliptical slice sampling [67] procedure discussed in Chapter 4, provides a better sampling procedure if the target variable has a zero mean Gaussian prior. The prior on the parameters here is a normal scale mixture prior and it is straight-forward to employ the Elliptical slice sampling procedure to obtain the posterior values of $\boldsymbol{\theta}$ based on the conditional likelihood function $p(y|x;\boldsymbol{\theta})$ in (6.3).

6.4.3 Sampling scale variables

The scale mixture variables are sampled one at a time. First the ϕ values are sampled given $\boldsymbol{\theta}$, $\boldsymbol{\eta}$ and σ^2. The conditional posterior of ϕ can be block sampled efficiently by independently sampling each ϕ_l from an Inverse Gaussian Distribution [98, 101] as in (6.33). The density function of an Inverse Gaussian distributed variable χ is given as $p(\chi;\mu) \propto exp\left\{-\frac{(\chi-\mu)^2}{2\mu^2\chi}\right\}$.

$$\phi_l \,|\boldsymbol{\theta}, \boldsymbol{\eta}, \sigma^2 \sim IG\left(\frac{\eta_l \sigma^2}{|\theta_l|}, 1\right) \tag{6.33}$$

The $\boldsymbol{\eta}$ variables, given the parameters, are sampled next using the count matrices N^{LBL}, N^{TRN} and N^{OBS} obtained when sampling the hidden states. The standard HDP posterior computation techniques [44, 76] discussed in Chapter 4, can be used to obtain first the global HDP variables β and ν and then η^{LBL}, η^{TRN} and η^{OBS} from these variables. Finally, the σ^2 variable has a conjugate prior and a technique similar to the one in [101] can be used to obtain the posterior estimates. The inference algorithm is outlined in Table 6.1.

6.4.4 Prediction

The posterior parameter samples collected during training are used during prediction. Let there be M posterior samples for $\boldsymbol{\theta}$. Given a new test sequence $\hat{x}$, and a posterior sample $\boldsymbol{\theta}_{(m)}$ the class label can be predicted as in (6.34). The mode of the class labels predicted for all the posterior samples can be used as the final label.

$$\hat{c} = \underset{y=1\ldots C}{\operatorname{argmax}}\, p(y|\hat{x};\boldsymbol{\theta}_{(m)}) \tag{6.34}$$

Table 6.1: Posterior Inference Algorithm

Input:	Training pairs of observation sequences and their corresponding class labels.
Output:	Samples of posterior parameters.

1. Sample initial values of the scale variables $\sigma^2, \beta, \nu, \boldsymbol{\eta}$ and ϕ from their respective distributions based on the hyper parameter values.
2. Sample the parameters $\boldsymbol{\theta}$ using elliptical slice sampling. The likelihood function in (6.3) and the potential function in (6.9) are used in this process. The prior for the parameters is determined from the scale variables as in (6.24).
3. For each training example, sample hidden state sequences $\boldsymbol{z}$ using the forward-backward algorithm. The $\boldsymbol{\theta}$ values sampled above are used in (6.9).
4. Compute the count matrices N^{LBL}, N^{TRN} and N^{OBS} from the sampled hidden state sequences.
5. Sample ϕ given the current values of the parameters $\boldsymbol{\theta}$ according to (6.33).
6. Sample the HDP variables β and ν followed by η^{LBL}, η^{TRN} and η^{OBS} and σ^2 as outlined in Section 6.4.2.
7. Sample the hyper parameters $\gamma, \alpha, \kappa^{LBL}, \kappa^{TRN}$ and κ^{OBS}.
8. Repeat from step (2) to collect more samples.

6.5 Experiments

The nonparametric HCRF model is evaluated on the recently released KARD [102] dataset. The dataset contains a variety of actions performed by a single human subject. The videos were captured using a Kinect sensor in an office scene that contains a desk, a phone, a coat rack and a waste bin. The distance between the subjects and the sensor was about 2 to 3 meters. The dataset is made up of 540 action sequences for about a total of one hours' worth of videos. The frame rate is 30 frames per second and the image resolution is 640x480 pixels. Both the RGB and depth image sequences are available, along with the estimated joint positions of the human skeleton.

The dataset contains 18 different actions – *horizontal arm wave, high arm wave, two hand wave, catch cap, high throw, draw x, draw tick, toss paper, forward kick, side kick, take umbrella, bend, hand clap, walk, phone call, drink, sit down* and *stand up*. Each action is performed by 10 different subjects and is repeated three times. One of the subjects was a female. Some example actions from this dataset is shown in Figure 6.3.

Each frame contains 15 3D joint positions with coordinates (x, y, z) in a world coordinate frame. In order to ensure invariance to uniform translation of the body, joint positions relative to the torso are used for computing the features. Although the HCRF model allows the hidden states to depend on observations from multiple frames, the joint positions from a single frame are used here. The total number of relative joint positions in a frame is 42. The feature vectors

computed from the joint positions relative to the torso are projected to a lower dimensional vector space using Principal Component Analysis (PCA) [35] and the projected vector is used as the final feature. The first d components that capture at least 90% of the total variance are used. The information in the depth channel or the RGB channel is not used in the experiments.

Figure 6.3: KARD dataset samples. *Top*: Actions *horizontal arm wave, forward kick, hand clap, drink* and *take umbrella* from the KARD dataset [102] are shown. The skeleton joint positions are overlaid on top of the depth images. *Bottom*: Some frames from the RGB video for the *catch cap* action is shown. Only the 3D joint positions are used in the experiments.

The experiments are conducted both for the setting in which an instance of the subject has already been seen during training and for the setting where the subject is new and is being seen for the first time during prediction. The former is referred as S-Seen while the latter as S-New. In the S-Seen setting, about two thirds of the instances corresponding to all subjects are used for training while in the S-New setting about 60% of the subjects are used for training. The rest of the training examples are used for testing in both the S-Seen and S-New settings. The actions are grouped into three different sets as in [102]. Table 6.2 shows the grouping. The three sets are in increasing order of difficulty with Set C, the most difficult one, containing actions that are very similar.

In the first iteration during posterior inference, all the hyper parameters are initialized from their respective priors. The HDP concentration parameters were all given a vague gamma prior similar to [44] and were re-sampled after each sampling iteration. The first 1500 samples were

discarded and a total of 100 samples were collected. The elliptical slice sampler had a further burn-in period of 10 iterations. Convergence was verified by examining the number of training pairs that were misclassified. The upper bound on the number of states was set to 40. During prediction, the class label is predicted for each posterior sample and the final label is selected based on the mode. The posterior inference procedure for the HCRF uses the forward-backward algorithm when block sampling the state sequence. This can be calculated in a time $O(CTK^2)$ where C is the total number of classes, T is the length of the sequence and K is the upper bound on the number of states.

Table 6.2: The three different action sets in the KARD dataset.

Set A	Set B	Set C
horizontal arm wave	*high arm wave*	*draw tick*
two hand wave	*side kick*	*drink*
bend	*catch cap*	*sit down*
phone call	*draw tick*	*phone call*
stand up	*hand clap*	*take umbrella*
forward kick	*forward kick*	*toss paper*
draw x	*bend*	*high throw*
walk	*sit down*	*horizontal arm wave*

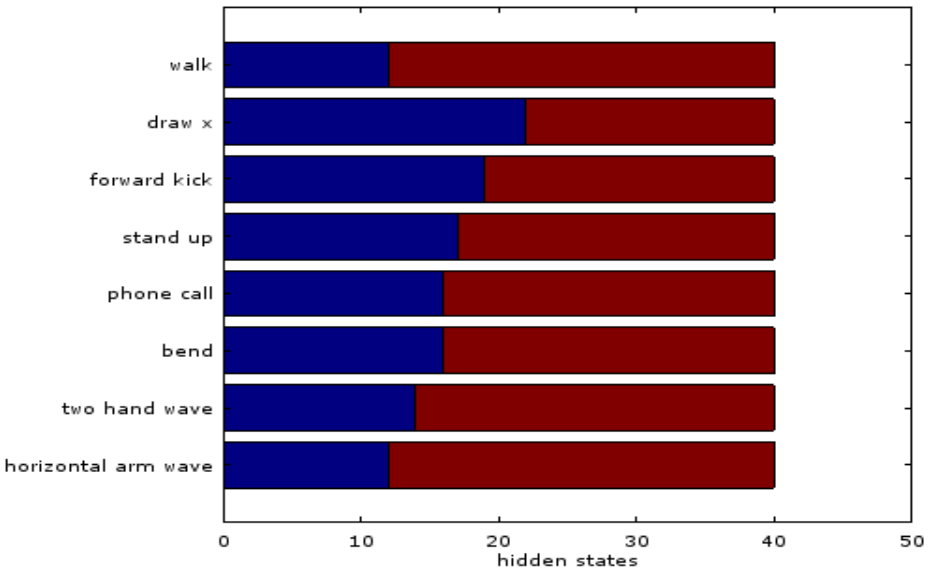


Figure 6.4: Hidden state instantiation. The number of hidden states that are actually used is smaller than the upper bound on the number of states (40 here). These states are from a posterior sample obtained when training Set A of the KARD dataset.

The number of hidden states that were instantiated during training was much smaller than the upper bound on the number of states. This fact is due to the nonparametric HDP prior that uses only a subset of the states to explain the training data. The actual numbers of states that were used for the different action classes in Set A during a sampling iteration are shown in Figure 6.4.

The effect of using the Laplacian prior can be seen in Figure 6.5. The figure shows a histogram of the parameter values in a posterior sample. It can be seen that many parameter values are close to zero indicating that sparse estimates are produced. This is desirable in a model such as HCRF that has a large number of parameters.

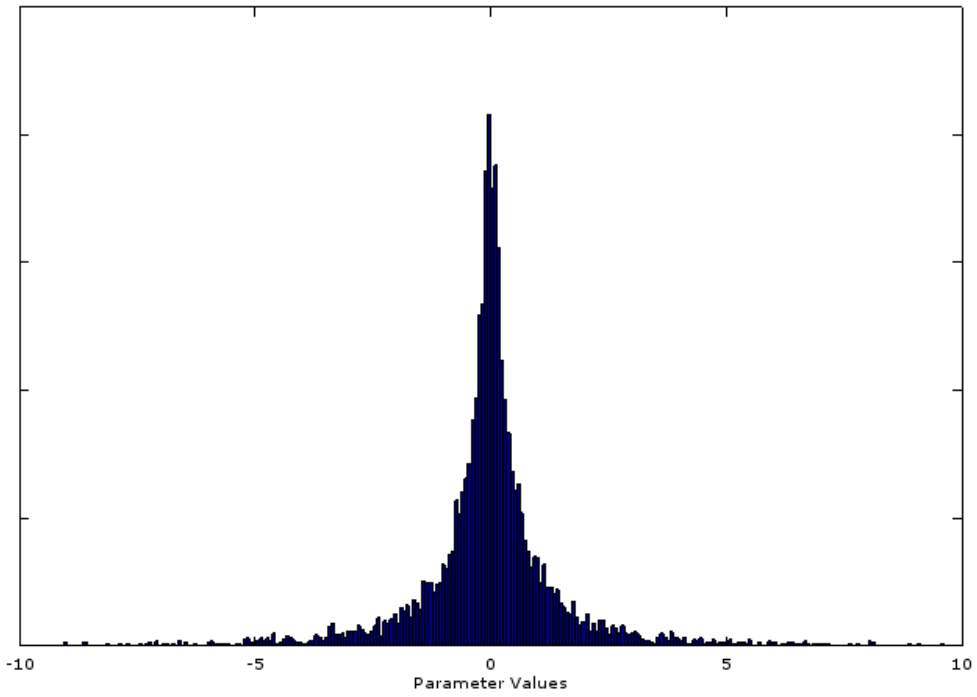


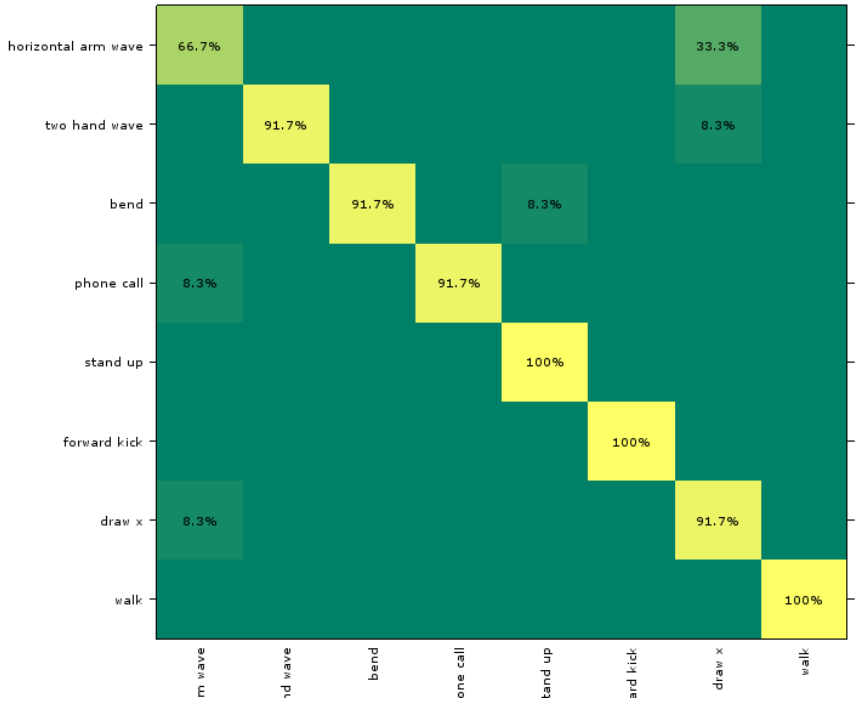
Figure 6.5: Histogram plot of the parameter values in a posterior sample. The sample was collected when training the actions in Set B for KARD dataset. The plot shows that most of the parameter values are concentrated near zero resulting in a sparse model.

Results

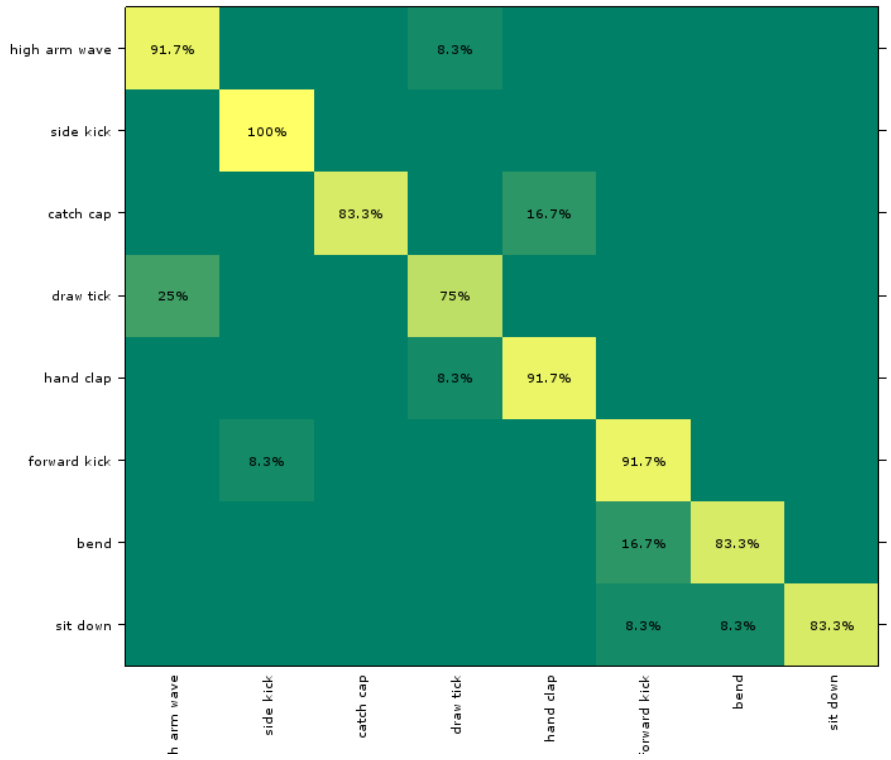
The classification results for this dataset are presented in Figure 6.6. The three confusion matrices correspond to the three different action sets Set A, Set B and Set C. As before, the diagonal entries show the percentage of correctly predicted examples. These results are for the S-New setting where the instance of the subject has not been seen during training.

The classification labels are mostly accurate for Set A. The largest error is due to the *horizontal arm wave* action being mislabelled as the *draw x* action. There are more misclassifications in Set B than in Set A. Actions that are incorrectly labelled include the *draw tick* action, which is confused with the *high arm wave* action and the *catch cap* action, which is confused with the

hand clap action. Finally, there are more off-diagonal entries in the results for Set C. This is understandable because this set is the most difficult of the three sets and contains many actions that involve similar sets of joints. The *draw tick* action is particularly challenging to classify since the arm movements in this action is very similar to a *wave* action.



(a)



(b)

(c)

Figure 6.6: KARD dataset classification results. The classification results for the dataset shown using a confusion matrix for the three different sets of action groupings. (a), (b) and (c) correspond to Set A, Set B and Set C respectively. The diagonal entries show the percentage of correct predictions for a particular class.

Table 6.3: Summary of classification results for the KARD dataset.

Method	Set	S-Seen Accuracy (%)	S-New Accuracy (%)
Posture Analysis [102]	Set A	95.1	93.0
	Set B	89.9	90.1
	Set C	84.2	81.7
This work	Set A	93.8	91.6
	Set B	92.7	87.5
	Set C	82.2	78.1

A summary of the classification results for this dataset is provided in Table 6.3. The accuracy for the S-Seen setting is better than that of S-New setting. This is expected because in the former

setting a subject performing an action has already been seen by the classifier during training. The latter is more challenging because the action style of a new subject may differ from the instances used when training the classifier. The overall classification accuracy is close to the state-of-the-art results reported in [102] even though the latter performs a more sophisticated posture analysis procedure. In [102], a multi-class Support Vector Machine (SVM) is used to first obtain a discriminative representation of the configuration of the joints and then a set of HMMs is trained for classification. The method proposed here does not perform any such explicit analysis on the joint positions and is intended to be more generically applicable. Further, the method proposed here compares favourably in terms of complexity with [102]. The training procedure in [102] involves training multiple SVMs and multiple HMMs. An exhaustive grid search on the parameter space using leave-one-out-cross-validation is employed. This involves repeating the training procedure multiple times using different combinations of parameters. Such a procedure is not necessary here.

Cornell Activity Dataset

The nonparametric HCRF model is also evaluated on the Cornell Activity [85] dataset discussed in Chapter 5. This dataset also contains videos obtained using the Kinect sensor. The activities in this dataset are *rinsing mouth*, *brushing teeth*, *wearing contact lens*, *talking on phone*, *drinking water*, *opening pill container*, *cooking (chopping)*, *cooking (stirring), talking on couch*, *relaxing on couch*, *writing on whiteboard* and *working on computer*. They are grouped into locations bathroom, bedroom, kitchen, living room and office and the classifier is trained on each group.

Table 6.4: Cornell activity dataset - Classification accuracy (in %) for the nonparametric HCRF.

Location	Activities	S-Seen	S-New
bathroom	*rinsing mouth. brushing teeth*, *wearing contact lens*	91.7	83.3
bedroom	*talking on phone*, *drinking water*, *opening pill container*	75.0	58.3
kitchen	*drinking water*, *opening pill container*, *cooking (chopping)*, *cooking (stirring)*	75.0	62.5
living room	*talking on phone*, *drinking water*, *talking on couch*, *relaxing on couch*	81.2	75.0
office	*talking on phone*, *drinking water*, *writing on whiteboard, working on computer*	87.5	68.7

Similar to the KARD dataset, both the S-Seen setting where the instance of the subject has been seen during training and the S-New setting where the subject is encountered for the first time

during prediction are evaluated. The initializations are similar to the ones used for the KARD dataset. The number of burn-in iterations during posterior sampling for this dataset was 800 and as before 100 samples were collected.

The classification results for this dataset are furnished in Table 6.4. The overall classification accuracy for this dataset using the nonparametric HCRF model was 96.0% for the S-Seen setting and 84.7% for the S-New setting. This classification accuracy is less than the results reported in the previous chapter for the nonparametric H-HMM model. However, the nonparametric HCRF model is much simpler in structure when compared with the H-HMM model.

6.6 Conclusion

Discriminative learning methods that model the classification rules directly are often better suited for classification problems than generative methods. The CRF combines the advantage of a discriminative approach with the expressive power of graphical models for sequential data classification. The inclusion of hidden states in a CRF provides a more powerful HCRF representation that can capture latent structures in the data. This chapter has proposed a HCRF model in which the hidden states are automatically inferred from the data. This avoids the usual trial-and-error techniques needed to discover the appropriate number of hidden states and instead uses a principled nonparametric method based on the Dirichlet Process. The use of a Bayesian paradigm in which the HCRF parameters are assigned a Laplace-HDP prior provides a realistic characterization of the uncertainty in parameter estimates. The proposed inference procedure is efficient and it leads to sparse posterior samples. The good results achieved with the model for action classification confirm the utility of this approach.

In many scenarios, obtaining ground truth labels is expensive, but plenty of unlabelled examples are available. It is not straight-forward to use unlabelled examples with conditional models. Hence it is difficult to perform semi-supervised learning with CRF. In contrast, the models proposed in Chapters 4 and 5 can work with both fully-labelled and partially-labelled datasets. Further, the use of hierarchical structures with CRF is not well studied unlike the generative equivalents. Models such as hierarchical HMM and even the more general Dynamic Bayesian Networks are more popular than CRF models in the literature. Although the Bayesian paradigm used here for learning the HCRF parameters provides advantages such as model averaging, it is computationally demanding. This may be an issue if there are a large number of classes that must be considered simultaneously.

7. Conclusions

With cameras becoming ever more popular and huge amounts of video data accumulating every day, there is a compelling need to understand the content in a video automatically. Recognizing human actions is an important step towards generating automatic descriptions of video content. Important applications that motivate interest in this topic include automated surveillance, content retrieval and natural human computer interfaces.

Automatic action recognition requires the application of complex spatiotemporal concepts. The research in this thesis has focused on this challenging problem. Three different techniques for action classification in depth videos are presented. This chapter summarizes these techniques and the main contributions of this thesis. Some limitations of the proposed methods are discussed in Section 7.2. Several paths for future research, both in the action recognition side and in the graphical model side are outlined in Section 7.3. The thesis concludes with some final remarks.

7.1 Summary

Motion patterns in humans are governed by a hierarchy of joints connecting the bones that form the skeleton. By knowing the positions of these joints and observing their evolution over time, actions can be distinguished from each other. However, reliably estimating the joint positions from conventional colour images is non-trivial. There is an inherent loss of information when a 3D scene is projected into a 2D image. Even locating a person may be difficult due to background clutter. Variations in size, appearance, colour, texture and lighting further compound the problem.

A recent breakthrough [14] in pose estimation was achieved with the use of depth images. These depth images, acquired using low-cost active 3D sensing technology, do not suffer from the image effects associated with colour images. In particular, it is easier to perform background subtraction and reliably extract the human silhouette in depth images. With the pose estimation algorithm integrated into hardware, 3D joint positions are now available in real-time. The research in this thesis uses the skeleton joint positions obtained from depth images to characterize human actions.

The state-space graphical models provide an effective mechanism for capturing the temporal dynamics of the joint positions. These models employ a set of discrete valued state variables that compactly represent the observed data and use a graph based representation to simplify

the description of the dependencies over many variables. Popular models used in sequential pattern recognition such as Hidden Markov Model (HMM) and Hidden Conditional Random Field (HCRF) are examples of graphical models.

An important limitation of the state-space graphical models is that the number of states must be fixed in advance. This number is not known in most applications and a model selection procedure, which evaluates several models and chooses one based on a model comparison metric, is employed. Such procedures do not adapt well to changes in data complexity. Instead of dealing directly with this combinatorial challenge, nonparametric methods fit a single model that estimates the number of states automatically from data.

A central theme in this thesis is the use of nonparametric state-space graphical models for sequential data classification. Three different types of model were described in this thesis along with the derivation of tractable inference procedures for each type. The proposed inference mechanisms use efficient block sampling techniques. The applicability of these models were demonstrated for action classification.

The models are general enough to be applied to other sequence labelling problems besides action sequences. The main contributions of this thesis are summarized below.

In Chapter 4, a discriminative nonparametric HMM was proposed. Unlike classical HMM, the number of hidden states is not fixed in advance in this model. The model extends the canonical nonparametric HMM with an additional class specific hierarchical level and formulates the distributions for parameters as transformations from a shared base distribution. These formulations allow sharing information across the action classes. Further, the model is augmented with a discriminative term that can be used during inference to ensure that the learned parameters produce good classification results. The model was evaluated for action classification on two publicly available depth video datasets using the skeleton joint positions as features.

In Chapter 5, a nonparametric Hierarchical HMM (H-HMM) that is suitable for classifying human activities was developed. An activity is decomposed into a set of coarse actions and each action in turn is decomposed into a set of granular poses. This organization allows the actions and poses to be reused across the activity classes. By regressing the activity labels only on the actions, a simplified classification model is produced. A nonparametric H-HMM with an unbounded number of action and pose states captures the hierarchical structure of the data. The flexible formulation in this model allows additional data channels to be incorporated seamlessly. The efficacy of the model was demonstrated on two datasets in which the skeleton joint positions

and the information in the depth image patches around the joint positions was used to define the features.

In Chapter 6, a nonparametric HCRF model for classifying actions was described. Unlike generative models such as HMM, the HCRF models the classification rules directly. The proposed nonparametric extension avoids specifying in advance the number of hidden states in the HCRF and infers it automatically from data. Further, the posterior distribution for the HCRF parameters is estimated. This fully Bayesian treatment of the training procedure provides a realistic characterization of the uncertainty in parameter estimates. An efficient inference technique using elliptical slice sampling leads to sparse posterior samples of parameter values. Good classification results are achieved in two different depth video datasets using this model.

7.2 Limitations

The action recognition methods proposed in this thesis rely on the skeleton joint positions estimated by the Kinect sensor. The sensor's range, field of view and environmental constraints may restrict the ability to obtain pose information. Additionally, the pose estimates may not be accurate in some situations. For example, when a person is partially occluded or if the person is not in an upright position, the sensor may not produce reliable skeleton estimates. These constraints make the Kinect sensor ineffective when the camera is mounted in a high position, as in surveillance scenarios, with the humans not directly facing the sensor. With future developments in depth sensing technology, these issues may be solved.

There are instances in which the proposed methods misclassify actions. This is particularly evident for actions that involve very similar motion patterns. Detecting the objects that a person interacts with and including this contextual information will benefit the model. The proposed models do not handle variations in gender, size and appearance between the humans performing an action. They also do not account for occlusions or changes in view point. These issues may be addressed by improving the feature extraction procedure.

The graphical models proposed in this thesis are not suitable to model complex events, for example events containing multiple hierarchical levels or involving more than one person. The models need to be extended by introducing new variables to handle these situations at a cost of increased complexity. Additionally, the models described here assume that the actions are segmented into their corresponding classes. This restricts their applicability to offline action recognition.

7.3 Future Research

There are several future research opportunities that arise directly out of this thesis. They are discussed below both from the perspective of action recognition and in terms of extensions to the nonparametric graphical models.

Further Evaluations

In addition to the datasets used in the experiments in this thesis, there are other publicly available datasets [5, 114] that contain skeleton information and are applicable for evaluating the accuracy of action classification. The datasets range from gaming actions to daily activities. They vary in terms of the action complexity, the number of actions and subjects, the number of times the actions were repeated and the video length. Some datasets contain additional information about the objects a person interacts with when performing an action. There are also datasets that contain depth and skeleton data captured simultaneously by multiple sensors for evaluating cross-view action recognition. The use of depth images is an emerging research area and more datasets containing complex activities have become available of late.

Combining the features used in other methods with the classification algorithms proposed here is one possible extension of the work in this thesis. Methods that use the relative geometry of subsets of joints and those that derive view invariant features are particularly interesting. The benefits of enhancing the linear dimension reduction method to manifold learning can also be investigated. The models in Chapter 4 and Chapter 5 support semi-supervised learning and the utility of including unlabelled examples can also be evaluated.

The models proposed in this thesis are generic in that they are applicable to other sequence classification problems. Evaluating the classification accuracy of these models on other sequence labelling tasks is another interesting research line.

Sensor Fusion

Action recognition in depth videos is affected by occlusion, camera position and field of view. Wearable inertial sensors such as accelerometers and gyroscopes provide alternative data in the form of accelerations and angular velocities that are useful in recognizing actions. These sensors can operate in unconstrained environments and are not affected by some of the challenges that vision based sensors face. However, they do suffer from limitations such as sensor drift and power requirements. No single sensor can cope with all situations. Combining the data from both depth and inertial sensors may improve action recognition performance [113].

In addition to using the depth information, it may be beneficial to include the colour images captured by conventional RGB cameras for action recognition. The RGB images carry rich texture information and may be useful for identifying the objects in the scene.

Beyond Traditional Recognition

It is not enough to determine the type of action that is occurring in video. It is also important to identify where in the video a particular action is taking place i.e. we wish to perform both action recognition and action localization. The action recognition methods in this thesis assume that the video has been trimmed to contain only the action of interest. A recent trend [188] is to determine when an action starts and ends in untrimmed long videos.

Another new trend is first person activity recognition [189] in which an observer (e.g. a robot or a person wearing a sensor) is involved in an activity. The recognition aim here is to determine what others are doing to the observer from the observer's perspective. In contrast with conventional third person recognition, in which the sensor is static and away from the subjects, the sensor here undergoes ego-motion in first person videos.

Variational Inference

The inference procedures in this thesis use Gibbs sampling to draw posterior samples. Although Gibbs sampling provides theoretical guarantees of accuracy, it suffers from two problems. Firstly it can be slow on large data sets and hence does not scale well for inference on large-scale data. Secondly, the sampler requires monitoring the convergence of a Markov chain. These have led to a search for computationally less demanding and deterministic alternatives for which it is easier to assess convergence. For example, variational Bayesian methods [16] compute an approximation to the posterior distribution using a locally optimal analytical solution. Variational inference methods have been developed for Hierarchical Dirichlet Processes (HDPs) [177]. The methods scale well for large data sets and can be used in place of the Gibbs sampling used in this thesis.

Alternative Structures

The graphical model structures considered in this thesis are fairly simple discrete state-space Markov models. There are alternative structures that introduce additional variables to model complex dynamic phenomenon. For example, the semi-Markov models [92, 176] use explicit distributions for each state's duration. Continuous valued states may also be used to capture temporal dependencies that exhibit structural changes over time [41]. There have also been hierarchical generalizations [190] to the discriminative models such as Hidden Conditional Random Fields (HCRFs). Such models may be useful when representing activities that involve multiple persons and long running complex events.

Dirichlet Process Extensions

The Hierarchical Dirichlet Process (HDP) is used as the nonparametric prior in this thesis. Recent efforts pursue alternative nonparametric priors to model data generating process that change

over time or vary fluidly with some set of covariates. These nonparametric processes [174] model distributions over collections of measures that are indexed by locations in some metric space such as time or spatial position. Another interesting nonparametric prior is the Pitman-Yor process [18] which generalizes the Dirichlet Process (DP) with an additional parameter that provides more flexibility to describe power law tail behaviour. The Pitman-Yor process can be used to model sequential data without any Markov assumptions.

7.4 Final Remarks

This thesis has focused on vision based action recognition – a problem that is yet to be solved and which is a fundamental building block for intelligent vision systems. It presented three different and original constructions of nonparametric state-space graphical models in which the number of states was inferred from data. These models were applied to the classification of actions in depth videos, demonstrating the utility of the models for action recognition. The generic nature of the models makes them suitable for other sequential data classification tasks. The investigations in this thesis led to the publication of papers [192, 193, 194, 195]. There are several promising research directions that arise out of this thesis.

A. Active 3D Sensors

In many applications it is useful to acquire 3D geometric information about the objects in a scene. The intensity and colour information in a single image are of limited use because the pixel values relate only indirectly to the surface geometry. In a depth image, the pixel values indicate the distances between the points in a scene and the sensor. This parameterized family of distances is a convenient way to represent a surface. The depth images are also referred in the literature as range images, depth maps and 2.5D images [23].

There are several techniques available for acquiring depth images. The traditional mechanism to obtain a depth image is stereo vision. To begin with, the scene is captured concurrently using two or more cameras that are placed in different positions. Typically the cameras are mounted parallel to one another and are separated by a short distance. The relative position and orientation of the camera along with its optical characteristics are assumed to be known. Pairs of points on the projected images that correspond to the same points on the scene are then identified. Finally, the geometric principle of triangulation is applied to calculate the point in 3D space using the corresponding points and the camera positions [24].

The main difficulty with stereo vision is the need to find points in the first image that correspond to points in the second image. The number of pairs of corresponding points that can be found is affected by various factors including the image content itself. As a consequence, the depth information that is obtained using stereo is unreliable and computationally expensive to obtain.

Active 3D sensors provide a better alternative to the above passive stereo technique. These sensors project electromagnetic energy such as light waves on to the scene and construct the depth information from a recording of the reflected energy. Only one camera is required. Further, they are relatively insensitive to the illumination conditions and texture effects. Two popular active 3D sensing techniques are structured light and time-of-flight.

A.1 Structured Light Imaging

Structured light based sensors [21] use a projector to emit a known pattern of light into a scene. The patterns can be formed by horizontal or vertical lines, or by dots or by a grid. A camera observes the distortion of the reflected light pattern caused by the shapes of the objects in the scene. By analysing these distortions, the surface shape can be reconstructed. The same triangulation principle used in stereo vision is applied here as well in order to infer the depth values. However, instead of finding corresponding points between two images, the

displacement of the patterns captured by the camera is compared against a known reference pattern. This matching is straight forward. The relative distance between a point in the projected pattern and the displaced position in the captured image is used to determine the depth value. An illustration is provided in Figure A.1.

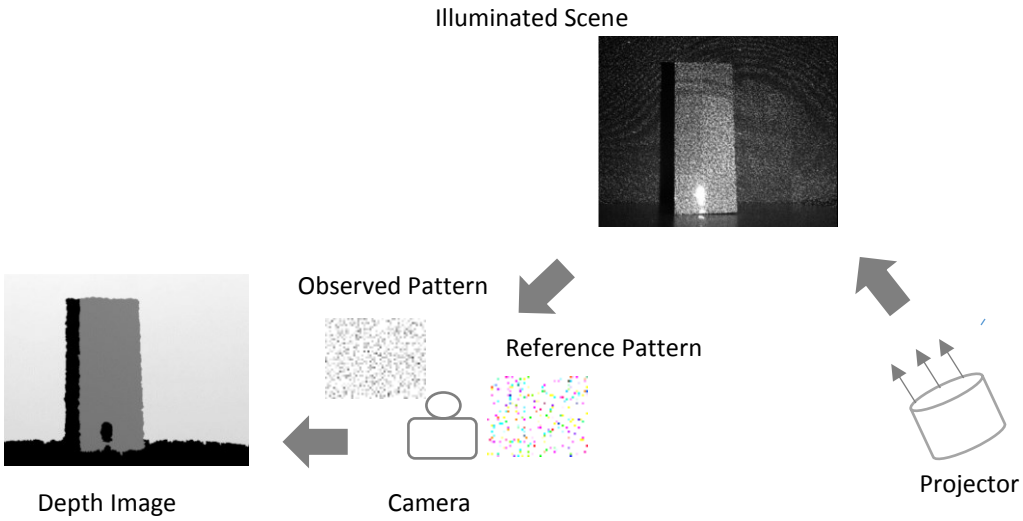


Figure A.1: Structured Light Imaging. A projector illuminates a scene with known pattern of light, in this case consisting of dots. The observed pattern is compared with a reference pattern to produce the depth image.

The first version of Kinect sensor, which produces the depth images used in this research, is based on the structured light principle [22]. It is composed of a near infrared laser projector that includes a laser diode at 850nm wavelength and a monochrome camera which is an infrared sensor. A horizontal bar connected to a base contains the camera and the projector, which are placed with a distance of about 7.5cm between them. The projector uses a known pattern of a large number of infrared dots called a *speckle pattern* to illuminate the scene. The reflected speckles are captured by the infrared camera. A reference plane at a known distance from the camera is assumed to be available. The position of a speckle in the infrared image is shifted if the object on which the speckle is projected lies ahead or behind the reference plane. The distance of the object to the sensor can then be determined by measuring these shifts. The depth estimation process is explained in detail below.

The projected image pattern of the speckles for a known depth is available to the Kinect sensor. Let Z_R be this known depth denoting the distance between the camera and an object that is on a point R on a reference plane. Let the camera and the projector be separated by a baseline b and the focal length of the camera be f. The f, b and Z_R values are determined by calibration. The depth values at each pixel of a new image observed by the camera must be computed. Let the object be on a point K on the object plane where the object plane may be behind or in front

of the reference plane. At each pixel in the observed image, a small correlation window is used to compare the local image pattern at the pixel with the known image pattern corresponding to the reference plane. This matching procedure provides an offset d from the known depth for the point K in terms of pixels. Let D denote the displacement of the point K in object space. Given f, b and Z_R, the triangulation principle [20] is applied to calculate the depth Z_K of the point K as follows:

$$\frac{D}{b} = \frac{Z_R - Z_K}{Z_R} \qquad \frac{d}{f} = \frac{D}{Z_K}$$
$$Z_K = \frac{Z_R}{1 + \frac{Z_R}{fb} d} \tag{A.1}$$

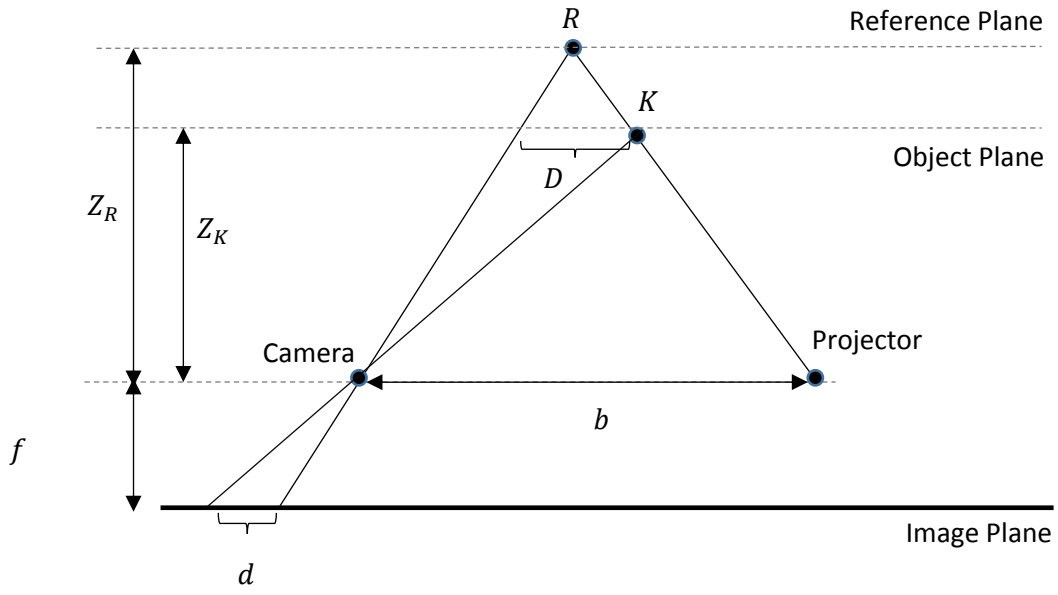


Figure A.2: Depth computation in Kinect. The distance to the reference plane Z_R, the focal length f, the displacement in the image plane d and the baseline b are used to calculate the depth value Z_K of a point K.

An illustration is provided in Figure A.2. By simply tracking a reflected speckle's horizontal coordinate in the observed image, the depth value at a pixel is estimated. The Kinect sensor has an operational range from 0.8 meters to 3.5 meters and highly accurate depth information can be estimated for distances up to 1.2 meters [25]. However, the sensor cannot be used in environments containing near infrared sources. In particular, the sensor cannot be used outdoors. The sensor has a field of view of 57° horizontally and 43° vertically. The depth estimates are obtained at high frame rates in near real time unlike passive stereo vision.

A.2 Time-of-flight Imaging

Surface coordinates can be determined directly based on the radar time-of-flight principle. The idea here is to illuminate a scene with infrared light using a projector and measure the time taken for the light to be reflected back to an image sensor. The use of high energy light reduces the interference from other infrared sources. A single sensor is sufficient and triangulation is not needed. However, an accurate mechanism to measure the roundtrip travel time of the light signals is required.

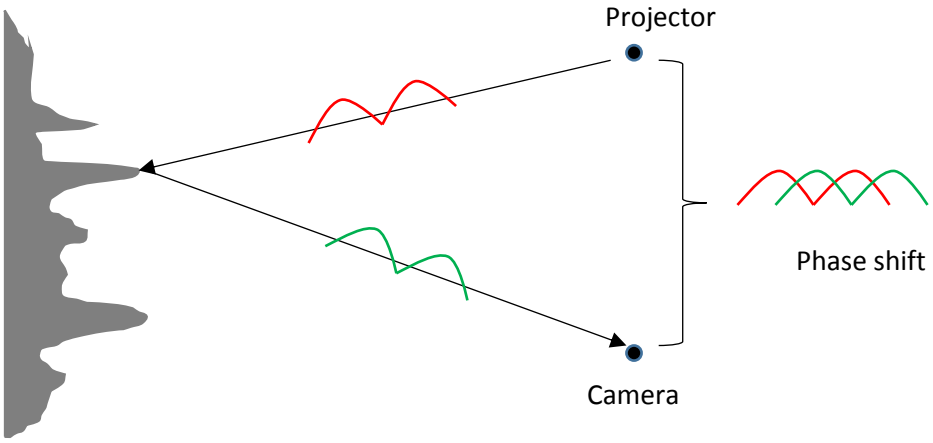


Figure A.3: Time-of-flight principle. A modulated infrared light beam emitted from a projector appears phase shifted in the camera. The distance between the camera and a target surface can be calculated from this shift.

Continuous wave modulation is used in many time-of-flight sensors. Instead of measuring the roundtrip travel time directly, the phase difference between the emitted and observed light signal is measured. When a scene is illuminated using infrared intensity modulated periodic light, the reflected optical signal undergoes a time shift. This time shift is recorded as an equivalent phase shift by the camera. This phase shift is proportional to the distance between the camera and the reflecting surface. The observed optical signal on the camera is correlated with a reference signal and the phase shift is computed using several correlation measurements obtained by varying the illumination and reference signals [22, 191]. Figure A.3 provides an overview of this process.

The second version of Kinect sensor uses time-of-flight imaging. It can estimate depth values for distances up to 8 meters and has an in built ambient light rejection mechanism that allows it to be used outdoors. It also has an improved field of view of 70° horizontally and 60° vertically. The older Kinect sensor cannot measure distances in the spaces between the speckles and its depth precision is lower than that of the newer sensor. The new sensor can derive depth estimates for

even very thin objects which are in effect invisible to the old sensor [26]. The research in this thesis uses depth images captured by the older Kinect sensor. The human body parts are large enough and hence the older version of the sensor is sufficient to provide depth estimates for these parts.

The depth images produced by the active 3D sensors do contain some errors. Surfaces that do not perfectly reflect the incident light could lead to gaps or holes in the measured depth values. The depth estimates are imprecise for objects that are far from the camera. The orientation of the object surface relative to the sensor also influences the quality of measured depth values. The sensor may give noisy depth values at the object boundaries because it is difficult to observe the projected light on the surfaces at the edges due to occlusions and shadows. However, the depth estimates acquired using these cameras are at real time and more reliable than those obtained using passive stereo. Further, these sensors are available at an affordable price. Even though the Kinect sensor was designed originally for gaming, they have been used innovatively by the research community in a number of scenarios such as attaching the sensor to a robot and using it for navigation and interaction with people.

B. Pose Estimation

The rigid articulated structure of the human skeleton governs the body motion using the various joints that connect the body parts. An established way to represent the configuration of a human body during the course of a motion is to use this underlying skeletal structure. Pose estimation is the process of recovering the skeleton body joints from images. It remains a difficult task despite many years of research. The position and orientation of a rigid object in a 3D space is specified using six independent parameters. As other objects are connected to it, more parameters are needed. The human body contains a large number of body parts and requires no less than 20 independent parameters to define its configuration. The configuration space is exponential in the number of joints making it harder to recover the poses. In addition, the diversities in the size, shape, style and gender induce significant variations. The effects of lighting, colour, texture and occlusions compound the problem.

The conventional approach to pose estimation exploits the kinematic constraints and tracks the body motion over time [55]. It involves initializing a canonical pose (e.g. limbs spread out) and tracking the pose changes using the temporal coherence from one image to the next in a sequence of images. This solution suffers from two major issues – the need to adopt an initialization pose and the frequent loss of track which requires re-initialization. It is preferable to have a solution that does not require an initial pose and uses only a single image to avoid errors due to tracking. Such a solution is not possible with the classical intensity or colour images.

The use of depth images provides an opportunity to estimate pose from a single image. They greatly simplify the task of background subtraction and facilitate extraction of an unambiguous human silhouette. Further, they help to overcome a lack of training data. With the enormous range of human body shape and size, it is important to have a training dataset that represents these variations. Since real images are often expensive to obtain, computer graphics techniques are typically used to obtain synthetic images. Unlike intensity images, the depth images are not hampered by colour and texture variations caused by clothing, hair or skin. Hence it is easier to build a large training dataset of synthetic depth images [9].

A natural approach for pose estimation is to identify the body parts that are spatially near to the skeletal joints of interest. The pose estimation problem can be reformulated as a body part labelling problem in which the pixels in the depth image are assigned to the appropriate body

parts. This formulation is consistent with other object recognition approaches [27] in which an object is modelled by a collection of parts that are arranged in a deformable configuration. In order to address the body part labelling problem, the depth image features must be designed and an appropriate classifier must be selected. There are many successful solutions available for classifying the pixels in an image [47]. However, the need to perform pose estimation in real time precludes the use of computationally expensive features and classifiers.

The work in [14] provides an ingenious method that uses simple yet discriminative depth comparison features and a randomized ensemble of decision trees to label the body parts in real time. A feature $f_n(x;\theta_n)$ at pixel x is the difference between the depth values at pixels $x+u$ and $x+v$ where the parameter $\theta=(u,v)$ contains the offsets u and v. Each feature provides a small amount of information about the locations of the body parts. For example, the feature f_1 in Figure B.1 (a) will give large positive responses for pixels near the top of the body while it will be close to zero for pixels in the centre of the body. A large number (about 2000) of these scale and translation invariant features are used.

The features are evaluated on multiple decision trees at each pixel to accurately identify the body part. To classify pixel x, each decision tree is traversed from the root node to leaf node by repeatedly evaluating a decision function. The decision function compares a feature value $f_n(x;\theta_n)$ to a pre-learned scalar threshold τ_n. If the function evaluates to 0, the path branches to the left child, otherwise to the right child. Each leaf node of the tree contains a distribution over the body part labels that represents a learned prediction model. The distributions are averaged together for all the decision trees to determine the final classification label. Finally, the per pixel label information is pooled across the pixels and the positions of the skeletal joints in the 3D world space are estimated. Figure B.1 describes the workflow.

In the above technique, the features do not need any pre-processing and only a small number of arithmetic operations are involved. When used in conjunction with a number of decision trees, this technique is sufficient to discriminate the body parts. Further, both the features and the classifier can be evaluated in parallel on each pixel in a computationally straight forward way. Hence the entire process is implemented in a GPU and takes under 5ms to determine the skeletal joint positions in an image. A key aspect to the success of this method is the size and composition of the training data. A large database of ground truth pose data obtained using motion capture technology and realistic depth images of humans synthesized using a graphics pipeline are used to construct a training corpus of one million images. This dataset contains good coverage of variations in pose, shape, clothing, hair and occlusions.

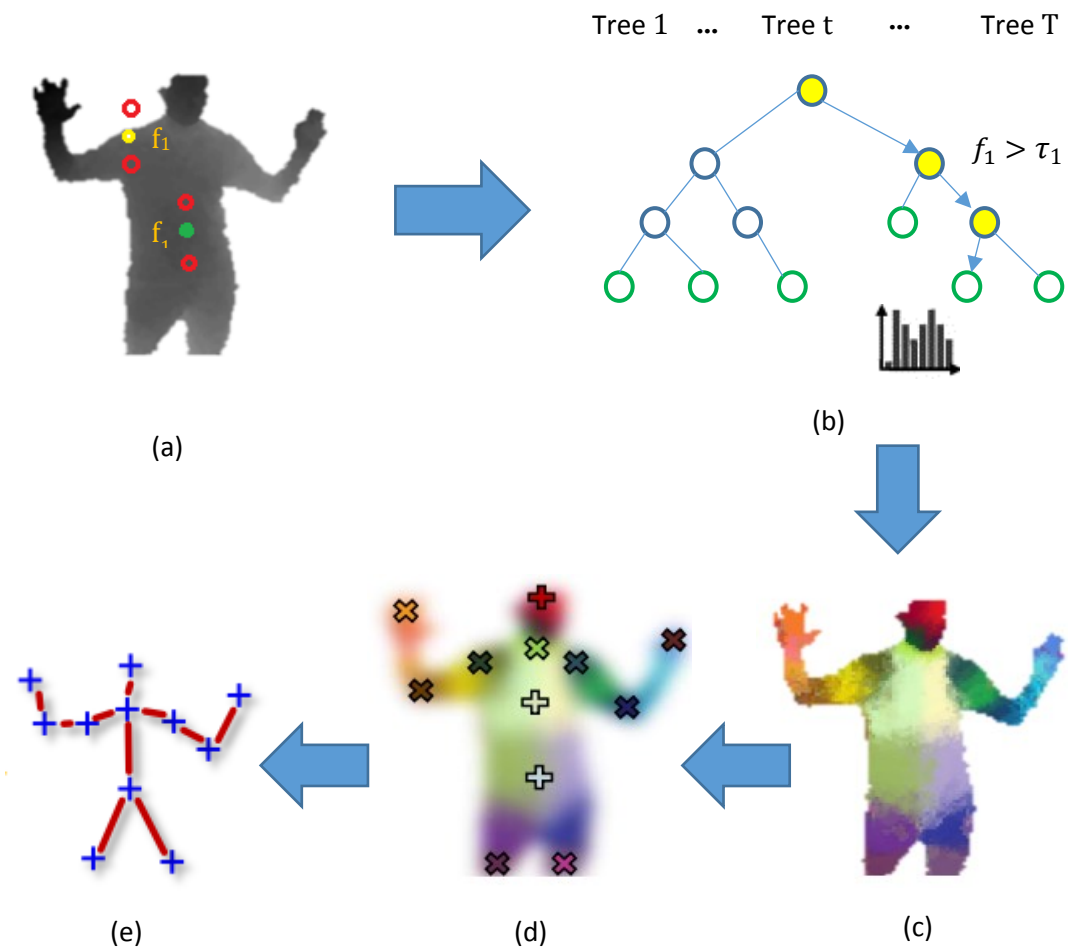


Figure B.1: Pose estimation pipeline. (a) Each feature f_1 is a difference between the values of two depth pixels (red) offset from a base pixel (yellow or green). (b) An ensemble of decision trees, such that each leaf node contains a learned distribution over the part labels. Each non-terminal node performs a simple test on the feature value using a decision function. The nodes highlighted in yellow illustrate a path chosen when testing the feature values corresponding to the pixel in yellow. (c) The image with per-pixel label. (d) The 3D joint positions estimated from the labels. (e) The skeleton structure with the 3D joint positions [14, 28].

The datasets used in the experiments in this thesis come annotated with the 3D joint positions in each video frame. These joint positions were determined by a procedure similar to the above technique.

C. Bayesian Approach

The mathematical framework of probability theory allows reasoning about uncertainty. A probability distribution assigns a probability to sets of possible outcomes of an experiment. The distribution is used for summarizing information and drawing conclusions. The Bayesian approach [34] provides a means of combining prior knowledge with the knowledge obtained from experiments.

C.1 Probability Model

The first step in the Bayesian approach is to choose a probability model for the data. This involves choosing a probability distribution for a random variable X that takes values in a set $\mathcal{X}$. The set $\mathcal{X}$ could be countably finite, in which case X is discrete or it could be a subset of $\mathbb{R}^n$, $n \in \mathbb{N}$. Typically, the distribution that is chosen is a member of a class of distributions called the *exponential family* [35]. Distributions from the exponential family can summarize a large dataset using a fixed number of parameters. Let the distribution of X depend on parameters θ taking values in a parameter space Θ. Let $p(x|\theta)$ denote the probability density function (pdf) for some $x \in \mathcal{X}$. The exponential family of densities is given by:

$$p(x|\theta) = h(x)g(\theta)\exp\{\theta^T t(x)\} \tag{C.1}$$

Here θ are the parameters of the distribution, $t(x)$ is a function of the data called the sufficient statistic, $g(\theta)$ ensures that the distribution is normalized and $h(x)$ is a known function. Both x and θ can be vectors, in which case $\theta^T t(x)$ is the scalar product of vectors. An important characteristic of the sufficient statistic is that it encapsulates all the information necessary to derive any estimate of the parameters given the data. Many standard distributions are members of the exponential family. The distributions that are relevant in this thesis are briefly mentioned.

Let us consider a binary random variable. The probability distribution over $\{0, 1\}$ is the *Bernoulli* distribution with a parameter ϱ that is the probability of observing 1. The probability mass function $p(x|\varrho)$ is written as follows:

$$p(x|\varrho) = \varrho^x(1 - \varrho)^{1-x} \qquad x \in \{0, 1\} \tag{C.2}$$

Let us consider a categorical random variable with K possible values. The *Multinomial* distribution on $\{1, 2, \dots K\}$ is given by

$$p(x|\boldsymbol{\varrho}) = \prod_{k=1}^{K} \varrho_k{}^{\mathbb{I}(x=k)} \qquad x \in \{1, 2, \dots K\} \qquad \text{(C.3)}$$

where $\boldsymbol{\varrho} = (\varrho_1 \dots \varrho_K)$, ϱ_k is the probability that $x = k$ and $\sum_k \varrho_k = 1$. Here $\mathbb{I}(x = k)$ is an indicator function that evaluates to 1 if $x = k$, 0 otherwise. The Bernoulli distribution is a special case of the Multinomial distribution.

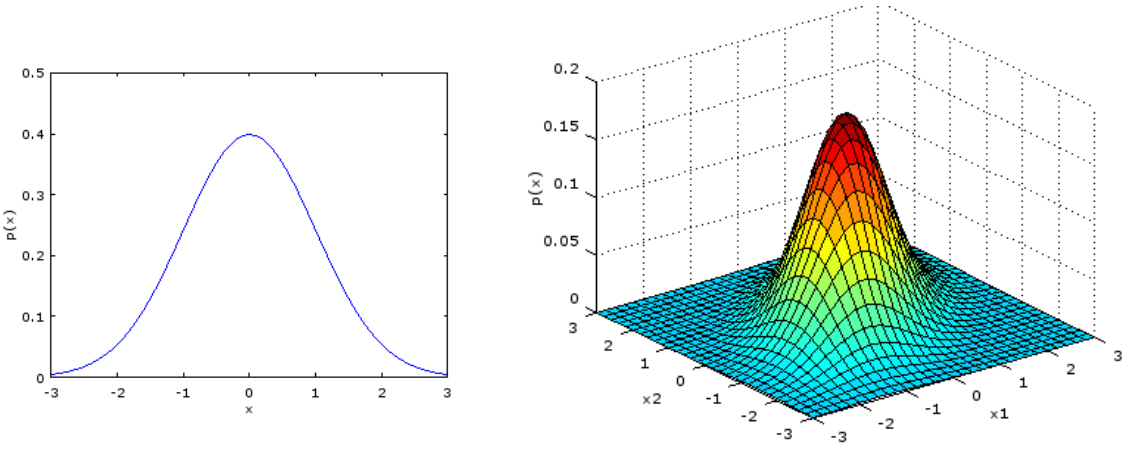


Figure C.1: Gaussian density plots. *Left*: The X axis represents a univariate continuous random variable and the Y axis represents the probability density value. *Right*: A Gaussian density for a two dimensional random variable with the Z axis representing the density value.

One of the most important probability distributions for modelling a continuous random variable is the *Gaussian* distribution. Two examples of Gaussian probability density functions are shown in Figure C.1. For the case $x \in \mathbb{R}$, the distribution has two parameters: a mean μ and variance σ^2. It is defined as:

$$p(x|\,\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\, exp\left\{\frac{-(x-\mu)^2}{2\sigma^2}\right\} \qquad \text{(C.4)}$$

The Gaussian distribution for a d dimensional continuous random variable is referred as the *multivariate Gaussian* distribution. It has parameters mean $\mu \in \mathbb{R}^d$ and the positive definite covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$ with $d(d+1)/2$ independent parameters. The multivariate Gaussian density for $x \in \mathbb{R}^d$ is defined as follows:

$$p(x|\,\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}}\, exp\left\{\frac{-1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\} \qquad \text{(C.5)}$$

Here $|\Sigma|$ is the determinant of Σ. In many applications, the covariance matrix is restricted to be diagonal, which reduces the number of parameters for Σ to d. The Gaussian distribution is referred to as the normal distribution, and denoted by $\mathcal{N}(x;\ \mu, \sigma^2)$ if x is univariate and $\mathcal{N}(x;\ \mu, \Sigma)$ for multivariate x. The notation $X \sim \mathcal{N}(\mu, \Sigma)$ is used to indicate that the random variable X has a Gaussian distribution.

Finally, the *Gamma* distribution is parameterized by a location parameter a and a scale parameter b. It is given as:

$$p(x|\ a, b) = \frac{1}{(a-1)!}\ b^a x^{a-1}\ exp\{-bx\} \tag{C.6}$$

C.2 Posterior Analysis

In the Bayesian approach, the parameters themselves are treated as random variables. After choosing a probability distribution $p(x|\ \theta)$ in which θ are the model parameters, a distribution $p(\theta)$ that captures any prior knowledge about the model parameters must be defined. This prior distribution can be interpreted as the uncertainty in θ before observing the data x. If very little is known about the prior distribution of θ, then a so called un-informative prior can be used.

The prior distribution assigned to θ may in turn depend on a new set of parameters η known as the hyper-parameters. The flexibility of the Bayesian framework allows a hyper-parameter to have its own prior (hyper-prior) with yet another set of parameters (hyper-hyper-parameters) and so on, resulting in a hierarchical model. For now, let us assume that these hyper-parameters η are set to some fixed value.

It is also important to construct the likelihood function from the observed data. The likelihood is the joint probability of the observed data, viewed as a function of the parameters. It is assumed that a dataset of observed values $\mathcal{D} = \{x^1, \dots x^N\}$ is obtained such that the x^n are independent samples from $p(x|\theta)$. The likelihood function $L(\theta|\mathcal{D})$ is defined by:

$$L(\theta|\mathcal{D}) = p(x^1, \dots x^N|\theta) = \prod_{n=1}^{N} p(x^n|\theta) \tag{C.7}$$

The fundamental problem in Bayesian analysis is to infer the probability distribution of the parameters given the dataset of observed values. This distribution $p(\theta|\mathcal{D}, \eta)$, referred as the posterior distribution, is obtained by applying the Bayes theorem as follows:

$$p(\theta|x^1,\dots x^N,\eta) = \frac{p(x^1,\dots x^N|\theta)\; p(\theta|\eta)}{\int p(x^1,\dots x^N|\theta')\; p(\theta'|\eta)\; d\theta'} \propto\; p(\theta|\eta)\prod_{n=1}^{N} p(x^n|\theta) \tag{C.8}$$

The posterior distribution captures the knowledge about θ that is contained in $\mathcal{D}$. The predictive likelihood of a future observation $\hat{x}$ is obtained by integration over θ,

$$p(\hat{x}|x^1,\dots x^N,\eta) = \int p(\hat{x}|\theta,\eta)\; p(\theta|x^1,\dots x^N,\eta)\; d\theta \tag{C.9}$$

This is unlike Maximum Likelihood Estimation (MLE) based methods where a single point estimate for θ is obtained by choosing the value that maximizes the likelihood function. The use of expected values leads to predictions which are often more robust than those obtained from a single point estimate of θ.

C.3 Conjugate Priors

A computationally tractable mechanism is needed to compute the posterior distribution over the parameters and the predictive likelihood of new observations. If $p(\theta|\eta)$ is chosen from an arbitrary family of prior distributions, then the integrals in (C.8) and (C.9) may be intractable. However, there is a family of prior distributions for which the posterior distribution and predictive likelihood can be computed analytically.

A prior distribution $p(\theta|\eta)$ is called a *conjugate* prior for the likelihood function $p(x|\theta)$, if the posterior distribution $p(\theta|x,\eta)$ is in the same family as the prior distribution.

$$p(\theta|x,\eta) \propto p(x|\theta)\; p(\theta|\eta) \propto\; p(\theta|\bar{\eta}) \tag{C.10}$$

The posterior distribution has the same algebraic form as the prior distribution and is described by an updated set of hyper-parameters $\bar{\eta}$. A conjugate prior provides a closed form expression to evaluate the posterior without the need to perform numerical integration.

The distributions described in the previous section all have conjugate priors [36] and these conjugate priors themselves are in the exponential family of distributions. The conjugate prior for the Bernoulli distribution with a parameter ϱ is the *Beta* distribution. The probability density function for the Beta distribution is defined as follows:

$$p(x|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\, x^{a-1}(1-x)^{b-1} \qquad a,b>0 \qquad \text{(C.11)}$$

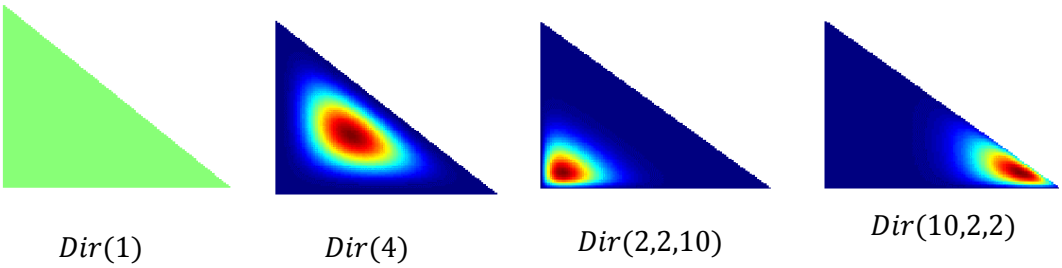


Figure C.2: Dirichlet distribution plots. The Dirichlet densities are visualized in the simplex $(\mathfrak{x}_1, \mathfrak{x}_2, 1-\mathfrak{x}_1-\mathfrak{x}_2)$. *Left to Right*: Symmetrical priors with parameter value 1, resulting in a uniform distribution. Prior with parameter value 4, resulting in the probability mass dispersed among all the categories. Biased priors with the probability mass concentrated around a particular category.

Here $\Gamma(n) = (n-1)!$ is the gamma function of a positive integer n and a, b are the parameters of the distribution. If the Bernoulli distribution parameter ϱ is $\varrho \sim Beta(a,b)$, then the distribution of ϱ given a set of observed values $\{x^1, \dots x^N\}$ is also a Beta distribution with updated parameters.

$$\varrho|x^1, \dots x^N, a, b \sim Beta(\bar{a}, \bar{b})$$
$$\bar{a} = a + \sum_{i=1}^{N} x^i \qquad \bar{b} = b + N - \sum_{i=1}^{N} x^i \qquad \text{(C.12)}$$

The conjugate prior for the Multinomial distribution is the *Dirichlet* distribution. Let $\boldsymbol{x} \sim Dir(\boldsymbol{a})$ denote a K dimensional vector $\boldsymbol{x} = (\mathfrak{x}_1 \dots \mathfrak{x}_k \dots \mathfrak{x}_K), 0 \leq \mathfrak{x}_k \leq 1, \sum_k \mathfrak{x}_k = 1$ obtained from a Dirichlet distribution with parameters $\boldsymbol{a} = (a_1 \dots a_K)$ written in the following form:

$$p(\boldsymbol{x}|\boldsymbol{a}) = \frac{\Gamma(\sum_k a_k)}{\prod_k \Gamma(a_k)} \prod_{k=1}^{K} \mathfrak{x}_k{}^{a_k-1} \qquad a_k > 0 \qquad \text{(C.13)}$$

By letting the Multinomial distribution parameter $\boldsymbol{\varrho} = (\varrho_1 \dots \varrho_K)$ to be $\boldsymbol{\varrho} \sim Dir(\boldsymbol{a})$, the posterior distribution of $\boldsymbol{\varrho}$ is also obtained from a Dirichlet distribution with updated values of $\boldsymbol{a}$.

$$\boldsymbol{\varrho}|x^1,\dots x^N,\boldsymbol{a})\sim Dir(a_1+n_1,\dots a_K+n_K)$$
$$n_k=\sum_{n=1}^{N}\mathbb{I}(x^n=k) \tag{C.14}$$

The notation $Dir(a_0)$ is used to denote the Dirichlet distribution in which all the K parameters of the Dirichlet distribution take the same value $a_k={}^{a_0}/_{K}$. This symmetric Dirichlet distribution is used when there is no prior knowledge distinguishing the categories. Figure C.2 shows examples of the Dirichlet distribution for different values of the parameters in the simplex defined by setting $K=3$.

The conjugate prior for the expected value of a Gaussian distribution with fixed covariance is the Gaussian distribution itself. Let the mean parameter μ of a multivariate Gaussian distribution with a known covariance Σ have the distribution $\mu\sim\mathcal{N}(\mu_0,\Sigma_0)$. The posterior distribution for μ is obtained as follows:

$$\mu|x^1,\dots x^N,\mu_0,\Sigma_0,\Sigma\sim\mathcal{N}(\overline{\mu_0},\overline{\Sigma_0})$$
$$\overline{\Sigma_0}=({\Sigma_0}^{-1}+N\,\Sigma^{-1})^{-1}\qquad\qquad\overline{\mu_0}=\overline{\Sigma_0}({\Sigma_0}^{-1}\mu_0+\Sigma^{-1}\sum_n x^n) \tag{C.15}$$

The conjugate prior for the covariance parameter of the Gaussian distribution with fixed mean is the *Inverse-Wishart* distribution. Let the notation $X\sim IW(\nu,\Delta)$ denote a random variable X that has a Inverse-Wishart distribution, where Δ denotes a $d\times d$ positive definite matrix and ν is a parameter that denotes the degrees of freedom. The pdf of an assignment $x\in\mathbb{R}^{d\times d}$ to X is written in the following form:

$$p(x|\nu,\Delta)\propto|x|^{-\frac{\nu+d+1}{2}}exp\left\{-\frac{1}{2}tr(\Delta x^{-1})\right\} \tag{C.16}$$

Here tr is the trace of a matrix. Let the covariance parameter Σ of a multivariate Gaussian distribution with a known mean μ have an Inverse-Wishart prior denoted as $\Sigma\sim IW(\nu,\Delta)$. The posterior value for Σ is obtained as follows:

$$\Sigma|x^1,\dots x^N,\nu,\Delta,\mu\sim IW(\bar{\nu},\bar{\Delta})$$
$$\bar{\nu}=\nu+N\qquad\qquad\bar{\Delta}=\Delta+\sum_n(x^n-\mu)(x^n-\mu)^T \tag{C.17}$$

D. Graphical Models

The distributions discussed in Appendix C are limited in the range of behaviours that they can represent. A richer family of distributions, can be obtained by introducing latent variables. These latent variables enable the construction of sophisticated models that can represent complex data patterns. With models now including many random variables, it is important to capture the relationship between these variables in a format that is amenable to probabilistic reasoning. The relationship between these variables can be represented more effectively by using graphs as data structures. This section surveys basic graph theory concepts and introduces the two main types of graphical models. It also describes sequential data modelling with a focus on the specific models used in this research. For in depth discussions, see [15] and [16].

D.1 Bayesian and Markov networks

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a finite set of nodes (also known as vertices) $\mathcal{V}$ and edges (also known as links) $\mathcal{E}$. An edge $(i, j) \in \mathcal{E}$ connects a pair of nodes $i, j \in \mathcal{V}$. In a *directed graph* all the edges are directed. The edge (i, j) connects a parent node i to a child node j. It is pictorially represented by an arrow with its tail originating at the parent node. In an *undirected graph* all the edges are undirected and an edge $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$. It is represented by a line between two nodes.

A *clique* is a set of nodes such that all pairs of nodes in the set have an edge between them. A *path* is a sequence of edges that connects two nodes. If there is a path between every pair of nodes then the graph is said to be *connected*. A path which starts and ends with the same node is a *cycle*. We are only interested in the graphs that do not have cycles and in which any two nodes are connected by exactly one path.

In a graphical model, each graph node represents a random variable and the edges represent the probabilistic relationships between the variables. Given a list of random variables $\boldsymbol{X} = (X_1, \dots X_i \dots X_N)$, the node $i \in \mathcal{V}$, with $\mathcal{V} = \{1,2, \dots N\}$, corresponds to the random variable X_i that takes outcomes from a set which can be either discrete or continuous. Let $\boldsymbol{x}$ denote the realization of $\boldsymbol{X}$ with x_i being the value assigned to the random variable X_i. The joint probability density function $p(\boldsymbol{x})$ depends on the distributions of the x_i and the graph structure. The key advantage of using the graphical model is that in many cases the distribution over $\boldsymbol{X}$ can be factorized into a product of distributions each one of which depends on a very small subset of the variables X_i.

Two random variables A and B are said to be independent, if for all assignments a and b, $p(a, b) = p(a)p(b)$. The notion of independence is useful because it allows probabilistic reasoning in isolation. While independent random variables do not occur commonly, *conditional independence,* in which two random variables become independent when conditioned on a third variable, is observed frequently. The conditionally independent variables can be eliminated when performing inference. This results in a simplified model that is computationally tractable. The missing edges in a graphical model encode the conditional independence relationships between the variables.

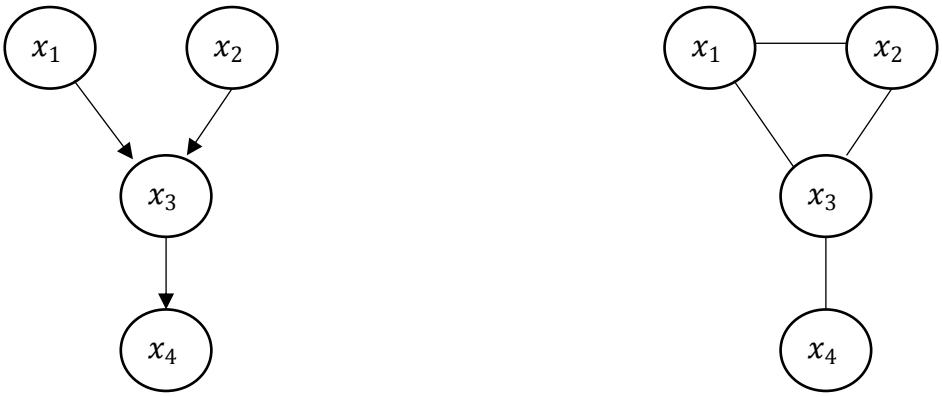


Figure D.1: Graphs showing the relationships between random variables. *Left*: Directed graph *Right*: Undirected graph

A directed acyclic graph describes the way in which each random variable is conditioned by the other random variables. A node in a directed graph is independent of its ancestors given its parents. A *Bayesian network* is a graphical model that uses a directed acyclic graph to characterize the joint distribution over the variables in the model. The joint distribution is factorized into a product of local conditional distributions that is governed by the parent-child relationship between the variables in the directed graph. Let $\mathcal{P}(i)$ denote the set of parent nodes of the node i. The joint distribution is decomposed as a product of the conditional distribution for each node given its parents and the probability density function is written as follows:

$$p(\boldsymbol{x}) = \prod_{i=1}^{N} p(x_i \mid \mathcal{P}(i)) \tag{D.1}$$

If a node j does not have any parent nodes, then instead of the conditional distribution simply $p(x_j)$ is used. Applying this decomposition, the joint density for the directed graph in Figure D.1 can be written as:

$$p(\boldsymbol{x}) = p(x_1)\, p(x_2)\, p(x_3 \mid x_1, x_2)\, p(x_4 \mid x_3) \tag{D.2}$$

It is evident from the formulation that x_4 is independent of x_1, x_2 conditioned on x_3. This conditional independence is denoted as $x_4 \perp x_1, x_2 \mid x_3$ to indicate that $p(x_4|x_1, x_2, x_3) = p(x_4 \mid x_3)$. The model structure is simplified and it is easier to learn the model.

A *Markov network* is a graphical model that uses an undirected graph to characterize the joint distribution over the model variables. As in a Bayesian network, a product of local functions is used to express the joint distribution. However, these local functions are not required to have a probabilistic interpretation. Further, unlike the parent-child relationship in the Bayesian network, the conditional independence here is governed by a set of cliques that determine the conditional independence of certain variables in the graph. Let C be the set of all the maximal cliques in an undirected graph $\mathcal{G}$. It is not possible to add any other node to a clique in C such that the resulting graph is still a clique. Let $\psi_c\big(x(c)\big)$ denote a non-negative potential function where c denotes a clique. Each ψ_c depends only on a subset $X(c) \subseteq \boldsymbol{X}$ of the random variables. The realization of $X(c)$ is $x(c)$. In a Markov network, the joint distribution is decomposed as a product of the potential functions corresponding to the cliques as follows:

$$p(\boldsymbol{x}) = \frac{1}{\mathcal{Z}} \prod_{c \in C} \psi_c\big(x(c)\big) \tag{D.3}$$

The constant term $\mathcal{Z}$, also called the *partition function*, ensures that the distribution $p(\boldsymbol{x})$ is correctly normalized. If $\boldsymbol{x}$ is discrete valued, then $\mathcal{Z}$ is given by:

$$\mathcal{Z} = \sum_{\boldsymbol{x}} \prod_{c \in C} \psi_c\big(x(c)\big) \tag{D.4}$$

In the undirected graph of Figure D.1, there are two cliques $\psi_{1,2,3}$= $\{x_1, x_2, x_3\}$ and $\psi_{3,4} = \{x_3, x_4\}$ with the variable x_4 being independent of x_1, x_2 conditioned on its neighbour x_3. The probability density function can be expressed as a function of the variables in the cliques as follows:

$$p(\boldsymbol{x}) = \frac{1}{\mathcal{Z}} \psi_{1,2,3}(x_1, x_2, x_3) \psi_{3,4}(x_3, x_4) \tag{D.5}$$

Both the Bayesian networks and the Markov networks can produce a factorization of a multivariate distribution function based on the conditional independence property. However,

they make different conditional independence assertions and there are families of probability distributions that are captured by one but not the other [29].

D.2 Sequential Data Modelling

In many applications, the input data is a sequence of observations in which there is an embedded structure which contains useful information. Some examples of such sequences include a time series of stock market prices, a DNA strand, a text sentence and a video. In this work, the sequences are assumed to be measurements observed at regular intervals in a discretized time line. Sequential data modelling involves tracking the evolution of these observations over time and performing probabilistic reasoning on them. The graphical models discussed above provide a framework for capturing the behaviour of sequential data.

Let $\boldsymbol{x} = \{x_t\}_{t=1}^{T}$ denote a set of input values observed at T different time instants. The notation $x_{1:T}$ is used sometimes to refer to $\boldsymbol{x}$. Each x_t is the realization of the random variable X_t at time instant t. The different random variables correspond to the nodes in a graph. The conditional independence relationships between these nodes are described by the graph edges. The joint density function of this input sequence factorizes as follows in a directed graph structure[3]:

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{1:t-1}) \tag{D.6}$$

If the Markov assumption is made, then the graph need not contain edges into variables at time $t+1$ from variables at time instants $t-1$ or earlier, as shown in Figure D.2.The Markov assumption is written in the form:

$$x_{t+1} \perp x_{1:t-1} \mid x_t \qquad 1 \leq t < T \tag{D.7}$$

The pdf $p(x_{1:T})$ in (D.6) reduces to

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{t-1}) \tag{D.8}$$

If an undirected graph was used instead of a directed graph to model the input values, then the pdf $p(x_{1:T})$ is written in the form

$$p(x_{1:T}) \propto \prod_{t=1}^{T-1} \psi_{t+1,t}(x_{t+1}, x_t) \tag{D.9}$$

[3] When $t = 1$, it is assumed that $p(x_t)$ is simply $p(x_1)$

where $\psi_{i,j}(x_i, x_j)$ denotes the potential function. Each $\psi_{i,j}$ depends only on the pair x_i and x_j.

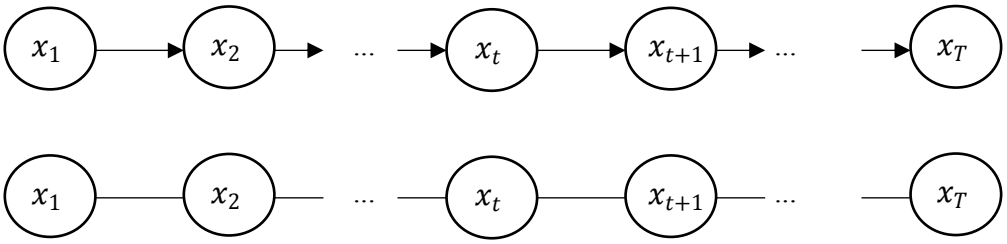


Figure D.2: Markov assumption. *Top*: A directed graph representation of $\boldsymbol{x}$ expressing the conditional independence between the future and past, given the present. *Bottom*: The Markov chain represented in an undirected graph.

D.3 Message Passing

When performing inference in a graphical model, the various statistical quantities such as likelihoods and conditional probabilities must be computed from a joint probability distribution. In particular, computing a marginal distribution – the distribution of a subset of variables conditioned on another subset, is often necessary to perform probabilistic reasoning. The graphs in Figure D.2 have a tree structure. Hence exact inference can be performed efficiently through a technique called *message passing*. The idea is to exploit the conditional independence relationship encoded in the local structure of the graph and pass real valued functions called messages between neighbouring nodes.

In the directed graph shown in Figure D.2, assume that the $\boldsymbol{x}$ values are discrete for the moment. This assumption will be relaxed to allow discrete or continuous observations later. The joint distribution over $\boldsymbol{x}$ factorizes as follows:

$$p(\boldsymbol{x}) = p(x_1)p(x_2|x_1) \dots p(x_t|x_{t-1})p(x_{t+1}|x_t) \dots (x_{T-2}|x_{T-1})p(x_T|x_{T-1}) \tag{D.10}$$

Given the joint distribution, computing a marginal distribution $p(x_t)$ involves summation of all possible values over all variables except x_t. The required marginal is written as:

$$p(x_t) = \sum_{x_1}\sum_{x_2} \dots \sum_{x_{t-1}}\sum_{x_{t+1}} \dots \sum_{x_T} p(\boldsymbol{x}) \tag{D.11}$$

Let each of the T discrete variables take K possible values. A naïve approach that evaluates the joint distribution for each term and performs summation explicitly is infeasible since there are

K^T values for $\boldsymbol{x}$. Instead, let us exploit the conditional independence relationship to determine a suitable order over which to sum. For instance, the summation over x_1 involves only the conditional distribution $p(x_2|x_1)$ and so this summation can be performed first to give a function of x_2.

$$m_{1,2}(x_2) = \sum_{x_1} p(x_2|x_1)p(x_1) \tag{D.12}$$

When performing a summation over x_2, this term $m_{1,2}(x_2)$ can be used without re-computing the summation over x_1. Consequently we now have:

$$m_{2,3}(x_3) = \sum_{x_2} p(x_3|x_2)m_{1,2}(x_2) \tag{D.13}$$

In the notation $m_{i,j}(x_j)$, the index i refers to the variable being summed and the index j refers to the other variable in the summation. Similarly, the summation over x_T involves only the conditional distribution $p(x_T|x_{T-1})$. The same technique can be used to compute the summation over x_T first and use this information when summing over x_{T-1}.

$$m_{T,T-1}(x_{T-1}) = \sum_{x_T} p(x_T|x_{T-1}) \tag{D.14}$$

$$m_{T-1,T-2}(x_{T-2}) = \sum_{x_{T-1}} p(x_{T-1}|x_{T-2})m_{T,T-1}(x_{T-1}) \tag{D.15}$$

The above technique distributes the summations efficiently by working inwards from the outer most nodes. Using this technique, the summations involved when computing the marginal $p(x_t)$ in (D.11) reduces to:

$$p(x_t) = m_{t-1,t}(x_t)m_{t+1,t}(x_t) \tag{D.16}$$

Note that each $m_{i,j}$ term contains a set of K elements, one for each possible value of x_j. Hence a product of two $m_{i,j}$ terms is interpreted as an element-wise multiplication that produces K elements. These $m_{i,j}$ terms can be evaluated recursively as,

$$m_{t-1,t}(x_t) = \sum_{x_{t-1}} p(x_t|x_{t-1})m_{t-2,t-1}(x_{t-1}) \tag{D.17}$$

$$m_{t+1,t}(x_t) = \sum_{x_{t+1}} p(x_{t+1}|x_t)m_{t+2,t+1}(x_{t+1}) \tag{D.18}$$

The marginal distribution $p(x_t)$ for the undirected graph in Figure D.2 can be computed using the same mechanism as above with the conditional distribution being replaced by the potential function in an undirected graph.

$$m_{t-1,t}(x_t) = \sum_{x_{t-1}} \psi_{t-1,t}(x_{t-1}, x_t) m_{t-2,t-1}(x_{t-1}) \tag{D.19}$$

$$m_{t+1,t}(x_t) = \sum_{x_{t+1}} \psi_{t+1,t}(x_{t+1}, x_t) m_{t+2,t+1}(x_{t+1}) \tag{D.20}$$

$$p(x_t) \propto m_{t-1,t}(x_t) m_{t+1,t}(x_t) \tag{D.21}$$

The $m_{i,j}$ terms can be interpreted as a generic message or information passed from node i to node j. Figure D.3 provides an illustration. The marginal distribution at a node is the product of incoming messages to the node. The message for a node variable is obtained by multiplying the incoming message to the variable by the conditional probability (or potential function in the case of an undirected graph) involving the variable and a neighbouring variable in the graph.

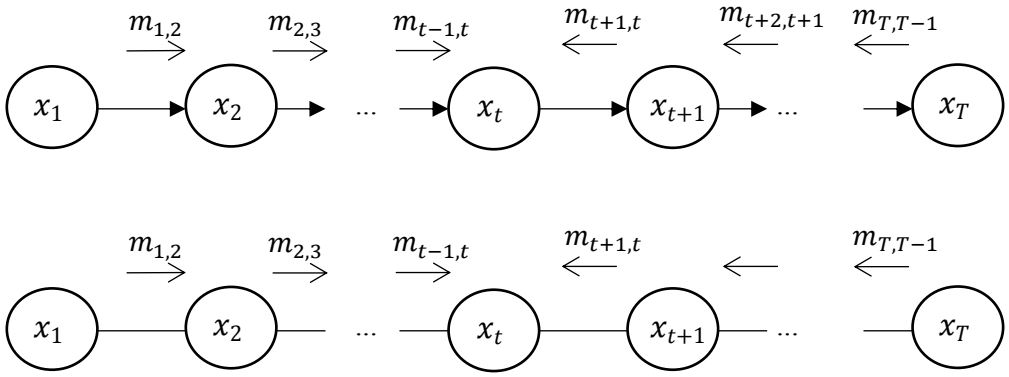


Figure D.3: Message Passing. The marginal distribution for x_t is obtained by multiplying the $m_{t-1,t}$ and $m_{t+1,t}$ messages. These messages can be evaluated recursively, working from the outer most nodes towards the node x_t. *Top*: Directed graph. *Bottom*: Undirected graph.

As shown in (D.17), each time we sum over the values of a random variable, the variable is eliminated from the distribution. The summation works inwards into the graph. The computational cost of calculating a marginal distribution by re-using the messages is $O(TK^2)$, which scales linearly with the number T of variables. This is in contrast with the brute force summation that is exponential in the number of variables.

E. Approximate Inference

In order to apply the probabilistic models in a Bayesian context, it is necessary to evaluate the posterior distribution of the latent variables given the observed data. However, the introduction of the latent variables results in a posterior distribution that has a highly complex form. Bayesian inference tasks such as prediction and computation of posterior parameter estimates relies on integration and there are often no closed form expressions readily available. The algebra becomes overwhelmingly cumbersome if the posteriors are evaluated analytically while numerical integration is not practical in high dimensional spaces. Except in the simplest cases, it is infeasible to perform exact inference and we need to resort to some form of approximation.

E.1 Simulation Methods

Simulation techniques are a successful form of approximate inference that has enabled widespread use of Bayesian methods. The general idea is to draw samples from some approximate distribution and then improve these draws to converge towards the target posterior distribution. In most situations, the posterior distribution is required primarily for evaluating expectations in order to make predictions. The desired integral involved in performing prediction can be formulated as an expectation with respect to a probability distribution.

Let θ be a set of continuous random variables (parameters) and let $p(\theta)$ be a probability distribution over possible values of θ. Let $f(\theta)$ be a function for which the expectation with respect to $p(\theta)$ is required. For example, $f(\theta)$ could be a likelihood function and $p(\theta)$ could be the posterior distribution. The integral is formulated as follows:

$$\mathbb{E}_{p(\theta)}[f(\theta)] = \int f(\theta)p(\theta)d\theta \tag{E.1}$$

The integral is replaced with a summation if θ is a set of discrete variables. The simulation methods are based on obtaining N independent samples $\theta^1, \dots, \theta^N$ from $p(\theta)$ and estimating the expectation by:

$$\begin{aligned}\mathbb{E}_{p(\theta)}[f(\theta)] &= \lim_{N\to\infty} \frac{1}{N}\sum_{n=1}^{N} f(\theta^n) \\ &\cong \frac{1}{L}\sum_{l=1}^{L} f(\theta^l)\end{aligned} \tag{E.2}$$

The estimate of $\mathbb{E}_{p(\theta)}[f(\theta)]$ becomes accurate as L increases.

The *Markov Chain Monte Carlo* (MCMC) methods [15, 16, 36] provide a general framework to obtain the samples $\theta^1, \dots, \theta^L$ from $p(\theta)$. The idea here is to sample sequentially, with a sample θ^{l+1} being obtained from a proposal distribution $q(\theta|\theta^l)$ that depends on the current sample θ^l. The proposal distribution is chosen such that it can be sampled in a straight-forward way. For instance, Gaussian distribution could be chosen. At each iteration, a candidate sample θ^* is accepted with some probability or the current sample θ^l is used. As we proceed through the process, the distribution from which the samples are obtained becomes closer to the target distribution $p(\theta)$. The sequence of samples θ^1, θ^2 … forms a Markov Chain with

$$q\left(\theta^{l+1}\middle|\theta^1, \dots, \theta^l\right) = q\left(\theta^{l+1}\middle|\theta^l\right) \qquad 1 \le l \le L-1 \tag{E.3}$$

E.2 Gibbs Sampling

Gibbs sampling [37, 38] is a widely applicable MCMC method that is useful in many multidimensional problems. The Gibbs sampler cycles through the variables. A sample for the current variable is obtained from a distribution that is conditioned on the values of remaining variables. Let the set of variables θ be divided into D components $\theta = (\theta_1, \dots \theta_d \dots \theta_D)$, in a fixed order. Each θ may contain one or more of the individual variables in θ. At an iteration l of the Gibbs sampler, a sample for the d^{th} component θ_d^l is drawn from a distribution conditioned on all the other components except d.

$$\begin{aligned} \theta_d^l &\sim p(\theta_d|\theta_{\setminus d}^l) \\ \theta_{\setminus d}^l &= \theta_1^l, \dots \theta_{d-1}^l, \theta_{d+1}^{l-1} \dots \theta_D^{l-1} \end{aligned} \tag{E.4}$$

Thus a new value for a component is obtained according to a distribution that is based on the latest values of all the other components. Table E.1 provides an outline of the algorithm.

Table E.1: Gibbs Sampling Algorithm

Input: Initial sample of a set of variables θ with D components $\theta^1 = (\theta_1, \dots \theta_D)$ **Output:** Samples of the components
1. For $l = 1 \dots L$ 2. Sample $\theta_1^{1+1} \sim p\left(\theta_1\middle|\theta_2^l, \dots \theta_D^l\right)$.

3.	Sample $\theta_2^{1+1} \sim p\left(\theta_2 \middle\vert \theta_1^{l+1}, \theta_3^l, \dots \theta_D^l\right)$.
4.	…
5.	Sample $\theta_D^{1+1} \sim p\left(\theta_D \middle\vert \theta_1^{l+1}, \dots \theta_{d-1}^{l+1}\right)$.
6.	End For

Gibbs sampling is particularly efficient if conditionally conjugate priors can be used. Consider for example two Gaussian random variables X_0 and X_1 that are mixed to give a random variable X in the following manner:

$$\mathrm{X}_0 \sim \mathcal{N}(\mu_0, \Sigma_0) \qquad \mathrm{X}_1 \sim \mathcal{N}(\mu_1, \Sigma_1)$$
$$X = Z\,\mathrm{X}_0 + (1 - Z)\mathrm{X}_1 \tag{E.5}$$

Here Z is a Bernoulli random variable that takes the value 1 with probability ϱ. It indicates the Gaussian from which X is drawn. The joint density function is written as

$$p(x^1 \dots x^N, z^1 \dots z^N, \varrho, \mu_0, \Sigma_0, \mu_1, \Sigma_1) = \left(\prod_{n=1}^{N} p(x^n | z^n, \mu_{z^n}, \Sigma_{z^n}) p(z^n | \varrho)\right) \times p_0(\varrho)\; p_0(\mu_0, \Sigma_0)\; p_0(\mu_1, \Sigma_1) \tag{E.6}$$

where x^n is an observed value, z^n is the Gaussian from which x^n was drawn and p_0 denotes a prior distribution. The set of variables is $\theta = (\mu_0, \Sigma_0, \mu_1, \Sigma_1, \varrho, z^1 \dots z^N)$. When sampling the posterior $p(\theta | x^1 .. x^N)$, Gibbs sampling is applied. The variables are sampled one at a time, with the assumption that the values of the other variables are available. A conjugate prior can then be assigned to the variable being sampled and the posterior for a variable can be computed analytically. The Gibbs sampler for the above two mixture Gaussian cycles through as follows:

(i) Sample $\mu_0 | \Sigma_0, z^1 \dots z^N, x^1 \dots x^N$. By assigning the mean parameter a Gaussian prior, the posterior values can be sampled from an updated Gaussian distribution as per equation (C.15).

(ii) Sample $\Sigma_0 | \mu_0, z^1 \dots z^N, x^1 \dots x^N$. By assigning the covariance parameter an Inverse-Wishart prior, the posterior values can be sampled from an updated Inverse-Wishart distribution as per equation (C.17).

(iii) Sample μ_1 and Σ_1 in a similar way to (i) and (ii) respectively.

(iv) Sample $\varrho | z^1 \dots z^N$. By assigning ϱ a Beta prior, the posterior values can be sampled as per equation (C.12).

(v) Sample $z^1 \dots z^N | \varrho$ by sampling from the Bernoulli distribution.

The Gibbs sampler relies on defining a conditional distribution $p(\theta_k|\theta_{\setminus k})$ from which it is easy to draw samples. In the above case sampling is easy because conjugate priors are used. Since Gibbs sampling allows the variables to be partitioned, it is ideally suited for inference in graphical models.

E.3 Slice Sampling

It is not always possible to have a conjugate prior for a variable, as observed in the subsequent chapters. One option is to choose an appropriate proposal distribution that will lead to efficient sampling. An alternative option is to treat a multivariate variable as univariate and apply Gibbs sampling to draw samples for each dimension of the variable given the value of other dimensions. Both these options need special coding and sometimes complex parameter tuning.

Slice sampling [39] provides a way of sampling from a density function which is known up to a scale factor. It uses the principle that one can sample from a univariate distribution by sampling uniformly from the region under the density curve. Let θ be the variable of interest and let $f(\theta)$ be a function that is proportional to the density of θ. The slice sampling technique involves augmenting θ with an additional auxiliary variable u and then drawing samples from the joint (θ, u) space and ignoring the u values. The algorithm has the following steps:

(i) Given a sample θ^t, evaluate $f(\theta^t)$ and sample u uniformly in the range $0 \leq u \leq f(\theta^t)$. The auxiliary variable u represents a horizontal "slice" of the distribution.

(ii) Define a region $\theta_{min} \leq \theta^t \leq \theta_{max}$ around the sample θ^t that encompasses as much of the slice as possible.

(iii) Draw a new sample $\theta^{t+1} \sim \{\theta' \in \theta_{min} \leq \theta^t \leq \theta_{max} : u < f(\theta')\}$.

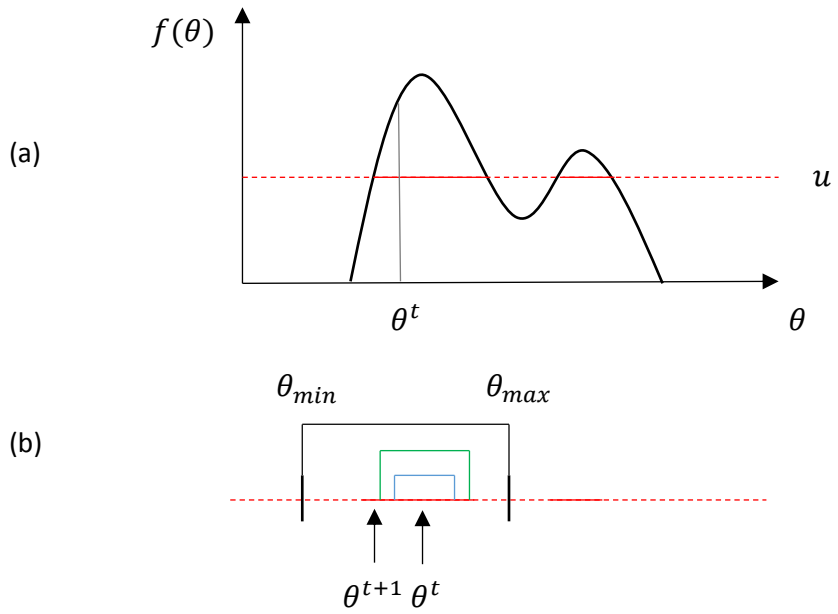


Figure E.1: Slice Sampling. (a) The solid horizontal line in red defines a slice through the distribution and is chosen uniformly in the region $0 \leq u \leq f(\theta^t)$ using the current sample θ^t. (b) A new sample is drawn from the region $\theta_{min} \leq \theta^t \leq \theta_{max}$. The region is obtained by expanding a region around the current sample (blue and green lines) until the end points are outside a slice (black line). When sampling from this region, if a candidate value lies outside the slice (solid red line) the region is shrunk further to use this value as θ_{min} or θ_{max}. Otherwise, the value is accepted.

The main challenge in the slice sampling algorithm is to find the region mentioned in step (ii) that contains all or much of the slice. There is a need to balance between obtaining a large sampling region, thereby making large moves in the space of θ and having only a minimal region outside the slice for the sampling to be accurate. A plausible approach to the choice of the region involves an expansion and contraction process. First a region around the current sample with a width w is tested to determine if the end points lie within the slice. If not, then the region is extended by increments of w until the end points lie outside the slice. A candidate value θ' is then chosen uniformly in this region and if it lies within the slice it is accepted. If it lies outside the slice, then the region is shrunk to contain θ' as the end point. An illustration is provided in Figure E.1.

References

[1] Dimitrova, N., Zhang, H. J., Shahraray, B., Sezan, I., Huang, T., & Zakhor, A. "Applications of video-content analysis and retrieval." IEEE Multimedia 9, no. 3, (2002): 42-55.

[2] O'Hara, K., Gonzalez, G., Sellen, A., Penney, G., Varnavas, A., Mentis, H., & Carrell, T et al. "Touchless interaction in surgery." Communications of the ACM 57, no. 1 (2014): 70-77.

[3] Atmosukarto, I., Ghanem, B., Ahuja, S., Muthuswamy, K., & Ahuja, N. "Automatic recognition of offensive team formation in american football plays." Computer Vision and Pattern Recognition Workshops (2013), pp. 991-998.

[4] Aggarwal, J. K., & Ryoo, M. S. "Human activity analysis: A review." ACM Computing Surveys (CSUR) 43, no. 3 (2011): 16.

[5] Aggarwal, J. K., & Xia, L. "Human activity recognition from 3d data: A review." Pattern Recognition Letters 48 (2014): 70-80.

[6] Chen, L., Wei, H., & Ferryman, J. "A survey of human motion analysis using depth imagery." Pattern Recognition Letters 34, no. 15 (2013): 1995-2006.

[7] Plagemann, C., Ganapathi, V., Koller, D., & Thrun, S. "Real-time identification and localization of body parts from depth images." Robotics and Automation (ICRA), (2010), pp. 3108-3113.

[8] Marr, D., Poggio, T., Hildreth, E. C., & Grimson, W. E. L. "A computational theory of human stereo vision." Proceedings of the Royal Society of London B: Biological Sciences 204, no. 1156 (1979): 301-328.

[9] Shotton, J., Girshick, R., Fitzgibbon, A., Sharp, T., Cook, M., Finocchio, M., & Blake, A et al. "Efficient human pose estimation from single depth images." Pattern Analysis and Machine Intelligence 35, no. 12 (2013): 2821-2840.

[10] Zhang, Z. "Microsoft Kinect sensor and its effect." IEEE MultiMedia 19, no. 2, (2012): 4-10.

[11] Xia, L., Chen, C. C., & Aggarwal, J. K, 2012, June. "View invariant human action recognition using histograms of 3d joints." Computer Vision and Pattern Recognition Workshops, (2012), pp. 20-27.

[12] Johansson, G. "Visual perception of biological motion and a model for its analysis." Attention, Perception, & Psychophysics 14, no. 2 (1973): 201-211.

[13] Puce, A., & Perrett, D. "Electrophysiology and brain imaging of biological motion." Philosophical Transactions of the Royal Society of London B: Biological Sciences 358, no. 1431 (2003): 435-445.

[14] Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., & Moore, R et al. “Real-time human pose recognition in parts from single depth images”. Communications of the ACM 56, no. 1 (2013): 116-124.

[15] Koller, D., & Friedman, N. Probabilistic Graphical Models: Principles and Techniques. MIT press, 2009.

[16] Bishop, C. M. Pattern Recognition and Machine Learning. Springer, 2006.

[17] Ghahramani, Z. "An introduction to hidden Markov models and Bayesian networks." International Journal of Pattern Recognition and Artificial Intelligence 15, no. 01 (2001): 9-42.

[18] Orbanz, P., & Teh, Y. W. "Bayesian nonparametric models." Encyclopedia of Machine Learning, (2010), pp. 81-89, Springer US.

[19] Teh, Y. W. "Dirichlet process." Encyclopedia of Machine Learning, (2010), pp. 280-287.

[20] Khoshelham, K. "Accuracy analysis of Kinect depth data." ISPRS workshop laser scanning, (2011), vol. 38, no. 5, p. W12.

[21] Sansoni, G., Trebeschi, M., & Docchio, F. "State-of-the-art and applications of 3D imaging sensors in industry, cultural heritage, medicine, and criminal investigation." Sensors 9, no. 1 (2009): 568-601.

[22] Sarbolandi, H., Lefloch, D., & Kolb, A. "Kinect range sensing: Structured-light versus time-of-flight Kinect." Computer Vision and Image Understanding, 139, (2015): 1-20.

[23] Trucco, E., & Verri, A. Introductory Techniques for 3-D Computer Vision. Vol. 201. Englewood Cliffs: Prentice Hall, 1998.

[24] Hartley, R., & Zisserman, A. Multiple View Geometry in Computer Vision. Cambridge University Press, 2003.

[25] Time-of-Flight and Kinect Imaging. Retrieved from campar.in.tum.de/twiki/pub/Chair/TeachingSs11Kinect/2011-DSensors_LabCourse_Kinect.pdf

[26] The Science Behind Kinects or Kinect 1.0 versus 2.0. Retrieved from gamasutra.com/blogs/DanielLau/20131127/205820/The_Science_Behind_Kinects_or_Kinect_10_versus_20.php

[27] Felzenszwalb, P. F., & Huttenlocher, D. P. "Pictorial structures for object recognition." International Journal of Computer Vision 61, no. 1 (2005): 55-79.

[28] Kohli, P., & Shotton, J. "Key developments in human pose estimation for Kinect." Consumer Depth Cameras for Computer Vision, (2013), pp. 63-70.

[29] Jordan, M. I. "Graphical models." Statistical Science (2004): 140-155.

[30] Barber, D. Bayesian Reasoning and Machine Learning. Cambridge University Press, 2012.

[31] Jordan, M. I., & Weiss, Y. "Probabilistic inference in graphical models." Handbook of Neural Networks and Brain Theory (2002).

[32] Rabiner, L. R., & Juang, B. H. "An introduction to hidden Markov models." IEEE ASSP Magazine, 3, no. 1 (1986): 4-16.

[33] Sutton, C., & McCallum, A. "An introduction to conditional random fields." Machine Learning 4, no. 4 (2011): 267-373.

[34] Glickman, M. E., & van Dyk, D. A. "Basic Bayesian methods." Topics in Biostatistics (2007): 319-338.

[35] Duda, R. O., Hart, P. E., & Stork, D. G. Pattern Classification. John Wiley & Sons, 2012.

[36] Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. Bayesian Data Analysis. Chapman & Hall, 2014.

[37] Resnik, P., & Hardisty, E. “Gibbs sampling for the uninitiated”. No. CS-TR-4956. Maryland Univ College Park Inst. for Advanced Computer Studies, 2010.

[38] Geman, S., & Geman, D. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images." Pattern Analysis and Machine Intelligence 6, (1984): 721-741.

[39] Neal, R. M. "Slice sampling." Annals of Statistics (2003): 705-741.

[40] Sudderth, E. B. "Graphical models for visual object recognition and tracking." PhD diss., Massachusetts Institute of Technology, 2006.

[41] Fox, E. B. "Bayesian nonparametric learning of complex dynamical phenomena." PhD diss., Massachusetts Institute of Technology, 2009.

[42] Antoniak, C. E. "Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems." The Annals of Statistics (1974): 1152-1174.

[43] Sethuraman, J. "A constructive definition of Dirichlet priors." Statistica Sinica (1994): 639-650.

[44] Teh, Y. W., Jordan, M. I., Beal, M. J., & Blei, D. M. "Hierarchical Dirichlet processes." Journal of the American Statistical Association (2012).

[45] Hjort, N. L., Holmes, C., Müller, P., & Walker, S. G, eds. Bayesian Nonparametrics. Vol. 28. Cambridge University Press, 2010.

[46] Orbanz, P. "Lecture notes on Bayesian nonparametrics." Journal of Mathematical Psychology 56 (2012): 1-12.

[47] Li, S. Z. Markov Random Field Modeling in Image Analysis. Springer Science & Business Media, 2009.

[48] Birney, E. "Hidden Markov models in biological sequence analysis." IBM Journal of Research and Development 45, no. 3.4 (2001): 449-454.

[49] Chou, W., & Li, L. "A minimum classification error (MCE) framework for generalized linear classifier in machine learning for text categorization/retrieval." Machine Learning and Applications, (2004), pp. 26-33.

[50] Genon-Catalot, V., Jeantheau, T., & Larédo, C. "Stochastic volatility models as hidden Markov models and statistical applications." Bernoulli 6, no. 6 (2000): 1051-1079.

[51] Jelinek, F. Statistical Methods for Speech Recognition. MIT press, 1997.

[52] DeCaprio, D., Vinson, J. P., Pearson, M. D., Montgomery, P., Doherty, M., & Galagan, J. E. "Conrad: gene prediction using conditional random fields." Genome research 17, no. 9 (2007): 1389-1398.

[53] Koppula, H., & Saxena, A. "Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation." International Conference on Machine Learning (2013), pp. 792-800.

[54] Shen, D., Sun, J. T., Li, H., Yang, Q., & Chen, Z. "Document summarization using conditional random fields." International Joint Conference on Artifical Intelligence, (2007), vol. 7, pp. 2862-2867.

[55] Poppe, R. "Vision-based human motion analysis: An overview." Computer Vision and Image Understanding 108, no. 1 (2007): 4-18.

[56] VidCon 2015 Haul: Trends, Strategic Insights, Critical Data, and Tactical Advice. Retrieved from reelseo.com/vidcon-2015-strategic-insights-tactical-advice/

[57] How many cameras in the UK?. Retrieved from securitynewsdesk.com/how-many-cctv-cameras-in-the-uk/

[58] Forsyth, D. A., Arikan, O., & Ikemoto, L. Computational Studies of Human Motion: Tracking and Motion Synthesis. Now Publishers Inc, 2006.

[59] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., & Berg, A. C et al. "Imagenet large scale visual recognition challenge." International Journal of Computer Vision 115, no. 3 (2015): 211-252.

[60] DARPA Mind's eye program : Broad agency announcement. Retrieved from fbo.gov/

[61] Biliński, P. T. "Human action recognition in videos." PhD diss., Université Nice Sophia Antipolis, 2014.

[62] Kinect for Xbox One. Retrieved from xbox.com/en-US/xbox-one/accessories/kinect-for-xbox-one

[63] Kasteren, T. L. M. "Activity recognition for health monitoring elderly using temporal probabilistic models." PhD diss., Informatics Institute, 2011.

[64] Lasserre, J. A., Bishop, C. M., & Minka, T. P. "Principled hybrids of generative and discriminative models." Computer Vision and Pattern Recognition, (2006), vol. 1, pp. 87-94.

[65] Steinwart, I, and Christmann, A. Support Vector Machines. Springer Science & Business Media, 2008.

[66] Paisley, J., Wang, C., & Blei, D. M. "The discrete infinite logistic normal distribution." Bayesian Analysis 7, no. 4 (2012): 997-1034.

[67] Murray, I., Adams, R. P., & MacKay, D. J. "Elliptical slice sampling." arXiv preprint arXiv:1001.0175 (2009).

[68] Sudderth, E. B., Torralba, A., Freeman, W. T., & Willsky, A. S. "Describing visual scenes using transformed objects and parts." International Journal of Computer Vision 77, no. 1-3 (2008): 291-330.

[69] Van Gael, J., Saatci, Y., Teh, Y. W., & Ghahramani, Z. "Beam sampling for the infinite Hidden Markov Model." International conference on Machine learning, (2008), pp. 1088-1095.

[70] Kalli, M., Griffin, J. E., & Walker, S. G. "Slice sampling mixture models." Statistics and Computing 21, no. 1 (2011): 93-105.

[71] Ishwaran, H., & Zarepour, M. "Markov chain Monte Carlo in approximate Dirichlet and beta two-parameter process hierarchical models." Biometrika 87, no. 2 (2000): 371-390.

[72] Cha, S. H. "Comprehensive survey on distance/similarity measures between probability density functions." International Journal of Mathematical Models and Methods in Applied Sciences, 4(1), (2007):300–3007.

[73] Gershman, S. J., & Blei, D. M. "A tutorial on Bayesian nonparametric models." Journal of Mathematical Psychology 56, no. 1 (2012): 1-12.

[74] Li, W., Zhang, Z., & Liu, Z. "Action recognition based on a bag of 3d points." Computer Vision and Pattern Recognition Workshops, (2010), pp. 9-14.

[75] Microsoft Developer Network, “Joint orientation, Kinect for Windows” Retrieved from msdn.microsoft.com/en-us/library/hh973073.aspx

[76] Fox, E. B., Sudderth, E. B., Jordan, M. I., & Willsky, A. S. "A sticky HDP-HMM with application to speaker diarization." The Annals of Applied Statistics (2011): 1020-1056.

[77] Dalal, N. and Triggs, B., 2005, June. "Histograms of oriented gradients for human detection." Computer Vision and Pattern Recognition, (2005), vol. 1, pp. 886-893.

[78] Zhu, Y., Chen, W., & Guo, G.. "Evaluating spatiotemporal interest point features for depth-based action recognition." Image and Vision Computing 32, no. 8 (2014): 453-464.

[79] Devanne, M., Wannous, H., Berretti, S., Pala, P., Daoudi, M., & Del Bimbo, A. "3-D human action recognition by shape analysis of motion trajectories on Riemannian manifold." IEEE Transactions on Cybernetics 45, no. 7 (2015): 1340-1352.

[80] Ohn-Bar, E., & Trivedi, M. "Joint angles similarities and HOG2 for action recognition." Computer Vision and Pattern Recognition Workshops, (2013), pp. 465-470.

[81] Fine, S., Singer, Y., & Tishby, N. "The hierarchical hidden Markov model: Analysis and applications." Machine Learning 32, no. 1 (1998): 41-62.

[82] Krishnapuram, B., Carin, L., Figueiredo, M. A., & Hartemink, A. J. "Sparse multinomial logistic regression: Fast algorithms and generalization bounds." Pattern Analysis and Machine Intelligence 27, no. 6 (2005): 957-968.

[83] Murphy, K. P., & Paskin, M. A. "Linear-time inference in hierarchical HMMs." Advances in Neural Information Processing Systems 2 (2002): 833-840.

[84] Heller, K. A., Teh, Y. W., & Görür, D. "Infinite hierarchical hidden Markov models." International Conference on Artificial Intelligence and Statistics, (2009), pp. 224-231.

[85] Sung, J., Ponce, C., Selman, B., & Saxena, A. "Human activity detection from RGBD images." Pattern, Activity, and Intent Recognition 64 (2011).

[86] Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B., & Weber, A. "Documentation mocap database HDM05.", Technical report, No. CG-2007-2, ISSN 1610-8892, Universität Bonn (2007).

[87] Song, S., & Xiao, J. "Tracking revisited using rgbd camera: Unified benchmark and baselines." International Conference on Computer Vision, (2013), pp. 233-240.

[88] Wang, J., Liu, Z., Wu, Y., & Yuan, J. "Learning actionlet ensemble for 3D human action recognition." Pattern Analysis and Machine Intelligence 36, no. 5 (2014): 914-927.

[89] Hu, J. F., Zheng, W. S., Lai, J., & Zhang, J. "Jointly learning heterogeneous features for RGB-D activity recognition." Computer Vision and Pattern Recognition, (2015), pp. 5344-5352.

[90] Shan, J., & Akella, S.. "3D human action segmentation and recognition using pose kinetic energy." Advanced Robotics and its Social Impacts (ARSO), (2014), pp. 69-75.

[91] Kurihara, K., Welling, M., & Teh, Y. W. "Collapsed variational Dirichlet process mixture models." International Joint Conferences on Artificial Intelligence, vol. 7, (2007), pp. 2796-2801.

[92] Johnson, M. J., and Willsky, A. S. "Bayesian nonparametric hidden semi-Markov models." The Journal of Machine Learning Research 14, no. 1 (2013): 673-701.

[93] Figueiredo, M. A. "Adaptive sparseness for supervised learning." Pattern Analysis and Machine Intelligence 25, no. 9 (2003): 1150-1159.

[94] Jordan, A. "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes." Advances in Neural Information Processing Systems 14 (2002): 841.

[95] Quattoni, A., Wang, S., Morency, L. P., Collins, M., & Darrell, T. "Hidden conditional random fields." IEEE Transactions on Pattern Analysis & Machine Intelligence 10 (2007): 1848-1852.

[96] Gunawardana, A., Mahajan, M., Acero, A., & Platt, J. C. "Hidden conditional random fields for phone classification." INTERSPEECH (2005), pp. 1117-1120.

[97] Qi, Y., Szummer, M., & Minka, T. P. "Bayesian conditional random fields." International Workshop on Artificial Intelligence and Statistics, (2005), pp. 269-276.

[98] Bhattacharya, A., Pati, D., Pillai, N. S., & Dunson, D. B. "Dirichlet–Laplace priors for optimal shrinkage." Journal of the American Statistical Association 110, no. 512 (2015): 1479-1490.

[99] Kotz, S., Kozubowski, T., & Podgorski, K. The Laplace Distribution and Generalizations: A Revisit with Applications to Communications, Economics, Engineering, and Finance. Springer Science & Business Media, 2012.

[100] Eltoft, T., Kim, T., & Lee, T. W. "On the multivariate Laplace distribution." Signal Processing Letters 13, no. 5 (2006): 300-303.

[101] Park, T., & Casella, G. "The Bayesian lasso." Journal of the American Statistical Association 103, no. 482 (2008): 681-686.

[102] Gaglio, S., Re, G. L., & Morana, M. "Human activity recognition process using 3-D posture data." IEEE Transactions on Human-Machine Systems 45, no. 5 (2015): 586-597.

[103] Dittmar, D. "Slice Sampling." TU Darmstadt. Retrieved from ausy. informatik. tu-darmstadt. de/uploads/Teaching/RobotLearningSeminar/Dittmar_RLS_2013. pdf (2013).

[104] Yamato, J., Ohya, J., & Ishii, K. "Recognizing human action in time-sequential images using hidden Markov model." Computer Vision and Pattern Recognition, (1992), pp. 379-385.

[105] Aggarwal, J. K., and Cai, Q. "Human motion analysis: A review." Nonrigid and Articulated Motion Workshop, (1997), pp. 90-102.

[106] Moeslund, T. B., Hilton, A., & Krüger, V.. "A survey of advances in vision-based human motion capture and analysis." Computer Vision and Image Understanding 104, no. 2 (2006): 90-126.

[107] Turaga, P., Chellappa, R., Subrahmanian, V. S., & Udrea, O. "Machine recognition of human activities: A survey." Circuits and Systems for Video Technology, IEEE Transactions on 18, no. 11 (2008): 1473-1488.

[108] Weinland, D., Ronfard, R., & Boyer, E. "A survey of vision-based methods for action representation, segmentation and recognition." Computer Vision and Image Understanding 115, no. 2 (2011): 224-241.

[109] Chaquet, J. M., Carmona, E. J., & Fernández-Caballero, A. " A survey of video datasets for human action and activity recognition." Computer Vision and Image Understanding 117, no. 6 (2013): 633-659.

[110] Cheng, G., Wan, Y., Saudagar, A. N., Namuduri, K., & Buckles, B. P. "Advances in human action recognition: A survey." arXiv preprint arXiv:1501.05964 (2015).

[111] Han, J., Shao, L., Xu, D., & Shotton, J. "Enhanced computer vision with Microsoft Kinect sensor: A review." IEEE Transactions on Cybernetics 43, no. 5 (2013): 1318-1334.

[112] Ye, M., Zhang, Q., Wang, L., Zhu, J., Yang, R., & Gall, J. "A survey on human motion analysis from depth data." Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications, (2013), pp. 149-187.

[113] Chen, C., Jafari, R., & Kehtarnavaz, N. "A survey of depth and inertial sensor fusion for human action recognition." Multimedia Tools and Applications (2015): 1-21.

[114] Presti, L. L., & La Cascia, M. "3D skeleton-based human action classification: A survey." Pattern Recognition 53, no. C (2016): 130-147.

[115] Davis, J. W., & Bobick, A. E.. "The representation and recognition of human movement using temporal templates." Computer Vision and Pattern Recognition, (1997), pp. 928-934.

[116] Wang, Y., Huang, K., & Tan, T. "Human activity recognition based on r transform." Computer Vision and Pattern Recognition, (2007), pp. 1-8.

[117] Blank, M., Gorelick, L., Shechtman, E., Irani, M., & Basri, R. "Actions as space-time shapes." International Conference on Computer Vision, (2005), vol. 2, pp. 1395-1402.

[118] Yu, E., & Aggarwal, J. K. "Human action recognition with extremities as semantic posture representation." Computer Vision and Pattern Recognition Workshops, (2009), pp. 1-8.

[119] Qian, H., Mao, Y., Xiang, W., & Wang, Z. "Recognition of human activities using SVM multi-class classifier." Pattern Recognition Letters 31, no. 2 (2010): 100-111.

[120] Ni, B., Wang, G., & Moulin, P. "Rgbd-hudaact: A color-depth video database for human daily activity recognition." Consumer Depth Cameras for Computer Vision, (2013), pp. 193-208.

[121] Jalal, A., Uddin, M. Z., Kim, J. T., & Kim, T. S. "Recognition of human home activities via depth silhouettes and $\Re$ transformation for smart homes." Indoor and Built Environment (2011): 1420326X11423163.

[122] Yang, X., Zhang, C., & Tian, Y. "Recognizing actions using depth motion maps-based histograms of oriented gradients." International Conference on Multimedia, (2012), pp. 1057-1060.

[123] Vieira, A. W., Nascimento, E. R., Oliveira, G. L., Liu, Z., & Campos, M. F. "Stop: Space-time occupancy patterns for 3d action recognition from depth map sequences." Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, (2012), pp. 252-259.

[124] Oreifej, O., & Liu, Z. "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences." Computer Vision and Pattern Recognition, pp. 716-723. 2013.

[125] Efros, A. A., Berg, A. C., Mori, G., & Malik, J. "Recognizing action at a distance." International Conference on Computer Vision, (2003), pp. 726-733.

[126] Ikizler-Cinbis, N., & Sclaroff, S. "Object, scene and actions: Combining multiple features for human action recognition." European Conference on Computer Vision (2010): 494-507.

[127] Ballin, G., Munaro, M., & Menegatti, E. "Human action recognition from rgb-d frames based on real-time 3d optical flow estimation." Biologically Inspired Cognitive Architectures, (2013), pp. 65-74.

[128] Laptev, I. "On space-time interest points." International Journal of Computer Vision 64, no. 2-3 (2005): 107-123.

[129] Oikonomopoulos, A., Patras, I., & Pantic, M. "Spatiotemporal salient points for visual recognition of human actions." IEEE Transactions on Systems, Man, and Cybernetics 36, no. 3 (2005): 710-719.

[130] Rapantzikos, K., Avrithis, Y., & Kollias, S. "Spatiotemporal saliency for event detection and representation in the 3D wavelet domain: potential in human action recognition." International Conference on Image and Video Retrieval, (2007), pp. 294-301.

[131] Klaser, A., Marszałek, M., & Schmid, C. "A spatio-temporal descriptor based on 3d-gradients." British Machine Vision Conference, (2008), pp. 275-1.

[132] Laptev, I., Marszałek, M., Schmid C., & Rozenfeld, B. "Learning realistic human actions from movies." Computer Vision and Pattern Recognition, (2008), pp. 1-8.

[133] Emonet, R., Varadarajan, J., & Odobez, J. M. "Temporal analysis of motif mixtures using Dirichlet processes." Pattern Analysis and Machine Intelligence, (2014), 36(1), 140-156.

[134] Zhu, Y., Chen, W., & Guo, G. "Evaluating spatiotemporal interest point features for depth-based action recognition." Image and Vision Computing 32, no. 8 (2014): 453-464.

[135] Dollár, P., Rabaud, V., Cottrell, G., & Belongie, S. "Behavior recognition via sparse spatio-temporal features." Visual Surveillance and Performance Evaluation of Tracking and Surveillance, (2005), pp. 65-72.

[136] Zhang, H., & Parker, L. E. "4-dimensional local spatio-temporal features for human activity recognition." Intelligent Robots and Systems, (2011), pp. 2044-2049.

[137] Xia, L. and Aggarwal, J.K. "Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. " Computer Vision and Pattern Recognition, (2013), pp. 2834-2841.

[138] Rahmani, H., Mahmood, A., Huynh, D. Q., & Mian, A. "HOPC: Histogram of oriented principal components of 3D pointclouds for action recognition." European Conference on Computer Vision, (2014), pp. 742-757.

[139] Zhao, Y., Liu, Z., Yang, L., & Cheng, H. "Combing rgb and depth map features for human activity recognition." Signal & Information Processing Association Annual Summit and Conference, (2012), pp. 1-4.

[140] Cheng, Z., Qin, L., Ye, Y., Huang, Q., & Tian, Q. "Human daily action analysis with multi-view and color-depth data." European Conference on Computer Vision Workshops and Demonstrations, (2012), pp. 52-61.

[141] Wang, J., Liu, Z., Chorowski, J., Chen, Z. and Wu, Y., "Robust 3d action recognition with random occupancy patterns." European Conference on Computer Vision, (2012), pp. 872-885.

[142] Sigal, L. "Human pose estimation." Computer Vision (2014), pp. 362-370.

[143] Kuettel, D., Breitenstein, M. D., Van Gool, L., & Ferrari, V. "What's going on? Discovering spatio-temporal dependencies in dynamic scenes." Computer Vision and Pattern Recognition, (2010), pp. 1951-1958.

[144] Poppe, R. "Vision-based human motion analysis: An overview." Computer Vision and Image Understanding 108, no. 1 (2007): 4-18.

[145] Hospedales, T., Gong, S., & Xiang, T. "A Markov clustering topic model for mining behaviour in video." International Conference on Computer Vision, (2009), pp. 1165-1172.

[146] Varadarajan, J., Emonet, R., & Odobez, J. M. "A sequential topic model for mining recurrent activities from long term video logs." International Journal of Computer Vision, (2013), 103(1), 100-126.

[147] Yang, X., & Tian, Y. "Eigenjoints-based action recognition using naive-Bayes-nearest-neighbor." Computer Vision and Pattern Recognition Workshops, (2012), pp. 14-19.

[148] Ellis, C., Masood, S. Z., Tappen, M. F., Laviola Jr, J. J., & Sukthankar, R. "Exploring the trade-off between accuracy and observational latency in action recognition." International Journal of Computer Vision 101, no. 3 (2013): 420-436.

[149] Raptis, M., Kirovski, D., & Hoppe, H. "Real-time classification of dance gestures from skeleton animation." SIGGRAPH/Eurographics Symposium on Computer Animation, (2011), pp. 147-156.

[150] Miranda, L., Vieira, T., Martínez, D., Lewiner, T., Vieira, A. W., & Campos, M. F. "Online gesture recognition from pose kernel learning and decision forests." Pattern Recognition Letters 39 (2014): 65-73.

[151] Lillo, I., Soto, A., & Niebles, J. "Discriminative hierarchical modeling of spatio-temporally composable human activities." Computer Vision and Pattern Recognition, (2014), pp. 812-819.

[152] Zhang, C., & Tian, Y. "Rgb-d camera-based daily living activity recognition." Journal of Computer Vision and Image Processing 2, no. 4 (2012): 12.

[153] Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., & Bajcsy, R. "Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition." Journal of Visual Communication and Image Representation 25, no. 1 (2014): 24-38.

[154] Sempena, S., Maulidevi, N. U., & Aryan, P. R. "Human action recognition using dynamic time warping." International Conference on Electrical Engineering and Informatics, (2011), pp. 1-5.

[155] Vemulapalli, R., Arrate, F., & Chellappa, R. "R3DG features: Relative 3D geometry-based skeletal representations for human action recognition." Computer Vision and Image Understanding, (2016).

[156] Evangelidis, G., Singh, G., & Horaud, R. "Skeletal quads: Human action recognition using joint quadruples." International Conference on Pattern Recognition, (2014), pp. 4513-4518.

[157] Gowayyed, M. A., Torki, M., Hussein, M. E., & El-Saban, M. "Histogram of oriented displacements (HOD): Describing trajectories of human joints for action recognition." International Joint Conference on Artificial Intelligence, (2013).

[158] Hussein, M. E., Torki, M., Gowayyed, M. A., & El-Saban, M. "Human action recognition using a temporal hierarchy of covariance descriptors on 3D joint locations." International Joint Conference on Artificial Intelligence, (2013), vol. 13, pp. 2466-2472.

[159] Yun, K., Honorio, J., Chattopadhyay, D., Berg, T. L., & Samaras, D. "Two-person interaction detection using body-pose features and multiple instance learning." Computer Vision and Pattern Recognition Workshops, (2012), pp. 28-35.

[160] Eweiwi, A., Cheema, M. S., Bauckhage, C., & Gall, J. "Efficient pose-based action recognition." Asian Conference on Computer Vision, (2014), pp. 428-443.

[161] Climent-Pérez, P., Chaaraoui, A. A., Padilla-López, J. R., & Flórez-Revuelta, F. "Optimal joint selection for skeletal data from RGB-D devices using a genetic algorithm." Advances in Computational Intelligence, (2012), pp. 163-174.

[162] Anirudh, R., Turaga, P., Su, J., & Srivastava, A. "Elastic functional coding of human actions: from vector-fields to latent variables." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2015), pp. 3147-3155.

[163] Slama, R., Wannous, H., Daoudi, M., & Srivastava, A. "Accurate 3D action recognition using learning on the Grassmann manifold." Pattern Recognition 48, no. 2 (2015): 556-567.

[164] Jiang, H. "Discriminative training of HMMs for automatic speech recognition: A survey." Computer Speech & Language 24, no. 4 (2010): 589-608.

[165] He, X., Deng, L., & Chou, W. "Discriminative learning in sequential pattern recognition." Signal Processing Magazine 25, no. 5 (2008): 14-36.

[166] Yu, D., and Deng, L. "Large-margin discriminative training of hidden Markov models for speech recognition." International Conference on Semantic Computing, (2007), pp. 429-438.

[167] Nguyen, N. T., Phung, D. Q., Venkatesh, S., & Bui, H. "Learning and detecting activities from movement trajectories using the hierarchical hidden Markov model." Computer Vision and Pattern Recognition, (2005), vol. 2, pp. 955-960.

[168] Natarajan, P., and Nevatia, R. "Coupled hidden semi Markov models for activity recognition." IEEE Workshop on Motion and Video Computing, (2007), pp. 10-10.

[169] Dai, P., Di, H., Dong, L., Tao, L., & Xu, G. "Group interaction analysis in dynamic context." IEEE Transactions on Systems, Man, and Cybernetics, 38, no. 1 (2008): 275-282.

[170] Wang, S. B., Quattoni, A., Morency, L. P., Demirdjian, D., & Darrell, T., "Hidden conditional random fields for gesture recognition." Computer Vision and Pattern Recognition, (2006), vol. 2, pp. 1521-1527.

[171] Zhang, J., & Gong, S. "Action categorization with modified hidden conditional random field." Pattern Recognition, (2010), 43(1), 197-203.

[172] Lin, T. H., Kaminski, N., & Bar-Joseph, Z. "Alignment and classification of time series gene expression in clinical studies." Bioinformatics 24, no. 13 (2008): i147-i155.

[173] Müller, P., & Quintana, F. A. "Nonparametric Bayesian data analysis." Statistical Science (2004): 95-110.

[174] Foti, N. J., & Williamson, S. A. "A survey of non-exchangeable priors for Bayesian nonparametric models." Pattern Analysis and Machine Intelligence, 37, no. 2 (2015): 359-371.

[175] Alexiou, I., Xiang, T., & Gong, S. "Learning a joint discriminative-generative model for action recognition." International Conference on Systems, Signals and Image Processing, (2015), pp. 1-4.

[176] Huggins, J. H., & Wood, F. "Infinite structured hidden semi-Markov models." arXiv preprint arXiv:1407.0044 (2014).

[177] Hughes, M. C., Kim, D. I., & Sudderth, E. B. "Reliable and scalable variational inference for the hierarchical dirichlet process." Artificial Intelligence and Statistics Conference, (2015).

[178] Hu, D. H., Zhang, X. X., Yin, J., Zheng, V. W., & Yang, Q. "Abnormal activity recognition based on HDP-HMM models." International Joint Conference on Artificial Intelligence, (2009), pp. 1715-1720.

[179] Bargi, A., Xu, D., Yi, R., & Piccardi, M. "An adaptive online HDP-HMM for segmentation and classification of sequential data." arXiv preprint arXiv:1503.02761 (2015).

[180] Kooij, J. F., Englebienne, G., & Gavrila, D. M. "A non-parametric hierarchical model to discover behavior dynamics from tracks." European Conference on Computer Vision, (2012), pp. 270-283.

[181] Gael, J. V., Teh, Y. W., & Ghahramani, Z. "The infinite factorial hidden Markov model." Advances in Neural Information Processing Systems, (2009), pp. 1697-1704.

[182] Valera, I., Ruiz, F. J., & Perez-Cruz, F. "Infinite factorial unbounded hidden Markov model for blind multiuser channel estimation." International Workshop on Cognitive Information Processing, (2014), pp. 1-6.

[183] Mcauliffe, J. D., & Blei, D. M. "Supervised topic models." Advances in Neural Information Processing Systems, (2008), pp. 121-128.

[184] Dai, A. M., & Storkey, A. J. "The supervised hierarchical Dirichlet process." Pattern Analysis and Machine Intelligence 37, no. 2 (2015): 243-255.

[185] Bousmalis, K., Zafeiriou, S., Morency, L. P., & Pantic, M. "Infinite hidden conditional random fields for human behavior analysis." Neural Networks and Learning Systems 24, no. 1 (2013): 170-177.

[186] Bousmalis, K., Zafeiriou, S., Morency, L.P., Pantic, M. and Ghahramani, Z., "Variational hidden conditional random fields with coupled Dirichlet process mixtures. " Machine Learning and Knowledge Discovery in Databases, (2013), pp. 531-547.

[187] Lin, L., Wang, K., Zuo, W., Wang, M., Luo, J., & Zhang, L. "A deep structured model with radius–margin bound for 3D human activity recognition." International Journal of Computer Vision (2015): 1-18.

[188] Shou, Z., Wang, D., & Chang, S. F. "Action Temporal Localization in Untrimmed Videos via Multi-stage CNNs." arXiv preprint arXiv:1601.02129 (2016).

[189] Ryoo, M. S., & Matthies, L. "First-person activity recognition: feature, temporal structure, and prediction." International Journal of Computer Vision (2015): 1-22.

[190] Sugiura, T., Goto, N., & Hayashi, A. "A discriminative model corresponding to hierarchical HMMs." Intelligent Data Engineering and Automated Learning, (2007), pp. 375-384.

[191] Shao, L., Han, J., Kohli, P., & Zhang, Z., (Eds.) Computer Vision and Machine Learning with RGB-D Sensors. Springer, 2014.

[192] Raman, N., Maybank, S. J., & Zhang, D. "Action classification using a discriminative non-parametric Hidden Markov Model." International Conference on Machine Vision, (2013), vol. 9067, pp. 10.

[193] Raman, N. & Maybank, S. J. "Action classification using a discriminative multilevel HDP-HMM." Neurocomputing 154 (2015): 149-161.

[194] Raman, N. & Maybank, S. J. "Activity recognition using a supervised non-parametric hierarchical HMM." Neurocomputing 199 (2016): 163-177.

[195] Raman, N. & Maybank, S. J. "Non-parametric hidden conditional random fields for action classification." IEEE International Joint Conference on Neural Networks, preprint, (2016).