Hierarchical Multi-channel Hidden Semi Markov Graphical Models for Activity Recognition

Pradeep Natarajan

Speech, Language and Multimedia Business Unit,
Raytheon BBN Technologies,
Cambridge, MA 02138
pradeepn@bbn.com

Ramakant Nevatia

Institute for Robotics and Intelligent Systems,
University of Southern California,
Los Angeles, CA 90089-0273
nevatia@usc.edu

Abstract

Recognizing human actions from a stream of unsegmented sensory observations is important for a number of applications such as surveillance and human-computer interaction. A wide range of graphical models have been proposed for these tasks, and are typically extensions of the generative hidden Markov models (HMM) or their discriminative counterpart, conditional random fields (CRF). These extensions typically address one of three key limitations in the basic HMM/CRF formalism - unrealistic models for the duration of a sub-event, not encoding interactions among multiple agents directly and not modeling the inherent hierarchical organization of activities. In our work, we present a family of graphical models that generalize such extensions and simultaneously model event duration, multi agent interactions and hierarchical structure. We also present general algorithms for efficient learning and inference in such models based on local variational approximations. We demonstrate the effectiveness of our framework by developing graphical models for applications in automatic sign language (ASL) recognition, and for gesture and action recognition in videos. Our methods show results comparable to state-of-the-art in the datasets we consider, while requiring far fewer training examples compared to low-level feature based methods.

Keywords: Hierarchical Graphical Models, Action Recognition, Duration
1. Introduction

Automated systems for modeling and recognizing daily activities can have a wide range of applications in surveillance, assistive technologies and intelligent environments. To this end, a significant amount of research has been devoted to represent, annotate and recognize the activities performed by a user. There are two key challenges faced by researchers in this area - 1) Developing robust low-level features for effectively capturing information from images and other sensory data 2) Developing appropriate models for bridging the gap between low-level features and high-level concepts while modeling errors and uncertainty. While several features have been proposed for the first challenge, our focus here is on addressing the second challenge using probabilistic graphical models.

The graphical models considered in literature are either generative and model the joint probability \( P(X, Y) \) between the observations \( X \) and hidden variables \( Y \), or discriminative and model the conditional probability \( P(Y|X) \). Generative models typically generalize hidden Markov models (HMM)[1] and are especially useful in situations where we only have a small amount of unlabeled or partially labeled training data. Discriminative models on the other hand generalize conditional random fields (CRF)[2] and have shown strong performance in action recognition when large amounts of annotated training data is available [3][4]. HMMs and CRFs have been widely used in action recognition and other sequential data analysis applications due to their simplicity and well understood algorithms for learning and inference. Despite their success, the basic HMM/CRF representation suffers from three key limitations:

- The first-order Markov assumption causes the probability of staying in a state to decrease exponentially with time, which is unrealistic (e.g. a person can keep walking for an arbitrary amount of time).

- While activities occur at different levels of abstraction, there is no direct way to model this in an HMM/CRF.

- Use of single variable state representation makes it difficult to model activities involving multiple interacting agents.

A simple way to apply HMMs for domains that require modeling of hierarchy, duration or multi-channel interactions is to expand the state space to account for
these variations. Then, we can apply the well understood Viterbi and Baum-Welch algorithms [1] for inference and parameter learning respectively. While this preserves the simplicity of the basic HMM/CRF framework, it will cause the size of the state space to grow exponentially, which in turn will make inference intractable since the Viterbi inference algorithm is quadratic with respect to the state space size. Further, this will also exponentially increase the number of parameters to learn which in turn increases the amount of training data needed to avoid over fitting. For example, if we have $C$ agents and each agent can be in one of $N$ states, modeling it with a simple HMM would require $N^C$ states. This implies that the inference and learning algorithms will have $O(TN^{2C})$ complexity where $T$ is the length of the observation sequence. This model would also have $O(N^{2C})$ parameters that requires a much larger set of training examples. Several extensions have been proposed in literature to address some of these challenges [5][6][7][4], but only limited work has been done to address them simultaneously [8][9]. Further, exact inference and learning in such models remain intractable, which limits their applicability.

In our work, we introduce a family of graphical models called hierarchical multi-channel semi-Markov graphical models (HM-HSGM) that simultaneously address these limitations. This allows mapping of logic based event definitions to probabilistic graphical models. We consider several hierarchical, multi-channel architectures for modeling multiagent actions and duration modeling using the semi-Markov model [1] as well as a variable transition model [10]. We also present novel and efficient learning and inference algorithms based on local variational approximations [11]. Together, this generalizes the models and algorithms developed for directed and undirected graphical models in previous literature. We demonstrate our approach for recognition of gestures in American sign language (ASL) and actions involving extensive articulation of the human pose. In both these domains, multiple agents (two hands for ASL, different limbs for pose articulation) perform a coordinated series of primitive actions to complete a composite action. Further each primitive gesture/action has a characteristic duration. Thus all three limitations of HMMs need to be addressed simultaneously in these domains.

For the ASL task, we consider three directed graphical models, namely CHSMM, HS-PaHMM and HPaHSMM that are special cases of HM-HSGM (section 6). Our approach produces $\approx 55\%$ improvement over PaHMM [12] and CHMM [5], which do not model hierarchical structure or event duration. For action recognition, we consider a novel directed graphical model called HVT-HMM (section 7). We show state-of-the-art performance, using models trained on only 1-2 examples on the standard Weizmann dataset [13], while running at real time speeds.
The rest of the paper is organized as follows - in section 2 we review related work in activity recognition and graphical models, in section 3 we formalize the notations for our graphical model framework, in section 4 we present efficient inference algorithms, in section 5 we present learning algorithms, in section 6 we demonstrate our approach for continuous sign language recognition, and in section 7 we present results for gesture and action recognition. Conclusions and future directions are given in section 8.

2. Related Work

Activity recognition has been an active subject of research in many communities. Existing work in this area can be broadly classified into two categories - in the first, the focus is on extracting low-level features from sensory data, particularly from videos which can then be used to discriminate among actions of interest. In the second, the focus is on efficiently representing and learning high-level constraints which can then be mapped to low-level features. Several low-level features have been proposed based on foreground blobs [14], optical flow [15], edge [16], shape+flow [17], spatio-temporal interest points [18] and dense oriented histograms [19]. Our focus in this paper is primarily on the second challenge of high-level modeling, and we will review these methods in greater detail.

Bayesian networks (BN) were used in [20][21] for recognition of structured, multi-person actions. Dynamic Bayesian networks (DBN) were used in [22] to simultaneously link broken trajectories and recognize complex events. Context-free grammars (CFGs) were used in [23] to develop a layered approach that first recognizes pose, then gestures and finally the atomic and complex events with the output of the lower levels fed to the higher levels. Stochastic context-free grammars (SCFGs) were used in [24] for representing and recognizing group activities. However, the likelihood of observation sequences in SCFGs are highly sensitive to model parameters, and inference has cubic time complexity [26].

While each of these formalisms have been successfully applied in various domains, Hidden Markov Models (HMM) and their extensions have by far been the most popular in activity recognition. Besides their simplicity, they also have well understood learning and inference algorithms making them well suited for a wide range of applications. HMMs were used to model complex gestures in American Sign Language (ASL) in [25] by modeling the actions of each hand. Coupled hidden Markov models (CHMM) were used in [5] to model multiple interacting processes and demonstrated them for recognizing tai-chi gestures. Parallel hidden Markov models (PaHMM) were introduced in [12] which recognized ASL ges-
tures by modeling each hand’s action as an independent HMM. The hierarchial hidden Markov model (HHMM) [26] was adopted in [6] for monitoring daily activities. The abstract hidden Markov model (AHMM) [27] is a related extension where each state of the HMM depends upon a hierarchy of actions. Explicit duration models using hidden semi-Markov models (HSMMs) were explored in [7] to recognize video events. The switching hidden semi-Markov model (S-HSMM), which is a two-layered extension of HSMM was presented in [8] and applied to activity recognition and abnormality detection.

In contrast to the generative HMMs, discriminative models like Conditional Random Fields (CRFs) have become increasingly popular due to their flexibility and improved performance when a large amount of labeled training data is available. CRFs were applied for contextual motion recognition in [3] and showed encouraging results. Hidden conditional random fields (HCRFs) presented in [28], introduced a 2-layer extension to the basic CRF framework for recognizing segmented gestures. The Latent Dynamic Conditional Random Fields (LDCRF) [4] extended HCRFs further for recognizing gestures from a continuous unsegmented stream. The hierarchical semi-conditional random field (HSCRF), developed in [9] simultaneously models hierarchical structure and state duration within a discriminative framework.

In this paper, we generalize these different generative and discriminative graphical models as well as their algorithms for learning and inference. In particular we show how our model generalizes the specific models we explored in our previous work [29][30][31] and present results that demonstrate their utility.

3. Model Definition and Parameters

Graphical models combine probability with graph theory to provide a general framework for modeling the joint distribution of a set of random variables $X = (X_1, ..., X_d)$. The variables are represented using the nodes in a graph, and the edges represent the relationships between them. In undirected graphical models like CRFs, based on the Hammersley-Clifford theorem (e.g. [2]) the joint probability of the variables $X$ is given by:

$$P(X|\theta) = \frac{1}{Z(\theta)} \prod_{c \in C} \psi_c(X|\theta_c)$$

where $\psi_c(X|\theta_c)$ is a potential function defined over the cliques $c$ in the graph, and $\theta_c$ are that parameterize the potential function. In a directed graphical model like
HMM, the joint probability $P(X)$ is factorized as:

$$P(X|\theta) = \prod_{i=1}^{d} P(x_i|x_{pa_i}, \theta_i)$$  \hspace{1cm} (2)

where $x_{pa_i}$ are the parents of $x_i$. In most applications, the random variables are divided as $X=(X_v,X_h)$ corresponding to the observed/visible variables $X_v$ and hidden variables $X_h$. The inference task involves finding the values for the hidden variables $X_h$ that maximize the joint probability $P(X_h, X_v|\theta)$ (generative training) or the conditional probability $P(X_h|X_v, \theta)$ (discriminative training). Learning involves finding the optimal parameters $\theta$ based on the training data. The various graphical models considered in literature in effect factorize the joint or conditional probability distribution and exploit the independence properties implied by the factorization for efficient inference and parameter learning.

When graphical models are used to analyze a temporal sequence of observations, the basic structure of the graph repeats over time. HMMs and their discriminative counterpart CRFs are the simplest graphical models for temporal analysis, with a single hidden variable at each time instant that takes a value from a set of discrete states. They are specified by the tuple:

$$\lambda = (Q, O, A, B, \pi)$$  \hspace{1cm} (3)

where, $Q$ is the set of possible states and $O$ is the set of observation symbols. In a HMM, $A$ is the state transition probability matrix ($a_{ij} = P(q_{t+1} = j|q_t = i)$), $B$ is the observation probability distribution ($b_j(k) = P(o_t = k|q_t = j)$) and $\pi$ is the initial state distribution. It is straightforward to generalize this model to continuous (like gaussian) output models by parameterizing the observation and transition probabilities. Since HMM and CRF form a discriminative/generative pair [32], in CRFs $A$, $B$ and $\pi$ correspond to the parameters of the transition, observation and initial potentials respectively.

This basic structure can be extended to model complex multi-scale structure by including a hierarchy of hidden states to obtain hierarchical hidden Markov models (HHMM) [26], or their discriminative counterparts such as hidden CRF (HCRF) [28] and latent dynamic CRF (LDCRF) [4]. Formally a model with $H$ levels can be represented by the tuples:

$$\lambda_h = (Q_h, O_h, A_h, B_h, \pi_h) \hspace{0.5cm} h = 1..H$$  \hspace{1cm} (4)

The parameters $A_h, B_h, \pi_h$ can depend not only on the state in level $h$ but also on the other levels, typically the parent level ($h+1$) and the child level ($h-1$).
The first order Markov assumption used in HMM/CRF implies that the duration probability of a state decays exponentially. The hidden semi-Markov models (HSMM) \[1\] and semi-CRF \[33\] alleviate this problem by introducing explicit state duration models. This model (\(\lambda''\)) can be specified by the tuple:

\[
\lambda'' = (Q, O, A, B, D, \pi)
\]  

(5)

Here \(D\) is a set of parameters that model the time spent in a particular state. Inference in HSMMs/semi-CRF have \(O(T^3)\) complexity for sequences of length \(T\). An alternate variable transition model was proposed in \[10\] where \(D\) is a set of parameters of the form \(P(i|j,d)\), where \(i, j\) are possible values for the state variable \(q\) and \(d\) is the duration for which the state has been \(j\). With this formulation inference complexity reduces to \(O(T)\) at the potential cost of increasing the number of parameters.

Many interesting processes have multiple interacting agents and several multi-channel graphical models have been proposed to model these. These extensions basically generalize the HMM/CRF state to be a collection of state variables \((S_t = S^1_t, .., S^C_t)\). In their most general form, such extensions can be represented as:

\[
\lambda''' = (Q^C, O^C, A^C, B^C, \pi^C)
\]  

(6)

where \(Q^c\) and \(O^c\) are the possible states and observations at channel \(c\) respectively and \(\pi^c\) represents the initial probability of channel \(c\)'s states. \(A^C\) contains the transition potentials/probabilities over the composite states \([q^1_{t+1}, .., q^C_{t+1}], [q^1_t, .., q^C_t]\), and \(B^C\) contains the observation potentials/probabilities over the composite states. In this form, the learning as well as inferencing algorithms are exponential in \(C\), and also result in poor performance due to over-fitting and large number of parameters that need to be learnt. The various multi-channel extensions typically introduce simplifying assumptions that help in factorizing the transition and observation probabilities. Factorial hidden Markov models (FHMM) \[34\] factor the hidden state into multiple variables which are nominally coupled at the output. This allows factorizing \(A^C\) into \(C\) independent \(N \times N\) matrices (\(N=\text{number of states in each channel}\)):

\[
P([q^1_{t+1}, .., q^C_{t+1}]|[q^1_t, .., q^C_t]) = \prod_{c=1}^{C} P(q^c_{t+1}|q^c_t)
\]  

(7)

while the observation probabilities \(B^C\) are left unfactorized. Parallel hidden Markov models (PaHMM) \[12\] factor the HMM into multiple independent chains and
hence allow factorizing both $A^C$ and $B^C$. Thus we have,

$$P([q^C_{t+1}, ..., q^C_t], [q^1_t, ..., q^C_t]) = \prod_{c=1}^{C} P(q^c_{t+1}|q^c_t)$$

$$P([o^1_t, ..., o^C_t], [q^1_t, ..., q^C_t]) = \prod_{c=1}^{C} P(o^c_t|q^c_t)$$

(8)

Coupled hidden Markov models (CHMM) [5] on the other hand factor the HMM into multiple chains where the current state of a chain depends on the previous state of all the chains. In CHMMs, each channel has its own observation sequence and hence $B^C$ can be factorized. Like FHMM and PaHMM, they allow each channel to evolve independently and hence we have,

$$P([q^C_{t+1}, ..., q^C_t]| [q^1_t, ..., q^C_t]) = \prod_{i=1}^{C} P([q^i_{t+1}| q^1_t, ..., q^C_t])$$

(9)

This can be further factorized using the representation theorem [11]. This states that given a conditional probability $P(x_i|p_{ai})$ and a variational parameter $\lambda$, there exist non-negative pairwise potentials

$$\overline{\psi}(x_i, x_j|\beta), \ \underline{\psi}(x_i, x_j|\beta) \ \forall j \in pa_i$$

such that

$$P(x_i|p_{ai}) = \max_{\beta} \prod_{j \in pa_i} \overline{\psi}(x_i, x_j|\beta) = \min_{\beta} \prod_{j \in pa_i} \underline{\psi}(x_i, x_j|\beta)$$

(10)

This implies that there exists a function $\psi(q^i_{t+1}, q^j_t|\beta)$ that will allow us to approximate $P([q^i_{t+1}| q^1_t, ..., q^C_t])$ in 9 as:

$$P(q^i_{t+1}| q^1_t, ..., q^C_t) \approx \prod_{j=1}^{C} \psi(q^i_{t+1}, q^j_t|\beta)$$

(11)

where, $\beta$ is an adjustable parameter that controls the quality of approximation. Note that the factorization in (10) is only an existence proof and it is in general difficult find a suitable $\psi$. However, for the logistic regression model that we use for the conditional probability $P(q^i_{t+1}| q^1_t, ..., q^C_t)$, there exist known transformations [11] that we leverage for additional factorization. We can define undirected
graphical models with the same structure as FHMM, PaHMM and CHMM by using potential functions instead of probabilities.

Each of the above extensions address one of the limitations of HMM/CRF and presents a solution that is suited to some specific application domains. In more recent work, the Switching-HSMM (S-HSMM)[8] and HSCRF [9] simultaneously address duration modeling and hierarchical organization. S-HSMM represents activities in 2-layers with the lower layer containing a HSMM and the upper layer containing a Markov model. Thus we have:

\[\lambda^{S-HSMM}_{\text{lower}} = (Q_{\text{lower}}, O_{\text{lower}}, A_{\text{lower}}, B_{\text{lower}}, D_{\text{lower}}, \pi_{\text{lower}})\]

\[\lambda^{S-HSMM}_{\text{upper}} = (Q_{\text{upper}}, O_{\text{upper}}, A_{\text{upper}}, B_{\text{upper}}, \pi_{\text{upper}})\]

The hierarchical multi-channel hidden semi-Markov graphical models (HM-HSGM) that we propose combine duration modeling, multi-channel interactions and hierarchical structure into a single model structure. In the most general form, they can be described by a set of parameters of the form:

\[\lambda^{HM-HSGM}_{h} = (Q_{h}^{c}, O_{h}^{c}, A_{h}^{c}, B_{h}^{c}, D_{h}^{c}, \pi_{h}^{c}) \quad h = 1..H\]

where, \(h \in 1..H\) is the hierarchy index, \(c\) is the number of channels at level \(h\), and the parameters have interpretations similar to before. Each channel at a higher level can be formed by a combination of channels at the lower level. Also, the duration models at each level are optional. Further, the channels at each level in the hierarchy maybe factorized using any of the methods discussed above (PaHMM, CHMM etc). It can be seen that \(\lambda^{HM-HSGM}\) presents a synthesis of \(\lambda', \lambda''\) and \(\lambda'''\).

4. Inference

Here we present a general algorithm for efficient inference in HM-HSGMs that leverage the model structure and approximations discussed in section 3. Our approach in effect generalizes the algorithms presented in [26][8] for inference in HHMM and S-HSMM respectively and allows inference in hierarchical models with multiple levels, with multiple channels and duration models at each level. In contrast, the algorithm in [26] allows only single channel models without duration models at each level in the hierarchy. On the other hand the algorithm in [8] considers only 2-level, single channel models. Further, our approach allows use of undirected graphical models with arbitrary potentials for modeling state transitions and durations while the algorithms in [26][8] are restricted to directed graphical models with probabilities.
4.1. Recursive Viterbi Descent (RVD)

A channel at level \( h \) in the HM-HSGM interacts only with channels in layers \( h-1 \) and \( h+1 \). The Recursive Viterbi Descent (RVD) algorithm uses this to factorize the probability distribution, and works by starting at states in the highest layer and recursively computing the Viterbi path at different time segments in the lower layers. In this algorithm, we assume that each channel in the lower level has single parent channel in the higher level and also each lower level state has a unique parent state. Thus we restrict the hierarchy to be trees rather than lattices.

In CRFs and HMMs, a single variable represents the state at time \( t \) and the best state sequence given an observation sequence \( \mathbf{O} = o_1..o_T \), is computed by calculating variables \( \delta_{i,t} \) as:

\[
\delta_{i,t} = \max_j \{ \delta_{j,t-1} + \ln \psi_t(j,i) \} + \ln \psi_o(i,o_t) \tag{13}
\]

where \( i \) and \( j \) are possible states. Here \( \delta_{i,t} \) denotes the log-likelihood of the maximum probability path to state \( i \) at time \( t \), \( \psi_t(j,i) \) and \( \psi_o(i,o_t) \) correspond to the transition and observation potentials/probabilities in CRF/HMM respectively. In the rest of the paper, we use terms of the form \( \psi \) to denote potentials in undirected models like CRFs and conditional probabilities in directed models like HMMs. Equation (13) describes the Viterbi algorithm for inference in HMMs. We have not included the partition function \( Z(\mathbf{O}) \) to normalize the likelihood computation in CRFs, since it is a constant for a given set of observations \( \mathbf{O} \).

In the HM-HSGMs, we have states at multiple levels \( h \) in the hierarchy. At each level, the state is represented by the values assigned to \( C_h \) variables, each corresponding to a channel at level \( h \). The inference is over time intervals \([t_{sc}, t_{ec}]\) in which channel \( c \) is in state \( ic \), instead of time instants \( t \). Each channel in level \( h \) has a parent channel in level \((h+1)\), and each state \( ic \) in that channel has a parent state \( p \) in the corresponding parent channel. Also, the probability that channel \( c \) is in state \( ic \) in interval \([tsc, tec]\) depends on the time \( \tau_{start} \) when the parent channel started in the parent state \( p \).

Thus generalizing the \( \delta_{i,t} \) variables in equation (13) to include these additional indices, at each level \( h \) in the hierarchy, we compute variables of the form \( \Delta_{h,c,p,\tau_{start}}^{[i_{c1}..i_{c2}]}[t_{sc1}..t_{sc2}][t_{ec1}..t_{ec2}] \), where \( c \) and \( p \) are the parent channel and state respectively in level \((h+1)\), \([c1..c2]\) are the set of channels in level \( h \) with \( c \) as parent and \([i_{c1}..i_{c2}]\) are possible states in channels \([c1..c2]\) respectively with \( p \) as parent state. \( \tau_{start} \) is the time at which the parent channel \( c \) started in parent state \( p \) and for an observation sequence of length \( T \), \( \tau_{start} \in [1..T] \). \( ts_{c'} \) and \( te_{c'} \) are start and end times of the interval during which channel \( c' \) in level \( h \) is in state \( i_{c'} \). With these
Figure 1: Illustration of Recursive Viterbi Descent with 2 channels at level $h$

definitions, we can generalize equation (13) to compute the $\Delta'$s as follows:

$$\Delta_{h,c,p}^{\text{start}}[i_{c} \ldots j_{c} \ldots i_{c}'] = \max_{j_{c}' \in \mathbb{C}_{h-1}, i_{c} \ldots j_{c} \ldots i_{c}'} \left\{ \Delta_{h-1,c',i}^{\text{start}}[i_{c} \ldots j_{c} \ldots i_{c}'] + \ln(\psi_{t}(i_{c} \ldots j_{c} \ldots i_{c}')) + \ln(\psi_{d}(i_{c}', t_{c}' - t_{s_{c}})) \right\}$$ (14)

where, $t_{s_{c}} - 1 \leq t_{c}'$, $\forall c'' \in [c_{1}..c_{2}], c'' \neq c'$, and $\psi_{t}$, $\psi_{d}$, $\psi_{o}$ correspond to the transition, duration and observation potentials.

Equation (14) looks quite dense due to the large number of variables, but has a fairly intuitive interpretation. We will explain using figure 1 which illustrates the algorithm for computing the $\Delta$'s at level $h$ with $C_{h} = 2$ channels. Given a $\Delta_{h,c,p}^{\text{start}}$, we first choose the channel $c'$ whose end time $t_{c'}$ is the smallest. Then, we add a "brick" of probability for some interval starting at $t_{c'} + 1$, by computing probabilities at the lower levels (Term 3). We take into account the
transition constraints through the transition potential \( \psi_t([i_{c1} \ldots j_{c1} \ldots i_{c2}], i_{c'}) \) and the duration constraints through the duration potential \( \psi_d(i_{c'}, te_{c'} - ts_{c'}) \). In order to find the best path, we take a max over the free variables. Note that we have factorized the transition probabilities like in equation (9). Thus we assume that the channels evolve independently.

At the highest level \( h = H \), there is no parent state and channel. Hence we drop the parent channel \( c \) and state \( p \) from the \( \Delta \) and set \( \tau_{\text{start}} = 1 \) (since we want infer the best path over the entire observation sequence). Thus we have:

\[
\Delta^{H, \tau_{\text{start}}=1} = \max_{i_{c1}', ts_{c1}', i_{c2}', te_{c2}'} \left\{ \Delta^{H, \tau_{\text{start}}=1} + \ln(\psi_t([i_{c1} \ldots j_{c1} \ldots i_{c2}], i_{c'})) + \right. \\
\left. \ln(\psi_d(i_{c'}, te_{c'} - ts_{c'})) + \max_{[i_{c1}', j_{c1}'] \in [i_{c1} \ldots i_{c2}], [ts_{c1}', ts_{c2}'] \in [ts_{c1} \ldots ts_{c2}]} \Delta^{H-1, c', p, \tau_{\text{start}}=1} \right\} 
\]

Also, at the lowest level \( h = 1 \), term 3 is replaced by the observation potential \( \ln(\psi_o(i_{c'}, O_{ts_{c'}} \ldots O_{te_{c'}})) \) in the interval \([ts_{c'}, te_{c'}]\) in state \( i_{c'} \).

\[
\Delta^{1, c, p, \tau_{\text{start}}} = \max_{i_{c1}', ts_{c1}', i_{c2}', te_{c2}'} \left\{ \Delta^{1, c, p, \tau_{\text{start}}} + \ln(\psi_t([i_{c1} \ldots j_{c1} \ldots i_{c2}], i_{c'})) + \ln(\psi_d(i_{c'}, te_{c'} - ts_{c'})) + \ln(\psi_o(i_{c'}, O_{ts_{c'}} \ldots O_{te_{c'}})) \right\} (16)
\]

where,

\[
\psi_o(i_{c'}, O_{ts_{c'}} \ldots O_{te_{c'}}) = \prod_{t=ts_{c'}}^{te_{c'}} \psi(i_{c'}, o_t)
\]

The maximum probability is given by \( \max_{[i_{c1} \ldots i_{c2}], [ts_{c1} \ldots ts_{c2}]} \Delta^{H, \tau_{\text{start}}=1} [T : T] \).

Suppose that there are a total of \( H \) levels and let the total number of channels in
any level $h$ be $C$. Computing the RHS in equation (14) has $O(N^C T^C)$ complexity. Since there are a total of $O(HCNT \times N^C T^C)$ $\Delta$ variables to compute, the overall complexity of the algorithm is $O(HCN^{2C+1} T^{3C+1})$.

4.2. Factored Recursive Viterbi Descent (FRVD)

While the RVD algorithm accurately computes the highest probability path, its time complexity is exponential in the number of channels $C$. This can be simplified for the independent channel (equation 8) and coupled channel (equation 11) factorizations of the transition probability, which allows us to factorize the $\Delta$’s into independent channels. Let $\delta_{i,c,p,\tau}^{h,c,p,\tau_{\text{start}}}$ denote the log-likelihood of the maximum probability path such that channel $c'$ in level $h$ is in state $i_{c'}$ from time $t_{s_{c'}}$ to $t_{e_{c'}}$. $c$ and $p$ are the parent channel and state and $\tau_{\text{start}}$ is the start time as before. Then,

$$
\delta_{i,c,p,\tau_{\text{start}}}^{h,c,p,\tau_{\text{start}}} = \begin{cases} 
\max_{i_{c'} \in [i_c, i_{c'}], t_{s_{c'}} \in [t_{s_{c'}}, t_{e_{c'}}]} & \Delta_{i_{c'}, c', t_{s_{c'}}, t_{e_{c'}}}^{h,c,p,\tau_{\text{start}}} \end{cases} \tag{17}
$$

which simply maximizes all the free variables in $\Delta$. Now, substituting (14) in (17),

$$
\delta_{i,c,p,\tau_{\text{start}}}^{h,c,p,\tau_{\text{start}}} = \begin{cases} 
\max_{i_{c'} \in [i_c, i_{c'}], t_{s_{c'}} \in [t_{s_{c'}}, t_{e_{c'}}]} & \Delta_{i_{c'}, c', t_{s_{c'}}, t_{e_{c'}}}^{h,c,p,\tau_{\text{start}}} \end{cases} \tag{18}
$$

$$
+ \ln(\psi_d(i_{c'}, t_{e_{c'}}, t_{s_{c'}})) + \max_{j \in [j_{c'}]} & \Delta_{j, t_{s_{c'}}, t_{e_{c'}}}^{h-1,c', j, t_{s_{c'}}} \end{cases} \tag{18}
$$
Now, if we use the independent non-interacting channel factorization as in equation (8), we have-

$$
\psi_t([i_{c_1}, j_{c'}], i_{c'}) = \psi_t(j_{c'}, i_{c'})
$$

(19)

Substituting (19) in (18) and rearranging,

$$
\delta^h_{c.p.\tau_{\text{start}}} = \max_{j_{c'}, ts'_{c'}, te_{c'}} \left\{ \max_{i_{c'}, ts_{c'}, te_{c'}} \left[ \Delta^h_{c.p.\tau_{\text{start}}} \right] + \ln(\psi_t(j_{c'}, i_{c'})) \right\}
$$

$$
+ \ln(\psi_d(i_{c'}, te_{c'} - ts_{c'})) + \max_{\begin{subarray}{c} i'_{c_{11}}, i'_{c_{12}} \end{subarray}} \left\{ \Delta^{h-1}_{c', j_{c'}, ts_{c'}} \right\}
$$

(20)

Substituting (17) for the $\Delta$ in the first term of (20) we get,

$$
\delta^h_{c.p.\tau_{\text{start}}} = \max_{j_{c'}, ts'_{c'}, te_{c'}} \left\{ \delta^h_{c.p.\tau_{\text{start}}(j_{c'}, ts'_{c'}, ts_{c'}, te_{c'})} + \ln(\psi_t(j_{c'}, i_{c'})) \right\}
$$

$$
+ \ln(\psi_d(i_{c'}, te_{c'} - ts_{c'})) + \max_{\begin{subarray}{c} i'_{c_{11}}, i'_{c_{12}} \end{subarray}} \left\{ \Delta^{h-1}_{c', j_{c'}, ts_{c'}} \right\}
$$

(21)

Also, if the channels are independent and do not interact, we can compute the $\Delta$'s in terms of the $\delta$'s as-

$$
\Delta^h_{c.p.\tau_{\text{start}}} = \sum_{c' = 1}^{c_2} \delta^h_{c.p.\tau_{\text{start}}} + \delta^h_{c.p.\tau_{\text{start}}(j_{c'}, ts'_{c'}, ts_{c'}, te_{c'})}
$$

(22)

Substituting (22) for the $\Delta$ in the third term of (21),

$$
\delta^h_{c.p.\tau_{\text{start}}} = \max_{j_{c'}, ts'_{c'}} \left\{ \delta^h_{c.p.\tau_{\text{start}}(j_{c'}, ts'_{c'})} + \ln(\psi_t(j_{c'}, i_{c'})) \right\}
$$

$$
+ \ln(\psi_d(i_{c'}, te_{c'} - ts_{c'})) + \sum_{c'' = 1}^{c_2} \delta^{h-1}_{c'', j_{c'}, ts_{c'}}
$$

(23)
where \( C_{i,c} \) is the number of channels in level \( h-1 \) with \( i,c \) as parent. This has an overall complexity of \( O(HC^2N^3T^4) \) which is also polynomial in all the variables. If we use the causally coupled factorization (11) instead of (8), by substituting (11) and (22) in (18) and rearranging we get:

\[
\delta_{i,c,p,T_{\text{start}} = 1}^{h} = \max_{j,c',t_s,c',t_e,c'} \left\{ \delta_{i,c,p,T_{\text{start}} = 1}^{h} + \ln(\psi_t(i,c',j,c',t_e,c')) \right\} + \\
\sum_{c' \in C_{pc}} \max_{j,c',t_s,c',t_e,c'} \left\{ \delta_{i,c,j,c',t_s,c',t_e,c'}^{h} + \ln(\psi_t(i,c',j,c',t_e,c')) \right\} + \\
\ln(\psi_d(i,c,c',t_s,c')) + \sum_{c' \in C_{pc}} \max_{j,c',t_s,c',t_e,c'} \delta_{i,c,j,c',t_s,c',t_e,c'}^{h-1} \\
\text{subject to } i,c,p,T_{\text{start}} = 1 \text{ and } c' \neq c
\]

(24)

where \( C_{pc} \) is the number of channels in level \( h-1 \) under state \( p \) in channel \( c \) at level \( h \). Term 1 in equation (24) corresponds to influence from the same channel, term 2 corresponds to influence from the other channels, term 3 corresponds to the duration probability and term 4 corresponds to the probability from the lower levels. Figure 2 illustrates the FRVD algorithm at level \( h \) with \( C = 2 \) channels.

The maximum probability given the observations is given by \( \sum_{c=1}^{H} \max_{i,c,t_s,c,T} \delta_{i,c,p,T_{\text{start}} = 1}^{h} \) and the best path can be computed by storing the indices that produce the maxima at each step in equation (24). With this approximation, the RHS in equation (24) takes \( O(CNT^2) \) and since there are a total of \( O(HCN \times N^3T^3) \) variables to compute, the overall complexity is \( O(HC^2N^3T^5) \) which is also polynomial.

### 4.3. Beam Search

While the FRVD algorithm significantly improves computation time by effectively factorizing the graphical model, the complexity of inference is still not linear in \( T \) making the computation expensive. If we could restrict the time spent in a particular state to be in some range \([M-Th, M+Th]\), then given \( T_{\text{start}} \) in equation (24), \( t_{s,c'}, t_{e,c'} \in [T_{\text{start}} + M-Th, T_{\text{start}} + M+Th] \). Thus there are only \( O(Th) \) possible values for \( t_{s,c'}, t_{e,c'} \). A similar reasoning applies for the RHS too and hence computing the \( \delta \)'s in equation (24) has \( O(HC^2N^3TMTh^3) \) and in equation (23) has \( O(HC^2N^3TMTh^2) \). We can simplify the complexity further by storing only the top \( K \) states for each interval \([t_{s,c'}, t_{e,c'}]\) in each channel \( c' \) at each level \( h \), and also pruning states whose probability is less than the maximum probability state by more than \( p \). Then the complexity becomes \( O(HC^2NK^2TMTh^3) \)
and $O(HC^2NK^2TMTh^2)$ for the coupled and independent channel factorizations respectively. In most applications, especially with Gaussian or Uniform duration models, $M, Th \ll T$ and also $k \ll N$ making the run time reasonable, even real-time under some conditions. Further, we can take advantage of additional constraints based on the specific model structure as well as application domain to further reduce the complexity.

4.4. Online Inference

One crucial issue with the algorithms for inference in graphical models is that the entire observation sequence must be seen before the state at any instant can be recognized. Thus even in cases where the algorithm runs at real time, the average latency (time taken to declare the state at a given instant) can be arbitrarily large. In simple HMMs or models that use the variable transition duration model, one simple solution is to use a fixed look ahead window, but the performance of this algorithm is poor. Instead we build on the online variable-window algorithm proposed in [35], which is demonstrated to be better than other approaches. At each frame we calculate the following function:

$$f(t) = M(t) + \gamma(t - t_0)$$  \hspace{1cm} (25)$$

where,
- $t_0$ is the last frame at which the states at different levels were output

- $M(t)$ can be any suitable function which is large if the potential at $t$ is concentrated around a single state and small if it is spread out

- The term $(t-t_0)$ penalizes latency

- $\gamma$ is a weight term that also functions as a normalization factor

With this metric, large values for $\gamma$ penalize latency while smaller values penalize errors. In our implementation we choose $M(t)$ to be \((1 - \sum_{c=1}^{H} \max_{i_c,t,s_c} \delta_{H,\tau_{start}}^{i_c,t,s_c,T})\) due to its simplicity. Alternate choices include the entropy of the state distribution but as our results show the simple choice performs satisfactorily. At each instant, if $M(t) < \gamma(t - t_0)$ we output the maximum probability state at $t$, and the states in the interval $[t_0, t]$ that lead up to it, and then repeat the procedure from $(t+1)$. This algorithm is shown to be 2-competitive\(^1\) in [35] for our choice of $M(t)$.

Note that the discussion in this section is valid only for models that do not use duration modeling or use the variable transition model. No known online inference algorithm exists for HSMM.

5. Parameter Learning

The choice of the parameter learning algorithm depends on whether the model is directed/undirected, and if the state variables are fully observed in training or not. In the rest of this section, we assume we have a training set $d = \{x_1, \ldots, x_N\}$ with $N$ examples. In directed graphical models, the parameters $\theta$ are represented using conditional probability tables that relates each node $X_i$ to its parents $X_{pa_i}$.

In the fully observed case where we have annotations for all the hidden variables, the maximum likelihood estimates for $\theta_i$ are simply tables containing normalized counts for each setting of $X_i$, given each setting of its parents, in the training set $d$. In this case, the parameters can be estimated independently from training data.

If the hidden variables are unobserved, we can use an Expectation-Maximization based algorithm for parameter estimation in directed HM-HSGMs by generalizing the Baum-Welch algorithm used for training HMMs. This will involve computation of forward and backward variables using methods similar to those presented in section 4. Details of this approach is presented in Appendix A. One issue with

\(^1\)The cost is no more than 2 times the cost of any other choice of $t$
Algorithm 1 Embedded Viterbi Learning

1: numTrain = number of training samples
2: \(P_{h,c,p}(i_{c'}) = \) Probability of starting in state \(i_{c'}\) in channel \(c'\), at level \(h\) with parent channel \(c\) and state \(p\).
3: \(P_{h,c,p}(j_{c'}|i_{c''}) = \) Probability of transitioning from state \(i_{c''}\) in channel \(c''\) to state \(j_{c'}\) in channel \(c'\), at level \(h\) with parent channel \(c\) and state \(p\).
4: \(P_{h,c,p}(d|i_{c'}) = \) Probability of spending a duration \(d\) in state \(i_{c'}\) in channel \(c'\), at level \(h\) with parent channel \(c\) and state \(p\).
5: \(P_{1,c,p}(O_k|i_{c'}) = \) Probability of observing symbol \(O_k\) in state \(i_{c'}\) in channel \(c'\) at the lowest level, with parent channel \(c\) and parent state \(p\).
6: for \(i = 1\) to numTrain do
7:     jointHMM \(\leftarrow\) String together HMMs of words/events forming training sequence \(i\).
8:     repeat
9:         maxPath \(\leftarrow\) State sequence of maximum probability path through jointHMM obtained using decoding algorithm.
10: \(\pi_{i_{c'}}^{h,c,p} \leftarrow\) No. of times channel \(c'\) is in state \(i_{c'}\) at level \(h\) with parent channel \(c\) and state \(p\) in maxPath.
11: \(n_{i_{c'},c}^{h,c,p} \leftarrow\) No. of times channel \(c'\) is in state \(i_{c'}\) at level \(h\) with parent channel \(c\) and state \(p\), in maxPath.
12: \(n_{i_{c'},j_{c'}}^{h,c,p} \leftarrow\) No. of times channel \(c'\) transitions from state \(i_{c'}\) to state \(j_{c''}\) at level \(h\) with parent channel \(c\) and state \(p\), in maxPath.
13: \(n_{i_{c'},d}^{h,c,p} \leftarrow\) No. of times state \(i_{c'}\) in channel \(c'\) spends duration \(d\) at level \(h\) with parent channel \(c\) and state \(p\), in maxPath.
14: \(n_{i_{c'},O_k}^{1,c,p} \leftarrow\) No. of times \(O_k\) is observed in state \(i_{c'}\) in channel \(c'\) at level \(h\) with parent channel \(c\) and state \(p\), in maxPath.
15: Re-estimate parameters using the following equations
16: \(P_{h,c,p}(i_{c'}) \leftarrow \pi_{i_{c'}}^{h,c,p} / \sum_{j_{c'}} \pi_{j_{c'}}^{h,c,p}\)
17: \(P_{h,c,p}(j_{c'}|i_{c'}) \leftarrow \frac{n_{i_{c'},j_{c'}}^{h,c,p}}{n_{i_{c'},c}^{h,c,p}}\)
18: \(P_{h,c,p}(d|i_{c'}) \leftarrow \frac{n_{i_{c'},d}^{h,c,p}}{n_{i_{c'},c}^{h,c,p}}\)
19: \(P_{1,c,p}(O_k|i_{c'}) \leftarrow \frac{n_{i_{c'},O_k}^{1,c,p}}{n_{i_{c'},c}^{h,c,p}}\)
20: until convergence
21: Split HMMs and update corresponding word/event HMMs.
22: end for

such an approach is that it slow to compute, and has numerical underflow issues
with long sequences. We take an alternate approach, where we first find the maximum probability path through the model using the FRVD algorithm. We then use the state sequence inferred to estimate parameters similar to the approach used in the fully observed case. We repeat this process until convergence. This approach is especially effective in constrained graphical structures like the left-right HMM (where each state can transition only to itself or to one state higher). A second challenge is that in many applications of interest to us, the training samples typically contain a sequence of words/events which are not segmented. We address this by stringing together the individual word/event models and then re-estimate parameters in the combined model. Algorithm 1 presents the pseudocode for the algorithm described here, which we call *Embedded Viterbi Learning*.

In undirected graphical models, the partition function $Z(\theta)$ couples all the parameters together even in the fully observed case. In this case, if all potential functions are from the exponential family, the optimization function for parameter learning is convex and is typically solved using gradient descent methods. Quasi-Newton methods such as L-BFGS have been particularly popular for this task. If some of the variables are unobserved however, the learning algorithm is intractable and general solutions to this task are not well understood.

6. Application to Continuous Sign Language Recognition

Figure 3: CHSMM Illustration- rounded boxes are primitive nodes, dark squares observations. $d_1$, $d_1 - 2$, $d_2$ are durations.
In this section, we demonstrate our approach in an application for automatically segmenting and recognizing American Sign Language (ASL) gestures from a continuous stream of data. Sign language recognition, besides being useful in itself, provides a good domain to test hierarchical multi-agent activities; Both hands go through a complex sequence of states simultaneously, each sign has distinct durations, and there is a natural hierarchical structure at the phoneme, word and sentence level. We considered the following three models in our experiments:

The Coupled Hidden Semi-Markov Model (CHSMM) (Figure 3) is a special case of $\lambda^h$ where each channel has states with explicitly specified duration models. This can be specified by the tuple:

$$\lambda^{CHSMM} = (Q^C, O^C, A^C, B^C, D^C, \pi^C)$$

where the parameters $Q^C, O^C, A^C, B^C, \pi^C$ are defined as before. $D^C$ contains a set of parameters of the form $P(d^c_i = k | q^c_i = i)$, i.e the probability of state $i$ in channel $c$ having a duration $k$. Further, $A^C$ and $B^C$ can be factorized in any of the methods discussed before. In this paper, we will focus on the causal coupling used in CHMMs.

The Hierarchical Semi-Markov Parallel Hidden Markov Model (HSPaHMM) (Figure 4) has 2 layers with multiple HMMs at the lower layer and HSMM at the upper layer and has the following set of parameters:

$$\lambda^{HSPaHMM}_{lower} = (Q^{lower}, O^{lower}, A^{lower}, B^{lower}, \pi^{lower})$$

$$\lambda^{HSPaHMM}_{upper} = (Q^{upper}, O^{upper}, A^{upper}, B^{upper}, D^{upper}, \pi^{upper})$$

Figure 4: HPaHSMM Illustration- circles are top-level nodes, rounded boxes lower-level nodes, dark squares observations. $d1, d1 - 2, d2$ are durations.
The Hierarchical Parallel Hidden Semi-Markov Model (HPaHSMM) (Figure 5) on the other hand contains multiple HSMMs at the lower layer and a single Markov chain at the upper layer and hence has the following set of parameters:

\[ \lambda_{\text{HPaHSMM lower}} = (Q_{\text{lower}}^C, O_{\text{lower}}^C, A_{\text{lower}}^C, B_{\text{lower}}^C, D_{\text{lower}}^C, \pi_{\text{lower}}^C) \]

\[ \lambda_{\text{HPaHSMM upper}} = (Q_{\text{upper}}, O_{\text{upper}}, A_{\text{upper}}, B_{\text{upper}}, \pi_{\text{upper}}) \]

The key difference between HSPaHMM and HPaHSMM is that HSPaHMM models the duration for the entire low-level PaHMM with the top-level HSMM state while HPaHSMM models the duration of each state in each low-level HSMM. Thus HSPaHMM requires fewer parameters, while HPaHSMM is a richer structure for modeling real events.

6.1. Experimental Results

For our experiments, we used a set of 50 test sentences from a larger dataset used in [12]; the sequences were collected using a MotionStar™ system at 60 frames per second, did not have any word segmentations and also had some noise at the beginning and end. We used a 10 word vocabulary, and each sentence is 2-5 words long for a total of 126 signs. The input contains the \((x, y, z)\) location of the hands at each time instant; from these we calculate the instantaneous velocities which are used as the observation vector for each time instant.

We model each word as 2-channel CHSMM, or PaHMM (for HSPaHMM) or 22
PaHSMM (for HPaHSMM) based on the Movement-Hold(MH) model [36] which breaks down each sign into a sequence of moves and holds. During a move some aspect of the hand is changed while during a hold all aspects are held constant. The MH model also identifies several aspects of hand configuration like location (chest, chin, etc), distance from body, hand shape, and kind of movement (straight, curved, round). With these definitions, we can encode the signs for various words in terms of constituent "phonemes". For example, in the word "I", a right-handed signer would start with his hand at some distance from his chest with all but his index finger closed and end at the chest. This can be encoded in the MH model as \((H(p0CH)M(strToward)H(CH))\), where \(p0CH\) indicates that the hand is within a few inches in front of the chest at the start, \(strToward\) indicates that hand moves straight perpendicular to the body and \(CH\) indicates that the hand ends at the chest. Similar transcriptions can be obtained for more complex 2-handed signs by considering both hands as well as hand shape. Table 1 shows the words in the lexicon and their corresponding phoneme transcriptions for the strong (right) hand.

We model the observation probabilities in the hold states as a normal distribution with \(\mu = 0\) while the move states are modeled as a signum function. Further, we set the inflection point of the signum to be the same as the Gaussian’s variance. The intuition behind this choice is that during the hold states the configuration of the hand remains constant with some random noise, while we have a move whenever the hand’s position changes above the noise threshold during an instant.

<table>
<thead>
<tr>
<th>Sign</th>
<th>Transcription</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>H-p0CH M-strToward H-CH</td>
</tr>
<tr>
<td>man</td>
<td>H-FH M-strDown M-strToward H-CH</td>
</tr>
<tr>
<td>woman</td>
<td>H-CN M-strDown M-strToward H-CH</td>
</tr>
<tr>
<td>father</td>
<td>H-p0FH M-strToward M-strAway M-strToward H-FH</td>
</tr>
<tr>
<td>mother</td>
<td>H-p0CN M-strToward M-strAway M-strToward H-CN</td>
</tr>
<tr>
<td>inform</td>
<td>H-iFH M-strDownRightAway H-d2AB</td>
</tr>
<tr>
<td>sit</td>
<td>S-m1TR M-strShortDown H-m1TR</td>
</tr>
<tr>
<td>chair</td>
<td>H-m1TR M-strShortDown M-strShortUp M-strShortDown H-m1TR</td>
</tr>
<tr>
<td>try</td>
<td>H-p1TR M-strDownRightAway H-d2AB</td>
</tr>
<tr>
<td>stupid</td>
<td>H-p0FH M-strToward H-FH</td>
</tr>
</tbody>
</table>

Table 1: Phoneme Transcriptions
We specified the duration models for the move states based on the distance between the starting configuration and the ending configuration, and the frame rate. To do this we separated the possible hand locations into 3 clusters - those around the abdomen (AB), those around the chest (CH) and those around the face/forehead (FH). We approximately initialized the hold state and intra-cluster transition times by looking at a few samples and set the inter-cluster transition time to be twice the intra-cluster transition time. We modeled the duration as a normal distribution centered around these means.

Table 2 shows the word accuracy rates. We did not include CHMM’s accuracy rates as the approximate decoding algorithm presented in [5] assumes that the probability mass at each channel is concentrated in a single state while calculating channel interactions. This assumption is not valid in our domain as many states can have nearly equal probabilities in the initial time steps and choosing only one of them prunes out the other words resulting in extremely poor performance.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHSMM</td>
<td>83.3% (N=126, D=5, S=14, I=2)</td>
<td>0.4</td>
</tr>
<tr>
<td>HPaHSMM</td>
<td>78.6% (N=126, D=6, S=17, I=4)</td>
<td>2.1</td>
</tr>
<tr>
<td>HSPaHMM</td>
<td>67.46% (N=126, D=11, S=22, I=8)</td>
<td>6.4</td>
</tr>
<tr>
<td>PaHMM</td>
<td>18.25% (N=126, D=7, S=23, I=73)</td>
<td>7.3</td>
</tr>
</tbody>
</table>

Table 2: Word Accuracy Rates(%) and Speed(fps)

These results indicate that including duration models significantly improves the results. HSPaHMM provides a good high-speed alternative to the more complex CHSMM and HPaHSMM. Further, HSPaHMM produces better results than PaHMM because the top-level HSMM restricts the number of word transitions and hence reduces the number of insertion errors. Our results were obtained without requiring any additional training data since the added model structure allows us to cleanly embed the domain constraints specified by the Movement-Hold model. Further we used only hand location as features, instead of detailed finger models.

7. Simultaneous Tracking and Recognition of Human Actions

In this section we develop a graphical model for representation and recognition of arm gestures and actions involving articulation of the full body. The HVT-HMM that we use for this has three layers with the top-most layer containing a single HMM for composite events. Each state in the top-level HMM corresponds to a
VT-HMM in the middle layer whose states in turn correspond to multiple HMMs at the track level for each degree of freedom. Using the notation described before, this can be formally represented by the set of parameters $\lambda_{HT-HMM}^c$ (composite layer), $\lambda_{HT-HMM}^p$ (primitive layer) and $\lambda_{HT-HMM}^x$ (track layer) as follows:

$$\begin{align*}
\lambda_{HT-HMM}^c & = (Q_c, O_c, A_c, B_c, \pi_c) \\
\lambda_{HT-HMM}^p & = (Q_p, O_p, A^d_p, B_p, \pi_p) \\
\lambda_{HT-HMM}^x & = (Q^C_x, O^C_x, A^C_x, B^C_x, \pi^C_x)
\end{align*}$$

(26)

Figure 6 illustrates the HVT-HMM.

**Pose Representation:** The pose $x$ of a person is represented using a 19D body model for the joint angles, with three additional dimensions for direction of translation(x,y,z) and one for scale(H) to give a total of 23 degrees of freedom (Figure 7). Note that we ignore the motion of the head, wrist and ankles as our image resolutions are generally too small to capture them. Each body part is represented as a cylinder, and the pose is fit to the image by projecting it to 2D.

We model each composite event with a single *left-right* VTHMM at the primitive layer and multiple HMMs (one for each moving part) at the track layer. If we denote the pose at time $t$ by $x_t$, the image by $I_t$, the foreground by $f_t$, and the projection of $x_t$ on $I_t$ by $Proj(x_t, I_t)$ we define the observation probability $P(I_t|x_t)$ to be the fraction of pixels in $Proj(x_t, I_t)$ that fall on $f_t$:

$$P(I_t|x_t) = \frac{|Proj(x_t, I_t) \cap f_t|}{|Proj(x_t, I_t)|}$$

(27)

We define the state transition probability at each of the three layers as follows:

At the **Composite Event Layer** transition probabilities are initialized uniformly as
the actors can perform actions in any order and it is generally difficult to obtain suitable training data to learn such high-level models.

At the **Primitive Event Layer**, we use a *left-right VTHMM*. Thus, at each instant a state can either transition to itself or to the next state in chain, and the probability of this transition varies with the time spent in the current state. We model this using a *sigmoid function* as follows:

\[
P(p_t | p_{t-1}) = \begin{cases} 
\frac{n}{1+e^{-d(p_{t-1} - \mu p_{t-1})/\sigma}} & \text{if } p_t = p_{t-1}, \\
\frac{n}{1+e^{-d(p_{t-1} - \mu p_{t-1})/\sigma}} & \text{if } p_t \geq p_{t-1} \\
0 & \text{otherwise}
\end{cases} \tag{28}
\]

where,

- \( n \) is a suitable normalization constant.
• \( p_t \) is the primitive event at time \( t \)

• \( p_t \geq p_{t-1} \) refers to the set of primitive events \( p_t \) that can occur after the primitive event \( p_{t-1} \)

• \( \mu_{p_{t-1}} \) denotes the average time spent in executing primitive action \( p_{t-1} \) and is learned from training data.

• \( d_{p_{t-1}} \) denotes the time spent in primitive \( p_{t-1} \) in the current execution of the action.

• \( \sigma \) is a noise parameter to allow for some variation in the actual end position of the actions. We set this to 20% of \( \mu_{p_{t-1}} \) in our implementation.

The intuition behind the choice of the sigmoid function is that it allows state transitions only when the primitive event has been executed for a duration close to the mean. This is because as \( d_{p_{t-1}} \) approaches \( \mu_{p_{t-1}} \), term 1 (for maintaining current state) in equation 28 decreases and term 2 (for transition to next state) increases as shown in Figure 8.

At the **Track Layer** we define the transition probability \( P(x_t|\{x_{t-1}, p_t\}) \) as follows:

\[
P(x_t|\{x_{t-1}, p_t\}) = \begin{cases} 
\frac{1}{2M_{p_t}} & \text{if } 0 \leq |x_t - x_{t-1}| \leq 2M_{p_t}, \\
0 & \text{otherwise}
\end{cases}
\]

(29)

where,
- $x_t$ is obtained from $x_{t-1}$ using a simple geometric transformation specified by $p_t$. For example, in the turn left gesture in figure 9, the first primitive corresponds to a clockwise rotation of the right arm, while the other body parts remain static.

- $|x_t - x_{t-1}|$ is the distance between $x_t$ and $x_{t-1}$, which depends on the geometric transformation specified by $p_t$. For example, for the first primitive event in the turn left gesture, this corresponds to the angle by which the right arm moves in a frame.

- $M_{p_t}$ is the mean distance the pose changes in one frame (average speed) under primitive $p_t$ and is learned during training. In the turn left gesture, this corresponds to the average angle by which the arm rotates.

This definition basically restricts the speed of a pose transition to at most $2M_{p_t}$. Based on these observation and transition probabilities, we estimated the parameters $\mu_i$ and $M_i$ using the embedded Viterbi algorithm presented in Algorithm 1. Further, since the actions in our experiments have typical primitive duration ($\mu_i$) and speed of action ($M_i$) across all instances, estimates from 1-2 samples for each action is sufficient to train the models.

7.1. Experiments

We tested our method in two domains - 1) For recognizing fourteen arm gestures used in military signalling and 2) For tracking and recognizing 9 different actions involving articulated motion of the entire body. All run time results shown were obtained on 3GHz, Pentium IV, running C++ programs.

7.1.1. Gesture Tracking and Recognition

In the first experiment, we demonstrate our method on the gesture datasets used in [16, 37], for examples in the domain of military signaling which consists of fourteen arm gestures shown in Figure 9. The dataset consists of videos of the fourteen gestures performed by five different people with each person repeating each gesture five times for a total of 350 gestures.

Each gesture is modeled by a 3-state VTHMM at the primitive layer which in turn corresponds to 2 track layer HMMs (one for each hand). We trained the model for each gesture using a randomly selected sequence and tested the variation in performance of the decoding algorithm with the pruning parameters $K$ and $p$. As can be seen from figure 10, while the accuracy increases slightly with larger $K$ and $p$, the speed decreases significantly. Since our video is at 30fps, we choose
Figure 9: Overlay Images for Arm Gestures

Figure 10: Variation in Accuracy(\%) and Speed(fps) with a)p b)K
Figure 11: Recognizing and tracking with variation in - a) Background and Lighting b) Style c) Random external motion d) Self-Oclusion

$K = 10$ and $p = 0.5$ for real-time performance. Our system gives an overall accuracy of 90.6%. We also tested our method with large variations in background, lighting, individual style, noise, occlusion and also with significant variation in the actual angle with the camera and observed no drop in performance. Figure 11 illustrates some of these variations. Next we replaced the primitive event layer of HVT-HMM with a simple HMM (to give a HHMM) and the semi-Markov HSMM (to give a HS-HMM) instead of VTHMM and tested them under the same training and pruning conditions. As can be seen from the results in table 3 including duration models in HVT-HMM produces a significant improvement in performance over an HHMM without the drop in speed seen in HS-HMMs. Further, the average latency of HVT-HMM is lower than that of HHMM because duration dependent state transitions restrict the set of possible states and hence $M(t)$ in equation 25 tends to be large. Since no known online decoding algorithm exists for an HSMM, the entire sequence must be seen before the state sequence is generated for HS-
### Table 3: Comparison of HVT-HMM, HHMM and HS-HMM Gesture Dataset

<table>
<thead>
<tr>
<th></th>
<th>Accuracy (%)</th>
<th>Speed (fps)</th>
<th>Avg. Latency (frames)</th>
<th>Max Latency (frames)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HVT-HMM</td>
<td>90.6</td>
<td>35.1</td>
<td>1.84</td>
<td>9</td>
</tr>
<tr>
<td>HHMM</td>
<td>78.2</td>
<td>37.4</td>
<td>4.76</td>
<td>14</td>
</tr>
<tr>
<td>HS-HMM</td>
<td>91.1</td>
<td>0.63</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

On the same dataset, [16] reports 85.3% accuracy on the first 6 gestures by testing and training on all 150 gestures (we have 97%). [37] reports 83.1% accuracy on the entire set. This approach uses edge contours instead of foreground silhouettes to allow for moving cameras, even though the data itself does not contain any such example; this may be partially responsible for its lower performance and hence a direct comparison with our results may not be meaningful. Another key feature of our method is that it requires just one training sample/gesture while [16, 37] require a `train : test` ratio of 4 : 1.

#### 7.1.2. Tracking and Recognizing Articulated Body Motion

In the second experiment, we demonstrate our methods on a set of videos used in [13] of actions that involve articulated motion of the whole body. The dataset contains videos (180*144, 25fps) of the 9 actions in Figure 12 performed by 9 different people. Each action is represented by a node in the top-level composite event graph. Each composite event node corresponds to a 2 or 4 node primitive event VTHMM, and each primitive node in turn corresponds to multiple track-level HMMs (one for each body part that moves). Thus the "walk" event has 4 primitive events (one for each half walk cycle) and 6 track HMMs (for the right and left shoulder, upper leg and lower leg), while the "bend" event has 2 primitives (for forward and backward motions) and 3 track HMMs (for the torso, knee and right shoulder) for each primitive. We learn the primitive and track level transition probabilities and also the average velocities for actions involving translation (walk, run, side, jump) using 1-2 examples per action.

Table 4 compares the performance of our approach with others and figure 13 shows some sample results. As can be seen, while the accuracy and speed are high, the latency is also higher compared to the gesture set. This is because the actions involve more moving parts and hence it takes longer to infer the state at any instant. Also note that even in cases where the initial joint estimates and pose tracking had errors, the recognition layer recovers. Next, we tested the robustness
Figure 12: Actions tested in Experiment 2 - a) Bend b) Jack c) Jump d) Jump-In-Place e) Run f) Gallup Sideways g) Walk h) Wave1 i) Wave2

Figure 13: Sample tracking and recognition results - a) Jack b) Jump c) Run d) Walk e) Gallop Sideways f) Bend g) Jump In Place h) Wave1 i) Wave2
<table>
<thead>
<tr>
<th></th>
<th>Accuracy (%)</th>
<th>Speed (fps)</th>
<th>Avg. Latency (frames)</th>
<th>Max Latency (frames)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HVT-HMM</td>
<td>100.0</td>
<td>28.6</td>
<td>3.2</td>
<td>13</td>
</tr>
<tr>
<td>HHMM</td>
<td>91.7</td>
<td>32.1</td>
<td>7.6</td>
<td>25</td>
</tr>
<tr>
<td>HS-HMM</td>
<td>100.0</td>
<td>0.54</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4: Comparison of HVT-HMM and HHMM Action Dataset

<table>
<thead>
<tr>
<th>Test Sequence</th>
<th>1st best</th>
<th>2nd best</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carrying briefcase</td>
<td>walk</td>
<td>run</td>
</tr>
<tr>
<td>swinging bag</td>
<td>walk</td>
<td>side</td>
</tr>
<tr>
<td>Walk with dog</td>
<td>walk</td>
<td>jump</td>
</tr>
<tr>
<td>Knees up</td>
<td>walk</td>
<td>run</td>
</tr>
<tr>
<td>Limp</td>
<td>jump</td>
<td>walk</td>
</tr>
<tr>
<td>Moon walk</td>
<td>walk</td>
<td>run</td>
</tr>
<tr>
<td>Occluded feet</td>
<td>walk</td>
<td>side</td>
</tr>
<tr>
<td>Full occlusion</td>
<td>walk</td>
<td>run</td>
</tr>
<tr>
<td>Walk with skirt</td>
<td>walk</td>
<td>side</td>
</tr>
<tr>
<td>Normal walk</td>
<td>walk</td>
<td>jump</td>
</tr>
</tbody>
</table>

Table 5: Robustness under occlusion, style variations and other factors

of our method to background, occlusions, style variations (carrying briefcase/bag, moonwalk etc) and also viewpoint (pan angle in the range $0^\circ - 45^\circ$). Our method is fairly robust to all these factors (Figure 14) and table 5 summarizes these results. The performance of our approach compares favorably with state-of-the-art results reported on the same set. These approaches are typically based on low-level features and require much higher $train : test$ ratios than ours, demonstrating the utility of modeling high-level constraints using graphical models.

8. Conclusions

We introduced a novel family of graphical models called hierarchical multi-channel hidden semi-Markov graphical models (HM-HSGMs) that can simultaneously model hierarchical structure, multi-agent interactions and event durations. We also presented efficient algorithms for inference in these models based on local variational approximations of the joint probability/potentials. Finally, we
presented fast training algorithms for parameter learning in directed HM-HSGMs using only a few partially annotated training examples. Together, this provides a powerful framework for action recognition in videos by modeling high-level constraints using graphical models. We demonstrated our framework by designing graphical models for continuous sign language recognition and for simultaneously tracking and recognizing human actions involving full body pose articulations. Our approach demonstrates state-of-the-art performance in all these domains. We plan to extend this work by developing graphical model based systems for action recognition in cluttered settings with camera motion, by combining them with successful low-level feature based methods.

Appendix A. Parameter Learning

Here we present an Expectation-Maximization algorithm for learning the various parameters that is similar to the Baum-Welch algorithm used to train HMMs. In the Baum-Welch algorithm for HMM, we compute forward and backward variables $\alpha_{i,t}$ and $\beta_{i,t}$ as follows:

$$\alpha_{i,t} = \sum_j \alpha_{j,t-1} \psi_t(j, i) \psi_o(i, o_t)$$ (A.1)

$$\beta_{i,t} = \sum_j \psi_t(i, j) \psi_o(j, o_{t+1}) \beta_{j,t+1}$$ (A.2)

where $\alpha_{i,t}$ denotes the total potential of being in state $i$ at time $t$, and $\beta_{i,t}$ is the total potential of the remaining observations $o_{t+1}..o_T$, starting in state $i$ at time $t$.

We generalize the $\alpha$ and $\beta$ variables to the HM-HSGMs similar to the way
we generalized the \( \delta \) variables in section 4.1. At each level \( h \) in the hierarchy, we need to compute forward variables of the form \( A_{h,c,p,\tau_{\text{start}}}^{[i1..i2][ts1..ts2][te1..te2]} \) where the indices have similar interpretations as in section 4.1. \( A_{h,c,p,\tau_{\text{start}}}^{[i1..i2][ts1..ts2][te1..te2]} \) denotes the total potential starting from \( t = \tau_{\text{start}} \) so that channel \( c' \) is in state \( i_{c'} \) in time interval \( [ts_{c'}, te_{c'}] \) \( \forall c \in 1..C_p \). We can write a recursive dynamic programming equation to compute the \( A \)'s as follows-

\[
A_{h,c,p,\tau_{\text{start}}}^{[i1..i2][ts1..ts2][te1..te2]} = \sum_{j,c,ts_{c'}} \left( A_{h,c,p,\tau_{\text{start}}}^{[i1..j..i2][ts1..ts_{c'}..ts2][te1..te_{c'}..te2]} \psi_t([i1..j..i2], i_{c'}) \times \psi_d(i_{c'}, te_{c'} - ts_{c'}) \times \sum_{c''} \left( A_{h-1,c',i_{c'},ts_{c'}}^{[i1..i_{c''}][ts1..ts_{c''}[te1..te_{c''}]} \right) \right)
\]

(A.3)

where, \( ts_{c'} - 1 \leq te_{c''}, \forall c'' \in [c1..c2], c'' \neq c' \).

Similarly we can define backward variables of the form \( B_{h,c,p,\tau_{\text{end}}}^{[i1..i2][ts1..ts2][te1..te2]} \) which denotes the total reverse potential from the ending time \( t = \tau_{\text{end}} \), so that channel \( c' \) is in state \( i_{c'} \) in time interval \( [ts_{c'}, te_{c'}] \) \( \forall c \in 1..C_p \). The \( B \)'s can be
computed as-

\[
B^{h,c,p,\tau_{\text{end}}}_{[i_{c1}..i_{c2}, [t_{s_{c1}}..t_{s_{c2}}, [t_{c_{e1}}..t_{c_{e2}}] = \]

\[
\psi_d(i_{c'}, te_{c'} - ts_{c'}) \times \sum_{j_c, te'_{c'}} \left\{ \psi_t([i_{c1}..i_{c2}], j_{c'}) \times B^{h-1,c',j_{c'}, te'_{c'}}_{[i_{c1}..i_{c2}, [t_{s_{c1}}..t_{s_{c2}}, [t_{c_{e1}}..t_{c_{e2}}] \right\} \right) \times \]

\[
\sum_{j_{c'}, te'_{c'}} \left\{ \psi_t([i_{c1}..i_{c2}], j_{c'}) \times B^{h,c,p,\tau_{\text{start}}}_{[i_{c1}..i_{c2}, [t_{s_{c1}}..t_{s_{c2}}, [t_{c_{e1}}..t_{c_{e2}}] \right\} \right) \right) \right) \right) \] (A.4)

where, \( te_{c'} + 1 \geq ts_{c''}, \forall c'' \in [c_1..c_2], c'' \neq c' \).

With these forward and backward variables, we can re-estimate the parameters at each layer similar to the standard Baum-Welch equation, but this can be time consuming and prone to overfitting. However, if we can factorize the transition potential \( \psi_t([i_{c1}..i_{c2}], j_{c'}) \) similar to equation (11) we can factorize the forward variables. Then, we need to compute variables \( \alpha^{h,c,p,\tau_{\text{start}}}_{i_{c'}, ts'_{c'}, te'_{c'}} \) which denotes the total potential such that channel \( c' \) in level \( h \) is in state \( i_{c'} \) from time \( ts_{c'} \) to \( te_{c'} \), starting from time \( t = \tau_{\text{start}} \). \( c \) and \( p \) are the parent channel and state and \( \tau_{\text{start}} \) is the start time as before. This can be done as:

\[
\alpha^{h,c,p,\tau_{\text{start}}}_{i_{c'}, ts_{c'}, te_{c'}} = \sum_{j_{c'}, ts'_{c'}} \left\{ \alpha^{h,c,p,\tau_{\text{start}}}_{j_{c'}, ts'_{c'}, te'_{c'}} \times \psi_t(j_{c'}, i_{c'}) \right\} \times \]

\[
\prod_{c''=1}^{C_{p_c}} \sum_{i_{c''}, ts''_{c''}, te''_{c''}} \left\{ \alpha^{h,c,p,\tau_{\text{start}}}_{i_{c''}, ts''_{c''}, te''_{c''}} \times \psi_t(i_{c''}, i_{c'}) \right\} \right) \right) \right) \] \times \psi_d(i_{c'}, te_{c'} - ts_{c'}) \times \prod_{c''=1}^{C_{c'}} \sum_{i_{c''}, ts''_{c''}, te''_{c''}} \alpha^{h-1,c',i_{c''}, ts''_{c''}, te''_{c''}} \] (A.5)

where \( C_{p_c} \) is the number of channels in level \((h - 1)\) under state \( p \) in channel \( c \) at level \( h \). Term 1 in equation (A.5) corresponds to influence from the same channel,
term 2 corresponds to influence from the other channels, term 3 corresponds to the duration potential and term 4 corresponds to the potential from the lower levels. Similarly, the backward variables $B$ can be factorized to compute variables $\beta_{i', c, \tau, end}^{h, c, p}$ which denotes the total reverse potential such that channel $c'$ in level $h$ is in state $i_{c'}$ from time $t s_{c'}$ to $t e_{c'}$, ending at time $t = \tau_{end}$.

$$
\beta_{i', c, \tau, end}^{h, c, p} = \sum_{j, t, e'_{c'}} \left\{ \beta_{j, t, e'_{c'} + 1, e'_{c'}}^{\tau, end} \times \psi_t (i_{c'}, j_{c'}) \right\} \times 
\prod_{c'' = 1}^{c'} \sum_{i_{c''}, t s_{c''}, e'_{c''}} \left\{ \beta_{i_{c''}, t s_{c''}, e'_{c''}}^{h, c, p, \tau, end} \times \psi_t (i_{c''}, i_{c'}) \right\} \times 
\psi_d (i_{c'}, t e_{c'} - t s_{c'}) \times C_{i', c'} \prod_{c'' = 1}^{c - 1} \alpha_{i, c'' - 1, t s_{c''}, e'_{c''}}^{h - 1, c', c'' - 1, t s_{c''}, e'_{c''}} (A.6)
$$

Also, if we use the independent channel factorization as in equation (8) instead of the causally coupled factorization, we can compute the forward and backward variables as:

$$
\alpha_{i, c, \tau, start}^{h, c, p} = \sum_{j, t, e'_{c'}} \left\{ \alpha_{j, t, e'_{c'} + 1, t s_{c'}}^{h, c, p, \tau, start} \times \psi_t (j_{c'}, i_{c'}) \right\} \times 
\psi_d (i_{c'}, t e_{c'} - t s_{c'}) \times \prod_{c'' = 1}^{c - 1} \alpha_{i, c'' - 1, t s_{c''}, e'_{c''}}^{h - 1, c', c'' - 1, t s_{c''}, e'_{c''}} (A.7)
$$

$$
\beta_{i, c, \tau, end}^{h, c, p} = \sum_{j, t, e'_{c'}} \left\{ \beta_{j, t, e'_{c'} + 1, t s_{c'}}^{h, c, p, \tau, end} \times \psi_t (i_{c'}, j_{c'}) \right\} \times 
\psi_d (i_{c'}, t e_{c'} - t s_{c'}) \times \prod_{c'' = 1}^{c - 1} \alpha_{i, c'' - 1, t s_{c''}, e'_{c''}}^{h - 1, c', c'' - 1, t s_{c''}, e'_{c''}} (A.8)
$$

In undirected graphical models like CRFs, the forward and backward variables are used for computing the partition function $Z$ during parameter estimation [2]. The exact equations for parameter estimation depends on how the transition ($\psi_t$), observation ($\psi_o$) and duration ($\psi_d$) potentials are parameterized. In directed graphical models extending HMMs that we considered in this paper, these potentials correspond to probabilities. Once we compute the forward and backward
variables at all levels, we can re-estimate these probabilities at each level \( h \) as follows with a suitable normalization factor:

1) **Initial Probability:** Let \( \pi_{i,c}^{h,c,p} \) denote the probability of starting in state \( i_c \) in channel \( c \) at level \( h \), and let the parent state and channel be \( p \) and \( c \) respectively. Then,
\[
\pi_{i,c}^{h,c,p} = \sum_{t_{end}, t_{c'}} \beta_{i_c,t_{end}, t_{c'}}^{h,c,p,t_{end}}
\]

2) **Transition Probability:** Let \( P_{h,c,p}^{h,c,p}(s_{t+1} = i_c' | s_t = j_{c''}) \) denote the probability of transitioning at level \( h \), to state \( i_{c'} \) in channel \( c' \) from state \( j_{c''} \) in channel \( c'' \) with the parent state and channel being \( p \) and \( c \) respectively. Then,
\[
P_{h,c,p}^{h,c,p}(s_{t+1} = i_c' | s_t = j_{c''}) = \sum_{t_{start}, t_{c'}} \alpha_{i_c',t_{start}, t_{c'}}^{h,c,p,tau_{start}} \times \sum_{t_{end}, t_{c'}} \beta_{j_{c''},t_{end}, t_{c''}}^{h,c,p,tau_{end}}
\]

3) **Duration Probability:** Let \( P_{e'}^{h,c,p}(d | s_t = i_{c'}) \) denote the probability of spending time \( d \) in state \( i_{c'} \) in channel \( c' \) at level \( h \) whose parent is state \( p \) in channel \( c \) at level \( h+1 \). Then,
\[
P_{e'}^{h,c,p}(d | s_t = i_{c'}) = \sum_{t_{start}, t_{c'}} \alpha_{i_{c'},t_{start}, t_{c'}}^{h,c,p,tau_{start}} \times \sum_{t_{end}, t_{c'}} \beta_{j_{c''},t_{end}, t_{c''}}^{h,c,p,tau_{end}}
\]

4) **Output Probability:** Observation probability is computed only at the lowest layer with \( h = 1 \). Let \( P_{o}^{1,c,p}(o = O_k | s_t = i_{c'}) \) denote the probability of observing
$O_k$ in state $i_{c'}$ whose parent is state $p$ in channel $c$. Then we re-estimate as,

$$P_{c'}^{1,c,p}(o_t = O_k|s_t = i_{c'}) = \left\{ \sum_{i_{c'},t's_{c'}} \alpha_{i_{c'},t's_{c'},t's_{c'}-1}^{1,c,p,t_{start}} \right\}$$

$$\sum_{t_{s_{c'}} \leq t \leq t_{s_{c'}} + d} \left\{ P_{c'}^{1,c,p}(d|i_{c'}) \prod_{t'=t_{s_{c'}},o_t = O_k} P_{c'}^{1,c,p}(o_{t'}|s_{t'} = i_{c'}) \right\} \sum_{j_{c'}^{\tau_{end}},t'e_{c'}^{\tau_{end}} + d + 1,t'e_{c'}} \beta_{j_{c'}^{\tau_{end}},t'e_{c'}^{\tau_{end}} + d + 1,t'e_{c'}}^{1,c,p,\tau_{end}}$$

(A.12)

We have skipped describing precise boundary conditions of the variables in the sums above for brevity. It can be seen that each of the equations (A.9)-(A.12) involve re-estimating parameters after summing out the free variables. Here $P$ and $P'$ correspond to the current and new estimate of the different probabilities, respectively. We re-estimate the probabilities using equations (A.9)-(A.12) until convergence.

References


[34] Z. Ghahramani, M. Jordan, Factorial hidden markov models, NIPS 8.

