Activity Recognition using a supervised non-parametric Hierarchical HMM

Natraj Raman[1] and S.J Maybank[2]
Department of Computer Science and Information Systems, Birkbeck, University of London

Abstract

The problem of classifying human activities occurring in depth image sequences is addressed. The 3D joint positions of a human skeleton and the local depth image pattern around these joint positions define the features. A two level *hierarchical* Hidden Markov Model (H-HMM), with independent Markov chains for the joint positions and depth image pattern, is used to model the features. The states corresponding to the H-HMM bottom level characterize the granular poses while the top level characterizes the coarser actions associated with the activities. Further, the H-HMM is based on a Hierarchical Dirichlet Process (HDP), and is fully non-parametric with the number of pose and action states inferred automatically from data. This is a significant advantage over classical HMM and its extensions. In order to perform classification, the relationships between the actions and the activity labels are captured using multinomial logistic regression. The proposed inference procedure ensures alignment of actions from activities with similar labels. Our construction enables information sharing, allows incorporation of unlabelled examples and provides a flexible factorized representation to include multiple data channels. Experiments with multiple real world datasets show the efficacy of our classification approach.

Keywords: activity classification; depth image sequences; hierarchical HMM; HDP; inference; multinomial logistic regression;

1 Introduction

Activity recognition involves automatic identification of interesting events that occur in a video. It has applications in diverse areas such as video synthesis, smart surveillance and human computer interaction. The recent advent of depth sensing technology that produces depth images in addition to the RGB images has offered opportunities to solve the challenging activity recognition problem. The depth images facilitate robust extraction of the human silhouette and the estimation of a human skeleton's 3D joint positions [1]. High level actions and activities can be inferred from these joint positions.

An activity is typically composed of a set of actions that occur over time. An action in turn is composed of a sequence of skeleton and object poses. The skeleton pose is a particular arrangement of the joint positions and the object pose is a specific representation of an object associated with the action. For example, a *rinse-mouth* activity may be composed of *drink* and *spit* actions. The *drink* action may involve skeleton poses corresponding to *lifting an arm* and the object pose may be a representation of a *mug*. The same pose may be present in different actions and the same action may be present in multiple activities. This composition allows the sharing of data across the activities and the learning of poses and actions 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. Thus, decomposing an activity into a set of actions and in turn an action into a set of poses enables information sharing and model simplification. The precise definition of the time scale for the actions

[1] nraman01@dcs.bbk.ac.uk Corresponding author Birkbeck, Malet St, London WC1E7HX, UK. T: +442076316700
[2] sjmaybank@dcs.bbk.ac.uk

and activities may depend on the task. In this work, evaluations are performed on fairly simple activities that span less than a minute. The joint positions extracted from a depth image are used for representing the skeleton poses. The object poses are represented using the information in the depth image patches around the joint positions.

A natural way to model a sequence of observations is to use a state-space model such as a Hidden Markov Model (HMM). In an HMM, discrete state variables are linked in a Markov chain by a state transition matrix and observations are drawn independently from a distribution conditioned on the state [2]. A simple HMM is not sufficient in our case, because there are two sequences at different levels – a top level for the coarse action sequence and a bottom level for the granular pose sequence. Intuitively, for a given action state at the top level, we have 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 of its own [3]. It generates sequences by recursively activating the sub-states of a state. In this context, when an action state is activated, it will use its own probabilistic model to emit a sequence of pose states with a pose state emitting an observation. We can flatten the H-HMM to a standard HMM by introducing a large number of states, however the inference of the activities would become intractable.

In the above 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 for circumventing this problem is to carry out training using different choices for the number of states and then to apply a model selection criterion. A better approach than this ad hoc procedure is to estimate the correct number of states *automatically* from the data.

The Dirichlet Process (DP) is a non-parametric Bayesian method used for mixture modelling. It estimates the number of mixture components automatically from data. Its extension, the Hierarchical Dirichlet Process (HDP), is used for modelling groups of data. A mixture model is produced for each group but all the groups share the same mixture components [4]. By drawing parallels from a HMM state to a group in grouped data, the HDP-HMM [5] can be viewed as a non-parametric variant of the classical HMM in which the number of hidden states is inferred from data. This paper uses a non-parametric extension to the H-HMM for modelling activities. The number of action states and the number of pose states are not bounded a priori. During inference these numbers are automatically estimated. The pose states use a factorized representation for the skeleton and object poses.

The above non-parametric H-HMM that models activities cannot be used for classification. A separate H-HMM can be trained for each activity class, but this would prohibit the sharing of actions and poses across the activity classes. In this work, a *single* H-HMM is trained for all the activities together. In order to perform 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 [6]. 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.

Our ***main contribution*** is the definition of a new factorized non-parametric H-HMM model integrated with multinomial logistic regression. We also propose a tractable inference procedure that is suitable for sequential data and conduct experiments on multiple real world datasets. This proposed model offers the following advantages – (a) The hierarchical composition of actions and poses enables information sharing and model simplification, (b) The non-parametric 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) Unlabelled examples can be used thus promoting semi-supervised learning. Although our model is generic and can be applied to other hierarchical sequence classification problems, our experiments focus on the activity classification problem. Figure 1 provides an overview of our approach.

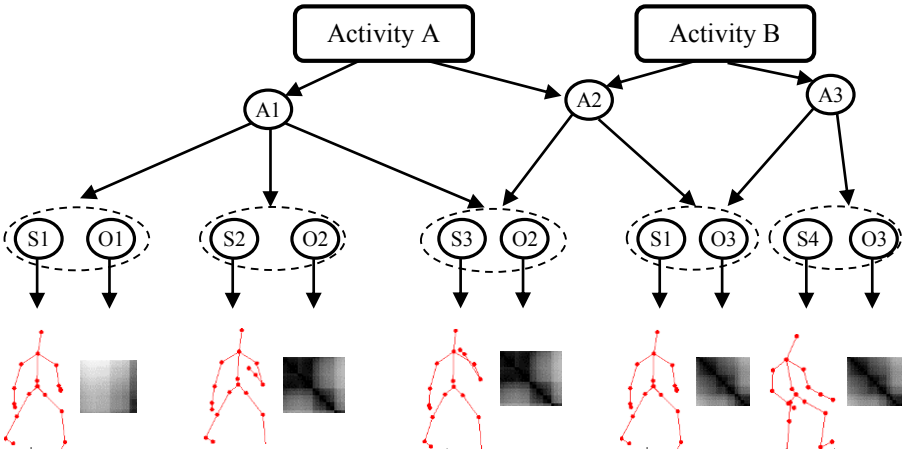

Figure 1: Activity Recognition Overview – Poses are learnt from observations and actions are learnt from poses. S1 to S4 represent the skeleton pose states, O1 to O3 the object pose states and A1 to A3 the action states. The skeleton poses are based on the 3D joint positions and the object poses are 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.

The paper is organized as follows. Section 2 briefly reviews the related work, section 3 provides relevant background, section 4 explains the model and section 5 contains the inference procedure. Experiments conducted on the activity datasets are produced in section 6 and section 7 is a conclusion.

2 Related Research

Human activity analysis is a very broad research area. The various techniques are reviewed in [7]. We focus specifically on approaches used for activity recognition in depth images. An overview of such approaches can be found in [8, 9 and 10].

Several activity classification methods rely on computing sophisticated features from the 3D joint positions. In [11], a conjunction of the features for a subset of joints, called an *actionlet,* is used to represent the interactions between joints. The temporal dynamics are captured using a Fourier temporal pyramid and a multiple kernel learning approach is used for classification. In [12], *heterogeneous features* are constructed from the skeleton, colour and depth patterns of an RGB-D image with the Fourier temporal pyramid used here as well for representing the temporal dynamics. A set of subspaces are then mined from these features in order to perform classification. In [13], a histogram based representation of the joint positions, called *HOJ3D*, is employed to describe human poses. A discrete HMM is then used to model a low dimensional projection of these features. A similar histogram based representation, but using 2D projections of 3D trajectories for describing displacements, called *HOD*, is used in [14]. In [15], a spatio-temporal representation, called *atomic action template*, is composed from key poses that are fed into classification models such as Support Vector Machine (SVM) and Random Forest. The key poses are identified based on changes in the kinetic energy of the joints. In [16], various interest point detectors such as Harris3D, cuboid, Hessian

etc. are combined with local feature descriptors such as SURF, HOG to create *STIP* features. A bag of words representation of these spatio-temporal descriptors are then used with an SVM to classify actions. In [17], the 3D point clouds in depth image sequences are directly processed in order to ensure viewpoint invariance. A descriptor called Histogram of Oriented Principal Components (*HOPC*) is encoded at each point. Using this descriptor, the spatio-temporal key points in the 3D point clouds are detected and an SVM is used for classification.

In contrast to the above methods, the method proposed in this paper could potentially benefit other sequences with a hierarchical structure. The model proposed here is agnostic to the features and in our experiments good results are obtained even with very simple features. Also, unlike some of the methods above, the sequential nature of the observations is an integral part of the model with the Markovian dynamics ensuring that noisy inputs are tolerated. Further, the factorized structure in the model provides flexibility to include other data channels.

The explicitly engineered features listed in the above methods are seldom applicable universally. Learning complex features automatically from data is an attractive alternative to hand-crafted features. Convolutional Neural Networks (CNN) [18], inspired by biological processes, use multiple layers of small neurons that overlap in order to represent an image. A hierarchy of trainable filters exploit the strong local spatial correlations present in the images and when combined with feature pooling operations, CNNs can automatically construct complex features from the raw inputs. Stellar results for static image recognition problems are reported [19] with CNN.

An additional temporal dimension must be incorporated into the CNN for action classification. This is to ensure that the motion information encoded across video frames is captured. In [20], multiple distinct convolutional operations are applied at the same location of the input in order to recognize human actions. Their 3D CNN architecture performs convolution on multiple channels of information generated from adjacent video frames and finally combines all this information. In [21], the CNN temporal connectivity pattern that takes best advantage of local motion information in videos is explored. Three patterns, namely *Early Fusion* that combines information on the pixel level across a time window, *Late Fusion* that uses two separate single frame networks and *Slow Fusion* that is a hybrid between the above two, are evaluated in a multi-resolution framework on the Sports-1M dataset for action classification. A different CNN architecture that uses two separate recognition streams, one for spatial data and another for temporal data, in order to classify actions, is investigated in [22]. The spatial stream CNN performs action recognition on the still images and the temporal stream CNN recognizes action from motion in the form of dense optical flow. This avoids the need to implicitly learn spatio-temporal features in the first layer of the CNN, which is often a difficult task. In [23], a Recurrent Neural Network (RNN) that uses internal states to exhibit dynamic temporal behaviour is combined with a CNN to model temporal information in videos. A visual CNN layer is connected to a long range temporal recursion layer that produces the output predictions for activity classification. This method allows sequential data of varying lengths for the input and output. A similar approach is employed in [24] with the output of a CNN fed into a RNN that uses Long Short-Term Memory (LSTM) cells. The actions in the videos here are explicitly modelled as ordered sequences of frames. By utilizing special gating procedures, the LSTM cells can learn representations from long input sequences.

The Markovian model used in this paper is fundamentally different to the above approaches based on neural networks. Although RNNs and CNNs offer the advantage of learning the features automatically from data, training them is often complex and computationally expensive. It is unclear if the CNN and RNN based models can produce good results with relatively low numbers of training examples. Further the H-HMM, being a generative model, allows the use of unlabelled examples.

Some approaches recast the action recognition problem as a statistical analysis on the shape space manifold. In [25], the trajectories described by the 3D joint positions are interpreted in a Riemannian manifold and an elastic metric is used to measure the similarity between 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. The shape analysis of curves is now extended to the trajectories. A similar approach is followed in [26] where a low-dimensional embedding of the actions is learnt from the high dimensional trajectories using a manifold functional variant of Principal Component Analysis (PCA). A warp invariant representation of the action sequences is provided using a representation called transport square root velocity function. The extension of manifold learning to dynamic points is non-trivial and a relatively new area. In contrast, it is well known that the HMM based models used here can capture temporal structure and handle noise effectively. It is also unclear whether hierarchical structures can be modelled and captured effectively in a manifold learning framework.

Instead of focusing on the accuracy of action recognition, some approaches focus on the recognition speed in order to scale up to large size problems. In [27], semantic texton forests, which act directly on pixels without using expensive descriptors, are applied to local space-time volumes to generate visual code words. A kernel k-means forest classifier is used for classification. In [28], a real-time action recognition system which integrates a fast random sampling method with local spatio-temporal features is presented. The local features are extracted from a computationally efficient local part model that includes both structure and order information. Motion information from compressed videos is used in [29] in order to determine local descriptors. In order to generate the visual code book, kd-forest approximate nearest neighbour search is employed. Unlike the above techniques, the focus of this paper is on recognition accuracy.

Undirected graphical models have also been used for activity recognition tasks. In [30], the spatio-temporal relations between human poses and objects are modelled using a Conditional Random Field (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 in the past and future, multiple possible graph structures are investigated based on dynamic programming techniques. In [31], group activities that involve both individual persons and the interactions between them are explored in a latent variable framework similar to Hidden CRF (HCRF). The structure of the hidden layer is implicitly inferred during learning. In [32], collective activities involving groups of people are recognized using a Hierarchical Random Field (HiRF). The higher order temporal structures in the videos are captured by using hierarchical dependencies between the variables and learning is specified in a max-margin framework. In contrast to above models that are based on Markov Random Fields, a directed graphical model is used in this paper.

Non-parametric Bayesian models is a vast area. We are interested in methods that are based on HDP [5, 33, 34 and 35] applied to vision problems [36]. In particular, we briefly review those HDP based works that target action and activity recognition in videos to illustrate the new aspects of our approach.

In [37], a HDP prior is used for a Markov switching model in order to perform online segmentation. An initial bootstrap phase is followed by an adaptive online phase in order to continuously segment and classify the videos. This is completely different to our model in which classification is achieved through logistic regression and the structure is hierarchical. The HDP priors have been used for detecting abnormalities in robotic assembly tasks [35] and activities [38]. In [38], abnormal activities are identified by applying the standard HDP-HMM, followed by a one-class SVM to filter out unlikely activities. This method combines the generative HDP with the discriminative SVM technique but in

separate phases. In contrast, we include logistic regression as an integral part of the model with the regression coefficients inducing the states during inference. A mixture of switching linear dynamic systems is used to discover actions and behaviours in [39]. Unlike our work, in which the objective is classification, the objective here is unsupervised learning based on hierarchical clustering. In [40], an additional level is added to the canonical HDP and parameter transformations that are learnt in a discriminative manner are used to classify actions. The use of a generative hierarchical HMM and logistic regression, both involving a different model structure and a different inference procedure, distinguishes our work from this.

Using a generative process and a linear model to capture the relationship between a group of observations and its associated label has been explored before. Relevant examples are the supervised Latent Dirichlet Allocation [41] and its non-parametric extension, supervised HDP [42]. These techniques were mainly used in topic models for document labelling. The use of an H-HMM and the application to a vision problem involving sequence classification makes our paper very different from [41] and [42]. In particular, the inference procedure in our model takes into consideration the factorized, hierarchical nature of the observations and the standard collapsed sampling used in document labelling cannot be used for sequential data.

The inference procedure described in this paper uses variants of the standard forward-backward algorithm [2] when sampling the state sequences. In particular Viterbi decoding is used when determining the state trajectory for test sequences. In [43] it is argued that the state sequence paths computed as the maximum of state probabilities in the Viterbi algorithm may not comprise valid paths and that the maximum probabilities may not be truly representative. Further the authors argue that such paths constitute an overly specific inference. They propose an alternative inference procedure called *state sequence analysis* that prunes the search space and uses a recursive relation to evaluate state sequence probabilities. Although this approach is interesting, their algorithm identifies only duration-free sequence of states. Since the forward-backward procedure is well-established, mature and computationally efficient we use it here.

To summarize, the literature contains work related to activity recognition, hierarchical structures, HDP based Markovian models and supervised non-parametric Bayesian methods, considered separately. In our approach, we bring together a hierarchical, factorized Markov model, multinomial logistic regression and non-parametric Bayesian techniques and derive inference procedures that can be applied to sequential data. Experiments conducted on both depth based and motion capture based datasets highlight the generic applicability of our approach.

3 Preliminaries

Background information on the classical HMM, hierarchical HMM, their Bayesian extensions and the non-parametric variant of H-HMM is provided.

3.1 Classical HMM

Let x_t be an observation at time t and let z_t be its corresponding hidden state. In a classical HMM, the hidden state sequence follows a first order Markov chain $z_t \perp z_{1:t-2} \mid z_{t-1}$ and the observations are conditionally independent given the current state i.e. $x_t \perp z_{1:t-1}, x_{1:t-1} \mid z_t$. There is a finite number K of hidden states and $z_t \in \{1,2 \dots K\}$.

The model is parameterized using distributions for state transitions and observation emissions. Specifically, the probability of transitioning to state k from j is given by $\pi_{j,k} = P(z_t = k | z_{t-1} = j)$ and $\pi_{0,k} = P(z_1 = k)$ is the initial probability of being in state k. Let $\mathcal{N}(\mu, \Sigma)$ be the normal

distribution with mean μ and covariance Σ. If the state at time t is k, then the observation x_t is drawn from a distribution $\mathcal{N}(\mu_k, \Sigma_k)$. Figure 2 provides an overview.

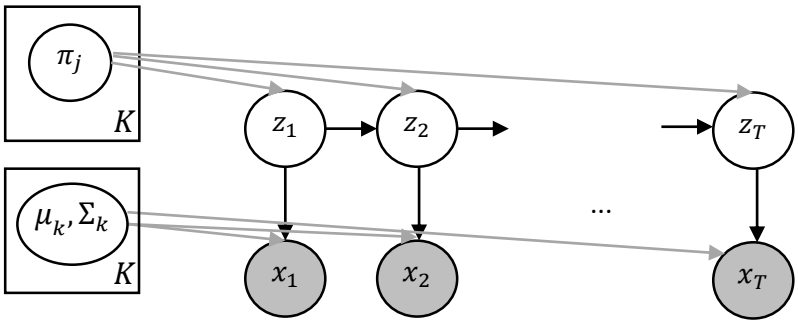

Figure 2: Graphical representation of the classical HMM. The states $z_{1:T}$ evolve based on the transition parameters π and the state at the previous time instant. Observations $x_{1:T}$ are generated from the emission distribution parameters (μ, Σ) of a given state.

3.2 Hierarchical HMM

In the hierarchical extension of the HMM [44], 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 it 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. In order to indicate whether a level has completed emitting its subsequence, let us use a binary variable f_t^l. When $f_t^l = 1$, the state one level above the current level is activated and when $f_t^l = 0$ the state at the current level and all levels above it remains unchanged.

The state transition probabilities of the H-HMM need additional conditioning on the states in the level above them. The probability of transitioning to state k from j for level l is now given by $\pi_{j,k}^l = P\left(z_t^l = k \middle| z_{t-1}^l = j, f_{t-1}^l = f', z_t^{1..l-1} = i\right)$ where i is the current state for all parent levels and $f' \in \{0,1\}$. Assuming that the states that emit observations are at the bottom level, the emission distribution remains identical to that of an HMM. Figure 3 provides an overview.

3.3 Bayesian extensions

For the Bayesian model of a classical HMM, it is necessary to introduce priors. Let the state transitions of a HMM have a Dirichlet prior $\beta \sim Dir(\frac{\gamma}{K} \ldots \frac{\gamma}{K})$ and $\pi_j \sim Dir(\alpha\beta_1 \ldots \alpha\beta_K)$. Here Dir is the Dirichlet [4] distribution, $\gamma, \alpha \in \mathbb{R}^+$ are hyper parameters and the intermediate β ensures that the transitions out of different states are coupled. Similarly, let the mixture means have a normal prior $\mu \sim \mathcal{N}(\mu_0, \Sigma_0)$ and let the covariance have an Inverse-Wishart prior [5, 39] $\Sigma \sim IW(\nu_0, \Delta_0)$.

For the H-HMM, the Dirichlet priors for transitions are now extended to multiple levels with $\beta^l \sim Dir(\frac{\gamma}{K} \ldots \frac{\gamma}{K})$ and $\pi_j^l \sim Dir\left(\alpha\beta_1^l \ldots \alpha\beta_K^l\right)$. The conditional probability of the binary variables when the level below has completed is $P\left(f_t^l = 1 \middle| f_t^{l+1} = 1, z_t^l, z_t^{l-1}\right) = \psi^{l, z_t^l, z_t^{l-1}}$. Here each ψ is a parameter that controls the chance of state transition at a particular level and can be assigned a beta prior $Beta(a, b)$ where $a, b \in \mathbb{R}^+$. The emission distribution prior remains identical to the prior of an HMM.

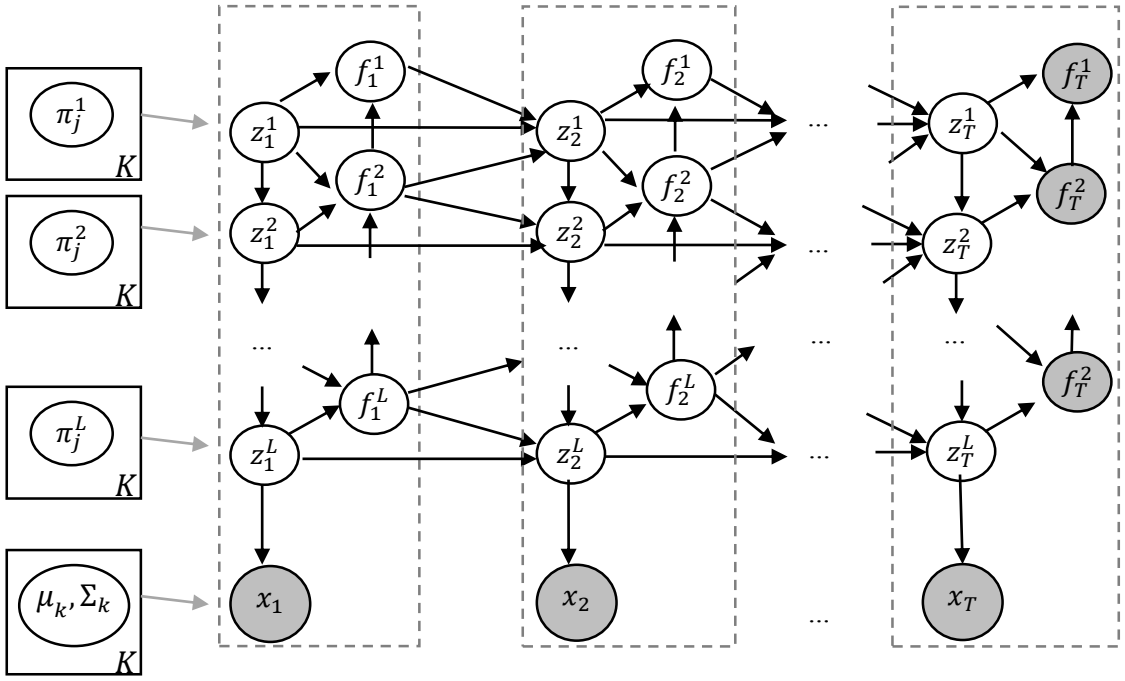

Figure 3: Graphical representation of a Hierarchical HMM [44] showing the states at various levels, the observations and the parameters. Each dotted rectangle groups the variables at a time instant. If the current level l has not finished ($f_{t-1}^l = 0$), then the state z_t^l is the same state z_{t-1}^l 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 (μ, Σ) are shown on the left.

3.4 Non parametric H-HMM

The Dirichlet Process (DP) is a useful nonparametric prior for mixture models. There is no upper bound on the number of components in the mixture. The Dirichlet process is denoted by $DP(\gamma, H)$ where H is a base measure and $\gamma \in \mathbb{R}^+$ is a concentration parameter that controls variability around H. A draw G_0 from a DP provides a probability distribution with infinitely many members. The almost sure discreteness of the drawn measures can be made explicit through a stick breaking process [4]. The distribution G_0 can be written in the following form.

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

$$\beta_k = \beta_k' \prod_{l<k} (1-\beta_l') \qquad \beta_k' \mid \gamma \overset{iid}{\sim} Beta(1,\gamma) \qquad \theta_k \mid H \overset{iid}{\sim} H$$

Here θ_k are the atoms drawn independently from the base distribution and β_k are the probabilities that define the mass on the atoms such that $\sum_{k=1}^{\infty} \beta_k = 1$. It is common to write the probability measure $\beta = \{\beta_k\}_{k=1}^{\infty}$ obtained from (1) as $\beta \sim GEM(\gamma)$.

The HDP is the hierarchical extension of the DP and 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 the groups to have different mixture proportions but share the same mixture components. Specifically,

the set $\{G_j\}_{j=1}^{J}$ of random distributions corresponding to J pre-specified groups are conditionally independent given a base global distribution G_0.

$$G_0 \mid \gamma, H \sim DP(\gamma, H) \qquad G_j \mid \alpha, G_0 \sim DP(\alpha, G_0) \tag{2}$$

Here G_j contains values drawn from G_0 that vary by an amount controlled by a concentration parameter α. Similar to (1) we can write the HDP using a stick breaking process as below with $\pi_j = \{\pi_{jk}\}_{k=1}^{\infty}$ and $\sum_{k=1}^{\infty} \pi_{jk} = 1$.

$$G_j = \sum_{k=1}^{\infty} \pi_{jk} \delta_{\theta_k}$$
$$\pi_{jk} = \pi'_{jk} \prod_{l<k} (1 - \pi'_{jl}) \qquad \pi'_{jk} \mid \alpha, \beta \overset{iid}{\sim} Beta\left(\alpha\beta_k, \alpha(1 - \sum_{l<k} \beta_l)\right) \tag{3}$$

The HDP can be used as a non-parametric prior for a set of mixture models with π_{jk} being interpreted as the probability that defines the mass on the atoms for the group j. An observation x_{jn} belonging to the j^{th} group is generated using a HDP as below.

$$\beta \mid \gamma \sim GEM(\gamma) \qquad \pi_j \mid \alpha, \beta \sim DP(\alpha, \beta) \qquad \theta_k \mid H \sim H$$
$$z_{jn} \mid \pi_j \sim \pi_j \qquad x_{jn} \mid z_{jn}, \{\theta_k\}_{k=1}^{\infty} \sim F(\theta_{z_{jn}}) \tag{4}$$

Here z_{jn} is a latent variable that indicates the mixture component, $GEM(\gamma)$ is the stick breaking process and F denotes a family of distributions parameterized by θ.

The generation process for the HDP is remarkably similar to the generation of a Bayesian HMM. The π_j in (4) are the transition probabilities for a HMM state j and F is a family of Gaussian distributions with $\theta_k = (\mu_k, \Sigma_k)$. Thus a non-parametric HMM can be represented using a HDP with an unbounded number of hidden states.

In the non-parametric H-HMM, each level has a separate HDP prior corresponding to the states of its parent level. Thus the number of states in each H-HMM level is unbounded. Note that we do not consider an alternate definition [33] in which the number of levels in the H-HMM is unbounded.

4 Activity Model

We are given training examples $X = \{x^n\}_{n=1}^{N}, Y = \{y^n\}_{n=1}^{N}$, where $x^n = x_1^n \dots x_T^n$ is an activity observation sequence and $y^n \in \{1 \dots C\}$ its corresponding activity label. An activity observation x_t contains skeleton based features $xs_t \in \mathbb{R}^{d_s}$ and object based features $xo_t \in \mathbb{R}^{d_o}$ with $x_t = (xs_t, xo_t)$. Discussion of the features is deferred to section 6. The objective is classification: given a new test activity sequence $\hat{x}$, the corresponding activity label $\hat{c}$ is predicted using the learned model parameters.

In the H-HMM model proposed here, we restrict the number of levels to two i.e. $L = 2$. 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 that produce the observations. A pose state at time t is factored into skeleton and object pose states zs_t and zo_t respectively. This factorization provides

flexibility to add new data channels if necessary. 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. This simplified model with only two levels ensures tractable inference, while retaining the benefits of a compact hierarchical structure.

The pose states always emit a single observation. Each observation consists of two samples from independent Gaussian distributions - one for the skeletion 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 do not directly depend upon the action states.

In order to perform classification, an activity label is modelled by multinomial logistic regression conditioned on the assignment of action states. Let $\overline{a}$ be the normalized empirical frequencies of the action states in a sequence of actions associated with 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 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.

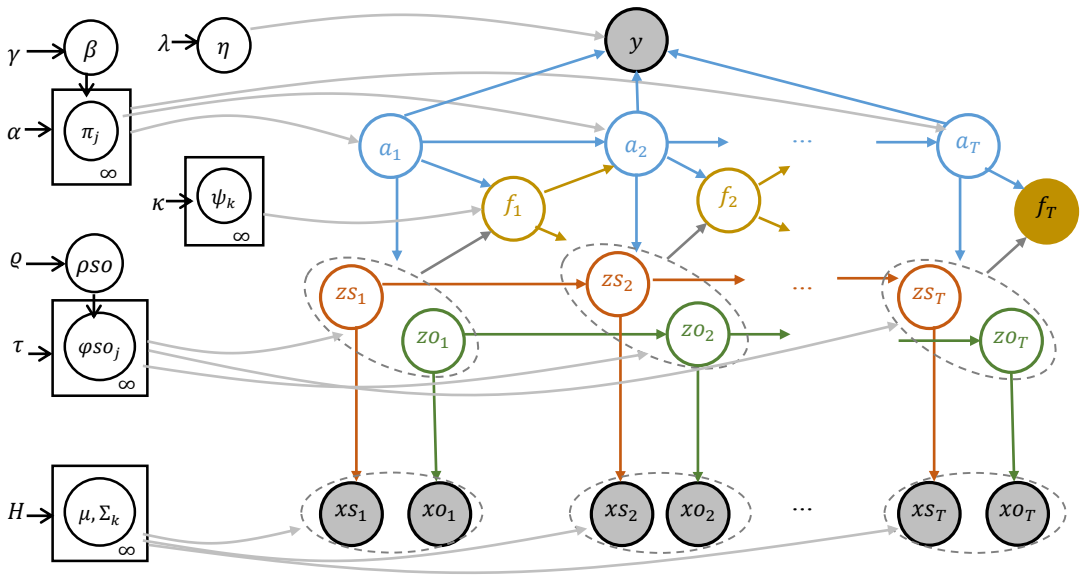

Figure 4: Graphical representation of the activity Model. The top level contains action states $a_{1:T}$ that emit the bottom level skeleton pose $zs_{1:T}$ and object pose $zo_{1:T}$ states which in turn emit the observations $x_{1:T}$. The binary variable f_t determines whether the action state remains unchanged or not. The action states determine the activity label y. The parameters and their priors are in the left side. Best viewed in colour.

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. A summary of the notation is given in Table 1 and Figure 4 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.

$$\begin{aligned} \beta \mid \gamma &\sim GEM(\gamma) \\ \pi_j \mid \alpha, \beta &\sim DP(\alpha, \beta) \qquad j = 1,2,\dots \end{aligned} \tag{5}$$

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.

$$\begin{aligned} \rho s^a \mid \varrho &\sim GEM(\varrho) && a = 1,2,\dots \\ \rho o^a \mid \varrho &\sim GEM(\varrho) && a = 1,2,\dots \\ \varphi s_j^a \mid \tau, a, \rho s^a &\sim DP(\tau, \rho s^a) && j = 1,2,\dots \\ \varphi o_j^a \mid \tau, a, \rho o^a &\sim DP(\tau, \rho o^a) && j = 1,2,\dots \end{aligned} \tag{6}$$

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 μo, Σo respectively.

$$\begin{aligned} \mu s_k \mid \mu_0^s, \Sigma_0^s &\sim \mathcal{N}(\mu_0^s, \Sigma_0^s) && k = 1,2,\dots \\ \Sigma s_k \mid \nu_0^s, \Delta_0^s &\sim IW(\nu_0^s, \Delta_0^s) && k = 1,2,\dots \\ \mu o_k \mid \mu_0^o, \Sigma_0^o &\sim \mathcal{N}(\mu_0^o, \Sigma_0^o) && k = 1,2,\dots \\ \Sigma o_k \mid \nu_0^o, \Delta_0^o &\sim IW(\nu_0^o, \Delta_0^o) && k = 1,2,\dots \end{aligned} \tag{7}$$

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

$$\psi^{a,ks,ko} \mid \kappa a, \kappa b \sim Beta(\kappa a, \kappa b) \qquad a, ks, ko = 1,2,\dots \tag{8}$$

5) Draw the regression coefficients for all the classes from a sparsity promoting Laplacian prior. Let $||\boldsymbol{\eta}||_1$ denote the l_1 norm. The Laplacian prior is given as follows.

$$\boldsymbol{\eta} \mid \lambda \sim \exp(-\lambda ||\boldsymbol{\eta}||_1) \tag{9}$$

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.

$$a_t^n \mid a_{t-1}^n, f_{t-1}^n, \pi \begin{cases} \sim \pi_{a_{t-1}^n} & if\ f_{t-1}^n = 1 \\ = \delta(a_t^n, a_{t-1}^n) & otherwise \end{cases} \tag{10}$$

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.

$$zs_t^n \mid a_t^n, f_{t-1}^n, \varphi s \sim \begin{cases} \varphi s_0^{a_t^n} & if\ f_{t-1}^n = 1 \\ \varphi s_{zs_{t-1}^n}^{a_t^n} & otherwise \end{cases} \tag{11}$$

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.

$$zo_t^n \mid a_t^n, f_{t-1}^n, \varphi o \sim \begin{cases} \varphi o_0^{a_t^n} & if\ f_{t-1}^n = 1 \\ \varphi o_{zo_{t-1}^n}^{a_t^n} & otherwise \end{cases} \tag{12}$$

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

$$f_t^n \mid a_t^n, zs_t^n, zo_t^n\ \psi \sim Ber(\psi^{a_t^n, zs_t^n, zo_t^n}) \tag{13}$$

11) Draw the skeleton and object observations from the normal distribution using the mean and covariance parameters drawn according to (7).

$$\begin{aligned} xs_t^n \mid zs_t^n, \mu s, \Sigma s &\sim \mathcal{N}(\mu s_{zs_t^n}, \Sigma s_{zs_t^n}) \\ xo_t^n \mid zo_t^n, \mu o, \Sigma o &\sim \mathcal{N}(\mu o_{zo_t^n}, \Sigma o_{zo_t^n}) \end{aligned} \tag{14}$$

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 (10) and the regression coefficients are drawn according to (9).

$$y^n \mid\ a_{1..T}^n, \boldsymbol{\eta} \sim Mult\left(\frac{\exp\left(\eta^{1\intercal}\overline{a}\right)}{\sum_{c'=1}^{C}\exp\left(\eta^{c'\intercal}\overline{a}\right)} \ldots \frac{\exp\left(\eta^{C\intercal}\overline{a}\right)}{\sum_{c'=1}^{C}\exp\left(\eta^{c'\intercal}\overline{a}\right)} \right) \tag{15}$$

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.

General	
x^n	The n^{th} training example sequence
y^n	The class of the n^{th} training example
x_t	An observation at time instant t
Action (Top Level)	
a_t	Action state at time instant t
$\overline{a}$	Empirical frequencies of the action states
β_k	Probability of transitioning to action state k
π_{jk}	Probability of transitioning to action state k given state j
Skeleton Pose (Bottom Level)	
xs_t	Skeleton pose observation at time instant t
zs_t	Hidden skeleton pose state at time instant t
ρs_k^a	Probability of transitioning to skeleton pose state k given action a
φs_{jk}^a	Probability of transitioning to skeleton pose state k given state j, action a
μs_k	Mean of Gaussian distribution corresponding to skeleton pose component k
Σs_k	Covariance of Gaussian distribution corresponding to skeleton pose component k
Object Pose (Bottom Level)	
xo_t	Object pose observation at time instant t
zo_t	Hidden object pose state at time instant t
ρo_k^a	Probability of transitioning to object pose state k given action a
φo_{jk}^a	Probability of transitioning to object pose state k given state j, action a
μo_k	Mean of Gaussian distribution corresponding to object pose component k
Σo_k	Covariance of Gaussian distribution corresponding to object pose component k
Finish Variable	
f_t	Binary variable indicating whether a sequence of actions is complete
$\psi^{a,ks,ko}$	Probability that an action a with skeleton state ks and object state ko is completed
Hyper Parameters	
γ	Concentration parameter, Hyper-prior for β
α	Concentration parameter, Hyper-prior for π
ϱ	Concentration parameter, Hyper-prior for $\rho s, \rho o$
τ	Concentration parameter, Hyper-prior for $\varphi s, \varphi o$
$\mu_0^s, \Sigma_0^s, \mu_0^o, \Sigma_0^o$	Hyper-prior for $\mu s, \mu o$
$\nu_0^s, \Delta_0^s, \nu_0^o, \Delta_0^o$	Hyper-prior for $\Sigma s, \Sigma o$
$\kappa a, \kappa b$	Hyper-prior for ψ
λ	Hyper-prior for the regression coefficients
Posterior Inference	
η	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
$\boldsymbol{a}$	Action state sequence from a sampling iteration
$V(.)$	Forward message value
$m(.)$	Backward message value

Table 1: Summary of Notations

5 Posterior Inference

Exact posterior inference is intractable in DP based models. Markov Chain Monte Carlo (MCMC) techniques, particularly Gibbs sampling, allow approximate inference by drawing posterior samples for all the variables. Gibbs sampling 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. 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). Table 2 lists the inference steps.

5.1 Sampling Hidden state sequence

Many DP based models sample one state at a time, given all other state assignments. This method, based on a Rao-Blackwellized Gibbs sampler [4], marginalizes over the unbounded number of state transitions. The observations are temporally coupled in sequential data and using the above procedure often exhibit slow mixing rates. 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.

In order to block sample the hidden state sequence, a weak limit approximation to DP is used [45]. Specifically, in (1), $Dir(\frac{\gamma}{K} \dots \frac{\gamma}{K})$ is used to sample the stick breaking weights. Here K is an upper bound on the number of components. As $K \to \infty$, the model's marginal distribution approaches the DP. The bound K is set to a large number. The HDP prior ensures that only a small subset of the K values is used. This truncated approximation is computationally efficient and allows using existing well-studied Bayesian message passing techniques. Consequent to this approximation, equations (5) and (6) become the following.

$$
\begin{aligned}
&\beta \mid \gamma \sim Dir(\frac{\gamma}{K^a} \dots \frac{\gamma}{K^a}) && \pi_j \mid \alpha, \beta \sim Dir(\alpha\beta_1, \dots, \alpha\beta_{K^a}) && j = 1, \dots, K^a \\
&\rho s^a \mid \varrho \sim Dir(\frac{\varrho}{K^s} \dots \frac{\varrho}{K^s}) && \varphi s_j^a \mid \tau, a, \rho s^a \sim Dir\left(\tau\rho s_1^a, \dots, \tau\rho s_{K^s}^a\right) && j = 1, \dots, K^s \\
&\rho o^a \mid \varrho \sim Dir(\frac{\varrho}{K^o} \dots \frac{\varrho}{K^o}) && \varphi o_j^a \mid \tau, a, \rho o^a \sim Dir\left(\tau\rho o_1^a, \dots, \tau\rho o_{K^o}^a\right) && j = 1, \dots, K^o
\end{aligned}
\tag{16}
$$

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 is provided as follows.

$$
\begin{aligned}
&p(xs_{1:T}, xo_{1:T}, y, a_{1:T}, zs_{1:T}, zo_{1:T}, f_{1:T} \mid \pi, \varphi s, \varphi o, \mu s, \Sigma s, \mu o, \Sigma o, \psi, \boldsymbol{\eta}) \\
&= p(y|a_{1:T}) \prod_{t=1}^{T} p(a_t|a_{t-1}, f_{t-1})\ p(zs_t|zs_{t-1}, a_t, f_{t-1})\ p(zo_t|zo_{t-1}, a_t, f_{t-1}) \\
&\qquad p(f_t|a_t, zs_t, zo_t)\ p(xs_t|zs_t)\ p(xo_t|zo_t)
\end{aligned}
\tag{17}
$$

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 [2] 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 are bounded here by the use of a truncated DP. The flattening and the weak limit approximation in (16), allows the use of variations of the forward-backward algorithm [2, 5] 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. The validity of this approximation is observed empirically in the experiments. Let $\boldsymbol{a}$ be a sampled action state sequence corresponding to an observation sequence in a previous sampling iteration. Let $V(.)$ define the forward message value for being in a specific combination of states (action: k^a, skeleton: k^s, object: k^o and finish:f') given the previous states as $k^{a'}, k^{s'}, k^{o'}, f''$. Then, this message value can be derived by substituting terms from equations (10) to (15) into (17).

$$V\left(k^{a'}, k^{s'}, k^{o'}, f'', k^a, k^s, k^o, f' \mid xs_t, xo_t, y, \boldsymbol{\eta}, \boldsymbol{a}\right) =$$
$$\left[\left[\delta(f'', 1)\, \pi_{k^{a'}, k^a}\ \varphi s_{0,k^s}^{k^a}\ \varphi o_{0,k^o}^{k^a}\right] + \left[\delta(f'', 0)\, \delta\left(k^a, k^{a'}\right) \varphi s_{k^{s'}, k^s}^{k^a}\, \varphi o_{k^{o'}, k^o}^{k^a}\right]\right] \tag{18}$$
$$Ber\left(f'; \psi^{k^a, k^s, k^o}\right) \mathcal{N}(xs_t;\ \mu s_{k^s}, \Sigma s_{k^s})\ \mathcal{N}(xo_t;\ \mu o_{k^o}, \Sigma o_{k^o})\ \frac{\exp\left(\eta^{y^\intercal} \overline{a}\right)}{\sum_{c'=1}^{C} \exp\left(\eta^{c'^\intercal} \overline{a}\right)}$$

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 $\boldsymbol{a}$. 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 message value, the hidden state at time t can then be sampled from the conditional distribution below.

$$p(a_t = k^a, zs_t = k^s, zo_t = k^o, f_t = f' \mid xs_t, xo_t, y, \boldsymbol{\eta}, \boldsymbol{a}) \propto$$
$$m_{t+1,t}(k^a, k^s, k^o, f')\, V(a_{t-1}, zs_{t-1}, zo_{t-1}, f_{t-1}, k^a, k^s, k^o, f' \mid xs_t, xo_t, y, \boldsymbol{\eta}, \boldsymbol{a}^{-t} + k^a) \tag{19}$$

Here $m_{t,t-1}$ is the backward 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 and is determined recursively over all possible states as follows.

$$m_{t,t-1}(k^a, k^s, k^o, f') =$$
$$\sum_{k^{a'} \in K^a} \sum_{k^{s'} \in K^s} \sum_{k^{o'} \in K^o} \sum_{f'' \in \{0,1\}} m_{t+1,t}\left(k^{a'}, k^{s'}, k^{o'}, f''\right) V(k^a, k^s, k^o, f', k^{a'}, k^{s'}, k^{o'}, f'') \tag{20}$$

5.2 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 [4] 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 [34] 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.

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 $\mathcal{X}S_{k^s}$. 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 (18)) 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 $\boldsymbol{\eta}$. Because of the normalization condition $\sum_c \exp(\eta^{c\top}\overline{a}) = 1$, the parameters need to be learnt only for $c-1$ classes. Using a Laplacian prior for $\boldsymbol{\eta}$ rather than a normal prior on the coefficients encourages coefficient values which are relatively large or near to zero. The sparseness of this prior leads to structural simplification and often results in generalization and better classification results [46]. The sparse multinomial logistic regression [6] 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. We use this procedure here for estimating the regression weights .

Given a test activity sequence during prediction, the action state sequence is inferred using Viterbi decoding from a posterior sample's parameters. The conditional likelihood term in (17) is not used in this process. The linear predictor is determined from the regression coefficients η and $\overline{a}$, with $\overline{a}$ computed from the inferred action state sequence. The label corresponding to the posterior sample is predicted using (21) and the final label is selected based on the mode.

$$\hat{c} = \underset{c}{\operatorname{argmax}} \frac{\exp(\eta^{c\top}\overline{a})}{\sum_{c'=1}^{C}\exp(\eta^{c'\top}\overline{a})} \tag{21}$$

Input: Training examples of activity sequences with their corresponding labels, Hyper parameters **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 (17) to (20). 3. Compute the count matrices from the sampled hidden state sequences as in section 5.2.

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.

Table 2: Posterior Inference Algorithm

6 Experiments

The activity model is evaluated against the Cornell Activity dataset [47] and the UTKinect-Action dataset [13]. These datasets based on depth images, contain various activities performed by different subjects. However the activities involve only one individual. The depth image sequences were recorded using a single Microsoft Kinect sensor. The datasets also contain annotated 3D joint positions of the human skeleton that were estimated from the depth images as explained in [1]. The estimated joint positions may have errors and we work with these noisy inputs. In addition, the model is also evaluated against a motion capture based dataset [48]. 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.

6.1 Cornell Activity Dataset

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. They were recorded in different locations viz. *office*, *kitchen*, *bedroom*, *bathroom* and *living room* of a regular household. Some examples are shown in Figure 5.

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). The projected vector is used as the skeleton features.

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) [49] 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 [50]. The descriptors obtained in this way from all the joints are concatenated and the resultant 2400 dimensional vector is projected using PCA to obtain the object based features.

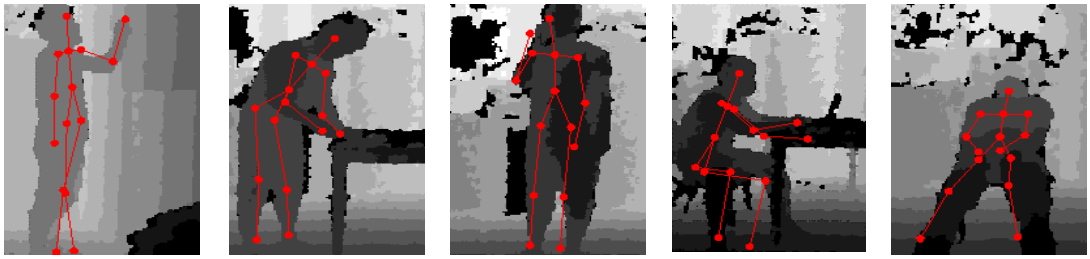

Figure 5: Sample activity images from Cornell Activity dataset [47]

To evaluate against this dataset, an experimental setup similar to [47] is followed. The activities are grouped based on their locations and classification is performed in each location group. This is done both for the setting where an instance of the subject has been seen (S-Seen) and when the subject is new (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. Each activity sequence spans about 45 seconds and on average the dataset provides 1400 frames of data per activity. In total, we work with 96 different sequences during the evaluation process. The sensor produces a depth image with a range of 1.2m to 3.5m and the image resolution in the dataset is 320x240.

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 [4], 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. K^a, the maximum number 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 we tried large values for K^a, K^s and K^o, we found that the value of $K^a = 20$ 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 0.8 times the empirical covariance respectively of the skeleton features. The hyper parameter v_0^s was given a large value (1000), which concentrates the mass of the prior in regions based on the data [5]. 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. PCA is applied to reduce dimensions for both the 315 dimensional skeleton feature vector and the 2400 dimensional object feature vector described above. The feature vectors are collected across the observation sequences for each of the skeleton and object features before applying PCA. The first d^s components of the skeleton features that capture at least 90% of the total variance are used and the first d^o components of the object features that capture at least 85% of the total variance are used. This amounted to 13 components for the skeleton features and 227 components for the object features. Increasing these thresholds did not improve the classification accuracy and reducing them by more than 10% resulted in a number of misclassifications.

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 [4], 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, we checked the change in the number of activated states and the difference in Gaussian distribution parameter values between iterations. 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 out-of-sample embedding is computed for a test sequence's skeleton and object features based on the components extracted from training examples. For each sample collected during training, based on the sample parameters, Viterbi decoding 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. To account for the randomness with MCMC sampling, the experiments were run more than 20 times and the mean performance is reported.

The block sampling procedure discussed in section 5.1 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 [5]. The block sampling procedure needs $O(TK)$ space in order to store the messages. On a 2.6GHz Intel Core i5 CPU machine, the average time taken to label a new test sequence with an appropriate class label from the information learnt during training, including the time for feature processing, was around 79 seconds giving an approximate frame rate of 17 fps for the sequences in this dataset. The test procedure can be extended by using a sliding window of frames to support online recognition. However, the focus of our work here has been on offline recognition and these settings were not evaluated.

Hierarchical Structure: The hierarchical nature of the model is examined. Figure 6 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 accurately define 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 7. 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.

During our evaluation, we 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 offers the advantage of estimating the state cardinality automatically based on the data.

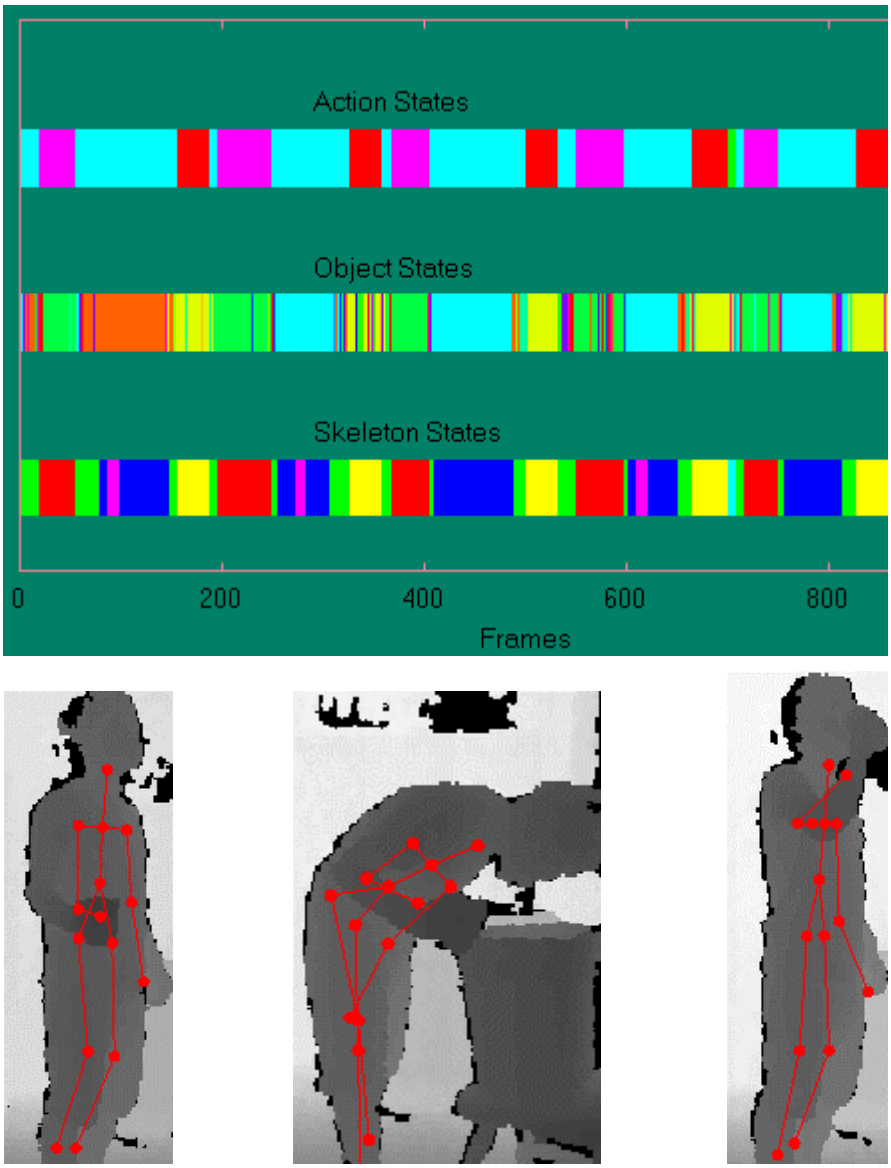

Figure 6: Learned hierarchical structure for the *rinsing mouth* activity. *Top:* The sampled skeleton, object and action states corresponding to an observation sequence are shown in a color coded format. The number of top level action states is fewer than the number of skeleton and object state combinations. 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.

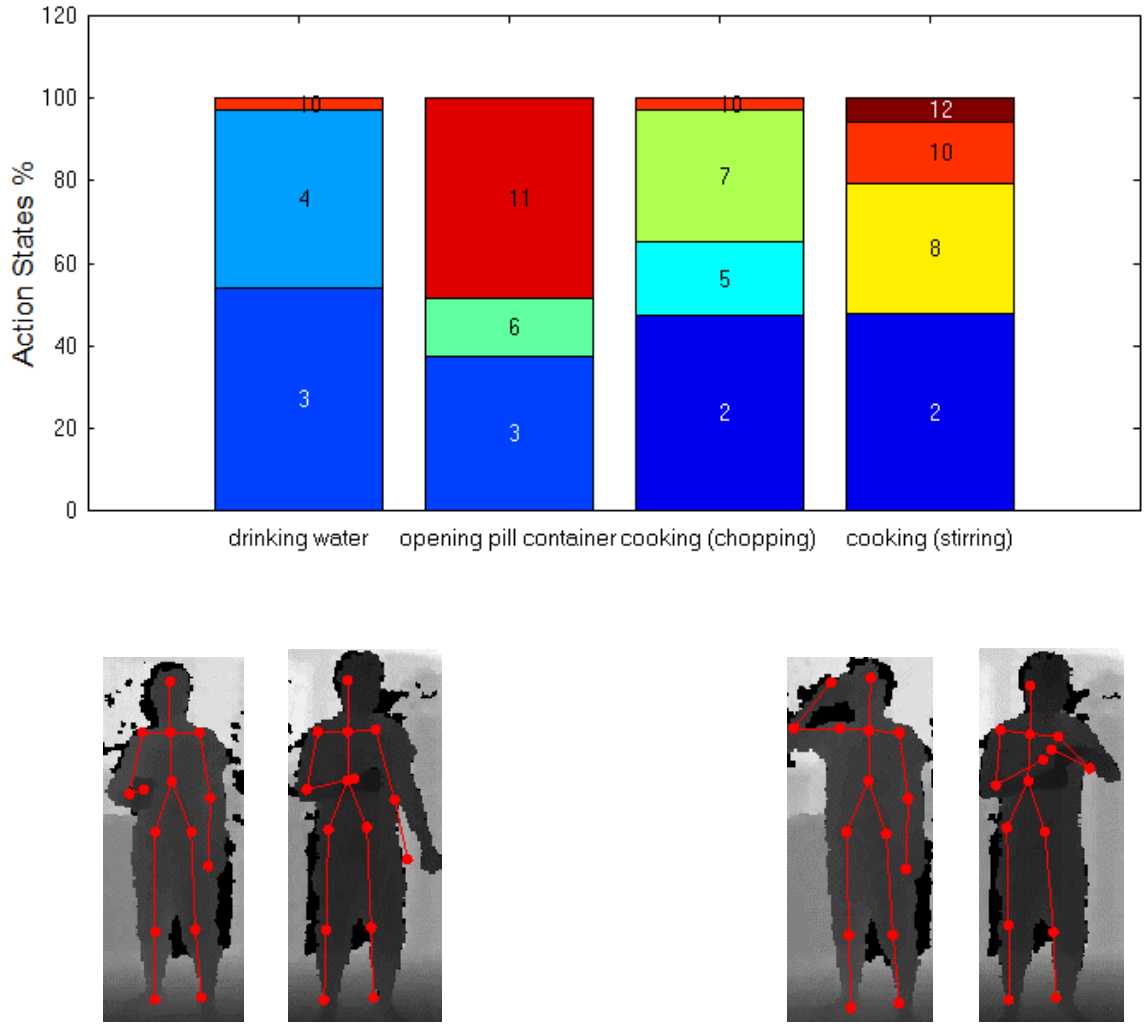

Figure 7: Action states for the activities involved in the *kitchen* location (*top*). There are 10 instantiated action states (2, 3, 4, 5, 6, 7, 8, 10, 11 and 12) for four activities in this sampled action state sequence. The stacked bars indicate the percentage of observations that were assigned to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

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 label. When training an HMM, the standard forward-filtering backward-sampling procedure is used to sample the state sequences. We collected 100 posterior samples for each HMM. During prediction, a test sequence's observation likelihood is computed against all the posterior samples and the mean likelihood is used. This is repeated for all the HMMs and the activity label is then selected based on the class conditional likelihood. Note that in a parametric HMM, the number of states must be specified in advance. Hence we try different numbers of states for each class. The results are averaged over all the location groups and shown in Table 3. The observed classification accuracy for the S-Seen setting was ***62.5***% and for S-New setting was ***47.7***%.

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

Table 3: Cornell activity dataset - Classical Parametric HMM classification accuracy

Parametric H-HMM: A classifier based on a two level parametric Hierarchical HMM [44] 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 4. The observed classification accuracy for the S-Seen setting was ***68.9***% and for S-New setting was ***56.4***%.

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 4: Cornell activity dataset - Parametric H-HMM classification accuracy

Unshared Parameters: The non-parametric H-HMM model in Section 4 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 remain factorized 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 and an average of ***82.1***% (±0.8%) and ***69.56***% (±1.3%) was observed for the S-Seen and S-New setting respectively.

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

Table 5: Cornell activity dataset - Classification accuracy (in %) for the model with regression excluded

Full Model: Finally, the results are evaluated against the full non-parametric 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. 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%*** with a standard deviation of 2.14% was observed. The results for the new subject setting is provided in Table 6.

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

Table 6: Cornell activity dataset - Classification accuracy (in %) for the full model.

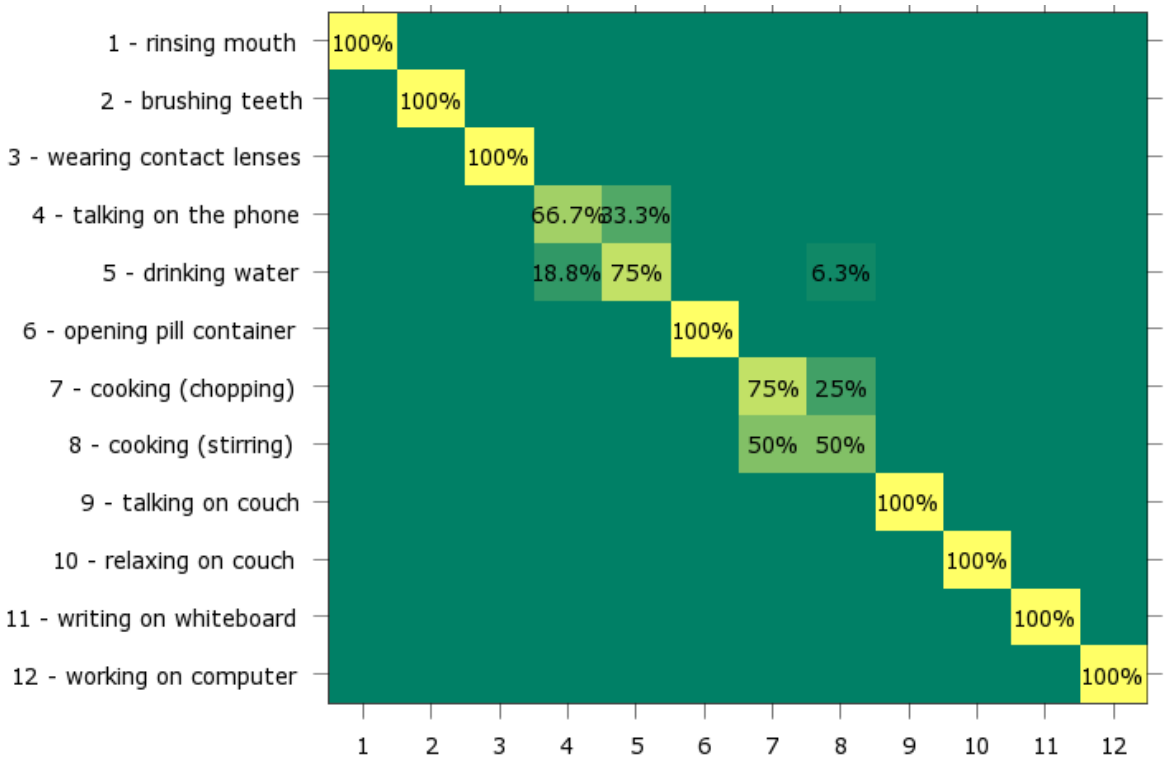

Figure 8: Cornell activity dataset - Confusion matrix for the full model with the S-New setting

	Method	S-Seen Accuracy (%)	S-New Accuracy (%)
Previous Works	STIP [16]	N/A	62.5
	MEMM [47]	84.3	64.2
	Actionlet [11]	94.1	74.7
	Heterogeneous Features [12]	N/A	84.1
	Action Templates [15]	100	91.9
Our Results	Parametric HMM	62.5	47.7
	Parametric H-HMM	68.9	56.4
	Unshared Parameters	82.1	69.5
	Full Model	100	85.4

Table 7: Cornell activity dataset - Comparisons.

Discussion: The confusion matrix in Figure 8 provides an overview of the classification performance for each activity label. The classifier labels are accurate for most of the classes. The mislabelling occurs for activities that involve very similar movements. For example, the two cooking activities are mislabelled in some instances. A comparison of all the results is provided in Table 7. The under-performance of the simple parametric HMM and H-HMM is expected. The usefulness of the non-parametric prior and the factorized structure is evident from the significantly improved accuracy when compared with the outcomes of its parametric counter-part. The full model evaluation that includes regression for classification with parameter sharing, outperforms the rest. When compared with previous works in the literature, our approach is equivalent to the state-of-the-art for the setting where a subject has been seen. For the new subject setting, our approach outperforms all the other existing approaches except the action templates [15]. The work in [15] 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, our focus 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.

6.2 UTKinect-Action Dataset

This dataset contains 10 actions - *walk, sit-down, stand-up, pick-up, carry, throw, push, pull, wave* and *clap-hands*. Each action was performed indoors by ten different subjects 9 males and 1 female, and were repeated twice. The actions have significant intra-class variations and were recorded from different views.

There are 20 3D joint positions with coordinates (x, y, z) in a world coordinate frame. All 190 combinations of the joint pairs are used to compute a 570 dimensional vector which is projected using PCA as described in section 6.1. Only skeleton based features are used for this dataset and this indirectly evaluates a variation in the model.

The experiments are conducted for the case in which the subject has been seen before and for subjects seen for the first time. For the former, a similar experimental setup to [13] is followed with a leave-one-out cross validation scheme. For the latter, 60% of subjects were used for training and the rest for testing. The dataset contains 200 action sequences and a total of 6220 frames with an average frame length of 32 per sequence. The depth image resolution is 320x240 and the depth range is 4 to 11 feet. The model parameters are initialized as discussed in section 6.1 and as before 100 samples are collected. The object related parameters are dropped and we used $K^a = 25$ and $K^s = 25$. The average time taken to label a test sequence was 550ms in this dataset.

As for the Cornell dataset, the results are evaluated for the classical HMM, parametric hierarchical HMM, unshared parameters and the full model. The classification accuracy for the setting where an instance of the subject has been seen is ***87.9%*** with a standard deviation of 1.01%. For the more

challenging new subject setting, an accuracy of ***84.0%*** with a standard deviation of 2.42% was observed.

	Method	S-Seen Accuracy (%)	S-New Accuracy (%)
Previous Works	HOJ3D [13]	90.9	N/A
	Shape Analysis [25]	91.5	N/A
	STIP [16]	N/A	81.0
	Discriminative HDP-HMM [40]	86.8	83.1
Our Results	Parametric HMM	69.7	60.5
	Parametric H-HMM	71.6	64.2
	Unshared Parameters	79.8	77.3
	Full Model	87.9	84.0

Table 8: UTKinect-Action dataset - Comparisons.

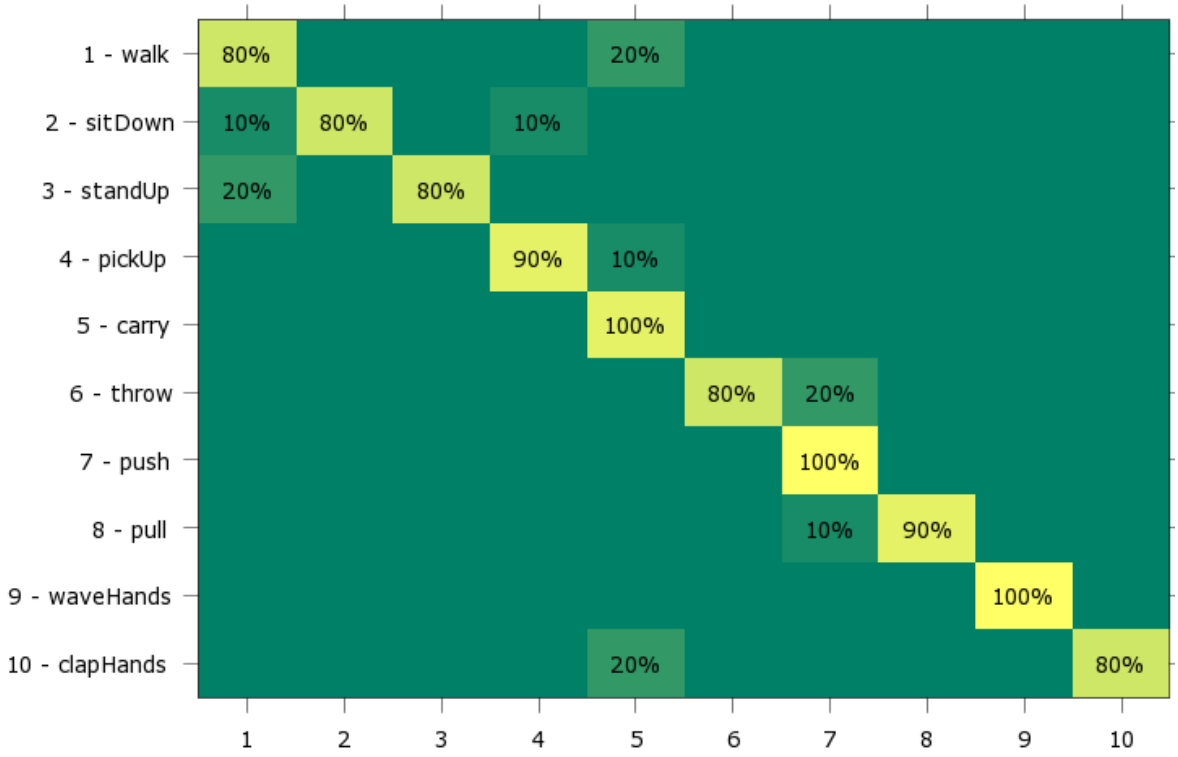

Figure 9: UTKinect-Action dataset - Confusion matrix for the full model with the S-Seen setting

Unlabelled Examples: In order to evaluate the semi-supervised learning setting, 20% of the subjects from all the actions were treated as unlabelled examples. The model was trained with only 40% of the subjects and prediction was done for the S-New setting with the rest of the subjects as before. The observed mean classification accuracy was 80.1%. The unlabelled examples were then used as part of the training set. When sampling the action state sequence for the unlabelled examples, the conditional likelihood term in (17) is not used since these examples do not have a corresponding label. The rest of the sampling procedure was similar as before. During prediction, the mean classification accuracy improved to 80.9%. Following a similar procedure for the S-Seen setting, 20% of the subjects were treated as unlabelled examples. In this case, including the unlabelled examples improved the mean accuracy from 82.5% to 85.6%. The standard deviations in the above results were all under 2.5%. To summarize, the inclusion of unlabelled examples improves the accuracy in both cases with a

marginal increase for the S-New setting and an increase of about 3% for the S-Seen setting. The classifier typically performs better when a subject's actions have been seen during training. Hence including the unlabelled examples, which would have otherwise been excluded in a fully supervised model, offers a distinctive advantage particularly in the S-Seen setting. The semi-supervised learning scheme is especially useful in real world scenarios where there may only be a few labelled examples, but many unlabelled examples available for training.

Discussion: The confusion matrix in Figure 9 provides an overview of the classification performance for each action label. As with Cornell dataset, the classifier labels are accurate for most of the classes and incorrect for some actions that involve very similar movements. For example, the *walk* action is incorrectly classified as *carry* since the body motions are very similar for these two actions. A comparison of all the results is provided in Table 8. As in the Cornell dataset, the non-parametric prior evidently outperforms the parametric variants and the full model evaluation that includes regression and parameter sharing, outperforms the rest. For the setting where the subject has been seen previously, our classification accuracy is marginally less than the results reported in [13] and [25]. Both these approaches use highly processed representations of the 3D joint positions while we focus on a generic approach. For the challenging setting where the subject has not been seen before, our method is better than the others. This shows that the classifier adapts to new observations much better than the competing classifiers.

6.3 Motion Capture Dataset

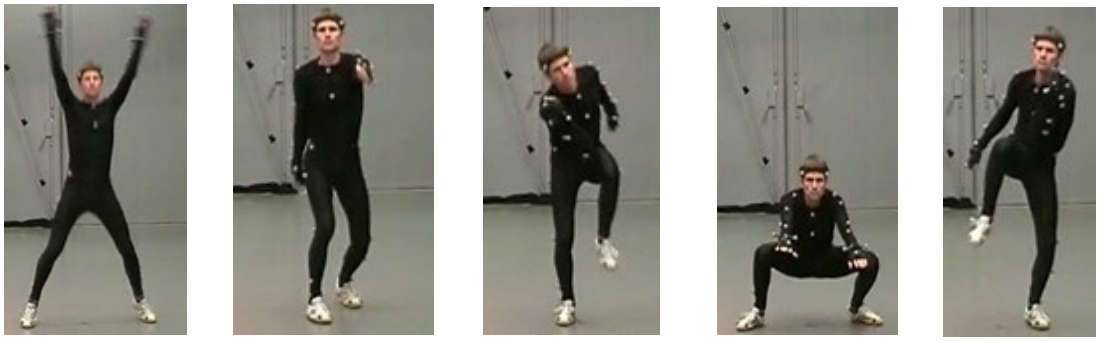

Figure 10: Samples poses from the *workout* activity in HDM05 dataset [48]

Method	Accuracy (%)
Parametric HMM	70.9
Parametric H-HMM	74.2
Unshared Parameters	85.0
Full Model	92.5

Table 9: HDM05 dataset - Results.

Finally, in order to verify the applicability of our generic approach, experiments are conducted on a motion capture dataset that does not involve depth images. The HDM05 [48] dataset contains joint trajectories captured by an optical marker based Vicon system. It contains different subjects performing a number of motion sequences that can be grouped into logical activities. The activities *badminton*, *dancing*, *grabbing/depositing* and *workout* are used to evaluate our 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 10 shows some example poses. We evaluated against 48 different activity sequences with each sequence containing an average of 4700 frames and lasting typically for 30 seconds. The unformatted C3D format data in which bone lengths are not

normalized is used rather than the AMC format data. The object features are excluded and only the skeleton features are used. The model parameters are initialized as discussed in section 6.1, with $K^a = 25$ and $K^s = 25$. It took an average of 57.4 seconds to label a test sequence with an appropriate class label using the information learnt during training. A high classification accuracy of ***92.5%*** (± 2.6%) is obtained with the full model. Table 9 provides the results.

The above experiments demonstrate the merits of using the proposed activity model. They show that the integration of a hierarchical Markov structure with regression is an effective mechanism for classifying activities. The model does misclassify some examples that have very similar motion patterns. A relevant example is the *sit down* and *pick up* actions in UTKinect-Action dataset. Our model may benefit if we detect the objects that a subject interacts with and include this contextual information. The model does not explicitly handle cases such as an activity varying significantly from one subject to another, occlusions of poses or poses out of view. These cases can be addressed by improving the feature extraction procedure and introducing new variables at the cost of increased complexity.

7 Conclusion

This paper has proposed a non-parametric hierarchical HMM structure to model activities with actions at the top level and factorized skeleton and object poses at the bottom level. The non-parametric extension enables the automatic inference of the number of states from data and the hierarchical structure facilitates information sharing and semi-supervised learning. The activity labels are regressed on the action states in order to perform classification. This results in model simplification. Experimental results comparable to the state-of-the-art on multiple datasets are produced thus highlighting the efficacy of this approach. In future, we intend to extend the model for classifying long running activities involving multiple subjects.

References

[1] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. "Real-time human pose recognition in parts from single depth images". *CVPR* (2011).

[2] Rabiner, L., & Juang, B. H. "An introduction to hidden Markov models". *ASSP Magazine*, IEEE 3, 4-16 (1986).

[3] Fine, S., Singer, Y., & Tishby, N. "The hierarchical hidden Markov model: Analysis and applications". *Machine learning*, 32(1), 41-62 (1998).

[4] Teh, Yee Whye, Michael I. Jordan, Matthew J. Beal, and David M. Blei, N. "Hierarchical dirichlet processes". *Journal of the American Statistical Association* 101.476 (2006).

[5] Fox, Emily B., Erik B. Sudderth, Michael I. Jordan, and Alan S. Willsky. "An HDP-HMM for systems with state persistence". *Proceedings of the 25th international conference on Machine learning*. ACM (2008).

[6] Krishnapuram, B., Carin, L., Figueiredo, M. A., & Hartemink, A. J. "Sparse multinomial logistic regression: Fast algorithms and generalization bounds". *Pattern Analysis and Machine Intelligence,* IEEE 27(6), 957-968 (2005).

[7] Aggarwal, J. K., and Michael S. Ryoo. "Human activity analysis: A review". *ACM Computing Surveys (CSUR)* 43.3: 16 (2011).

[8] Han, Jungong, Ling Shao, Dong Xu, and Jamie Shotton. "Enhanced Computer Vision with Microsoft Kinect Sensor: A Review". *IEEE Transactions on Cybernetics* (2013).

[9] Aggarwal, J. K., & Xia, L. "Human activity recognition from 3d data: A review". *Pattern Recognition Letters*, 48, 70-80 (2014).

[10] Ye, M., Zhang, Q., Wang, L., Zhu, J., Yang, R., & Gall, J. “A survey on human motion analysis from depth data”. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications* pp. 149-187, (2013).

[11] Wang, J., Liu, Z., Wu, Y., & Yuan, J. “Learning Actionlet Ensemble for 3D Human Action Recognition”. *Pattern Analysis and Machine Intelligence* (2014).

[12] Hu, J. F., Zheng, W. S., Lai, J., & Zhang, J. “Jointly Learning Heterogeneous Features for RGB-D Activity Recognition”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. (2015).

[13] Xia, L., Chen, C. C., & Aggarwal, J. K. “View invariant human action recognition using histograms of 3d joints”. *CVPRW* (2012).

[14] Gowayyed, M. A., Torki, M., Hussein, M. E., & El-Saban, M. “Histogram of oriented displacements (HOD): describing trajectories of human joints for action recognition”. *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press (2013).

[15] Shan, J. & Akella, S. “3D Human Action Segmentation and Recognition using Pose Kinetic Energy”. In *IEEE Workshop on Advanced Robotics and its Social Impacts* (2014).

[16] Y. Zhu, W. Chen, and G. Guo. "Evaluating spatiotemporal interest point features for depth-based action recognition." *Image and Vision computing* (2014).

[17] H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian, "HOPC: Histogram of Oriented Principal Components of 3D Pointclouds for Action Recognition," *ECCV* (2014).

[18] Zeiler, M. D., & Fergus, R. “Visualizing and understanding convolutional networks.” *ECCV* (2014).

[19] Krizhevsky, A., Sutskever, I., and Hinton, G.E. “Imagenet classification with deep convolutional neural networks.” *NIPS* (2012)

[20] S. Ji, W. Xu, M. Yang, and K. Yu. “3D convolutional neural networks for human action recognition.” *PAMI* (2013)

[21] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. “Large-scale video classification with convolutional neural networks.” CVPR (2014)

[22] Simonyan, K., & Zisserman, A. “Two-stream convolutional networks for action recognition in videos”. NIPS (2014).

[23] Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., & Darrell, T. “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description.” CVPR (2015).

[24] N. J. Yue-Hei, H. Matthew, V. Sudheendra, V. Oriol, M. Rajat, and T. George. “Beyond short snippets: Deep networks for video classification.” CVPR (2015).

[25] M. Devanne, H. Wannous, S. Berretti, P. Pala, M. Daoudi, and A. Del Bimbo, "3D human action recognition by shape analysis of motion trajectories on Riemannian manifold," IEEE Trans. on Cybernetics (2014).

[26] Anirudh, R., Turaga, P., Su, J., & Srivastava, A. “Elastic Functional Coding of Human Actions: From Vector-Fields to Latent Variables.” CVPR (2015).

[27] Yu, T. H., Kim, T. K., & Cipolla, R. “Real-time Action Recognition by Spatiotemporal Semantic and Structural Forests.” BMVC (2010).

[28] Shi, F., Petriu, E., & Laganiere, R. “Sampling strategies for real-time action recognition.” CVPR (2013).

[29] Kantorov, V., & Laptev, I. “Efficient feature extraction, encoding, and classification for action recognition.” CVPR (2014).

[30] H. Koppula and A. Saxena, "Learning spatio-temporal structure from RGB-D videos for human activity detection and anticipation," ICML (2013).

[31] Lan, T., Wang, Y., Yang, W., Robinovitch, S.N., Mori, G.: Discriminative latent models for recognizing contextual group activities. TPAMI (2012)

[32] Amer, M. R., Lei, P., & Todorovic, S. “Hirf: Hierarchical random field for collective activity recognition in videos.” ECCV (2014).
[33] Heller, K. A., Teh, Y. W., & Görür, D. “Infinite hierarchical hidden Markov models”. International Conference on Artificial Intelligence and Statistics (pp. 224-231) (2009).
[34] Johnson, M. J., & Willsky, A. S. “Bayesian nonparametric hidden semi-markov models”. Journal of Machine Learning Research, 14(1), 673-701 (2013).
[35] Di Lello, E., De Laet, T., & Bruyninckx, H. "Hierarchical dirichlet process hidden markov models for abnormality detection in robotic assembly." NIPS (2012).
[36] Sudderth, E. B., Torralba, A., Freeman, W. T., & Willsky, A. S. “Describing visual scenes using transformed objects and parts”. IJCV (2008).
[37] Bargi, Ava, R. Y. D. Xu, and Massimo Piccardi. “An Infinite Adaptive Online Learning Model for Segmentation and Classification of Streaming Data”. Pattern Recognition (2014).
[38] Hu, D. H., Zhang, X. X., Yin, J., Zheng, V. W., & Yang, Q. “Abnormal Activity Recognition Based on HDP-HMM Models”. IJCAI (2009).
[39] Kooij, Julian FP, Gwenn Englebienne, and Dariu M. Gavrila. “A non-parametric hierarchical model to discover behavior dynamics from tracks”. ECCV (2012).
[40] Raman, N., & Maybank, S. J. “Action classification using a discriminative multilevel HDP-HMM”. Neurocomputing (2015).
[41] Mcauliffe, J. D., & Blei, D. M. “Supervised topic models”. Advances in neural information processing systems pp. 121-128 (2008).
[42] Dai, A., & Storkey, A. “The supervised hierarchical Dirichlet process”. Pattern Analysis and Machine Intelligence, IEEE 37(2), 243-255 (2014).
[43] Y.Grinberg, T.J.Perkins. “State Sequence Analysis in Hidden Markov Models.” 31st Conference on Uncertainty in Artificial Intelligence (2015).
[44] Murphy, K. P., & Paskin, M. A. "Linear-time inference in hierarchical HMMs." Advances in neural information processing systems (2002).
[45] Ishwaran, H., & Zarepour, M. "Exact and Approximate Sum Representations for the Dirichlet Process." Canadian Journal of Statistics (2002).
[46] Figueiredo, M. A. "Adaptive sparseness for supervised learning." Pattern Analysis and Machine Intelligence (2003).
[47] Sung, J., Ponce, C., Selman, B., & Saxena, A. “Human Activity Detection from RGBD Images”. In AAAI workshop on Pattern, Activity and Intent Recognition (2011).
[48] Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B., & Weber, A, "Documentation mocap database hdm05". (2007)
[49] N. Dalal and B. Triggs. “Histograms of Oriented Gradients for Human Detection”. CVPR (2005).
[50] Song, S., & Xiao, J. “Tracking revisited using rgbd camera: Unified benchmark and baselines”. In IEEE International Conference on Computer Vision (2013).