Model-based motion clustering using boosted mixture modeling

Vladimir Pavlovic
Dept. of Computer Science
Rutgers University
Piscataway, NJ 08854

Abstract

Model-based clustering of motion trajectories can be posed as the problem of learning an underlying mixture density function whose components correspond to motion classes with different statistical properties. We propose a general framework for boosted modeling of mixtures of parametric densities. A density is represented as a parametric mixture of kernels where mixture components are now being added recursively, one at a time, until a best fit to data occurs and an optimal number of mixture components is selected. Optimal ML and MAP solutions to this problem are found using the functional gradient techniques. Unlike traditional mixture modeling techniques, the new method does not rely on random parameter initialization and exhaustive exploration of varying model orders (such as the number of mixture components.) The method justifies parameter estimation of new mixture components independently of that of the rest of the mixture, thus allowing tractable use of complex kernels such as HMMs or switching linear dynamic models. The relationship to traditional parametric EM-based mixture modeling algorithms is established. We demonstrate the utility of the new algorithm on the problem of discovering motion sequence clusters. Our generative modeling framework has an important advantage over nonparametric approaches in that it can be used for classification as well as synthesis of the learned motion categories.

1. Introduction

Recent years have witnessed a tremendous growth in the amount of gathered and stored motion data. In parallel with the successes in basic object tracking and analysis techniques, a need has arisen for automatic grouping and organization of this data. Identification of important motion patterns and their modeling have become crucial for the analysis of this wealth of data in areas such as visual surveillance and monitoring. At the same time, the ability to obtain complex generative models of human or animal motion has opened new doors for the models' use in motion synthesis and computer animation.

In this paper, we focus on the problem of automatically finding clusters of similar object motions within a database of motion sequences as well as leering generative models of these clusters. Previous approaches to motion clustering have often relied on nonparametric clustering methods, e.g, []. Model-based approaches usually employed HMMs in k-means [16, 21, 28] or EM-based mixture learning frameworks [2]. However, the model-based methods often suffer from several problems. The EM learning with many complex kernels such as HMMs may be prohibitively expensive if it attempts to *simultaneously* optimize all cluster models. Moreover, it is often very sensitive to the initial choice of model parameters that may lead to poor clustering and cluster models. Finally, the approaches face a complex task of estimating the number of motion clusters. Solutions to this problem traditionally hinge on *independent* learning and *exhaustive* evaluation of models of different orders (c.f., [24]) or hierarchical pruning techniques. However, computational complexity again comes into play.

To address these problems, we propose a model-based clustering approach using a mixture modeling framework. We derive a *recursive* mixture estimation algorithm that uses the estimates of the simpler mixture models to infer or *boost* the ones of higher complexity. Parameters of the new component can be estimated *independently* of the parameters of the simpler models. Furthermore, the components are added until an optimal model order is reached.

2. Recursive maximum likelihood estimation algorithm for mixtures of densities

In this section we present a new recursive mixture modeling algorithm. The algorithm resembles the well-known boosting algorithms for classification of [26, 9]. We extend a similar formalism to the problem of recursive mixture density estimation.

Let X be the space of data generated by some unknown distribution D, and let $\{x_1, x_2, \ldots, x_M\}$ be a set of samples from D. Furthermore, let $f \in \mathcal{F}$ be a kernel function on X, and F be a mixture density represented as a convex combination of f:

$$F(x) = \sum_{k=1}^{K} w_k f_k(x), \quad \sum_{k=1}^{K} w_k = 1, \;\; w_k \geq 0.$$

Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)
1063-6919/04 $20.00 © 2004 **IEEE**

Our goal is to find the density F that minimizes the cost functional

$$J(F) = \sum_{i=1}^{M} -a_i \log F(x_i),$$

where a_i is a set of fixed weights. Without loss of generality we present the rest of this analysis with uniform weights $a_i = 1/M$. Assume F (of a certain order K) is known. We now want to find a new f such that when added to F, it decreases the cost J but keeps the new mixture in the convex subspace for some small $\varepsilon > 0$. In other words,

$$J((1-\varepsilon)F + \varepsilon f) < J(F), \tag{1}$$

Such an f then needs to be in a direction in the functional space that most decreases J. Hence, f has to be a direction of *negative functional gradient* of J, $f = -\nabla_F J(F) = \frac{1}{F}$. However, to guarantee that the new F still belongs to the space of convex combinations of kernels, it has to be constrained to the largest projection of $\nabla J(F)$ onto that space, which is equivalent to

$$\begin{aligned} f_k^* &= \arg\max_f < -\nabla_F J(F), f - F > \\ &= \arg\max_f < \frac{1}{F}, f - F > \\ &= \arg\max_f \frac{1}{M^2} \sum_{i=1}^{M} \frac{f(x_i) - F(x_i)}{F(x_i)} \\ &= \arg\max_f \sum_{i=1}^{M} \frac{f(x_i)}{F(x_i)} \end{aligned} \tag{2}$$

In the first order approximation, to decrease J, we choose f^* that maximizes (2). As long as $\left\langle f - F, \frac{1}{F} \right\rangle > 0$ the cost J will decrease with the addition of f^*. Once the optimal component f^* has been selected, the optimal combination coefficient can be determined as

$$w^* = \arg\min_w J((1-w)F + wf^*) \tag{3}$$

The general Algorithm 1 summarizes the above steps.

Equation 2 is a general form of a weighted maximum likelihood estimation and the resulting f^* is also known as a (weighted) M-estimator [12] for the sample set $\{x_1, \ldots, x_M\}$. Hence, it is important to note that f does not have to belong to a traditional family of distributions (such as Gaussian or gamma) but can be a more general robust function kernel [12].

More importantly, recursive Equation 2 reveals an intuitive and appealing way the "simpler" distribution F_{k-1} influences the estimation of the "new" kernel f_k. Samples with low probability F_{k-1} are given higher weights than the highly likely samples (according to current model F_{k-1}). Hence, the new kernel *focuses* on those samples that have been poorly explained by the simpler distribution F_{k-1}! Illustrative examples of this property will be shown in Section 5. We also note that similar property is exhibited by other additive models. For instance, boosted classifiers focus on samples that have been incorrectly classified by simpler models [26].

input : A set of samples $X = \{x_1, x_2, \ldots, x_M\}$.
output : A mixture model $F = \sum_{k=1}^{K} w_k f_k$.
begin
  $k = 1$
  Pick initial f_k
  $w_k = 1$
  $F_k = f_k$
  $J_k = -\log(F_k)$
  for $k = 2, 3, \ldots$ **do**
    $f_k = \arg\max_f \sum_{i=1}^{M} \frac{f(x_i)}{F_{k-1}(x_i)}$
    if $\sum_{i=1}^{M} \frac{f_k(x_i) - F_{k-1}(x_i)}{F_{k-1}(x_i)} < 0$ **then**
      break
    end
    $w_k = \arg\min_w -\log\left((1-w)F_{k-1} + wf_k\right)$
    $F_k = (1-w)F_{k-1} + w_k f_k$
    Optional: Refine all previously estimated w_k, f_k.
  end
end

Algorithm 1: Recursive mixture modeling algorithm.

The algorithm can also determine an *optimal number* of mixture components through its stopping criterion determined by the condition

$$\sum_{i=1}^{M} \frac{f^*(x_i) - F_{k-1}(x_i)}{F_{k-1}(x_i)} < 0. \tag{4}$$

When such condition occurs, the new component f^* will not, in the first order approximation, be able to further decrease the cost functional J and the recursion should terminate.

Finally, the general Algorithm 1 allows for an optional joint reestimation of mixture weights and kernels w, f, after the initial selection of w^*, f^*. The reason for the reestimation lies in the first order approximation used for selection of f^* which may not lead to the truly optimal f. In practice this adjustment can be implemented through a number of runs of the EM algorithm whose initial conditions are now determined by F_{k-1}, w^* and f^*.

2.1. Example: recursive estimation of Gaussian mixtures

To illustrate the general recursive algorithm we consider its particular form for Gaussian mixture models. In a Gaus-

sian mixture, the kernel function is a Gaussian distribution $f(x) = \mathcal{N}(x, \mu, \Sigma)$ parameterized by the mean μ and variance Σ.

Estimation of new Gaussian components takes on a particularly simple form. By letting $\alpha_{ki} = \frac{1}{F_k(x_i)}$, Eq. 2 becomes

$$(\mu_k, \Sigma_k) = \arg\max_{(\mu,\Sigma)} \sum_{i=1}^{M} \alpha_{k-1,i} \mathcal{N}(x_i, \mu, \Sigma) \qquad (5)$$

The solution to this optimization problem can be found in many ways, but one iterative solution is particularly appealing

$$\begin{aligned} \beta_{k,i,j-1} &= \alpha_{k-1,i} \mathcal{N}(x_i, \mu^*_{k,j-1}, \Sigma^* k, j-1) \\ \mu^*_{k,j} &= \frac{\sum_{i=1}^{M} x_i \beta_{k,i,j-1}}{\sum_{i=1}^{M} \beta_{k,i,j-1}} \\ \Sigma^*_{k,j} &= \frac{\sum_{i=1}^{M} (x_i - \mu^*_{k,j})(x_i - \mu^*_{k,j})^t \beta_{k,i,j-1}}{\sum_{i=1}^{M} \beta_{k,i,j-1}} \end{aligned} \qquad (6)$$

Initial estimates $\mu^*_{k,0}$ and $\Sigma^*_{k,0}$ can be obtained, for example, by substituting $\beta_{k,i,-1} = \alpha_{k-1,i}$.

Equations 6 are very similar to the M-estimate update equations for Gaussian kernels (see, for instance, [12] and also resemble the Gaussian mixture EM equations (c.f., [19]). The main difference is the presence of the weight factors α that stem from the previous simpler mixture F_{k-1}.

3. Regularized recursive mixture algorithm

Maximum likelihood estimation of mixture models has one significant disadvantage: a theoretically optimal number of mixture components is equal to the number of data samples M. This overfitting property is a general characteristic of the MLE. To circumvent the problem, regularized or integral likelihood estimates are sought instead [24]. In this section we will extend the recursive MLE mixture algorithm to handle a specific regularization case.

One possible approach to regularized estimation is to consider Laplace approximation of the integral density $F^i(x) = \int F(x|\theta)p(\theta)d\theta$. The Laplace approximation is given as $F^i(x) \approx F^i_{Laplace}(x) = F(x|\theta_{ML})p(\theta_{ML})(2\pi)^d |H(\theta_{ML})|^{1/2}$, where H denotes the Hessian of $-\log F(x|\theta) - \log p(\theta)$ evaluated at θ_{ML}. We will use the compact notation for this approximation, $F^i(x) \approx F(x|\theta)G(\theta)$. Here θ is a shorthand for all model parameters, f and w inclusive. Hence, Equations 7 and 8 hold and the first order functional approximation is for a class of J such that $J(xy) = J(x)J(y)$. The selection of f^*_K, w^*_K that minimize $J(F^i_K)$ is, in general, difficult because G-functionals depend on both of those parameters.

To proceed we instead consider a specific but very common approximation know as the Bayesian Information Criterion or BIC [27]. There, $G(K) = M^{d/2}$, where d denotes the total number of parameters of the K-component mixture. In this case G does not depend on the actual values of mixture parameters but only on their total number. Hence, the selection condition for the new component becomes

$$\begin{aligned} f^*_k &= \arg\max_f < \frac{1}{G(k-1)F_{k-1}}, f - g(k-1)F_{k-1} > \\ &= \arg\max_f \sum_{i=1}^{M} \frac{f(x_i)}{F_{k-1}(x_i)} \end{aligned} \qquad (9)$$

which is the same form as that of the MLE recursive algorithm in Equation 2. However, the stopping criterion changes to

$$\left\langle \frac{1}{G(k-1)F_{k-1}}, f - g(k-1)F_{k-1} \right\rangle > \frac{G(k) + G(k-1)}{w_K}. \qquad (10)$$

Note that in the BIC case the determination of the candidate optimal component f is still independent of w—however, the stopping criterion depends on both f and w. Otherwise, the two algorithms are identical.

3.1. Example: mixture of Gaussian hidden Markov models

Explosive growth in the number systems that gather and store data about the motion of objects, machines, vehicles, humans, animals, etc. has raised the need for general-purpose tools for grouping and organizing the motion patterns. For instance, methods for discovering clusters of similar motion sequences in these data sets would enable pattern discovery, anomaly detection, modeling, summarization, etc. Furthermore, knowledge of clusters could be exploited in data reduction, as well as in efficient methods for sequence indexing and retrieval.

One approach to modeling motion uses a mixture density model where each component represents a different motion type, modeled itself as a hidden Markov model [2]. Using our notation each kernel f_k now becomes a density modeled by an HMM:

$$\begin{aligned} f_k(x) = \sum_s & Pr(s_0|k) Pr(x_0|s_0, k) \\ & \prod_{t=1}^{T-1} Pr(s_t|s_{t-1}, k) Pr(x_t|s_t, k), \end{aligned} \qquad (11)$$

where x now denotes a *sequence* of T motion measurements $x = \{x_0, \ldots, x_{T-1}\}$ and s is the sequence of corresponding hidden states, $s = \{s_0, \ldots, s_{T-1}\}$, $s_t \in \{1, \ldots, S\}$.

In this example we consider Gaussian HMM kernels:

$$\begin{aligned} Pr(x_t|s_t = i, k) &= \mathcal{N}(x_t; \mu_{i,k}, \Sigma_{i,k}) \\ Pr(s_t = i|s_{t-i} = j, k) &= \Pi_k(i, j) \\ Pr(s_0 = i|k) &= \pi_k(i), \end{aligned}$$

$$
\begin{aligned}
F_K^i(x) &= \left[\sum_{k=1}^{K} w_k f_k\right] G(w_1,\ldots,w_K,f_1,\ldots,f_K) \\
&= \left[(1-w_K)F_{K-1}^i(x)G(w_1,\ldots,w_{K-1},f_1,\ldots,f_{K-1})^{-1} + w_K f_K\right] \\
&\quad \times G(w_1,\ldots,w_K,f_1,\ldots,f_K) \\
&= \left[(1-w_K)F_{K-1}^i(x)G(K-1)^{-1} + w_K f_K\right] G(K) \qquad (7)
\end{aligned}
$$

$$
\begin{aligned}
J(F_K^i) &= J\left(G(K)\right) + J\left((1-w_K)F_{K-1}^i G(K-1)^{-1} + w_K f_K\right) \\
&\approx J\left(G(K)\right) + J(G(K-1)^{-1}) + J(F_{K-1}^i) \\
&\quad -w_K\left\langle -\nabla J(F_{K-1}^i G(K-1)^{-1}), f_K - F_{K-1}^i G(K-1)^{-1}\right\rangle, \qquad (8)
\end{aligned}
$$

where (μ_i, Σ_i) are the mean and the variance of x_t in state i of the HMM, Π is the state transition conditional probability table (cpt) and π is the initial state distribution. Hence, each kernel is parameterized by the parameter set $\theta_k = (\pi_k, \Pi_k, \mu_{0,k}, \Sigma_{0,k}, \ldots, \mu_{S-1,k}, \Sigma_{S-1,k})$.

Recursive estimation of the mixture HMM follows the same steps outlined in Algorithm 1. Kernel k is recursively estimated from density of $k-1$ components as

$$\theta_k^* = \arg\max_{\theta_k} \sum_{m=1}^{M} \frac{f_k(x^{(m)}|\theta_k)}{F_{k-1}(x^{(m)})},$$

where we used $x^{(m)}$ to denote the $m-th$ sequence from the set of M sequences. Alternatively, one may estimate parameters of a Gaussian HMM on a set of nonuniformly weighted M data points with weights $\alpha_m = 1/F_{k-1}(x^{(m)})$. Hence, it can be shown that θ_k^* can be obtained from slightly modified traditional EM (Baum-Welch) HMM parameter estimates (c.f., [23]). For instance, estimates of the mean in the $i-th$ state are obtained using a modified M-step in equation (12). Note that the modified M-step differs from the traditional HMM M-step in the factor α_m, the weight introduced by the $k-1$ mixture model. It is also important to stress that the estimation of the $k-th$ HMM kernel is itself recursive because it involves a modified EM-based HMM parameter estimation—e.g., $Pr(s_t^{(m)} = i|x^{(m)}, \theta_k^*)$ depends on the previous estimate $\mu^{i,k}$.

Finally, a BIC-regularized recursive HMM mixture model can be obtained by adding a regularization term $G(k)$. For instance, a full covariance, unrestricted transition HMM, the regularization term is

$$
\begin{aligned}
2\log G(k) &= (S-1)\log M + \\
&\quad (S-1)S\log((T-1)M) + \left(S + SD + SD^2\right)\log(TM),
\end{aligned}
$$

where T is the average length of M sequences and D is the dimension of x_t, e.g., $x_t \in \mathcal{R}^D$.

4. Prior work

Methods for clustering time-series data have been proposed recently, particularly, in the statistics and data mining communities. For example, the Fourier transform has been applied in clustering certain types of time-series data [1], splines have been used to cluster functional data [13], and mixtures of regression models have been used to fit trajectories generated by multiple processes [8]. In the computer vision community clustering of motion data has been used for classification and prediction of pedestrian trajectories [14, 30], for detection of human actions in clutter [5], and for event-based analysis of long video sequences [30, 34].

Specific methods for clustering sequences using AR or hidden Markov models have been proposed in speech recognition [15], computational biology [6], and machine learning [28, 21, 16]. In computer vision, hidden Markov models have been successfully used in the supervised learning and recognition of specific actions [33], activities [4], interactions [3], gaits [10], and gestures [29]. In [2] the authors proposed a method for HMM-based motion clustering using a mixture-of-HMMs framework and the EM algorithm. While successful, the method relies on an explicit specification of the model order.

Our recursive algorithm for mixture estimation relies on functional gradient optimization of a general class of convex additive models and was first introduced in a simpler form in [22]. Similar approaches have been proposed in the past in [18, 7], albeit for the two different tasks of additive classification and regression modeling. An algorithm similar in some aspects to our own was proposed recently by [32]. There, the authors try to solve the same problem of recursive addition of mixture components. However, they were not able to directly solve the optimization problem stated in Equation 1 and resorted instead to a heuristic search. Other approaches to the general problem of modeling mixtures of Bayesian network models have been also

$$\mu^*_{i,k} = \frac{\sum_{m=0}^{M-1} \alpha_m Pr(x^{(m)}|\theta^*_k) \sum_{t=0}^{T_m-1} x_t^{(m)} Pr(s_t^{(m)} = i|x^{(m)}, \theta^*_k)}{\sum_{m=0}^{M-1} \alpha_m Pr(x^{(m)}|\theta^*_k) \sum_{t=0}^{T_m-1} Pr(s_t^{(m)} = i|x^{(m)}, \theta^*_k)}. \quad (12)$$

proposed by [31, 20, 11, 25].

4.1. Comparison with EM-Based mixture modeling

Our recursive algorithm can also be explained as a special case of the general EM algorithm for mixture models [19]. Consider, for instance, the expectation step for the component f of a mixture model $(1-\varepsilon)F + \varepsilon f$:

$$Pr(\text{component} f|x, \varepsilon, f, F) = \frac{\varepsilon f(x)}{(1-\varepsilon)F(x) + \varepsilon f(x)}, \quad (13)$$

where F, f, and ε are the estimates from a previous iteration of the EM algorithm. By letting $\varepsilon \to 0$ the component probability behaves as $f(x)/F(x)$, which is exactly the form used in the mixture estimation algorithm. Assuming an uninformative initial $f \sim 1$, the M step maximization of f would then attempt to optimize

$$\sum_{i=1}^{M} \frac{\log(f(x_i))}{F(x_i)}, \quad (14)$$

which is a bound on the log of the cost of Equation 2.

5. Experiments

We demonstrate the properties and usability of the recursive density modeling algorithm on the problem of clustering of motion sequences. We first demonstrate the algorithm on synthetic dataset and then show its application on real-world data.

We compare the results of our recursive algorithm with that of the standard EM-based exhaustive mixture modeling with a BIC prior [24].

5.1. Recursive modeling of hidden Markov model mixtures

To study the benefits of the recursive mixture modeling approach, we next consider a problem of modeling mixtures of dynamic stochastic models. In particular, we focus our attention on mixtures of hidden Markov models (HMMs) [23] because HMMs represent a general class of dynamic models often used in motion modeling.

Constructing mixtures of HMMs can be seen as "soft" parametric clustering of time-series data. Methods for clustering sequences or temporal data using hidden Markov models have been proposed in, for instance, speech recognition [15], machine learning [28, 16], and computer vision [2]. The problem is challenging because, in general, it involves concurrent estimation of complex kernels (HMMs) and the mixture order. Recursive clustering using mixtures of HMMs is appealing because it avoids the initialization issue and alleviates the need for concurrent kernel estimation.

We used an example of a mixture of three cyclic HMMs of different order to demonstrate the properties of the recursive mixture density estimation algorithm when applied to complex mixture kernels. A synthetic set of 200 sequences of varying length (2-4 times the average HMM cycle) was generated using the following three kernels:

$$\Pi_1 = \begin{bmatrix} .95 & 0 & .01 \\ .05 & .99 & 0 \\ 0 & .01 & .99 \end{bmatrix} \quad \mu_1 = \begin{bmatrix} -5 & 0 & 5 \end{bmatrix}$$
$$\Sigma_1 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \quad \pi_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}'$$

$$\Pi_2 = \begin{bmatrix} .95 & .01 \\ .05 & .99 \end{bmatrix} \quad \mu_2 = \begin{bmatrix} -10 & 0 \end{bmatrix}$$
$$\Sigma_2 = \begin{bmatrix} 1 & 1 \end{bmatrix} \quad \pi_2 = \begin{bmatrix} .99 & .01 \end{bmatrix}'$$

$$\Pi_3 = \begin{bmatrix} .95 & .01 \\ .05 & .99 \end{bmatrix} \quad \mu_3 = \begin{bmatrix} 0 & 10 \end{bmatrix}$$
$$\Sigma_3 = \begin{bmatrix} 1 & 1 \end{bmatrix} \quad \pi_3 = \begin{bmatrix} .99 & .01 \end{bmatrix}'$$

The data in the synthetic set consisted of equal proportions (1/3) of each of the three types. Examples of sequences generated from the three kernels are shown in Figure 1.

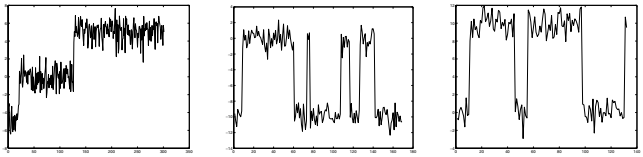


Figure 1. Examples of sequences generated by the three HMM kernels described in the text.

We were interested in two aspects of the recursive algorithm: 1) whether it can recover the generative model's structure (individual kernel model orders and the mixture order) and 2) how it performs compared to traditional mixture modeling methods. The first task required that the kernel estimation algorithm be modified to include a combinatorial search over the kernel HMM order (number of hidden states) as a part of optimization in Equation 2. We achieved this by exhaustively searching over kernel HMM orders 1 through N, with $N = 6$ in this example. Note that this search is in addition to the EM-based kernel parameter estimation described in Section 3.1.

To complete the second goal we used an exhaustive search over different mixture orders (one through eight). For each mixture order we randomly seeded initial sequence assignments to mixture clusters and then reestimated the mixture HMM using an extension of the EM algorithm, as described in [2].

Figure 2(a) shows the BIC-regularized model cost of mixture models of orders one through eight obtained using the recursive algorithm. It is clear from the above example that the algorithm recovers the correct number of mixture components (3) as that with the lowest regularized cost. Upon closer examination, it can be seen that the three kernel HMM models estimated in the optimal mixture closely resemble the original generative kernels, i.e., two of them have two and one kernel has three hidden states.

Results of comparison of the recursive algorithm with the traditional exhaustive search reveal that, on average, the recursive algorithm does not result in models with the lower total cost. This is shown in Figure 2(b). The lower cost signifies better fit of the model to the true underlying distribution. However, the recursive algorithm was able to recover the true mixture order. Moreover, such fit was achieved at a fraction of the computational cost of the exhaustive search, partially due to the lack of random initialization.

In the recursive algorithm, the estimation of a new HMM kernel k (from mixture of order $k-1$) required at most two iterations. The few iterations are due to a highly peaked weight distribution $\alpha_m = \frac{1}{F_{k-1}(x^{(m)})}$ that usually assigns non-zero weight to only one or two sequences in the set, as shown in Figure 2(c). This is not surprising given the high dimensionality (hence, sparseness) of the sequence space. Estimation of the $k-th$ kernel contribution, w_k, required at most seven iterations. The joint refinement step was not necessary as it often diluted the initial kernels. It also led to a very computationally efficient solution. On the other hand, in the case of an exhaustive search from 5 random initial assignments, each joint EM reestimation of the mixture parameters took from 12 up to 35 (maximum allowed by our implementation) iterations.

5.2. Clustering of human motion data

We conducted experiments with gait data that was collected for the Human ID project at the computational perception lab at Georgia Tech (http://www.cc.gatech.edu/cpl/projects/hid). The database consists of video sequences (and accompanying data files) of twenty walking human subjects taken under various viewing conditions (namely, different combinations of indoor/outdoor footage, side/angle view, and far/near view.) We used a subset of this database for our experiment: a total of 45 sequences of 15 subjects (3 sequences per subject), for which binary masks, extracted using a background subtraction algorithm, was available.

All sequences were taken under the same viewing conditions: indoor footage, and far side view. Fig. 3 shows example frames of the sequence pertaining to subject 3. Binary masks are shown below their corresponding frames.

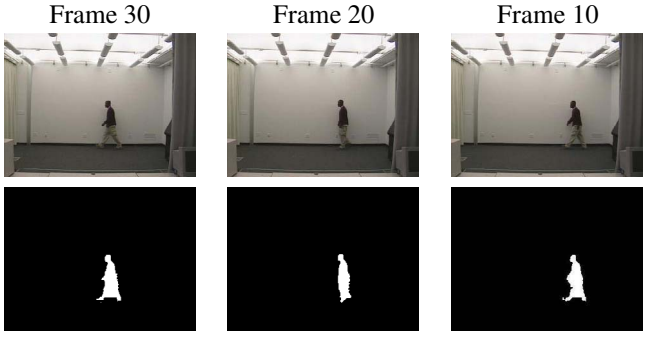


Figure 3. Example frames and corresponding silhouettes extracted from image sequences pertaining to subject 3.

For each binary mask we computed the aspect ratio of its bounding box. The resulting aspect ratio sequences for the 15 subjects are depicted in Figure 4(a).

Similar to the other tasks, the goal here was to learn the number of different motion styles corresponding to the 15 subjects in the set as well as parametric models of those styles. We modeled each mixture component as a Gaussian-observation HMM, analogous to the example in Section 3.1. Observation space was considered to be the space of aspect ratios and their temporal rates of change (velocities). As in Section 3.1 the selection of the best kernel k included a search over different kernel HMM orders, ranging from 1 to 8.

Figure 4(b) shows the optimal learned segmentation of the original motion set into motion styles. It is easy to see that the recursive algorithm *does not* recover the individual motion styles of the 15 subjects. Rather, the motion data is classified into one of four categories. Each category corresponds to similar motion patterns exhibited by several subjects. For instance, subjects 7, 9, 10, and 12 all have similar motion patterns that are all assigned to class 1. On the other hand, 11 and 14 belong together to class 4. Class 2 covers the largest number of subjects and seems to correspond to an average motion in the group. Addition of new motion classes, beyond 4, is prevented by the BIC regularization cost. Indeed, it can be seen that the additional mixture components do not correspond to obviously new motion classes, not already covered by the first four components.

6. Conclusions

In this paper we have presented a novel algorithm for recursive parametric estimation of mixture densities, applied to the problem of model-based clustering of motion

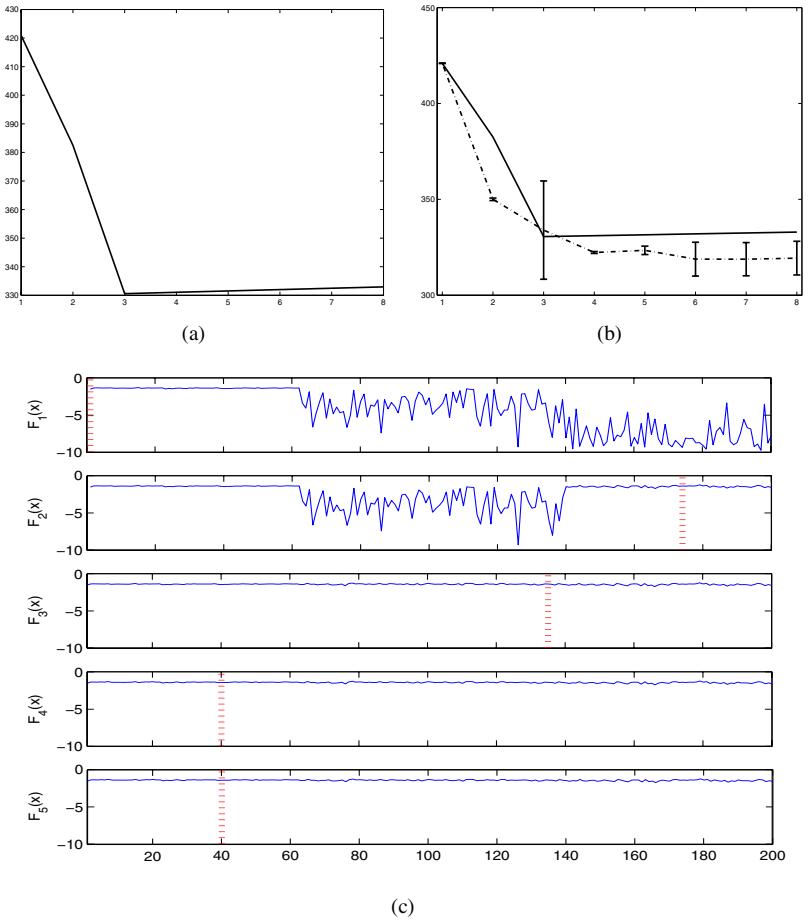


Figure 2. (a) BIC-regularized model cost for the synthetic data set. Note that the least cost corresponds to the mixture order (3) of the generating model. (b) Comparison of the model costs obtained using the recursive algorithm (solid line) and an exhaustive search from ten random initial assignments. Shown are also the one standard deviation intervals for the exhaustive search. (c) Illustration of the data selection process by the recursive algorithm. Shown are estimates of mixture models, $\log F_k(x)$ for five different recursion steps (models of order 1 through 5) and the entire training set of 200 sequences. Sequence 1-60, 61-140, and 141-200 belong to the three models, respectively. Also shown (vertical dashed bars) are sequences on which the current mixture component focuses on, based on weights α_m. Note how the models successively cover different motion groups, represented through high values of F_k.

sequences. The algorithm is based on a general functional derivative optimization of a class of convex additive models. Unlike the traditional EM-based algorithms, the recursive method estimates one mixture kernel at the time by focusing on the samples poorly modeled by previous mixtures. We present both the unregularized likelihood and the regularized BIC versions of the algorithm. A relationship to the EM is discussed. We illustrate the algorithm's promising performance on problems of clustering of synthetic as well as real-world motion sequence datasets.

We plan to further evaluate the algorithm on additional real-world motion datasets, as well as demonstrate the applicability of the obtained cluster models for motion synthesis. In particular, we will consider more complex clusters comprised of switching linear dynamic models whose utility for motion analysis and synthesis has been demonstrated in the past [17].

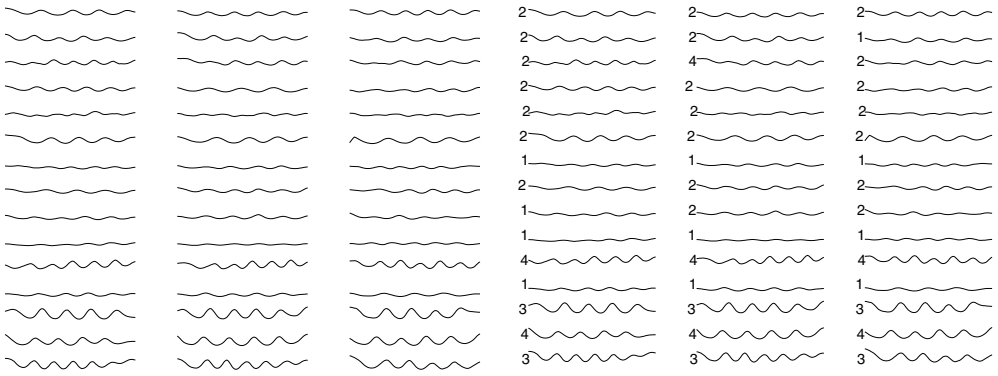
References

[1] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In *Proc. of the 4th International Conf. on Foundations of Data Organization and Algorithms*, pages 69–84, 1993.

[2] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic. Discovering clusters in motion time-series data. In *CVPR*, Madison, WI, 2003.

[3] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *CVPR97*, pages 994–999, 1997.

[4] C. Bregler. Learning and recognizing human dynamics in video sequences. In *CVPR97*, pages 568–574, 1997.



(a) Motion dataset (b) Learned grouping

Figure 4. Grouping of motion data. Panel (a) shows the aspect-ratio motion sequences of 15 subjects (top to bottom). Panel (b) depicts learned assignments of the motion trajectories into four optimal group, as determined by the recursive learning algorithm.

[5] F. Cuzzolin, A. Bissacco, R. Frezza, and S. Soatto. Towards unsupervised detection of actions in clutter. Technical Report CSD-2000/33, UCLA, 2001.

[6] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis*. Cambridge University Press, Cambridge, UK, 1998.

[7] J. Friedman. Greedy function approximation: a gradient boosting machine. Technical report, Dept. of Statistics, Stanford University, 1999.

[8] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In *Knowledge Discovery and Data Mining*, pages 63–72. ACM Press, 1999.

[9] T. J. Hastie and R. J. Tibshirani. *Generalized additive models*. Chapman and Hall, 1999.

[10] Q. He and C. Debrunner. Individual recognition from periodic activity using hidden markov models. In *HUMO00*, pages 47–52, 2000.

[11] D. Heckerman, C. Meek, and B. Thiesson. Staged mixture modeling and boosting. In *Proc. UAI*, 2002.

[12] P. J. Huber. *Robust statistics*. Wiley, 1981.

[13] G. James and C. Sugar. Clustering for sparsely sampled functional data. Information and Operations Management Dept., U. of Southern California, 2002.

[14] N. Johnson and D. Hogg. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14(8):609–615, 1996.

[15] B.-H. Juang and L. Rabiner. A probabilistic distance measure for hidden markov models. *AT&T Technical Journal*, 64(2):391–408, 1985.

[16] C. Li and Biswas. A bayesian approach to temporal data clustering using hidden markov models. In *International Conf. on Machine Learning*, pages 543–550, 2000.

[17] Y. Li, T. Wang, and H.-Y. Shum. Motion texture: A two-level statistical model for character motion synthesis yan. In *SIGGRAPH*, 2003.

[18] L. Mason, J. Baxter, P. Bartlett, and M. Frean. Functional gradient techniques for combining hypotheses. In *Advances in Large Margin Classifiers*. MIT Press, 1999.

[19] G. McLachlan. *Finite mixture models*. John Willey & Sons, Inc., 2001.

[20] M. Meila and T. Jaakkola. Tractable bayesian learning of tree belief networks. Technical Report CMU-RI-TR-00-1, Carnegie Mellon University, 2000.

[21] T. Oates, L. Firoiu, and P. Cohen. Using dynamic time warping to bootstrap HMM-based clustering of time series. In *Sequence Learning: Paradigms, Algorithms and Applications*. Springer, 2000.

[22] V. Pavlovic. Boosting distributions. Snowbird Learning Workshop, April 2002.

[23] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proc. of the IEEE*, volume 77(2), pages 257–285, 1989.

[24] S. J. Roberts, D. Husemeier, L. Rezek, and W. Penny. Bayesian approaches to gaussian mixture modeling. *IEEE PAMI*, 20(11):1133–1142, 1998.

[25] S. Rosset and E. Segal. Boosted density estimation. In *NIPS*, 2002.

[26] R. E. Schapire. A brief introduction to boosting. In *IJCAI*, Stockholm, Sweeden, 1999.

[27] G. Schwartz. Estimating the dimension of a model. *Annals of Statistic*, 6:461–464, 1978.

[28] P. Smyth. Clustering sequences with hidden markov models. In M. Mozer, M. Jordan, and T.Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, pages 648–654. MIT Press, 1997.

[29] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. In *SCV95*, pages 265–270, 1995.

[30] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *PAMI*, 22(8):747–757, August 2000.

[31] B. Thiesson, C. Meek, , and D. Heckerman. Learning mixtures of DAG models. Technical Report MSR-TR-98-12, Microsoft Research, 1997.

[32] J. Verbeek, N. Vlassis, and B. Kröse. Greedy gaussian mixture learning for texture segmentation. In *Proc. of Workshop on Kernel and subspace methods for computer vision on Int. Conf. on Artif. Neural Networks*, 2001.

[33] A. Wilson and A. Bobick. Parametric hidden markov models for gesture recognition. *PAMI*, 21(9):884–900, September 1999.

[34] L. Zelnik-Manor and M. Irani. Event-based analysis of video. In *CVPR01*, pages II:123–130, 2001.