Hierarchical Hidden Markov Models with General State Hierarchy

Hung H. Bui*
Artificial Intelligence Center
SRI International
333 Ravenswood Ave
Menlo Park, CA 94025, USA
bui@ai.sri.com

Dinh Q. Phung, Svetha Venkatesh
Department of Computing
Curtin University of Technology
GPO Box U 1987
Perth, Western Australia
{phungquo, svetha}@cs.curtin.edu.au

Abstract

The hierarchical hidden Markov model (HHMM) is an extension of the hidden Markov model to include a hierarchy of the hidden states. This form of hierarchical modeling has been found useful in applications such as handwritten character recognition, behavior recognition, video indexing, and text retrieval. Nevertheless, the state hierarchy in the original HHMM is restricted to a tree structure. This prohibits two different states from having the same child, and thus does not allow for sharing of common substructures in the model. In this paper, we present a general HHMM in which the state hierarchy can be a lattice allowing arbitrary sharing of substructures. Furthermore, we provide a method for numerical scaling to avoid underflow, an important issue in dealing with long observation sequences. We demonstrate the working of our method in a simulated environment where a hierarchical behavioral model is automatically learned and later used for recognition.

Introduction

Hierarchical modeling of stochastic and dynamic process has recently emerged as an important research issue in many different applications. The challenging problem is to design algorithms that are robust in noisy and uncertain dynamic environments, and at the same time take advantage of the natural hierarchical organization common in many real-world domains. To this end, the hierarchical hidden Markov model (HHMM) originally proposed by (Fine, Singer, & Tishby 1998) is an extension of the hidden Markov model to include a hierarchy of the hidden states. In this paper, we will refer to this as the FST model. Motivated by the inside-outside algorithm for probabilistic context-free grammar (PCFG) (Lari & Young 1990), Fine et al. presented a method for inference and expectation-maximization (EM) parameter learning in the HHMM with complexity T^3Sb^2, where T is the length of the observation sequence, S is the total number of hidden states, and b is the maximum number of substates for each state (branching factor). This model has recently been applied to a number of application domains, including handwritten character recognition (Fine, Singer, & Tishby 1998), robot navigation (Theocharous & Mahadevan 2002), behavior recognition (Luhr *et al.* 2003), video indexing (Xie *et al.* 2003) and information retrieval (Skounakis, Craven, & Ray 2003). In all of these domains, there exists a natural hierarchical decomposition of the modeled process, for example, in the way a sentence is written or a planned behavior is acted out. This property is shared by many other real-world domains, and thus an effective way of capturing this dynamic hierarchical abstraction can be very useful.

The original HHMM, however, has limitations that hinder its applicability. First, the original model considers only state hierarchies that have tree structures, disallowing the sharing of substructures among the high-level states. We argue here that substructure sharing is common in many domains. For example, in models for hand-written scripts, different word-level states should be able to share the same letter-level substates; in models of human behavior, different high-level intentions should be able to share the same set of lower-level actions. From a modeling perspective, sharing the substructures allows us to build more compact models, which facilitates more efficient inference and reduces the sample complexity in learning. Previous work on HHMM (Fine, Singer, & Tishby 1998; Murphy & Paskin 2001) allows sharing of substates only through explicit parameter tieing, although doing this is neither elegant nor efficient: one would need to expand a lattice state structure into a tree, which leads to the unnecessary exponential increase in the size of the representation and in the complexity of inference. To address this problem, we show that inference and EM learning can be done directly on the lattice state-structure, with the same complexity $O(T^3Sb^2)$. Furthermore, we present a method for numerical scaling to avoid underflow when dealing with a long observation sequence, a previously unaddressed issue. We demonstrate the working of our method in a simulated behavior recognition scenario.

This paper is motivated by earlier work. The idea that hierarchical decomposition of a stochastic dynamic process can be modeled in a dynamic Bayesian network (DBN) first appears in (Bui, Venkatesh, & West 2000; Murphy & Paskin 2001). Murphy and Paskin (2001) convert the HHMM to a DBN, and apply general DBN inference to the model to achieve complexity linear in time T, but exponential in the depth D of the HHMM. The same analysis applies to the "flattening" method, that is to convert an HHMM to an equivalent HMM for inference purposes (Xie *et al.* 2003). When the state hierarchy is strictly a tree, the number of states S is necessarily exponential in D. However, when considering a general lattice state hierarchy, the number of states S can be much smaller than $exp(D)$, or even linear in

*Supported by DARPA under contract NBCHD030010

D , depending on how much sharing is in the model. Thus, methods with complexity linear to the number of states like ours are useful, especially when the level of substate sharing is high. The T^3 complexity, however, means that our method is suitable only for batch learning. For online filtering, there exists an efficient approximate method for the abstract hidden Markov memory model (AHMEM) (Bui 2003), which is directly applicable to the HHMM. This can provide a fast approximation (linear in time and in the number of states of the model) for online recognition and classification.

HHMM: Model Definition and Parameters

In this section, we describe the structural model and parameter for the HHMM. We also briefly summarize the task involved in doing EM parameter reestimation in this model. The actual computational method is deferred until later sections.

Structural model and parameter

An HHMM is defined by a structural model ζ and a set of parameters θ. The structural model ζ specifies the model depth D, and for each $d \in 1..D$ a set of states at level d: Q^d. For convenience, we take the set Q^d to be $\{1..|Q^d|\}$. Level 1 is the top level and consists of a single state: $Q^1 = \{1\}$, while level D is the bottom one. For each state $q^d \in Q^d, d < D$, the structural model also specifies the set of children of q^d, denoted by $\text{ch}(q^d) \subset Q^{d+1}$. From this, the set of parents of q can be derived and is denoted by $\text{pa}(q^d)$. Note that the original HHMM assumes that $\text{pa}(q^d)$ always contains a single state; thus, the state hierarchy is strictly a tree. In addition to the states, we also specify an observation model. In the discrete observation case, we let $Y = \{1..|Y|\}$ be the set of observation symbols. It is straightforward to generalize to a Gaussian observation model, although the details are not shown here. Throughout the paper, the structural model ζ is assumed to be known and fixed; thus, we are not concerned with the problem of model selection.

Given such a structural model ζ, the parameter θ of the HHMM is specified as follows. For each level $d \in \{1..D-1\}$, $p \in Q^d$, $i, j \in \text{ch}(p)$: $\pi_i^{d,p}$ is the initial probability of the child i given the parent is p, $\text{A}_{i,j}^{d,p}$ is the transition probability from child i to child j, and $\text{A}_{i,\text{end}}^{d,p}$ is the probability that state p terminates given its current child is i. We require that $\sum_i \pi_i^{d,p} = 1$, $\sum_j \text{A}_{i,j}^{d,p} = 1$, and $\text{A}_{i,\text{end}}^{d,p} \leq 1$. FST transition parameters $a_{i,j}^{d,p}$ are slightly different from our definition, and can be obtained as $a_{i,j}^{d,p} = \text{A}_{i,j}^{d,p}(1 - \text{A}_{i,\text{end}}^{d,p})$. Finally, for $i \in Q^D$, $y \in Y$, $\text{B}_{y|i}$ is the probability of observing the symbol y given that the state at the bottom level is i.

Given the parameter θ, the HHMM defines a joint distribution over a set of variables that represent the evolution of the stochastic process over time. Following (Murphy & Paskin 2001), we specify this distribution by a DBN given in Figure 1. The variable q_t^d denotes the state at time t and level d; e_t^d is an indicator variable denoting if the state q_t^d has ended (i.e., its child has gone to an end state[1]).

[1]Note that q_t^d has a different interpretation than the same node in (Murphy & Paskin 2001): it takes a value over Q^d, the set of all states at level d, as opposed to just over the set of children of some parent state. Thus, in our DBN, q_t^d depends only on q_t^{d-1} and not on the entire set of states above it $q_t^{1:d-1}$.

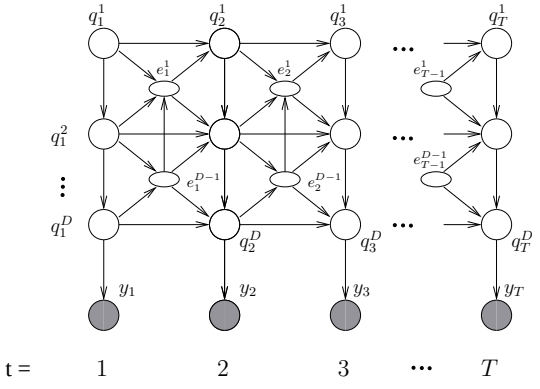


Figure 1: DBN for HHMM

Let $\mathcal{V} = \{q_{1:T}^{1:D}, e_{1:T}^{1:D-1}, y_{1:T}\}$ be the set of all variables. The HHMM defines a JPD over $\mathcal{V}$ following the factorization of the DBN[2]: $\Pr(q_{1:T}^{1:D}, e_{1:T}^{1:D-1}, y_{1:T}) \triangleq \prod_{v \in \mathcal{V}} \Pr(v \mid \text{parent}(v))$. For ease of notation, we also include the set of ending variables at the bottom level $\{e_t^D\}$, and fix their values to 1.

The conditional probability $\Pr(v \mid \text{parent}(v))$ is defined from the parameters of the HHMM and captures the evolution of the variables over time. A state can end only when the state below it has ended, and then does so with probability given by the termination parameter:

$$\Pr(e_t^d = 1 \mid q_t^{d+1} = i, q_t^d = p, e_t^{d+1}) = \begin{cases} \text{A}_{i,\text{end}}^{d,p} & \text{if } e_t^{d+1} = 1 \\ 0 & \text{if } e_t^{d+1} = 0 \end{cases}$$

A state stays the same until the next time if it does not end. If it ends and the parent state stays the same, it follows a transition to a new child-state of the same parent. Otherwise, it is initialized by a new parent-state:

$$\Pr(q_t^d = j \mid q_{t-1}^d = i, q_t^{d-1} = p, e_{t-1}^{d-1:d}) = \begin{cases} \delta(i,j) & \text{if } e_{t-1}^{d-1:d} = 00 \\ \text{A}_{i,j}^{d-1,p} & \text{if } e_{t-1}^{d-1:d} = 01 \\ \pi_j^{d-1,p} & \text{if } e_{t-1}^{d-1:d} = 11 \end{cases}$$

Sufficient statistics and EM

The previous two equations show that the HHMM parameter θ ties together different sets of parameters of the DBN. Thus, it is not surprising that the HHMM can be written in the exponential family form, with the sufficient statistics vector τ being the count of the different types of configurations in the data $\mathcal{V}$. In this case, maximum likelihood estimation of

[2]Since the top-level state does not end during the HHMM process, the set of well-defined events is restricted to those instantiations of $\mathcal{V}$ such that $e_1^{1:T-1}$ are false. An event that does not satisfy this property is not associated with any probability mass.

θ from a complete data set $\mathcal{V}$ reduces to setting the parameters to the normalized values of the sufficient statistics τ. Most of the time, however, we can observe only a subset $\mathcal{O}$ of the variables $\mathcal{V}$ so that the remaining variables $\mathcal{H} = \mathcal{V} \setminus \mathcal{O}$ are hidden. In this case, doing EM parameter reestimation reduces to first calculating the expected sufficient statistics (ESS) $\bar{\tau} = \mathrm{E}_{\mathcal{H}|\mathcal{O}}\, \tau$, and then setting the reestimated parameter $\hat{\theta}$ to the normalized value of $\bar{\tau}$.

Space restriction prevents giving the full set of sufficient statistics here. Rather, we describe only a subset of the sufficient statistics that corresponds to the transition parameter $\{\mathrm{A}^{d,p}_{i,j}\}$. Imagine the process of going through $\mathcal{V}$ and counting the number of occurrences of the configuration $\{q^{d+1}_{t+1} = j, q^{d+1}_t = i, q^d_{t+1} = p, e^{d:d+1}_t = 01\}$, resulting in the count $\tau(\mathrm{A})^{d,p}_{i,j}$. The corresponding ESS is then[3]:

$$\bar{\tau}(\mathrm{A})^{d,p}_{i,j} = \mathop{\mathrm{E}}_{\mathcal{H}|\mathcal{O}} \tau(\mathrm{A})^{d,p}_{i,j} = \sum_{t=1}^{T-1} \xi^{d,p}_t(i,j) \Big/ \Pr(\mathcal{O}) \qquad (1)$$

where the auxiliary variable $\xi^{d,p}_t(i,j)$ is defined as the probability $\Pr(q^{d+1}_{t+1} = j, q^{d+1}_t = i, q^d_{t+1} = p, e^{d:d+1}_t = 01, \mathcal{O})$. Most commonly, what we observe is the sequence $y_{1:T}$, together with the implicit fact that the HHMM does not end prior to T; thus our observation is $\mathcal{O} = \{y_{1:T}, e^1_{1:T-1} = \mathbf{0}\}$. In the rest of the paper, we will work explicitly with this set of observations.[4] Our method, however, also applies to an arbitrary set of observations in general.

The Asymmetric Inside-Outside Algorithm

From Equation (1), computing the ESS reduces to evaluating the set of auxiliary variables and the likelihood $\Pr(\mathcal{O})$. We now present a method for doing this.

Handling multiple parents

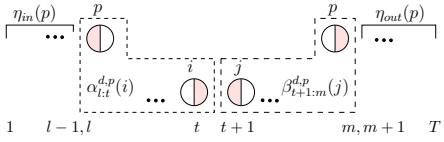


Figure 2: FST's decomposition.

We first describe the main intuition behind the FST method. Suppose we know that the state $q^d_t = p$ starts at time $l \le t$ and stops at time $m \ge t$ (when these times are not known, we can simply sum over all possible values of l, m). Fine et al. observe that the auxiliary variable $\xi^{d,p}_t(i,j)$ can then be factorized into the product of four different parts: the "in" part (η_{in}) consisting of the observations prior to l, the "forward" part (α) consisting of the observations from l to t, the "backward" part (β) consisting of the observations from $t+1$ to m, and the "out" part (η_{out}) consisting of the observations after m (Figure 2). Although not stated by Fine et al., implicit in this factorization is the assumption that q^d_t has a single parent state. When this does not hold, the unknown parent state of q^d_t destroys the conditional independence between the "in" and the "out" parts, and thus the factorization no longer holds.

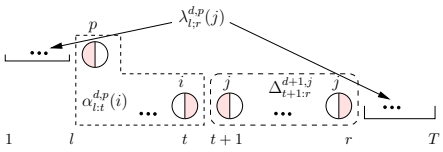


Figure 3: Our asymmetric inside-outside decomposition.

Our fix requires two modifications to the original method. First, rather than summing over the stopping time m of the state q^d_t, we sum over the stopping time r of the child state q^{d+1}_{t+1}. Second, we appeal to a similar technique used by the inside-outside algorithm in the PCFG, and group all the observations outside the interval $[l, r]$ into one, called the "outside" part (λ). Thus, the boundary between the inside and outside parts is not symmetrical, hence the term *asymmetric inside-outside*. The inside part is further factorized into an asymmetric inside part (same as FST's forward α), and an symmetric inside part ($\triangle$) (Figure 3). The formal definitions of these newly defined inside-outside variables are as follows[5]:

$$\begin{aligned}
\alpha^{d,p}_{l:r}(i) &\triangleq \Pr(\mathcal{O}_{in}, q^{d+1}_{\bullet r} = i, e^d_{l:r-1} = \mathbf{0} \mid \cdot q^d_l = p) \\
\triangle^{d,i}_{l:r} &\triangleq \Pr(\mathcal{O}_{in}, e^d_{l:r-1} = \mathbf{0}, e^d_r = 1 \mid \cdot q^d_l = i) \\
\lambda^{d,p}_{l;r}(i) &\triangleq \Pr(\mathcal{O}_{out} \mid \cdot q^d_l = p, e^d_{l:r-1} = \mathbf{0}, q^{d+1}_{\bullet r} = i) \Pr(\cdot q^d_l = p) \\
\underset{\sim}{\alpha}^{d,p}_{l:r+1}(i) &\triangleq \Pr(\mathcal{O}_{in}, \cdot q^{d+1}_{r+1} = i, e^d_{l:r} = \mathbf{0} \mid \cdot q^d_l = p)
\end{aligned}$$

where the dot in front of q^d_t represents the event $e^d_{t-1} = 1$, the dot after q^d_t represents the event $e^d_t = 1$, $\mathcal{O}_{in} = y_{l:r}$ is the set of observations "inside", and $\mathcal{O}_{out} = \{y_{1:l-1}, y_{r+1:T}, e^1_{1:T-1} = \mathbf{0}\}$ is the set of observations "outside". Intuitively, the asymmetric inside $\alpha^{d,p}_{l:r}(i)$ contains all the observations between when a parent state p starts and a child state i ends; the asymmetric outside $\lambda^{d,p}_{l;r}(i)$ contains all the remaining observations. The symmetric inside $\triangle^{d,i}_{l:r}$ contains all the observations between when a state i starts and when it ends. For convenience we also define a *started* α, denoted by $\underset{\sim}{\alpha}^{d,p}_{l:r+1}(i)$, which contains the observations from when a parent d starts at l to just before when a child i starts at $r+1$. A summary of these variables in concise diagrammatic forms is given in Figure 4.

[3] When multiple observation sequences are encountered, we need to also sum over the set of observation sequences in the equation.

[4] Note that Fine et al. assume that the observation also includes $e^T_1 = 1$. This limits training data to the set of complete sequences of observations from the start to the end of the HHMM generating process. Removing this terminating condition at the end allows us to train the HHMM with prefix data, e.g., observed data that does not have to last until the end of the process.

[5] The definitions for FST's auxiliary variables include informal use of words such as "started", "finished", and in some cases are incorrect. For example, the correct definition of their backward variable β should be $\Pr(\mathcal{O}_{in}, e^d_{l:r-1} = \mathbf{0}, e^d_r = 1 \mid \cdot q^{d+1}_l = i, q^d_l = p)$. Using the formal definitions of our auxiliary variables, all the equations in our paper can be verified formally. Space restrictions, however, prevent us from showing the detailed derivations.

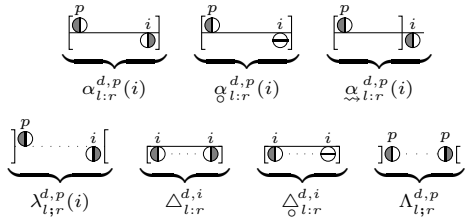


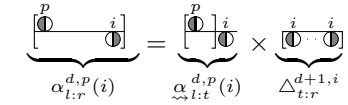
Figure 4: Diagrams for the inside and outside variables. Brackets denote the range of observations. Half-shaded circles denote starting/ending states. Crossed circles denote nonending states.

An important property of the HHMM is that given the event $\{\cdot q_l^d = p, e_{l:r-1}^d = \mathbf{0}, q_r^{d+1} = i\}$, which we call the *asymmetric boundary event*, the observations inside ($\mathcal{O}_{in}$) and outside ($\mathcal{O}_{out}$) of this asymmetric boundary are independent. With this property, we can expand the auxiliary variable ξ in terms of the newly defined variables:

$$\xi_t^{d,p}(i,j) = \sum_{l=1}^{t} \sum_{r=t+1}^{T} \left(\lambda_{l;r}^{d,p}(j) \alpha_{l:t}^{d,p}(i) a_{i,j}^{d,p} \triangle_{t+1:r}^{d+1,j} \right) \quad (2)$$

Calculating the inside-outside variables

The diagrammatic visualization of the inside and outside variables gives us a good tool for summarizing the recursive relations between these variables. Imagine trying to construct the diagram associated with $\alpha_{l:r}^{d,p}(i)$ using the other diagrams in Figure 4 as the building blocks. Let t be the starting time of the child state i that ends at r. We can then break the diagram of $\alpha_{l:r}^{d,p}(i)$ into two subdiagrams: one corresponds to $\underset{\rightsquigarrow}{\alpha}_{l:t}^{d,p}(i)$ and another one corresponds to $\triangle_{t:r}^{d+1,i}$:



The conditional independence property in the HHMM allows us to simply take the product of the two parts. Summing over the unknown time t then gives us the precise recursion formula for $\alpha_{l:r}^{d,p}(i)$:

$$\alpha_{l:r}^{d,p}(i) = \sum_{t=l}^{r} \underset{\rightsquigarrow}{\alpha}_{l:t}^{d,p}(i) \triangle_{t:r}^{d+1,i}$$

Recursion formulas for other variables can be derived intuitively in a similar way. The appendix provides a summary of all the formulas. Although not shown here, rigorous derivations can be worked out for each of them. Intuitively, each inside variable depends on other inside variables with smaller ranges, or variables at the lower level. The initialization of these variables can be derived straight from their definitions:

$$\underset{\rightsquigarrow}{\alpha}_{r:r}^{d,p}(i) = \pi_i^{d,p}, \triangle_{r:r}^{D,i} = \mathrm{B}_{y_r|i}$$

Based on this, dynamic programming can be used to compute all the inside variables bottom-up. Similarly, the outside variables can be computed top-down.

The remaining problem is to compute the likelihood $\Pr(\mathcal{O})$. Similar to FST, we sum over the asymmetric inside variable at the top level to yield

$$\sum_{i \in Q^2} \alpha_{1:T}^{1,1}(i) = \Pr(\mathcal{O}, e_T^1 = 1 \mid \cdot q_1^1 = 1) = \Pr(\mathcal{O}, e_T^1 = 1)$$

Unfortunately, this is not enough to give us the likelihood $\Pr(\mathcal{O})$. The missing term is $\Pr(\mathcal{O}, e_T^1 = 0)$, and we have not discussed how this can be obtained. To do this, we need to define a new variant of the asymmetric inside variable when the child state does not end at the right time index r. We call this the *continuing* α, denoted by $\underset{\circ}{\alpha}$:

$$\underset{\circ}{\alpha}_{l:r}^{d,p}(i) \triangleq \Pr(\mathcal{O}_{in}, q_r^{d+1} = i, e_r^{d+1} = 0, e_{l:r-1}^d = \mathbf{0} \mid \cdot q_l^d = p)$$

Similar recursion formulas can be established for $\underset{\circ}{\alpha}$ (see the appendix), which allow us to compute these variables bottom-up. It is then straightforward to verify that

$$\sum_{i \in Q^2} (\alpha_{1:T}^{1,1}(i) + \underset{\circ}{\alpha}_{1:T}^{1,1}(i)) = \Pr(\mathcal{O}) \quad (3)$$

Numerical Scaling

It is well-known that as the length of the observation sequence increases, naive implementations of the HMM will run into a numerical underflow problem. For this, a method for numerical scaling for the HMM has been derived (Rabiner 1989). It is thus imperative to address this same issue in the HHMM. Equation (1) reveals the source of the problem: both the numerator and the denominator of the RHS are joint probability of $O(T)$ variables that quickly go to zero as T becomes large. We can rewrite Equation (1) as $\bar{\tau}(\mathrm{A})_{i,j}^{d,p} = \sum_{t=2}^{T} \tilde{\xi}_t^{d,p}(i,j)$, where the scaled auxiliary variable $\tilde{\xi}_t^{d,p}(i,j)$ is defined as $\xi_t^{d,p}(i,j)/Pr(\mathcal{O})$, which is $\Pr(q_{t+1}^d = p, q_{t+1}^{d+1} = j, q_t^{d+1} = i, e_t^{d:d+1} = 01 \mid \mathcal{O})$. It is thus desirable to compute $\tilde{\xi}$ directly.

To do this, define the scaled factor at time t as follows: $c_1 = \Pr(y_1)^{-1}$, $c_t = \Pr(y_t, e_{t-1}^1 = 0 \mid y_{1:t-1}, e_{1:t-2}^1 = \mathbf{0})^{-1}$ for $t \geq 2$, so that $\prod_{k=1}^{t} c_k = \Pr(y_{1:t}, e_{1:t-1}^1 = \mathbf{0})^{-1}$. We proceed to scale up each inside and outside variable by the set of scaled factors in the observation range, that is,

$$\tilde{\alpha}_{l:r}^{d,p}(i) = \alpha_{l:r}^{d,p}(i) \prod_{k=l}^{r} c_k, \tilde{\triangle}_{l:r}^{d,i} = \triangle_{l:r}^{d,i} \prod_{k=l}^{r} c_k$$

$$\tilde{\lambda}_{l;r}^{d,p}(i) = \lambda_{l;r}^{d,p}(i) \prod_{k=1}^{l-1} c_k \prod_{k=r+1}^{T} c_k$$

and similarly for the other variables. Since each of the scaled variables effectively carries all the scaled factors within its range of observations, their product would carry the product of all the scaled factors. Thus, Equation (2) still holds if we replace each of the variables by its scaled version. By the same reason, the recursion formulas for the unscaled variables in the appendix also hold for their scaled counterparts.

We will now show that the scaled variables can be computed via dynamic programming in a bottom-up and left-right fashion. Assume that we have computed all the scaled inside variables up to the right index $r-1$. We can simply

apply the recursion formulas to calculate the scaled variables with the right index r. However, the remaining difficulty is in the initialization step $\tilde{\triangle}^{D,i}_{r:r} = \mathrm{B}_{y_l|i} c_r$ that requires knowledge of the scale factor c_r. Fortunately, we can get around this by a two-stage procedure. In the first stage, we calculate the new scaled variables from their usual recursion formulas, however, using the unscaled initialization $\triangle^{D,i}_{r:r} = \mathrm{B}_{y_l|i}$. The variables computed at this stage are partially scaled, i.e. they carry all the necessary scale factors except c_r. When we reach the top level, the partially scaled variables, denoted by $\ddot{\alpha}$, are

$$\ddot{\alpha}^{1,1}_{1:r}(i) = \alpha^{1,1}_{1:r}(i) \prod_{k=1}^{r-1} c_k, \quad \ddot{\alpha}^{1,1}_{\circ 1:r}(i) = \alpha^{1,1}_{\circ 1:r}(i) \prod_{k=1}^{r-1} c_k$$

Substituting these into Equation (3), we obtain

$$\sum_{i \in Q^2} (\ddot{\alpha}^{1,1}_{1:r}(i) + \ddot{\alpha}^{1,1}_{\circ 1:r}(i)) = \Pr(\mathcal{O}) \prod_{k=1}^{r-1} c_k = c_r^{-1} \qquad (4)$$

In the second stage, once c_r is obtained, it is multiplied into all the partially scaled variables calculated in the first stage. This gives us the final scaled variables at time index r.

Finally, the log likelihood $\log \Pr(\mathcal{O})$ can be obtained from the set of scale factors: $\log \Pr(\mathcal{O}) = -\sum_{k=1}^{T} \log c_k$.

Experimental Results

The first simulation is designed to to verify the EM parameter estimation in the presence of shared structures in the hierarchy (Figure 5). Based on a randomly initialized param-

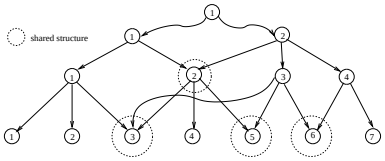


Figure 5: A 4-level hierarchy topology used in the simulation

eter θ, we generate 40 sequences of observation of length $T = 50$. Using the data, we estimate $\tilde{\theta}$. We found $\tilde{\theta}$ to be very close to θ. In general, the accuracy increases from top to bottom. This is not surprising because recovering the parameters at the higher hidden level is more difficult. An example is shown for the transition matrix $\mathrm{A}^{3,2}$ (estimated values in RHS).

$$\begin{bmatrix} 0.0459 & 0.2620 & 0.2475 & 0.4446 \\ 0.3716 & 0.2851 & 0.3233 & 0.0201 \\ 0.2296 & 0.2907 & 0.1835 & 0.2962 \end{bmatrix} \begin{bmatrix} 0.0472 & 0.2686 & 0.2549 & 0.4292 \\ 0.3682 & 0.2868 & 0.3272 & 0.0176 \\ 0.2324 & 0.3012 & 0.1896 & 0.2768 \end{bmatrix}$$

The second experiment demonstrates the advantage of our proposed method in comparison with the linear time method (Murphy & Paskin 2001). We construct two different topologies for testing. The first topology has 4 levels containing 3 states each (a total of 12 states excluding the top-level dummy state); the topology is fully connected, that is a state at each level is the parent of all the states at the level below. The second topology is similar except that it has five

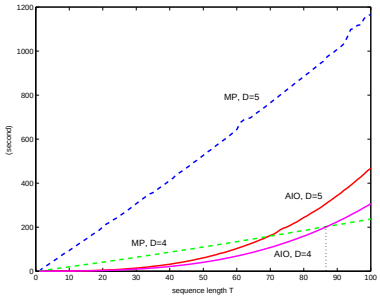


Figure 6: Computation time of MP method (Murphy & Paskin 2001) versus Asymmetric Inside-Outside (AIO) method

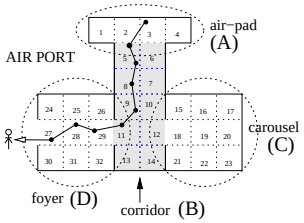


Figure 7: Airport simulation environment.

levels (and thus has 15 states). Figure 6 shows the computational time of the forward pass in one EM iteration using the two methods on two different topologies. While our method exhibits $O(T^3)$ complexity, it scales linearly when we increase the depth of the model. Adding an extra level means that our method only has to deal with an extra 3 states, where as the linear time method has to deal with 3 times the number of states due to its conversion to a tree structure.

The third experiment demonstrates the use of our method in learning a hierarchical model for movement trajectories in a simulated "airport" environment, shown in Figure 7. The airport is divided into four subregions: (A) the airpad, (B) the corridor, (C) the carousel, and (D) the foyer for entry and exit. At the top level, we are interested in three behaviors: (1) leaving the plane and exit the airport without collecting luggage (*exit-airport*), (2) collecting the luggage before exiting the airport (*pickup-luggage-exit*), and (3) friends picking up passengers (*meet-pickup*). This behaviors are built from a set of 9 behaviors at the lower level: *exit-airplane* (region A); *turn-left-foyer*, *turn-right-carousel*, *pass-left-right*, *pass-right-left* (region B); *collect-luggage*, *leave-carousel* (region C); and *enter-foyer*, *leave-foyer* (region D). The production level includes all the grid cells as its state space $(1 - 32)$. This results in a 4-level HHMM where each state corresponds to a behavior in the hierarchy. The lattice structure allows us to model sharing of subbehaviors among the high-level behaviors. For example, *exit-airport* and *pickup-luggage-exit* would share most of the lower-level subbehaviors, except that *exit-airport* does not involve entering the carousel.

A manually-specified HHMM is used to generate 30 sequences (length $= 30$) of observations as the training data set. The generating model is then thrown away, and the data is used to train a new HHMM. Except for prior knowledge

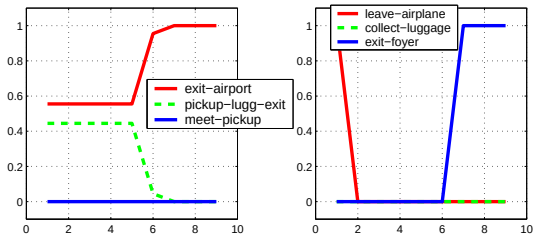


Figure 8: Online tracking result for top-level behaviors (left) and subbehaviors (right)

about the topology of the state hierarchy, learning is done completely unsupervised. Results similar to the first experiment are observed when comparing the original and estimated parameters. This allows us to then relabel the states of the learned model using "semantic" labels from the original model. To see how the learned model can be used for online tracking, we randomly generate trajectories for behaviors at the top level and examine the filtering distributions. For example, a random trajectory generated for *exit-airport* is shown in Figure 7. The filtering probability for this trajectory at two levels of behaviors is plotted in Figure 8. At the top level, the behavior *exit-airport* wins at the end, although the middle part is uncertain when it is not clear if the passenger will pick up the luggage (left diagram). For tracking lower-level behaviors, the graph shows a high probability for *leave-airplane* at the beginning, and a high probability for *exit-foyer* at the end (right diagram).

Conclusion

We have presented a method for parameter reestimation for the HHMM in the case where the state hierarchy is a general lattice. This allows us to learn models where the high-level states could share the same children at the lower level. Furthermore, we address an important issue when dealing with long observation sequences by providing a method for numerical scaling for the HHMM. Experimental results with simulated data show the potential of our method in building a hierarchical model of behavior from a data set of observed trajectories. We are currently applying our method to a real-world environment for learning hierarchical models of human movement behavior.

Appendix : Summary of the Formulas

We present the set of all formulas for computating the auxiliary variables defined for the HHMM. In all cases, except where explicitly stated, the level index d will range from 1 to $D-1$, where D is the depth of the model.

$$\begin{aligned}
\alpha^{d,p}_{l:r}(i) &\triangleq \Pr(y_{l:r}, q^{d+1}_{\bullet r}{=}\, i, e^{d}_{l:r-1}{=}\,\mathbf{0} \mid \cdot q^{d}_{l}{=}\, p) \\
&= \sum_{t=l}^{r} \underset{\rightsquigarrow}{\alpha}{}^{d,p}_{l:t}(i) \triangle^{d+1,i}_{t:r} \\
\underset{\circ}{\alpha}{}^{d,p}_{l:r}(i) &\triangleq \Pr(y_{l:r}, q^{d+1}_{r} = i, e^{d+1}_{r} = 0, e^{d}_{l:r-1} = \mathbf{0} \mid \cdot q^{d}_{l}{=}\, p) \\
&= \sum_{t=l}^{r} \underset{\rightsquigarrow}{\alpha}{}^{d,p}_{l:t}(i) \underset{\circ}{\triangle}{}^{d+1,i}_{t:r}
\end{aligned}$$

$$\begin{aligned}
\underset{\rightsquigarrow}{\alpha}{}^{d,p}_{l:r}(i) &\triangleq \Pr(y_{l:r-1}, \cdot q^{d+1}_{r}{=}\, i, e^{d}_{l:r-1} = \mathbf{0} \mid \cdot q^{d}_{l}{=}\, p) \\
&= \begin{cases} \sum_{j\in \mathrm{ch}(p)} \alpha^{d,p}_{l:r-1}(j) a^{d,p}_{j,i} & \text{if } r > l \\ \pi^{d,p}_{i} & \text{if } r = l \end{cases} \\
\triangle^{d,i}_{l:r} &\triangleq \Pr(y_{l:r}, e^{d}_{l:r-1} = \mathbf{0}, e^{d}_{r} = 1 \mid \cdot q^{d}_{l}{=}\, i) \\
&= \begin{cases} \sum_{s\in \mathrm{ch}(i)} \alpha^{d,i}_{l:r}(s) \mathrm{A}^{d,i}_{s,\mathsf{end}} \\ = \mathrm{B}_{y_r|i} & \text{if } d = D \end{cases} \\
\underset{\circ}{\triangle}{}^{d,i}_{l:r} &\triangleq \Pr(y_{l:r}, e^{d}_{l:r} = \mathbf{0} \mid \cdot q^{d}_{l} = i) \\
&= \begin{cases} \sum_{s\in \mathrm{ch}(i)} \left[\alpha^{d,i}_{l:r}(s) \left(1 - \mathrm{A}^{d,i}_{s,\mathsf{end}}\right) + \underset{\circ}{\alpha}{}^{d,i}_{l:r}(s)\right] \\ 0 \qquad \text{if } d = D \end{cases}
\end{aligned}$$

Define $\mathcal{O}_{out} \triangleq \{y_{1:l-1:}, y_{r+1:T}, e^{1}_{1:T-1} = \mathbf{0}\}$

$$\begin{aligned}
\Lambda^{d,p}_{l;r} &\triangleq \Pr(\mathcal{O}_{out} \mid \cdot q^{d}_{l}{=}\, p, e^{d}_{l:r-1}{=}\,\mathbf{0}, e^{d}_{r}{=}\,1) \Pr(\cdot q^{d}_{l}{=}\, p) \\
\lambda^{d,p}_{l;r}(i) &\triangleq \Pr(\mathcal{O}_{out} \mid \cdot q^{d}_{l}{=}\, p, e^{d}_{l:r-1}{=}\,\mathbf{0}, q^{d+1}_{\bullet r}{=}\, i) \Pr(\cdot q^{d}_{l}{=}\, p) \\
\Lambda^{1,1}_{1;T} &= 1, \qquad \lambda^{1,1}_{1;T}(i) = \mathrm{A}^{1,i}_{1,\mathsf{end}} \\
\lambda^{1,1}_{1;r}(i) &= \sum_{r'=r+1}^{\mathrm{RB}(1,1)} \sum_{j} \lambda^{1,1}_{1;r'}(j) \triangle^{2,j}_{r+1:r'} a^{1,1}_{i,j} \\
\Lambda^{d,p}_{l;r} &= \sum_{q\in \mathrm{pa}(p)} \sum_{l'=1}^{\mathrm{LB}(d,l)} \underset{\rightsquigarrow}{\alpha}{}^{d-1,q}_{l':l}(p) \lambda^{d-1,q}_{l';r}(p) \\
\lambda^{d,p}_{l;r}(i) &= \sum_{r'=r+1}^{\mathrm{RB}(d,r)} \sum_{j\in \mathrm{ch}(p)} \lambda^{d,p}_{l;r'}(j) a^{d,p}_{i,j} \triangle^{d+1,j}_{r+1:r'} + \Lambda^{d,p}_{l;r} \mathrm{A}^{d,p}_{i,\mathsf{end}}
\end{aligned}$$

where $\mathrm{LB}(d,l)$ and $\mathrm{RB}(d,r)$ are two functions used to compute the appropriate index boundary: $\mathrm{LB}(d,l) \triangleq l$ if $l > 1, d > 2$, and $\triangleq 1$ otherwise; $\mathrm{RB}(d,r) \triangleq T$ if $d < D-1, \triangleq (r+1)$ if $d = (D-1)$, and $\triangleq r$ if $r = T$.

References

Bui, H. H.; Venkatesh, S.; and West, G. 2000. On the recognition of abstract Markov policies. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-2000)*.

Bui, H. H. 2003. A general model for online probabilistic plan recognition. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*.

Fine, S.; Singer, Y.; and Tishby, N. 1998. The hierarchical Hidden Markov Model: Analysis and applications. *Machine Learning* 32.

Lari, K., and Young, S. J. 1990. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language* 4:35–56.

Luhr, S.; Bui, H. H.; Venkatesh, S.; and West, G. 2003. Recognition of human activity through hierarchical stochastic learning. In *IEEE International Conference on Pervasive Computing and Communication (PERCOM-2003)*.

Murphy, K., and Paskin, M. 2001. Linear time inference in hierarchical HMMs. In *NIPS-2001*.

Rabiner, L. R. 1989. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2):257–286.

Skounakis, M.; Craven, M.; and Ray, S. 2003. Hierarchical hidden Markov models for information extraction. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*.

Theocharous, G., and Mahadevan, S. 2002. Learning the hierarchical structure of spatial environments using multiresolution statistical models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Xie, L.; Chang, S.-F.; Divakaran, A.; and Sun, H. 2003. Unsupervised discovery of multilevel statistical video structures using hierarchical hidden Markov models. In *International Conference on Multimedia and Exhibition (ICME-2003)*.