Learning Hierarchical Hidden Markov Models for Video Structure Discovery

Lexing Xie[†], Shih-Fu Chang[†], Ajay Divakaran[‡], Huifang Sun[‡]

[†]Department of Electrical Engineering, Columbia University, New York, NY
{xlx, sfchang}@ee.columbia.edu
[‡]Mitsubishi Electric Research Labs, Murray Hill, NJ
{ajayd, hsun}@merl.com

Abstract

Structure elements in a time sequence are repetitive segments that bear consistent deterministic or stochastic characteristics. While most existing work in detecting structures follow a supervised paradigm, we propose a fully unsupervised statistical solution in this paper. We present a unified approach to structure discovery from long video sequences as simultaneously finding the statistical descriptions of structure and locating segments that matches the descriptions. We model the multilevel statistical structure as hierarchical hidden Markov models, and present efficient algorithms for learning both the parameters, as well as the model structure including the complexity of each structure element and the number of elements in the stream. We have also proposed feature selection algorithms that iterate between a wrapper and a filter method to partition the large feature pool into consistent and compact subsets, upon which the hierarchical hidden Markov model is learned. When tested on a specific domain, soccer video, the unsupervised learning scheme achieves very promising results: the automatically selected feature set includes the manually identified intuitively most significant feature, and the system automatically discovers the statistical descriptions of high-level structures, and at the same time achieves even slightly better accuracy in detecting discovered structures in unlabelled videos than a supervised approach designed with domain knowledge and trained with comparable hidden Markov models.

Keywords: video structure, statistical learning, unsupervised learning, feature selection, hierarchical hidden Markov model, hidden Markov model, model selection, MCMC

Contents

1 Introduction

In this paper, we present algorithms for jointly discovering statistical structures, identifying model orders, and finding informative low-level features from video in an unsupervised setting. Effective solutions to video indexing require detection and recognition of *structure* and *event* in the video, where *structure* represents the syntactic level composition of the video content, and *event* represents the occurrences of certain semantic concepts. We define structure as the repetitive segments in a time sequence that possess consistent deterministic or stochastic characteristics. This definition is general to various domains, and structures exist at multiple levels of abstraction. At the lowest level for example, structure can be the frequent triples of symbols in a DNA sequence, or the repeating color schemes in a video; at the mid-level, the seasonal trends in web traffics, or the canonical camera movements in films; and at a higher level, the genetic functional regions in DNA sequences, or the game-specific state transitions in sports video. Automatic detection of structures will help locate semantic events from low-level observations, and facilitate summarization and navigation of the content.

1.1 The structure discovery problem

The problem of identifying structure consists of two parts: finding a description of the structure (a.k.a *the model*), and locating segments that matches the description. There are many successful cases where these two tasks are performed in separate steps. The former is usually referred to as *training*, while the latter, *classification* or *segmentation*. Among various possible models, hidden Markov model (HMM) [Rab89] is a discrete state-space stochastic model with efficient learning algorithms that works well for temporally correlated data streams. HMM has been successfully applied to many different domains such as speech recognition, handwriting recognition, motion analysis, or genome sequence analysis. For video analysis in particular, different genres in TV programs were distinguished with HMMs trained for each genre [WLH00], and the high-level structure of soccer games (e.g. play versus break) was also delineated with a pool of HMMs trained for each category [XCDS02].

The structure detection methods above falls in the conventional category of supervised learning - the algorithm designers manually identify important structures, collect labelled data for training, and apply supervised learning tools to learn the classifiers. This methodology works for domain-specific problems at a small scale, yet it cannot be readily extended to diverse domains at a large scale. In this paper, we propose a new paradigm that uses fully unsupervised statistical techniques and aims at automatic discovery of salient structures and simultaneously recognizing such structures in unlabelled data without prior expert knowledge. Domain knowledge, if available, can be used to relate semantic meanings to the discovered structures in a post-processing

stage. Although unsupervised learning techniques dates back to several decades ago [JMF99], most of the data sets were treated as independent samples, while the temporal correlation between neighboring samples are largely unexploded. Classical time series analysis techniques has been widely used in many domains such as financial data and web stat analysis [ISZ99], where the problem of identifying seasonality reduces to the problem of parameter estimation in known order of ARMA model, and the order is determined with statistical tests on the correlation functions, yet this model does not readily adapt to domains with frequent discontinuities. New statistical methods such as Monte Carlo sampling has also appeared in genome sequence analysis [LAB+93], where unknown short motifs were recovered by finding a best multiple alignment among all protein sequences using Gibbs sampling techniques on a multinomial model, yet independence among amino acids in adjacent positions is still assumed. Only a few instances has been explored for video, however. Clustering techniques are used on the key frames of shots [YY96] or the principal components of color histogram [SZ99] to discover the story units in a TV drama, yet the temporal dependency of video was not formally modelled. In an independent work of Naphade and Huang [NH02], several HMMs were concatenated to identify temporally evolving events in films such as explosion and crashing, yet no specific quantitative results were reported.

Moreover, the computational front end in many real-world scenarios extracts a large pool of observations (i.e. features) from the stream, and at the absence of expert knowledge, picking a subset of relevant and compact features becomes a bottleneck. And automatically identifying informative features, if done, will improve both the learning quality and computation efficiency. Prior work in feature selection for supervised learning mainly divides into *filter* and *wrapper* methods according to whether or not the classifier is in-the-loop [KS96]. Many existing work address the supervised learning scenario, and evaluate the fitness of feature with regard to its information gain against training labels (*filter*) or the quality of learned classifiers (*wrapper*). For unsupervised learning on spatial data (i.e. assume samples are independent), Xing et. al. [XK01] iterated between cluster assignment and filter/wrapper methods for known number of clusters; Dy and Brodley [DB00] used scatter separability and maximum likelihood (ML) criteria to evaluate fitness of features. To the best of our knowledge, no prior work has been reported for our problem of interest: unsupervised learning on temporally dependent sequence with unknown cluster size.

1.2 Our approach

In this paper, we model the temporal dependencies in video and the generic structure of events in a unified statistical framework. Under certain dependency assumptions, we model the individual recurring events in a video as HMMs, and the higher-level transitions between these events as another level of Markov chain. This hierarchy of HMMs forms a Hierarchical Hidden Markov Model(HHMM), its hidden state inference and parameter estimation can be efficiently learned

using the expectation-maximization (EM) algorithm. This framework is general in that it is scalable to events of different complexity; yet it is also flexible in that prior domain knowledge can be incorporated in terms of state connectivity, number of levels of Markov chains, and the time scale of the states. In addition, Bayesian learning techniques are used to learn the model complexity automatically, where the search over model space is done with reverse-jump Markov chain Monte Carlo, and Bayesian Information Criteria is used as model prior. We use a combination of filter and wrapper methods for feature selection. The first step is to wrap information gain criteria around HHMM learning, and discover relevant feature groups that are more consistent to each other within the group than across the group; the second step is to find an approximate Markov blanket for each group, thus eliminating redundant features that does not contribute to uncovering the structure from sequence given its Markov Blanket; and the last step is to evaluate each condensed feature group with a normalized BIC, and rank the resulting models and feature sets with respect to their *a posteriori* fitness.

Evaluation against real video data showed very promising results: Evaluation against real video data showed a very promising result - the unsupervised approach automatically discovers the high-level structures, their statistical descriptions, and at the same time achieves even slightly better accuracy in detecting discovered structures in unlabelled videos than a supervised approach using domain knowledge and comparable HMM models. On two MPEG-7 soccer videos, the number of clusters that the algorithm converges to is mostly two or three, matching manually labelled classes with comparable accuracies in [XCDS03]; the optimal feature set includes the dominant color ratio, the intuitively the most distinctive feature.

The rest of this report is organized as follows: section 2 discusses the properties of structures in video and lists the assumptions used in modelling; section 3 presents the structure and semantics of the HHMM model; section 4 presents the inference and parameter learning algorithms for HHMM; section 5 presents algorithms for learning HHMM strucutre; section 6 presents our feature selection algorithm for unsupervised learning over temporal sequences; section 7 evaluates the results of learning with HHMM on soccer video data; section 8 summarizes the work and discusses open issues.

2 Modelling Video Structure

Our main attention in this paper, is on the particular domain of video, where the structures has the following properties: (1)Video structure is in a discrete state-space, since we humans understand video in terms of concepts, and concepts are discrete in nature; (2)The observations are stochastic, since segments of video seldom have exactly the same raw features even if they are conceptually similar ; (3)The sequence is highly correlated in time, since the videos are sampled

in a rate much higher than that of the changes in the scene.[1] In particular, we will focus our attention on the subset of *dense* structure for this paper. By *dense* we refer to the cases where competing structure elements can be modelled as the same parametric class, and representing their alternation would be sufficient for describing the whole data stream, i.e. there is no need for an explicit *background* model that delineates *sparse* events from the majority of the background.

Hence, we model stochastic observation in a temporally correlated discrete state space, and use a few weak assumptions to facilitate efficient computation. We assume that within each concept, states are discrete and Markov, and observations are associated with states under a fixed parametric form, usually Gaussian. Such assumptions are justified based on the satisfactory results from the previous works using supervised HMM of similar forms [WLH00, XCDS02]. We also model the transitions of concepts as a Markov chain at a higher level, this simplification will bring computational convenience at a minor cost of modelling power. In the basic algorithm described in section 4, we will also assume that the size of the model is given, and the model is learned over a pre-defined feature set. These two assumptions can be relaxed using the proposed model selection algorithms described in section 5, and feature selection criteria in section 6.

3 Hierarchical hidden Markov models

Base on the two-level Markov setup described above, we use two-level hierarchical hidden Markov model to model structures in video. In this model, the higher-level structure elements usually correspond to semantic concepts, while the lower-level states represents variations that can occur within the same concept, and these lower-level states in turn produce the observations, i.e., measurements taken from the raw video, with mixture-of-Gaussian distribution.

3.1 Structure of HHMM

Hierarchical hidden Markov model was first introduced by Fine, Singer and Tishby [FST98] as a natural generalization to HMM with hierarchical control structure. As shown in Figure 1(A), every higher-level state symbol correspond to a stream of symbol produced by a lower-level sub-HMM; a transition on the higher-level model is invoked only when the lower-level model enters an *exit* state (shaded nodes in Figure 1(A)); observations are only produced by the lower level states.

This bottom-up structure is general in that it includes several other hierarchical schemes as special cases. Examples include the discrete counterpart of the jump Markov model [DA01] with top-down(rather than bottom-up) control structure, or parallel stacking of left-right HMMs [NH02].

[1] In a soccer video for example, the high-level *concept* of a "goal" and the lower-level *concept* of a "close-up shot" exemplify themselves in the motion or color *observations* obtained from the video stream. The concept of "close up" is among the set of potential concepts {global, zoom-in, close-up}, and the color histogram of two close-up shots would be similar but not identical.

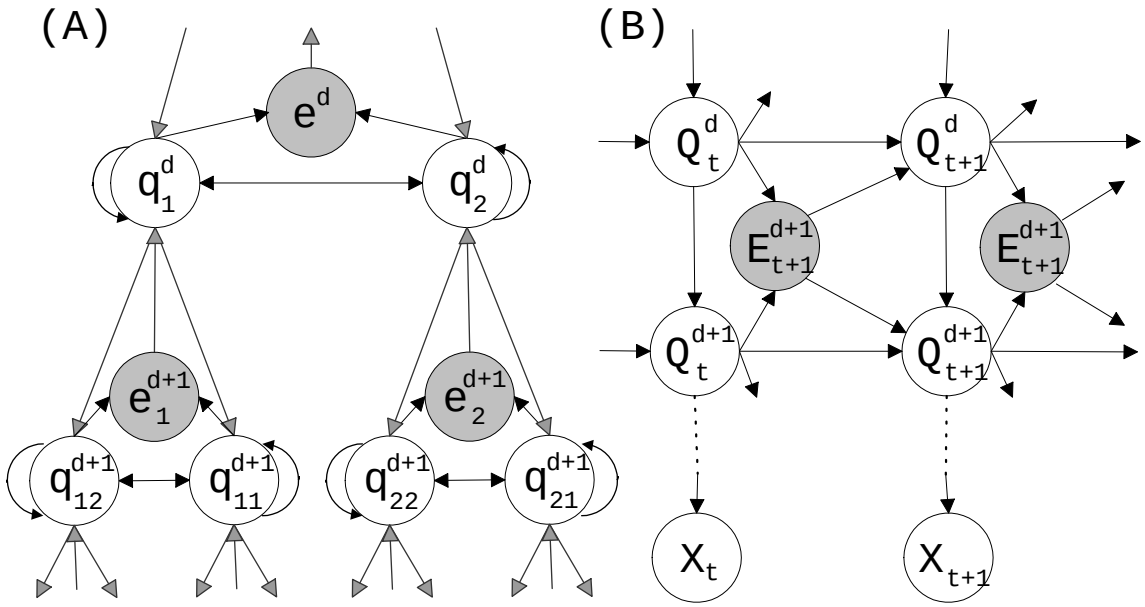

Figure 1: Graphical HHMM representation at level d and $d+1$ (A)Tree-structured representation; (B)DBN representations, with observations X_t draw at the bottom. Uppercase letters denote the states as random variables in time t, lowercase letters denote the state-space of HHMM, i.e. values these random variables can take in any time slice. Shaded nodes are auxiliary *exit* nodes that turns on the transition at a higher level, a state at level d is not allowed to change unless the corresponding $E^{d+1} = 1$.

The original work have also shown that the state inference can be done in $O(T^3)$ where T is the sequence length [FST98]. Equivalently, HHMM can be unrolled in time and represented as a Dynamic Bayesian Network (DBN) [MP01] (Figure 1(B)). In the DBN representation, the hidden states Q_t^d at each level $d = 1, \ldots D$, the observation sequence X_t, and the auxiliary *level-exiting* variables E_t^d completely specifies the state of the model at time t. Note E_t^d can be turned on only if all lower levels of $E_T^{d+1:D}$ are on. It is also shown that inference is empirically $O(T)$ using the junction tree algorithm for DBNs.

Prior applications of HHMM falls into two categories: (1) Supervised learning where manually segmented training data is available, hence each sub-HMM are learned separately on the segmented sub-sequences, and cross-level transitions and learned on the transition statistic across the subsequences. For example, extron/intron recognition in DNA sequences [HIS+00] and action recognition [IB00], and more examples summarized in [Mur01] falls into this category. (2) Unsupervised learning, where segmented data are not available for training, and parameters of different levels are jointly learned; (3)A mixture of the above, where alignment at a higher level is given, yet parameters still needs to be estimated across several levels. Existing instances of (2) is rare, yet (3) can be regarded as a generalization of (1) and (2), and examples abound: the celebrated application of building a speech recognition system with word-level annotation [Dep], text parsing and handwriting recognition [FST98].

4 Learning HHMM with EM

In this section, we first introduce convenient notations to represent the states and parameter set of an HHMM, followed by a brief overview on deriving the EM algorithm for HHMMs, and then we present the EM algorithms that compute the state inference and parameter estimation in $O(T)$.

4.1 Representing an HHMM

Denote the maximum state-space size of any sub-HMM as Q, we use the *bar notation* (equation1) to write the entire configuration of the hierarchical states from the top (level 1) to the bottom (level D) with a Q-ary D-digit integer, with the lowest-level states at the least significant digit:

$$k^{(D)} = q_{1:D} = \overline{(q_1 q_2 \dots q_D)} = \sum_{i=1}^{D} q_i \cdot Q^{D-i} \tag{1}$$

Here $1 \le q_i \le Q; i = 1, \dots, d$. We drop the superscript of k where there is no confusion, the whole parameter set Θ of an HHMM then consists of (1) Markov chain parameters λ^d in level d indexed by the state configuration $k^{(d-1)}$, i.e., transition probabilities A_k^d, prior probabilities π_k^d, and exiting probabilities from the current level e_k^d; (2) emission parameters B that specifies the distribution of observations conditioned on the state configuration, i.e., the means μ_k and covariances σ_k when emission distributions are Gaussian.

$$\begin{aligned} \Theta &= (\bigcup_{d=1}^{D} \{\lambda^d\}) \cup \{B\} \\ &= (\bigcup_{d=1}^{D} \bigcup_{i=1}^{Q^{d-1}} \{A_i^d, \pi_i^d, e_i^d\}) \cup (\bigcup_{i=1}^{Q^D} \{\mu_i, \sigma_i\}) \end{aligned} \tag{2}$$

4.2 Outline of the EM algorithm

Denote Θ the old parameter set, $\hat{\Theta}$ the new (updated) parameter set, then maximizing the data likelihood L is equivalent to iteratively maximizing the expected value of the complete-data log-likelihood Q:

$$\begin{aligned} Q(\hat{\Theta}, \Theta) &= E[\log(P(Q_1^T, X_1^T|\hat{\Theta}))|X_1^T, \Theta] && (3) \\ &= \sum_{Q_1^T} P(Q_1^T|X_1^T, \Theta) \log(P(Q_1^T, X_1^T|\hat{\Theta})) && (4) \\ &= L^{-1} \sum_{Q_1^T} P(Q_1^T, X_1^T|\Theta) \log(P(Q_1^T, X_1^T|\hat{\Theta})) && (5) \end{aligned}$$

Specifically, the "E" step evaluates this expectations, and the "M" step finds the value of $\hat{\Theta}$ that maximizes this expectation. So if we choose a proper hidden state space, and write out the

summation in equation (5) properly, then each of the unknowns will be delineated into separate summation term. Then when we take the partial derivative of eq. (5), each unknowns can be solved in closed form.

In order for the summation in eq. (5) to be separable for HHMMs, we need to define hyper-initial probabilities $\tilde{\pi}(k)$ and transition probabilities $\tilde{a}(k', k, d)$

$$\pi(k) = \prod_{d=1}^{D-1} \pi_{q_d}^d \tag{6}$$

$$\tilde{a}(k', k, d) = \prod_{i=D-1}^{d} e_{q_i'}^i \pi_{q_i}^i \cdot a^d(q_d', q_d) \tag{7}$$

The reason why we need an extra parameter d instead of just writing $\tilde{a}(k', k)$ is that we want $\tilde{a}$ to be in product-of-parameters form so when we take the logarithm in equation 5 it becomes sum-of-individual-parameters and only then the unknowns can be delineated when we take the partial derivative.

4.3 The forward-backward algorithm

Similar to the inference of HMMs, we define forward variables $\alpha_t(k)$ as the probability of the observations up to time t and HHMM in state k under the current model Θ(equation 8). And the data likelihood L can be computed iteratively using the forward variables, as shown in equations (9)-(11).

$$\alpha_t(k) \triangleq P(X_{1:t}, Q_t = k|\Theta) \tag{8}$$

Initialization:
$$\alpha_1(k) = \tilde{\pi}_k b_k(x_1); \quad k = 1, \ldots, Q^D \tag{9}$$

Iteration:
$$\alpha_{t+1}(k) = b_k(x_{t+1}) \textstyle\sum_{k'} \alpha_t(k') \tilde{a}(k', k, d); \quad t = 1, \ldots, T-1; k = 1, \ldots, Q^D \tag{10}$$

Termination:
$$L = P(X_{1:T}|\Theta) = \textstyle\sum_k \alpha_T(k) \tag{11}$$

Similarly, we can define and compute the backward variables $\beta_t(k)$, and they will become useful in parameter estimation.

$$\beta_t(k) \triangleq P(X_{t+1:T}|Q_t = k, \Theta) \tag{12}$$

Initialization:
$$\beta_T(k) = 1; \quad k = 1, \ldots, Q^D \tag{13}$$

Iteration:
$$\beta_t(k) = \sum_{k'} \sum_d \beta_{t+1}(k') b_{k'}(x_{t+1}) \tilde{a}(k, k', d); \quad t = T-1, \ldots, 1; k = 1, \ldots, Q^D \tag{14}$$

$$\tag{15}$$

4.4 The Viterbi algorithm

The Viterbi algorithm for decoding the optimum state sequence is very similar to the forward algorithm. The changes involved are: 1. Replace the summation with maximum in (10); 2. Keep track of the backpointers $\psi_t(k) = (k^*, e^*)$ where k^* is the "best-likelihood" states at $t-1$ that leads to state k at time t, and e^* is the corresponding *exit level* taking values from D to 2; 3. Do back tracing after the forward path, pick the "best-likelihood" from time T, and trace back to $t = 1$ according to ψ, recover the optimum path $(Q_t^*, E_t^*)_{t=1}^T$.

Initialization:
$$\psi_1(k) = 0; \delta_1(k) = \hat{\pi}_k b_k(x_1); \quad k = 1, \ldots, Q^D \tag{16}$$

Iteration:
$$\psi_{t+1}(k) = \arg \max_{\substack{1 \le k' \le Q^D \\ 1 \le d \le D}} \delta_t(k') \hat{a}(k', k, d) \tag{17}$$

$$\delta_{t+1}(k) = b_k(x_{t+1}) \cdot \max_{\substack{1 \le k' \le Q^D \\ 1 \le d \le D}} \delta_t(k') \hat{A}(k', k, d); \quad t = 1, \ldots, T-1; k = 1, \ldots, Q^D \tag{18}$$

Back Trace:
$$Q_T^* = \arg\max_k(\delta_T(k)); \quad E_T^* = 1; \tag{19}$$

$$(Q_t^*, E_t^*) = \psi_{t+1}(Q_{t+1}^*); \quad t = T-1, \ldots, 1; \tag{20}$$

4.5 Parameter estimation

Parameter learning for HHMM is done by maximizing the model likelihood given the observation sequence $X_{1:T}$, using the EM algorithm in the augmented hidden state-space $\{Q_t, E_t\}_{t=1}^T$ We define the state-occupancy variables $\gamma_t(k)$ and the auxiliary transition variables $\xi_t(k', k, d)$ for this purpose:

$$\begin{aligned} \gamma_t(k) &\triangleq P(Q_t = k | X_1^T, \Theta) \\ &= L^{-1} \cdot \alpha_t(k) \beta_t(k); \quad {\scriptstyle t=1,\ldots,T} \end{aligned} \tag{21}$$

$$\begin{aligned} \xi_t(k', k, d) &\triangleq P(Q_t = k', Q_{t+1} = k, F_t^{1:d} = 0, F_t^{d+1:D} = 1 | X_{1:T}, \Theta) \\ &= L^{-1} \alpha_t(k') \hat{A}(k', k) b_k(x_{t+1}) \beta_{t+1}(k); \quad {\scriptstyle t=1,\ldots,T-1} \end{aligned} \tag{22}$$

Then the parameter estimates can be obtained by marginalizing and normalizing the corresponding auxiliary variables. Estimation for parameters in equation (2) are as follows.

$$\text{Prior probability} \qquad \hat{\pi}_q^d(j) = \frac{\sum_{t=1}^{T-1} \sum_{q'} \sum_{q''} \sum_{i} \xi_t(\overline{(qiq')}, \overline{(qjq'')}, d)}{\sum_{t=1}^{T-1} \sum_{q'} \sum_{q''} \sum_{i} \sum_{j} \xi_t(\overline{(qiq')}, \overline{(qjq'')}, d)} \tag{23}$$

$$\text{Within-level transition probability} \qquad \hat{a}_q^d(i,j) = \frac{\sum_{t=1}^{T-1} \sum_{q'} \sum_{q''} \xi_t(\overline{(qiq')}, \overline{(qjq'')}, d)}{\sum_{t=1}^{T-1} \sum_{q'} \sum_{q''} \sum_{j} \xi_t(\overline{(qiq')}, \overline{(qjq'')}, d)} \tag{24}$$

$$\text{Level-exiting probability} \qquad \hat{e}_q^d(i) = \frac{\sum_{t=1}^{T-1} \sum_{q'} \sum_{k'} \xi_t(\overline{(qiq')}, k', d)}{\sum_{t=1}^{T} \gamma_t(k)} \tag{25}$$

$$\tag{26}$$

Note for the $d-1$-digit Q-ary number q, the constraint $q = Q_t^{1:d-1} = Q_{t+1}^{1:d-1}$ is inherent since we are only interested in the transitions made at level d, and $q' = Q_t^{d+1:D}$ and $q'' = Q_{t+1}^{d+1:D}$ are the states at levels lower than d. Also note the temporal dimension of γ and ξ are always marginalized out, so we only compute $\tilde{\gamma}(k) = \sum_t \gamma_t(k)$ and $\tilde{\xi}(k, k', d) = \sum_t \xi_t(k, k', d)$ in implementation and use them instead.

Assume each observation x_t is a row vector, then the means and covariances of state k at the bottom level can be estimates as:

$$\text{The mean} \qquad \hat{\mu}_k = \frac{\sum_{t=1}^{T} x_t \cdot \gamma_t(k)}{\sum_{t=1}^{T} \gamma_t(k)} \tag{27}$$

$$\text{The covariance} \qquad \hat{\Sigma}_k = \frac{\sum_{t=1}^{T} x_t^T x_t \cdot \gamma_t(k)}{\sum_{t=1}^{T} \gamma_t(k)} \tag{28}$$

4.6 Complexity of this algorithm

Complexity of this algorithm is in fact $O(T \cdot Q^{2D})$, since:

1. The size of the collapsed state-space of an HHMM is at most Q^D.

2. Each round of the forward (or Viterbi or backward) iterations involves going through each of the T time slices; and for each of the Q^D states in each time slice, the likelihood (or best incoming path or backward probability) is obtained by going through all possible previous (or the next) node. Hence the complexity of each round is $O(T \cdot Q^{2D})$.

3. Parameter estimation takes constant times forward/backward iterations, while computing likelihood and Viterbi decoding takes once and twice forward iterations, respectively. Hence the complexity of the EM algorithm is still $O(T \cdot Q^{2D})$, although the multiplicative constant does make a difference in implementation.

5 Bayesian model adaptation

In the HHMM learning algorithms described in Section 4, EM is known to converge to a local maxima of data likelihood, and the model size is predefined. And since searching for a global maxima in the likelihood landscape or searching through all possible model structure is intractable, we adopt randomized search strategies to address these issues. Markov chain Monte Carlo(MCMC) is a class of such algorithms that has been successfully used to solve high-dimensional optimization problems, especially the problem of Bayesian learning of model structure and model parameters [AdFDJ03]. Moreover, model selection can also be addressed in the same framework with reverse-jump MCMC (RJ-MCMC) [Gre95], by constructing reversible moves between parameter spaces of different dimensions. In particular, Andrieu et.al. [AdFD01] applied RJ-MCMC to the learning of radial basis function (RBF) neural networks by introducing birth-death and split-merge moves to the RBF kernels. This is similar to our case of learning variable number of Gaussians in the feature space that correspond to the emission probabilities. In this work, we deployed a MCMC scheme to learn the correct state-space size of the HHMM models, where the major random steps are jumping between different number of Gaussians, different number of upper-level states, and swapping the state association relationships of two children states.

5.1 Overview of MCMC for HHMM

MCMC for learning statistical models usually iterates between two steps: (1)The proposal step comes up with a new set of structure and model parameters based on the data and the current model(the *Markov chain*) according to certain *proposal distributions* (*Monte Carlo*); (2)The decision step computes acceptance probability α of the proposed new model using model posterior and proposal strategies, and then this proposal is *accepted* or *rejected* with probability α. MCMC will converge to the global optimum *in probability* if certain constraints [AdFDJ03] are satisfied

for the proposal distribution and if the acceptance probability are evaluated accordingly, yet the speed of convergence largely depends on the *goodness* of the proposals.

Model adaptation for HHMMs is choreographed as follows: (1)Based on the current model Θ, compute a probability profile $P_\theta = [p_{em}, p_{sw}, p_{st}, p_{sb}, p_{mt}, p_{mb}]$, then propose a move among the types {*EM, swap, split-top, split-bottom, merge-top, merge-bottom*} according to the profile P_θ. *EM* is regular parameter update; *Swap* involves swapping the parents of two lower level states associated with different higher-level nodes; *split/merge-bottom* means splitting the emission probability of one of the current bottom level states or merging two of them into one; and *split-top* would randomly partition one higher level state into two and assign its children to either one of the new high-level state, while *merge-top* would collapse two higher-level states into one. (2)Acceptance probability is then evaluated based on model posterior, computed with the Bayesian Information Criteria; for *split* and *merge*, the proposal likelihood and model space alignment also need to be taken into account.

Note we are using a mixture of the EM and MCMC, in place of full Monte Carlo update of the parameter set and the model, since EM is more efficient than full MCMC, and the convergence behavior doesn't seem to suffer in practice. In the following subsections, we will present the model adaptation algorithm in more detail.

5.2 The Model adaptation algorithm

In HHMM, model adaptation also involves similar moves such as the split/merge of Gaussian kernels that associates states in the lowest level with observations, we are not including birth/death moves since these moves can be reached with multiple moves of split/merge. Due to the multi-level structure of HHMM, however, we will need an additional *swap* move that changes the association of lower level states with its higher level parents, and split/merges of parent states that also incur appropriate actions in their children states. Here are the main steps:

1. Initialization, choose maximum sub-model size Q. set the entire set of model parameters Θ_K (with clustering and heuristic grouping).

2. At iteration i, as there are Q Gaussian kernels at the lower level. Use equations (29)–(34) to compute proposal probabilities $P_\theta = [p_{em}, p_{sw}, p_{st}, p_{sb}, p_{mt}, p_{mb}]$ for {*EM, swap, split-top, split-bottom, merge-top, merge-bottom*} respectively, according to preset simulation parameters: c^*, the maximum probability of proposing a particular type of move other than EM; $\hat{Q}_0$ and $\hat{Q}_1$ as the hyper-parameter for priors on the number of states at the higher and lower levels, respectively. Denote the current total number of states in the higher and lower levels as Q_0 and Q_1, respectively.

$$p_{sb} \quad = \quad c^* \cdot \min\{1, \hat{Q}_1/(Q_1+1)\}; \tag{29}$$

$$
\begin{aligned}
p_{st} &= c^* \cdot \min\{1, \hat{Q}_0/(Q_0+1)\}; && (30)\\
p_{mb} &= c^* \cdot \min\{1, (Q_1+1)/\hat{Q}_1\}; && (31)\\
p_{mt} &= c^* \cdot \min\{1, (Q_0-1)/\hat{Q}_0\}; && (32)\\
p_{sw} &= c^*; && (33)\\
p_{em} &= 1-(p_{sw}+p_{st}+p_{sb}+p_{mt}+p_{mb}). && (34)
\end{aligned}
$$

(a) propose a move type from {*EM, swap, split-top, split-bottom, merge-top, merge-bottom*} according to probability profile $P_\theta = [p_{em}, p_{sw}, p_{st}, p_{sb}, p_{mt}, p_{mb}]$;

(b) Update the model size and the parameter set according to Section 5.3;

(c) Compute the acceptance ratio r_k according to 5.4, and accept/reject this move with probability $\alpha_k = min\{1, r_k\}$.

3. Stop if converged, otherwise go to step 2

5.3 Computing different moves

EM is one regular hill-climbing iteration as described in section 4; and once a move type other than EM is selected, one (or two) states at a certain level are selected at random for swap/split/merge, and the parameters are modified accordingly:

- Swap the association of two states:
  Choose two states from the same level, each belongs to a different higher-level state, swap their higher-level association and the corresponding children states.

- Split a bottom state:
  Choose a state at random, perturb its mean as follows, where u_s $U[0,1]$, and η is a simulation parameter that ensures reversibility between split and merge moves.

$$
\begin{aligned}
\mu_1 &= \mu_0 + u_s\eta\\
\mu_2 &= \mu_0 - u_s\eta
\end{aligned} \qquad (35)
$$

- Merge two bottom states:
  Within the same higher-level states, choose two states at random, merge them by assigning the new mean as the average of the two.

$$
\mu_0 = \frac{\mu_1+\mu_2}{2}, \quad \text{if } |\mu_1-\mu_2| \le 2\eta \qquad (36)
$$

- Split one higher-level state (that has more than one children):
  Choose the state to split at random, split its children into two disjoint sets at random, update the corresponding multi-level Markov chain parameters accordingly.

- Merge two higher-level states:
  Randomly choose two states at the same level (that also belong to the same parents), merge them by making all the children of these two states the children of the merged state, and modify the transition probabilities according.

5.4 Computing the acceptance ratio

When moves are proposed to a space with identical size as the original, such as EM or state-swapping, the acceptance ratio is calculated as the ratio of posteriors after and before the move. When we use Bayesian information criteria (BIC, Eq. 37) [Sch78] as the posterior, the acceptance ratio simplifies into likelihood ratio.

$$BIC = \log(P(x|\Theta)) \cdot \lambda + \frac{1}{2}|\Theta| \log(T) \tag{37}$$

Intuitively, BIC is a trade-off between data likelihood $P(X|\Theta)$ and model complexity $|\Theta| \cdot \log(T)$ with weighting factor λ. Larger models are penalized by the number of free parameters in the model $|\Theta|$; yet the influence of the model penalty decreases as the amount of training data T increases, since $\log(T)$ grows slower than $O(T)$. The weighting factor λ is taken as $1/16$ in the simulations of this section as well as those in section 6.

$$r \stackrel{\triangle}{=} \text{(posterior ratio)} = \frac{exp(\widehat{BIC})}{exp(BIC)} = \frac{P(x|\hat{\Theta}_k)}{P(x|\Theta_k)} \tag{38}$$

When moves are proposed to a parameter space with different dimension, such as split or merge, we also need a proposal ratio term and a Jacobian term to align the spaces, and ensure detailed balance [Gre95], as shown in equations (39)–(42).

$$r_k \stackrel{\triangle}{=} \text{(posterior ratio)} \cdot \text{(proposal ratio)} \cdot \text{(Jacobian)} \tag{39}$$

$$r_{split} = \frac{P(k+1, \Theta_{k+1}|x)}{P(k, \Theta_k|x)} \cdot \frac{m_{k+1}/(k+1)}{p(u_s)s_k/k} \cdot J \tag{40}$$

$$r_{merge} = \frac{P(k, \Theta_k|x)}{P(k+1, \Theta_{k+1}|x)} \cdot \frac{p(u_s)s_{k-1}/(k-1)}{m_k/k} \cdot J^{-1} \tag{41}$$

$$J = \left|\frac{\partial(\mu_1, \mu_2)}{\partial(\mu_0, u_s)}\right| = \begin{vmatrix} 1 & \eta \\ 1 & -\eta \end{vmatrix} = 2\eta \tag{42}$$

6 Feature selection for unsupervised learning

Feature extraction schemes for audio-visual streams abound, and we are usually left with a large pool of diverse features without knowing which ones are relevant to the concepts in the data. A few features can be selected manually if expert domain knowledge exits, but more often we lack adequate domain knowledge, or the connection between high-level expert knowledge and low-level

features are not obvious. Moreover, the task of feature selection is divided into eliminating *irrelevant* features and *redundant* ones, where the former may disturb the classifier and degrade classification accuracy, the latter adds to computational burden without bringing in new information. Furthermore, in the unsupervised structure discovery scenario, different subsets of features may well represent different concepts, and they should be described with separate models rather than modelled jointly.

Hence the scope of our problem, is to select structurally relevant and compact feature subset that fits the HHMM model assumption in unsupervised learning over temporally highly correlated data streams.

6.1 The feature selection algorithm

Denote the feature pool as $F = \{f_1, \ldots, f_D\}$, the data sequence as $\mathbf{X}_F = X_F^{1:T}$, the feature selection algorithm proceeds through these general steps:

1. (Let $i = 1$ to start with) At the i-th round, produce a *reference set* $\tilde{F}_i \subseteq F$ at random, learn HHMM $\tilde{\Theta}_i$ on $\tilde{F}_i$ with model adaptation, perform Viterbi decoding of $\mathbf{X}_{\tilde{F}_i}$, get the *reference state-sequence* $\tilde{\mathbf{Q}}_i = \tilde{Q}_F^{1:T}$.

2. For each feature $f_d \in F \setminus \tilde{F}_i$, learn HHMM Θ_d of size $|\tilde{\Theta}_i|$, get the Viterbi state sequence $\mathbf{Q}_d$ compute the information gain (sec. 6.2) of each feature on the $\mathbf{Q}_d$ with respect to the reference partition $\tilde{\mathbf{Q}}_i$. Find the subset $\hat{F}_i \subseteq (F \setminus \tilde{F}_i)$ with significantly large information gain, and keep the union of our *reference set* and the *relevance set* $\bar{F}_i \triangleq \tilde{F}_i \cup \hat{F}_i$ for further processing.

3. Use Markov blanket filtering in sec. 6.3, eliminate redundant features within the set $\bar{F}_i$ whose Markov blanket exists. We're then left with a relevant and compact feature subset $F_i \subseteq \bar{F}_i$. Learn HHMM Θ_i again with model adaptation on X_{F_i}.

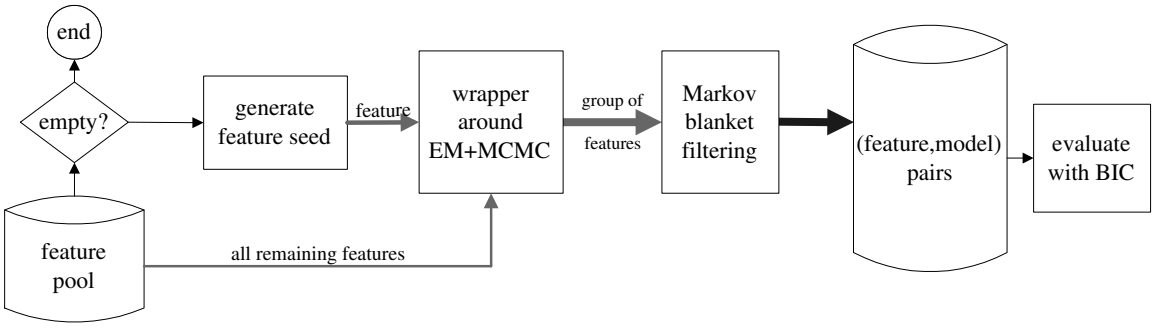

Figure 2: Feature selection algorithm overview

4. Eliminate the previous candidate set by setting $F = F \setminus \bar{F}_i$; go back to step 1 with $i = i+1$ if F is non-empty.

5. For each feature-model combination $\{F_i, \Theta_i\}_i$, evaluate their *fitness* using the normalized BIC criteria in sec. 6.4, rank the feature subsets, and interpret the meanings of the resulting clusters.

6.2 Evaluating information gain

Information gain [XK01] measures the degree of *agreement* of each feature to the reference partition. We label a partition Q of the original set $\mathbf{X}_F = X_F^{1:T}$ as integers $Q_F^t \in \{1, \ldots, N\}$, let the probability of each part be the empirical portion (eq. 43), and define similarly the conditional probability of the reference partition Q_0 given the partition Q_f induced by a feature f (eq.44). The information gain of feature f with respect to Q^0 is defined as eq. 45.

$$P_Q(i) = \frac{|\{t|q_t = i\}|}{T}; \quad i = 1, \ldots, N \tag{43}$$

$$P_{Q_0|Q_f}(i \mid j) = \frac{|\{t|(q_0^t, q_f^t) = (i,j)\}|}{|\{t|q_f^t = j\}|}; \quad i,j = 1, \ldots, N \tag{44}$$

$$I_g \quad \triangleq \quad H(P_{Q_0}) - \sum_f P_{Q_f} \cdot H(P_{Q_0|Q_f}) \tag{45}$$

Where $H(\cdot)$ is the entropy function. Intuitively, a higher information gain value for feature f suggests that the f-induced partition Q_f is more consistent with the reference partition Q_0.

6.3 Finding a Markov Blanket

After the previous wrapper step, we are left with a subset of features with consistency yet possible redundancy. A feature f is said to be redundant if the partition of the data set is independent to f given its *Markov Blanket* F_M. In prior works [KS96, XK01], Markov blanket is identified with the equivalent condition that the expected KL-divergence between class-posterior probabilities with or without f should be zero.

For unsupervised learning over a temporal stream however, this criteria cannot be readily employed since the temporal correlation prevents us from estimating the posterior distributions by just counting over every feature-label pair. Thus results in two difficulties: (1)The dependency between adjacent observations and class-labels makes the distribution of features and posterior distribution of classes multi-dimensional, and summing over them quickly becomes intractable; (2)We will not have enough data to estimate these high-dimensional distributions. We therefore

use an alternative necessary condition that the optimum state-sequence $C_{1:T}$ should not change conditioned on observing $F_M \cup f$ or F_M only. Additionally, as few if any features will have a Markov Blanket of limited size in practice, we sequentially remove features that induces the least change in state sequence given the change is small enough ($< 5\%$).

Note the sequential removal will not cause divergence of the resulting set [KS96]; and this step is a filtering step since we do not need to retrain the HHMMs for each $F_M \cup f$, Viterbi decoding on only the dimensions of interest would suffice.

6.4 Normalized BIC

Iterating over section 6.2 and section 6.3 results in disjoint small subsets of features $\{F_i\}$ that are compact and consistent with each other. The HHMM models $\{\Theta_i\}$ learned over these subsets are best-effort fits on the features, yet the $\{\Theta_i\}$s may not fit the multi-level Markov assumptions in section 2.

There are two criteria proposed in prior work [DB00], scatter separability and maximum likelihood (ML). Note the former is not suitable to temporal data since multi-dimensional Euclidean distance does not take into account temporal dependency, and it is non-trivial to define another proper distance measure for temporal data; while the latter is also known [DB00] to be biased against higher-dimensional feature sets. We use a normalized BIC criteria(eq. 46) as the alternative to ML, which trades off normalized data likelihood $\tilde{L}$ with model complexity $|\Theta|$. Note the former has weighting factor λ in practice; the latter is modulated by the total number of sample values $\log(DT)$; and $\tilde{L}$ for HHMM is computed in the same forward-backward iterations, except all the emission probabilities $P(X|Q)$ are replaced with $P'_{X,Q} = P(X|Q)^{1/D}$, i.e. normalized with respect to data dimension D, under the *naive-Bayes* assumption that features independent given the hidden states.

$$\widetilde{BIC} = \tilde{L} \cdot \lambda - \frac{1}{2}|\Theta| \log(DT) \tag{46}$$

Initialization and convergence issues exist in the iterative partitioning of the feature pool. The strategy for producing the random *reference set* $\tilde{F}_i$ in step (1) affects the result of feature partition, as different $\tilde{F}_i$ may result in different final partitions. If the dimension of $\tilde{F}_i$ is too low for example, the resulting structure may not be significant and it tends to result in many small feature clusters; on the other hand, if $\tilde{F}_i$ is too large, structures may become too complex, feature subsets maybe too few, and the the result will be hard to interpret.

7 Experiments and Results

Two TV soccer video clips taken from mpeg-7 CD are used to test the algorithm, clip *Korean* is 25 minutes long, 320x240 resolution, 29.97 frames per second; clip *Spain* is 15 minutes long at 352x288, 25 frames per second.

The evaluation basis for the unsupervised structure algorithm, including learning HHMM parameters and structure (section 3 and 5) are two semantic events labelled are *play* and *break* [XCDS02], defined according to the rules of soccer game. These two events are dense since they covers the whole time scale of the video, and distinguishing *break* from *play* will be useful for the viewers since *break* takes up about 40% of the screen time. Two manually selected features, dominant color ratio and motion intensity [XCDS02], uniformly sampled from the video stream every 0.1 seconds, are used.

Here we compare the learning accuracy of four different learning schemes against the ground truth. (1)Supervised HMM [XCDS02]: One HMM per semantic event is trained on manually segmented chunks; and then the video data with unknown boundary are first chopped into 3-second segments, where the data likelihood of each segment is evaluated with each of the trained HMMs; and final segmentation boundary is obtained after a dynamic programming step taking into account the model likelihoods and the transition likelihoods of the short segments from the segmented data. (2)Supervised HHMM: Individual HMMs at the bottom level of the hierarchy are learned separately, essentially using the models trained in (1); across-level and top level transition statistics are also obtained from segmented data; and then segmentation is obtained by decoding the Viterbi path from the hierarchical model on the entire video stream. (3)Unsupervised HHMM without model adaptation: An HHMM is initialized with known size of state-space and random parameters; the EM algorithm is used to learn the model parameters; and segmentation is obtained from the Viterbi path of the final model. (4)Unsupervised HHMM with model adaptation: An HHMM is initialized with arbitrary size of state-space and random parameters; the EM and RJ-MCMC algorithms are used to learn the size and parameters of the model; state sequence is obtained from the converged model with optimal size.

For algorithms (1)-(3), the model size is set to the optimal model size that algorithms (4) converges to, i.e. 6-8 bottom-level states per event. For supervised algorithms (1) and (2), K-means clustering and Gaussian mixture fitting is used to randomly initialize the HMMs; for unsupervised algorithms(3)(4), inital bottom-level HMMs are obtained with K-means and Gaussian fitting followed by a grouping algorithm based on temporal proximity. We run each algorithm for 15 times with random start, and compute the per-sample accuracy against manual labels. The median and semi-interquartile range [2] across multiple rounds are listed in table 7.

[2]Semi-interquartile as a measure of the spread of the data, is defined as half of the distance between the 75th and 25th percentile, it is more robust to outliers than standard deviation.

no.	super-vised?	model type	adap-tation?	accuracy	
				median	siq [2]
(1)	Y	HMM	N	75.5%	1.8%
(2)	Y	HHMM	N	75.0%	2.0%
(3)	N	HHMM	N	75.0%	1.2%
(4)	N	HHMM	Y	75.7%	1.1%

Table 1: Evaluation of learning results against ground truth using learning schemes (1)-(4) on clip *Korea*

Results showed that the unsupervised learning achieved very close results as the supervised learning case, this is quite surprising since the unsupervised learning of HHMMs is not tuned to the particular ground-truth. Yet the reason for this performance match can be attributed to the carefully selected feature set that well represents the events. Also note the comparison basis using supervised learning is actually a loose bound since unlike [XCDS02], the HMMs are learning and evaluated on the same video clip and results reported for (1)(2) are actually training accuracies.

The feature selection algorithm is tested on both *Korea* and *Spain*. A nine-dimensional feature vector sampled at every 0.1 seconds are taken as the initial feature pool, this include: Dominant Color Ratio (DCR) and Motion Intensity (MI), the least-square estimates of camera translation (MX, MY), and five audio features - Volume, Spectral roll-off (SR), Low-band energy (LE), High-band energy (HE), and Zero-crossing rate (ZCR). We run the feature selection + model learning algorithm on each video stream for five times, with one randomly selected initial *reference feature*. After eliminating degenerate cases such as there are only one feature in the resulting set, we look at the feature-model pair that has the largest *Normalized BIC* value as described in section 6.4.

For clip *Spain*, the selected feature set is {DCR, Volume}, there are two high-level states in the HHMM, each with five lower-level children. Evaluation against the *play/break* labels showed 74.8% accuracy. For clip *Korea*, the selected feature set is {DCR, MX}, with three high-level states and {7, 3, 4} children states respectively. If we assign each of the three clusters the majority ground-truth label it corresponds to (which would be {*play, break, break*} respectively), per-sample accuracy would be 74.5%. This three-cluster results actually matches the previous results [XCDS03] with fixed two clusters and manually-selected feature set {DCR, MI}, since the horizontal camera panning contribute to a majority of the whole motion intensity in soccer video, especially when the camera is tracking the ball movement in wide angle. The accuracies are comparable to their previous counterparts [XCDS03] without varying the cluster order or the feature set (75%). Yet the small discrepancy may due to:(1) Variability in EM, or the algorithm is yet to converge when maximum iteration is reached; (2)Possible inherent bias may still exist in equation 37 although we are using the same λ value for both algorithms.

8 Conclusion

In this paper we proposed algorithms for unsupervised discovery of structure from video sequences. We model the class of dense, stochastic structures in video using hierarchical hidden Markov models, the models parameters and model structure are learning using EM and Monte Carlo sampling techniques, and informative feature subsets are automatically selected from a large feature pool. When evaluated on a TV soccer clip against manually labelled ground truth, we achieved comparable results as supervised learning counterpart. Many open issues in stochastic structure discovery using HHMM remains, however: we would like to see the fitness of this model applied to other video domains; and it is very desirable to model sparse events within the current framework as well.

References

[AdFD01] Christophe Andrieu, Nando de Freitas, and Arnaud Doucet. Robust full bayesian learning for radial basis networks. *Neural Computation*, 13:2359–2407, 2001.

[AdFDJ03] Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, special issue on MCMC for Machine Learning, to appear 2003.

[DA01] Arnaud Doucet and C. Andrieu. Iterative algorithms for optimal state estimation of jump Markov linear systems. *IEEE Transactions of Signal Processing*, 49:1216–1227, 2001.

[DB00] Jennifer G. Dy and Carla E. Brodley. Feature subset selection and order identification for unsupervised learning. In *Proc. 17th International Conf. on Machine Learning*, pages 247–254. Morgan Kaufmann, San Francisco, CA, 2000.

[Dep] Cambridge University Engineering Department. Hidden Markov model toolkit (HTK). http://htk.eng.cam.ac.uk/.

[FST98] Shai Fine, Yoram Singer, and Naftali Tishby. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.

[Gre95] Peter J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[HIS+00] Mike Hu, Chris Ingram, Monica Sirski, Chris Pal, Sajani Swamy, and Cheryl Patten. A hierarchical HMM implementation for vertebrate gene splice site prediction. Technical report, Dept. of Computer Science, University of Waterloo, 2000.

[IB00] Yuri A. Ivanov and Aaron F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transaction of Pattern Recognition and Machines Intelligence*, 22(8):852–872, August 2000.

[ISZ99] Arun Iyengar, Mark S. Squillante, and Li Zhang. Analysis and characterization of large-scale web server access patterns and performance. *World Wide Web*, 2(1-2):85–100, 1999.

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[KS96] Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *International Conference on Machine Learning*, pages 284–292, 1996.

[LAB+93] Charles E. Lawrence, Stephen F. Altschul, Mark S. Boguski, Jun S. Liu, Andrew F. Neuwald, and John C. Wootton. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 8(262):208–14, October 1993.

[MP01] Kevin Murphy and Mark Paskin. Linear time inference in hierarchical HMMs. In *Proceedings of Neural Information Processing Systems*, Vancouver, Canada, 2001.

[Mur01] Kevin Murphy. Representing and learning hierarchical structure in sequential data. Unpublished manuscript, at http://www-anw.cs.umass.edu/ cs691t/SS02/readings.html, 2001.

[NH02] Milind Naphade and Thomas Huang. Discovering recurrent events in video using unsupervised methods. In *Proc. Intl. Conf. Image Processing*, Rochester, NY, 2002.

[Rab89] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, February 1989.

[Sch78] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 7:461–464, 1978.

[SZ99] Emile Sahouria and Avideh Zakhor. Content anlaysis of video using principal components. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(9):1290–1298, December 1999.

[WLH00] Yao Wang, Zhu Liu, and Jincheng Huang. Multimedia content analysis using both audio and visual clues. *IEEE Signal Processing Magazine*, 17(6):12–36, November 2000.

[XCDS02] Lexing Xie, Shih-Fu Chang, Ajay Divakaran, and Huifang Sun. Structure analysis of soccer video with hidden Markov models. In *Proc. Interational Conference on Acoustic, Speech and Signal Processing (ICASSP)*, Orlando, FL, 2002.

[XCDS03] Lexing Xie, Shih-Fu Chang, Ajay Divakaran, and Huifang Sun. Unsupervised discovery of multilevel statistical video structures using hierarchical hidden Markov models. Submitted for conference publication, Janurary 2003.

[XK01] Eric P. Xing and Richard M. Karp. Cliff: Clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. In *Proceedings of the Ninth International Conference on Intelligence Systems for Molecular Biology (ISMB)*, pages 1–9, 2001.

[YY96] Minerva Yeung and Boon-Lock Yeo. Time-constrained clustering for segmentation of video into story units. In *ICPR*, Vienna, Austria, 1996.