2003-03-26

Discovering Clusters in Motion Time-Series Data

Boston University Computer Science Tech. Report No. 2003-008, March 26, 2003. To appear in Proc. IEEE CVPR, June 2003.

Discovering Clusters in Motion Time-Series Data

Jonathan Alon, Stan Sclaroff, and George Kollios
Computer Science Department
Boston University
Boston, MA 02215

Vladimir Pavlovic
Computer Science Department
Rutgers University
Piscataway, NJ 08854

Abstract

A new approach is proposed for clustering time-series data. The approach can be used to discover groupings of similar object motions that were observed in a video collection. A finite mixture of hidden Markov models (HMMs) is fitted to the motion data using the expectation-maximization (EM) framework. Previous approaches for HMM-based clustering employ a k-means formulation, where each sequence is assigned to only a single HMM. In contrast, the formulation presented in this paper allows each sequence to belong to more than a single HMM with some probability, and the hard decision about the sequence class membership can be deferred until a later time when such a decision is required. Experiments with simulated data demonstrate the benefit of using this EM-based approach when there is more "overlap" in the processes generating the data. Experiments with real data show the promising potential of HMM-based motion clustering in a number of applications.

1. Introduction

In the past decade, there has been an explosive growth in the number systems that gather and store data about the motion of objects, machines, vehicles, humans, animals, etc. These data sets are collected and analyzed for a broad range of applications, too numerous to mention. The parameterization and dimensionality of the motion time series data can vary widely, depending on the particular input device, tracking method, motion model, relevant degrees of freedom, etc.

Given the size and diversity of these motion data archives, clearly, general-purpose tools are needed for grouping and organizing the motion patterns contained therein. 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.

In this paper, we focus on the problem of finding groups, or clusters of similar object motions within a database of motion sequences, and estimating motion time series models based on these groups. We employ a probabilistic model-based approach, where the data is assumed to have been generated by a finite mixture model. In particular, we assume that the observed sequences are generated by a finite mixture of hidden Markov models (HMMs), and estimate this mixture of HMMs via an Expectation Maximization (EM) formulation.

Previous approaches for HMM-based clustering employ a k-means formulation [11, 13, 16], which has the drawback that in each iteration, each sequence can only be assigned to a single cluster, and then only those sequences that are assigned to a particular cluster are used in the re-estimation of its HMM parameters. This can lead to problems when there is not a particularly good separation between the underlying processes that generated the groups of time series data.

In our proposed EM approach, a sequence can be partially assigned to all clusters, with the degree of cluster membership determined by the *a posteriori* probability, which depends via Bayes rule on the cluster *a priori* probability and the data likelihood. In contrast with k-means, the parameter estimates of a single HMM cluster are influenced by all the observation sequences with the corresponding *a posteriori* probabilities. The hard decision about the sequence class membership can be deferred until a later time when such a decision is required. In experimental evaluation, this EM-based formulation tends to yield improved accuracy over the k-means approach, particularly when there is more "overlap" in the processes that generated the data.

2. Related Work

Methods for clustering time-series data have been proposed recently, particularly, in the statistics and data mining communities. (See e.g., [4]). In the computer vi-

sion community clustering of motion data has been used mainly for classification and prediction of pedestrian trajectories [8, 18], and for event-based analysis of long video sequences [18, 19].

Methods for clustering sequences or temporal data using hidden Markov models have been proposed in speech recognition [9], computational biology [3], and machine learning [16, 11, 13]. In computer vision, hidden Markov models have been successfully used in the supervised learning and recognition of specific activities [2] and gestures [17]; however, to the best of our knowledge, unsupervised learning or clustering using hidden Markov models has received little or no attention.

A complete approach to HMM-clustering should address four key problems [11]: estimating the parameters of a single HMM cluster, selecting the number of HMM clusters, learning HMM structure (topology and size), and assigning sequences to clusters.

The fourth problem, and the focus of this paper is how to assign sequences to clusters. In previous approaches [11, 13] the sequences are assigned in a winner-take-all manner: a sequence is assigned to the HMM that most likely generated it. In contrast, the EM-based formulation given in this paper enables soft assignment, whereby each sequence is partially assigned to a cluster according to the cluster *a posteriori* probability given the sequence. This EM-based algorithm is much preferred in noisy data situations or when the underlying densities are with more "overlap" [10].

3. Background: HMMs

In this section, we give a brief review of hidden Markov models (HMMs). Our main purpose is to define notation used in our clustering formulation. For a detailed overview of HMMs, readers are directed to [14].

A complete specification of a first-order HMM with a simple Gaussian observation density is formally given by:

1. N states, $S = \{S_1, S_2, \ldots, S_N\}$.
2. The state transition probability distribution $A = \{a_{ij}\}$, where $a_{ij} = P(q_{t+1} = S_j | q_t = S_i), 1 \leq i, j \leq N$.
3. The observation density $b_j(O_t) = \mathcal{N}(O_t; \mu_j, \Sigma_j), 1 \leq j \leq N$, where μ_j, and Σ_j are the mean and covariance of the Gaussian of state j.
4. The initial state probability distribution, $\pi = \{\pi_i\}$, where $\pi_i = P(q_1 = S_i), 1 \leq i \leq N$

where O_t and q_t are the observation and state respectively at time t. It is common to use the compact notation

$$\lambda = (\pi, A, \{\mu_j\}, \{\Sigma_j\}),\ 1 \leq j \leq N \tag{1}$$

to indicate the complete parameter set of the model.

The problem of estimating the parameters of a HMM λ^* given L independent sequences $O^{(l)}, 1 \leq l \leq L$ can be cast as a maximum likelihood (ML) problem

$$\lambda^* = \arg\max_{\lambda} \prod_{l=1}^{L} P(O^{(l)}|\lambda). \tag{2}$$

Unfortunately, there is no known analytical way for finding the global ML solution. The well known Baum-Welch algorithm is an iterative procedure that can only guarantee convergence to a local maximum. It consists of the following re-estimation formulas [14]:

$$\bar{\pi}_i = \frac{\sum_{l=1}^{L} \gamma_1^{(l)}(i)}{L} \tag{3}$$

$$\bar{a}_{ij} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T-1} \xi_t^{(l)}(i,j)}{\sum_{l=1}^{L} \sum_{t=1}^{T-1} \gamma_t^{(l)}(i)} \tag{4}$$

$$\bar{\mu}_j = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_t^{(l)}(i) \cdot O_t}{\sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_t^{(l)}(i)} \tag{5}$$

$$\bar{\Sigma}_j = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_t^{(l)}(i) \cdot (O_t - \mu_j)(O_t - \mu_j)'}{\sum_{l=1}^{L} \sum_{t=1}^{T} \gamma_t^{(l)}(i)} \tag{6}$$

where $\gamma_t(i)$ is the probability of being in state S_i at time t, given the observation sequence O and the model λ, and $\xi_t(i,j)$ is the probability of being in state S_i at time t and state S_j at time $t+1$, given the observation sequence O and the model λ. These two variables can be computed by the forward-backward algorithm [1].

4. Finite Mixture of HMMs

In this section we present a specialization of the general framework for probabilistic model-based clustering, where the cluster models are HMMs. In HMM-based clustering it is assumed that an observation sequence O is generated according to a mixture distribution of M components. Let $P(O|\lambda^{(m)})$ be the probability distribution of the sequence O given the m'th HMM parameterized by $\lambda^{(m)}$, and let $z_l = (z_{l1}, \ldots, z_{lM})$ be the cluster membership vector for the l'th sequence, where $z_{lm} = 1$ if sequence $O^{(l)}$ was generated by the m'th HMM and 0 otherwise. Then, the z_{lm}'s can be treated in one of two ways: as fixed but unknown parameters, or alternatively as missing binary random variables. In the first case, we want to estimate the set of parameters $\theta = \{\lambda^{(m)}, z_l | 1 \leq m \leq M, 1 \leq l \leq L\}$ that maximize the likelihood function

$$\mathcal{L}_1(\theta) = \prod_{l=1}^{L} P(O^{(l)}|\lambda^{(z_l)}), \tag{7}$$

In the second case, each z_{lm} has a prior probability $p^{(m)}$ of being generated by the m'th HMM. In this case we want to estimate the set of parameters $\theta = \{\lambda^{(m)}, p^{(m)} | 1 \leq m \leq M\}$ that maximize the likelihood function

$$\mathcal{L}_2(\theta) = \prod_{l=1}^{L} \sum_{m=1}^{M} p^{(m)} P(O|\lambda^{(m)}), \tag{8}$$

It is well known that ML estimates cannot be found analytically for neither likelihood function, and one must resort to iterative procedures. K-means is an iterative procedure for finding the estimates for the first likelihood function (Eq. 7). In k-means, each iteration consists of two steps:

- Assign sequences $O^{(l)}$ to clusters $\lambda^{(m)}$

$$z_{lm} = \begin{cases} 1 & \text{if } m = \arg\max_i P(O^{(l)}|\lambda^{(i)}) \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

- Update estimates $\bar{\lambda}^{(m)}$

$$\bar{\pi}_i^{(m)} = \frac{\sum_{l=1}^{L} z_{lm} \gamma_1^{(lm)}(i)}{\sum_{l=1}^{L} z_{lm}} \tag{10}$$

$$\bar{a}_{ij}^{(m)} = \frac{\sum_{l=1}^{L} z_{lm} \sum_{t=1}^{T-1} \xi_t^{(lm)}(i,j)}{\sum_{l=1}^{L} z_{lm} \sum_{t=1}^{T-1} \gamma_t^{(lm)}(i)} \tag{11}$$

$$\bar{\mu}_j = \frac{\sum_{l=1}^{L} z_{lm} \sum_{t=1}^{T} \gamma_t^{(lm)}(i) \cdot O_t}{\sum_{l=1}^{L} z_{lm} \sum_{t=1}^{T} \gamma_t^{(lm)}(i)} \tag{12}$$

$$\bar{\Sigma}_j = \frac{\sum_{l=1}^{L} z_{lm} \sum_{t=1}^{T} \gamma_t^{(lm)}(i) \cdot (O_t - \mu_j)(O_t - \mu_j)'}{\sum_{l=1}^{L} z_{lm} \sum_{t=1}^{T} \gamma_t^{(lm)}(i)} \tag{13}$$

EM is an iterative procedure for finding the ML estimates for the mixture likelihood function (Eq. 8). Similar to the k-means algorithm, each iteration consists of two steps:

- E-Step

$$\begin{aligned} w_{lm} &\stackrel{def}{=} E\left[z_{lm}|\mathcal{O}, \theta\right] = Pr\left[z_{lm} = 1 | O^{(l)}, \lambda^{(m)}\right] \\ &= \frac{p^{(m)} P(O^{(l)}|\lambda^{(m)})}{\sum_{m=1}^{M} p^{(m)} P(O^{(l)}|\lambda^{(m)})}, \end{aligned} \tag{14}$$

where the last equality follows from Bayes law, $p^{(m)}$ is the a priori probability that $z_{lm} = 1$, and w_{lm} is the posterior probability that $z_{lm} = 1$ after observing $O^{(l)}$, and

- M-Step

$$\bar{p}^{(m)} = \frac{\sum_{l=1}^{L} w_{lm}}{\sum_{m=1}^{M} \sum_{l=1}^{L} w_{lm}} = \frac{\sum_{l=1}^{L} w_{lm}}{L} \tag{15}$$

and replace z_{lm} with w_{lm} in Eq. (10)-(13) throughout. We note that the computation of the posterior probabilities in the E-step (Eq. 14) requires special arithmetic manipulations to avoid numerical problems. Otherwise, the computation cost of EM and k-means is similar.

5. Model Selection

In the previous section, we have assumed that the number of mixture components M is known. The problem of estimating the "correct" number of clusters is a difficult one: a full Bayesian solution for obtaining the posterior probability on M, requires a complex integration over the HMM parameter space, as well as knowledge about the priors on the mixture parameters and about the priors on M itself. Often this integration cannot be solved in closed form, and Monte-Carlo methods and other approximation methods are used to evaluate it. However, these methods are computationally intensive. Other methods that sacrifice some accuracy for efficiency, are the penalized likelihood approaches, where the log-likelihood term is penalized by subtraction of a complexity term. We use such a method that tries to find a model with Minimum Description Length (MDL) [15]. Assuming all the models are equally likely *a-priori* we can write:

$$\log P(M|\mathcal{O}) \approx \log P(\mathcal{O}|M, \hat{\theta}) - \frac{d}{2} \log L, \tag{16}$$

where $P(M|\mathcal{O})$ is the approximate posterior distribution on M, $P(\mathcal{O}|M, \hat{\theta})$ is the data likelihood term (Eq. 7 for k-means and Eq. 8 for EM) given the ML estimates $\hat{\theta}$, and $\frac{d}{2} \log L$ is the MDL term, where $d = M + |\hat{\theta}|$ is the number of model parameters.

6. Experimental Evaluation

The two HMM-based clustering algorithms (EM and k-means) described in Section 4 were implemented in Matlab using a HMM toolbox [12]. The measure used for testing the validity of our clustering results is classification accuracy. The reason is that for all our experiments ground-truth was available, so the majority class of each cluster could be associated with the cluster itself, which enabled the computation of classification accuracy. When ground-truth is not available, or when it is not known whether the data points can be naturally clustered, other validity measures should be employed [7]. The purpose of the experiments with simulated data is to compare performance of the k-means and the EM-based approaches for clustering sequences with HMMs. The purpose of the experiments with real data is to demonstrate the usefulness of HMM-based clustering in a number of applications.

6.1 Experiment with Synthetic Data

In this experiment, 100 sequences of length 200 are generated from a 2-component HMM mixture (50 sequences from each component). Both HMMs are modeled with two

Method	Separation $\frac{\Delta\mu}{\sigma}$								
	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00
True	0	0	3	28	44	50	50	50	50
EM	0	0	0	1	22	46	49	47	47
k-means	0	0	0	0	21	29	30	44	49

Table 1. EM vs. k-means classification accuracy. Each entry in the table is the number of trials for which the method achieved the specified classification accuracy as a function of the separation between the observation Gaussian densities

states, and a 1-d Gaussian observation density, in a manner similar to [16]. Using Dynamic Time Warping (DTW) for initial clustering, both models were initialized in the standard way: uniform priors, uniform transition matrices, and means and variances were estimated using k-means. In this experiment the amount of overlap between the generating models was varied by varying the mean separation between the Gaussian outputs of the two states, in a similar way for both HMMs, leaving all the other parameters fixed. To avoid confusion between states and HMM clusters, we use superscripts to denote HMM clusters, and subscripts to denote states.

6.1.1 Experiment 1 : Sensitivity to Observation Noise

In this experiment the HMMs' dynamics were

$$A^{(1)} = \begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix}, \quad A^{(2)} = \begin{pmatrix} 0.4 & 0.6 \\ 0.6 & 0.4 \end{pmatrix}, \quad (17)$$

and for both HMMs the standard deviations of the Gaussians were kept fixed $\sigma_1 = \sigma_2 = 1$, the mean of the first state was kept fixed $\mu_1 = 0$, and the mean of the second state (for both HMMs) varied in the range $\mu_2 = (1.00, \ldots, 3.00)$, a total of nine values. This corresponds to a change in $\frac{\Delta\mu}{\sigma}$, which is the normalized mean separation between the two Gaussians. For each of the nine values of $\frac{\Delta\mu}{\sigma}$, 50 trials were run, and for each trial, classification accuracy was computed. Model selection was applied only to the data generated in the 50 trials corresponding to mean separation $\frac{\Delta\mu}{\sigma} = 3.00$. In 46 out of the 50 trials the EM algorithm correctly found 2 HMM clusters. The k-means algorithm found the correct number of clusters in all of the trials. We instantiated all other trials with a two-components HMM mixture.

The results are summarized in Table 1. Each entry in Table 1 is the number of trials (out of 50) for which the method achieved 90% classification accuracy. Classification results are given for the k-means and EM methods, and for reference we give the classification accuracy of the "true" HMMs that generated the sequences. These are the optimal classifiers, and their performance gives an upper bound on the possible classification accuracy.

The results from Table 1 indicate that when the separation between the Gaussians in state 1 and 2 is large enough (3.00), both k-means and EM achieve near optimal classification accuracy. When the separation between the Gaussians is very small (< 2.00), both approaches perform badly as expected, as the Gaussians become practically indistinguishable. Between these two extremes the EM seems to outperform k-means.

6.2 Experiments with Real Data

6.2.1 Experiment 2: Camera Mouse

In this experiment, we cluster 2D time-series data obtained via the Camera Mouse system [5]. As shown in Fig. 1(a), a correlation-based video tracking subsystem estimates the image position of a selected facial feature. In these experiments, the tip of the user's nose was tracked. Motion of the user's head/nose then drives an onscreen cursor, and thereby enables the user to control a hierarchical spelling interface, as shown in Fig. 1(b). In the top-level menu of the hierarchical spelling interface, the alphabet is divided into five sub-alphabets. By moving the cursor, the user selects that sub-alphabet which contains the desired letter, and then a second menu appears that allows the user to pick the desired letter from the sub-alphabet.

For this particular experiment, the subjects used the Camera Mouse system to spell the words: "athens", "berlin", "london", "boston", and "paris". The number of sequences obtained for each word were 3, 4, 4, 5, 4 respectively, yielding a total of 20 sequences. The average length of a sequence is around 1100. The shortest sequence length is 834, and the longest one is 1719. Figure 2 shows four sequences: two corresponding to the word "berlin", and two corresponding to the word "london".

The model selection criterion was tested with values of M varying in the range between 1 and 6. Since the sequences were relatively long the likelihood term dominated the MDL term, and a the maximum value of 6 was selected

(a)

(b)

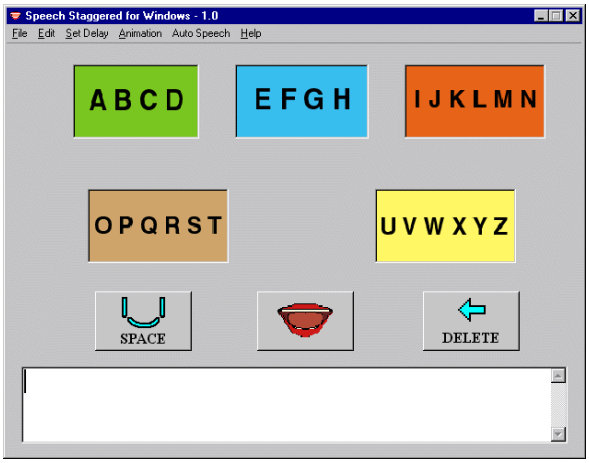

Figure 1. Spelling via the CameraMouse interface. (a) The correlation-based video tracking system estimates the image position of a selected facial feature. In these experiments, the tip of the user's nose was tracked. (b) Motion of the user's nose then moves the cursor on the screen to spell words via a hierarchical spelling interface as described in the text.

as the "correct" number of clusters for both the EM and k-means algorithms. The sequences corresponding to the word "boston" were distributed among two clusters by both algorithms. In what follows, we chose to ignore the estimated number of clusters and initialized the algorithms with the true number of classes, 5.

Two different initializations were tested: (1) initialization with DTW (in a similar manner to [13]), and (2) random initialization. With DTW, 16 out of the 20 sequences were initially correctly classified, and using this as initial classification, both HMM-clustering approaches were able to move one more sequence (corresponding to the word 'boston') to its correct class, so 17 out of 20 were eventually correctly classified. With random initialization only 10 out of the 20 sequences were initially correctly classified, and after HMM-clustering 16 out the 20 sequences were correctly classified. We noticed that the EM and k-means performed essentially the same, because the sequences lengths were long enough, causing the likelihood ratios to dominate the prior probabilities ratios, and therefore causing the EM soft posteriors to approach the k-means hard posteriors.

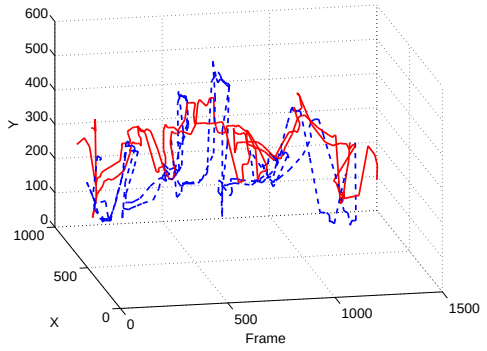

Figure 2. Four tracking sequences: two corresponding to the word "berlin"(blue dashed), and two corresponding to the word "london"(red solid). The graph depicts the x and y position of the user's nose.

6.2.2 Experiment 3: Georgia-Tech Gait database

We conducted experiments with Gait data that was collected for the Human ID project at the computational perception lab at Georgia Tech. This database is available online at http://www.cc.gatech.edu/cpl/projects/hid/. This database consists of video sequences of 20 walking human subjects taken under various viewing conditions 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. Fig. 3 shows example frames of two out of the 45 sequences pertaining to subjects 3 and 5. Binary masks are shown below their corresponding frames. For each binary mask we computed the aspect ratio of its bounding box, and thus for each of the 45 sequences we obtained an aspect ratio time-series. An average smoothing filter of size five was applied to the time-series. The resulting smoothed aspect ratio time-series for subjects 3 and 5 are depicted in Fig. 4.

The model selection criterion was tested with 5, 10, and 15 clusters, the true number of classes. The EM algorithm selected the value $M = 5$, and the k-means algorithm selected $M = 10$. Model selection underestimated the number of clusters because the aspect ratio is not the most discriminative feature, and in such cases MDL will tend to simplify the model even more. In what follows, we chose to ignore the estimated number of clusters and initialized the algorithms with the true number of classes, 15.

In order to reduce the sensitivity of the HMM-clustering algorithm to the initial grouping, we tried two initializations: Smyth's method [16], and Dynamic time warping. Both methods use agglomerative hierarchical clustering,

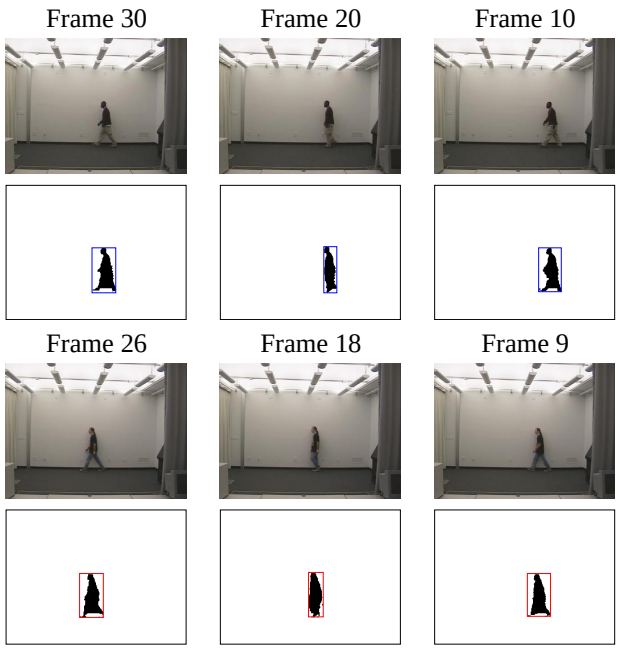

Figure 3. Example frames and corresponding silhouettes extracted from image sequences pertaining to subjects 3 and 5.

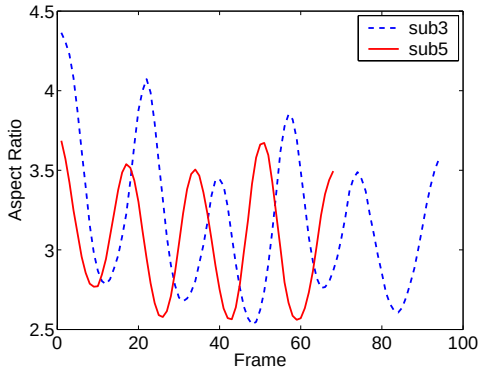

Figure 4. Smoothed aspect ratio signals for subjects 3 (blue dashed) and 5 (red solid).

	Method	Classification Accuracy (%)
Supervised	1-NN leave-one-out	76
Unsupervised DTW init.	DTW init.	51
	K-means	69
	EM	69
Unsupervised Smyth's init.	Smyth's init.	71
	K-means	69
	EM	69

Table 2. Classification results for Georgia-tech gait database.

and the clusters are formed by cutting the hierarchy at level 15. Smyth's method uses the KL divergence as the dissimilarity measure, while DTW uses the warping distance as the dissimilarity measure. The results of the agglomerative algorithms were used as initial clusterings, and were refined using the two HMM partition-based clustering algorithms, namely k-means and EM.

The input time-series depicted in Fig. 4 can be modeled as sine waves plus noise. We therefore selected a periodic HMM structure that consists of four states corresponding to the sine valley, zero crossing up, peak, and zero crossing down. This is a more compact representation than the one typically used to represent periodic signals with HMMs [6, 13]. The typical model consists of as many states as observations in a single period. The features we used are the aspect ratio and its first derivative.

The results of this experiment are summarized in Table 2. The results indicate that the EM and k-means procedures improve the initial clusters obtained from DTW. Smyth's procedure yields good initial clusters. EM and k-means assigned a few more sequences to clusters with similar sequences, thus reducing the number of clusters and consequently the classification accuracy. The classification accuracies of the clustering algorithms are very similar to the classification accuracy obtained using supervised learning. In other words, the clustering-based classifier achieved comparable performance, but without the need for class labelings provided in the supervised learning approach. This is an encouraging result.

7 Conclusion and Future Work

In this paper we presented an EM-based algorithm for clustering sequences using a mixture of HMMs, and compared it to the k-means algorithm typically used in this context. Our experiments show that though the EM and k-means approaches generally have similar performance, the EM-based approach produces better estimates when there is more "overlap" in the underlying densities.

In future work, we plan to improve the clustering algorithm by incorporating a robust method to handle outliers, and a method for learning the structure of the HMMs. Our main goal then is to make use of motion clusters for efficient indexing and retrieval of similar motion sequences.

Acknowledgments

This research was supported in part by the Office of Naval Research under grants N000140310108 and N000140110444, and the National Science Foundation under grant IIS-0208876 and CAREER Award 0133825.

References

[1] L. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *Ann. Math Stat.*, 41:164–171, 1970.

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

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

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

[5] J. Gips, M. Betke, and P. Fleming. The camera mouse: Preliminary investigation of automated visual tracking for computer access. In *Rehabilitation Engineering and Assistive Technology Society of North America*, 2000.

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

[7] A. Jain and R. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

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

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

[10] M. Kearns, Y. Mansour, and A. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Uncertainty in Artificial Intelligence*, pages 282–293, 1997.

[11] 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.

[12] K. Murphy. HMM toolbox for Matlab. http://www.cs.berkeley.edu/∼murphyk/Bayes/hmm.html.

[13] 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.

[14] 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.

[15] J. Rissanen. Hypothesis selection and testing by the MDL principle. *The Computer Journal*, 42(4):260–269, 1999.

[16] 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.

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

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

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