Spectral Learning of Infinite Mixture of Hidden Markov Models for Human Action Recognition

Y. Cem Sübakan¹, Oya Çeliktutan¹, A. Taylan Cemgil², Bülent Sankur¹

¹ Electrical Engineering, ² Computer Engineering
Boğaziçi University
İstanbul, Bebek 34342
{cem.subakan, oya.celiktutan, taylan.cemgil, bulent.sankur}@boun.edu.tr

Abstract

In this work, we approach the sequence clustering and classification problem using an infinite mixture. For each class, we learn a mixture of multiple observation Hidden Markov Models (HMM). The exact inference requires the evaluation of an intractable integral over HMM parameters. Assuming that the observed sequences are sufficiently long, we approximate the intractable integral by a spectral method based on the approach by Hsu et.al [1]. We apply the resulting algorithm on human action recognition problem. We show that doing clustering in the training phase improves the classification accuracy, as many human action sequences tend to be multimodal. Our results also suggest an improved accuracy when compared to a more conventional Expectation Maximization approach.

1 Introduction

Sequence clustering can be defined as assigning a given sequence to a discrete class label in an unsupervised manner. It is an important problem due to its wide range of application areas like, bioinformatics, finance, information retrieval [2, 3]. In this work, we approach the sequence clustering problem using Hidden Markov Models.

In literature, most of the existing HMM clustering methods (e.g. [4]) use parameter estimation techniques such as expectation maximization (EM) algorithm. This approach is tedious and cumbersome since, to compute the likelihood of a sequence one has to do parameter estimation which requires proper initialization and a sufficient number iterations. We propose a clustering algorithm that employs the spectral Hidden Markov Model learning algorithm proposed in [1], that bypasses the parameter estimation procedure and gives a unique likelihood estimation. We also do clustering in a non-parametric manner to deal with the cases where the number of clusters is unknown.

As a test application, we classify human actions, such as walking or jumping [5], given video sequences of the scene. In the most basic setting, one can try to represent each action by a single Hidden Markov Model. However, in reality, human action instantiations, show great variability depending upon the subject, the environment, the presence of others, viewing angle, intention etc. Hence a single representative prototype may not suffice. It is best then to envision clusters of action instances, which basically results in mixture models for each action. We experimentally show that learning mixture models of actions improves the classification accuracy.

Each action sequence is represented by a matrix, where each column gives the features of motion within different blocks of the subject’s bounding box. These observations are assumed to be independently occurring, in other words, spatial statistical dependence, if any, between block observations is not modeled, as otherwise the state-space becomes combinatorially large, making the problem intractable. For this reason, we derive a spectral algorithm for multiple observation Hidden Markov Model.
2 Multiple Observation HMMs

A multiple observation HMM is defined as follows:

\[ p(x_{1:T}, h_{1:T} | \theta) = \prod_{t=1}^{T} p(h_t | h_{t-1}) \prod_{i=1}^{I} p(x_{i,t} | h_t) \]  

(1)

Here, \( x_{i,t} \in \{1, \ldots, W\} \) are observations for time \( t \in \{1, \ldots, T\} \) and multiple observations are indexed by \( i \in \{1, \ldots, I\} \). \( h_{1:T} \) denotes the hidden state sequence, where \( h_t \in \{1, \ldots, M\} \). We denote the model parameters as \( \theta = (\pi, O_t, A) \), which are defined as follows:

\[ p(h_1 = u|h_0) = p(h_1 = u) = \pi_u \]  

(2)

\[ p(h_t = u|h_{t-1} = j) = A_{u,j} \]  

(3)

\[ p(x_{i,t} = k|h_t = u) = O_{t,k,u} \]  

(4)

Given the parameters, the observed data likelihood is given by:

\[ p(x_{1:T} | \theta) = \sum_{h_{1:T}} p(x_{1:T}, h_{1:T} | \theta) \]  

(5)

When the parameters \( \pi, O_t, A \) are regarded as \( M \times 1, W \times M \) and \( M \times M \) matrices, we can write the observed data likelihood as follows: (Note that \( 1^T \) is a row vector only consisting of 1’s.)

\[ p(x_{1:T} | \theta) = 1^T \left( \prod_{t=1}^{T} A \prod_{i=1}^{I} L_{x_{i,t}} \right) \pi \]  

(6)

where \( L_{x_{i,t}} = \text{diag}(O_i(x_{i,t}, :)) \). \( O_i(x_{i,t}, :) \) means that we take the \( x_{i,t} \)th row of \( O_i \).

2.1 Spectral learning for Multiple Observation HMM’s

Inspired by the work in [6], which derive a spectral method for learning arbitrary latent trees, we derive spectral learning algorithm for multiple observation HMMs. To make the likelihood in (6) computable in terms of observable quantities, we write it as follows:

\[ p(x_{1:T} | \theta) = 1^T \left( U_{1} O_{1} \right)^{-1} \prod_{t=1}^{T} \left( U_{t} A (U_{t}^T O_{t})^{-1} \right) \prod_{i=1}^{I} \left( U_{pi} O_{pi} \right) L_{x_{i,t}} \left( U_{pi}^T O_{pi} \right)^{-1} \left( U_{pi}^T O_{pi} \right)^{-1} \pi \]  

(7)

We choose the columns of \( U_i \) as the left singular vectors of \( p(X_{i,t}, X_{\lambda i,t}) \), computed by taking its thin SVD. We adopt the ordering notation in [6] for observations. Note that \( \lambda i = \text{mod}(i, I) + 1 \) and \( \rho i = \text{mod}(i - 2, I) + 1 \). For example, for \( I = 5 \), \( \lambda 3 = 1 \), \( \lambda 4 = 4 \), \( \rho 1 = 5 \), \( \rho 4 = 3 \). The required terms \( 1^T, A, L_{x_{i,t}}, \pi \) can be computed in terms of observable quantities as follows: \(^1\)

\[ \tilde{A} = U_{1}^T p(X_{1,t}, X_{1,1}) (U_{1} p(X_{1,t}, X_{1,1}))^\dagger \]  

(8)

\[ \tilde{L}_{x_{i,t}} = U_{pi}^T p(X_{pi,t}, X_{\lambda i,t}) (U_{pi} p(X_{pi,t}, X_{\lambda i,t}))^\dagger \]  

(9)

\[ \tilde{1}^T = p(X_{1,t}) (U_{1}^T p(X_{1,t}, X_{1,1}))^\dagger \]  

(10)

\[ \tilde{\pi} = U_{1}^T p(X_{1}) \]  

(11)

Writing the observation \( x \) as \( X \) with a capital letter in eqs. (8), (9), (10), (11) means that we consider all possible outcomes of \( p(x = l), l \in \{1, \ldots, W\} \) to form a vector \( p(X) \) (or similarly a matrix or tensor). Note that in eqn. (7), we chose the term between \( 1^T \) and \( \tilde{A} \) as \( U_{1}^T O_{1} \) for convenience (We could have also chosen \( U_{1}^T O_{1} \) for instance). Since the choice for this term induces a particular choice for the estimation in (8) for \( \tilde{A} \), one has to choose this term carefully so that \( p(X_{1,2}, X_{\lambda 1,1}) \) contains substantial state transition information.

\(^1\)Proofs can be found from http://ycemsubakan.files.wordpress.com/2012/07/proofsupplement.pdf
3 Mixture of HMM’s

In the mixture context, we refer to the \(n\)’th observed sequence \(x_{1:1:T:n}\) as \(x_n\). The sequence index is \(n \in \{1, \ldots, N\}\). We denote the parameters of the \(k\)’th cluster with \(\theta_k\), which are drawn from a prior distribution \(H\). We denote the cluster indicator variables for each \(x_n\) with \(r_n \in \{1, \ldots, K\}\).

We use the notation \(r_n^{-n}\), which denotes all \(r_{1:N}\), except the \(n\)’th item. The mixing proportions are given with \(c\). \(K\) cluster finite mixture model for HMMs can be written as follows:

\[
\begin{align*}
    c &\sim \text{Dirichlet}(\alpha/K, \ldots, \alpha/K), \\
    \theta_k &\sim H, \\
    r_n | c &\sim \text{Discrete}(c), \\
    x_n | r_n, \theta_{1:K} &\sim \prod_k p(x_n | \theta_k)^{r_n=k}
\end{align*}
\]

Note that \(p(x_n | \theta_k)\) quantifies the likelihood of the \(n\)’th sequence for the HMM in the \(k\)’th cluster. We can easily integrate the mixing proportions out. Then, the full conditional of the cluster indicator variables can be derived as follows:

\[
p(r_n = k | r_{1:N}^{-n}, x_{1:N}) = \frac{N_k^{-n} + \alpha / K}{N + \alpha - 1} \int p(x_n | \theta_k)p(\theta_k | \{x_l : l \neq n, r_n = k\})d\theta_k
\]

\(N_k^{-n}\) denotes the number of sequences present in the \(k\)’th cluster except the \(n\)’th sequence. As the eq. (12) suggests, we have to integrate parameters \(\theta_k\) out using the posterior of \(\theta_k\) for a collapsed Gibbs sampler. Unfortunately this integral is intractable since we also have to integrate out the latent state sequences \(h_{1:T}\). If we assume that the posterior of the parameters is already very peaked, we can make the following approximation:

\[
\int p(x_n | \theta_k)p(\theta_k | \{x_l : l \neq n, r_n = k\})d\theta_k \approx p(x_n | \theta_k^{\text{ML}})
\]

Note that the maximum likelihood (ML) estimate in the right hand side can be easily computed from the given sequences using the spectral learning algorithm suggested in section 2. Given a particular partitioning \(r_{1:N}\), using the sequences in cluster \(k\), one can uniquely estimate a \(\theta_k\), for each cluster \(k\). This means that each partitioning automatically induces a unique \(\theta_k\) for each cluster. To conclude, we basically approximate the predictive density in (12) just by partitioning the data and computing the empirical statistics required in (8), (9), (10) from the given sequences. A Gibbs sampler that makes use of this idea is given in algorithm 1.

**Algorithm 1** Gibbs Sampler for Finite HMM Mixtures

```plaintext
Randomly initialize \(r_{1:N}\)
for \(e = 1 \rightarrow E\) do
    for \(n = 1 \rightarrow N\) do
        \(r_n^* = r_n\)
        sample \(r_n \sim p(r_n | \text{others})\)
        if \(r_n^* \neq r_n\) then
            Update \(\theta_{r_n^*}\) and \(\theta_{r_n}\)
        end if
    end for
end for
```

In algorithm 1, \(r_n^*\) is the cluster assignment at the previous iteration \(e-1\). If \(r_n \neq r_n^*\), for a data item \(x_n\), we simply update the cluster memberships and the parameters accordingly: We simply extract the counts of \(x_n\) from the previous cluster \(r_n^*\) and add then to the cluster \(r_n\). Note that this algorithm requires us to know \(K\) beforehand. To avoid this need, we formulate below an infinite mixture of Hidden Markov models.

3.1 Infinite Mixture of HMM’s

To formulate an infinite mixture of HMMs, we consider the full conditionals of the indicator variables. If we take \(K\) to infinity, and lump all the empty clusters together [7], then we obtain the following full conditional for an assignment to an empty cluster:

\[
p(r_n = K+1 | r_{1:N}^{-n}, x_{1:N}) = \frac{\alpha}{N + \alpha - 1} \int p(x_n | \theta_{K+1})p(\theta_{K+1})d\theta_{K+1}
\]

(14)
According to (14), we have to integrate out the parameters using the prior density. This integral is also intractable. To approximate it, we do the following: We gather all the sequences \( x_{1:N} \) to be clustered and estimate a likelihood for \( x_n \). This somewhat approximates the marginal density of the data \( p(x_n) \): The integral in (14) is a weighted average of the likelihood, weighted by our prior knowledge. We approximate this average with the likelihood of a sequence \( x_n \), estimated using all of the sequences \( x_{1:N} \). Mathematically, the approximation is:

\[
\int p(x_n | \theta_{K+1}) p(\theta_{K+1}) d\theta_{K+1} \approx p(x_n | \theta^{all})
\]

We denote the parameter set learned from all of the sequences \( x_{1:N} \) by \( \theta^{all} \). For an assignment to an already occupied cluster, eq. (12) remains the same except that in the first term, \( \alpha/K \) vanishes. Combining these results, the overall algorithm can be summarized as follows: As in algorithm 1, we start with a random partitioning \( r_{1:N} \). At each iteration, for each data item, we first compute the approximations to the full conditionals in (12) and (14). If the sampled \( r_n \) value is lower than the number of active clusters \( K \), we update the cluster \( r_n \) by adding the counts of sequence \( x_n \) to eqs. (8), (9), (10), (11) and update the old cluster by extracting the counts of this sequence. If the sampled \( r_n \) is \( K + 1 \), we increase \( K \) by one. We form a new cluster using the statistics of the sequence \( n \). Again, we extract counts of \( x_n \) from the old cluster. If during this process, one of the clusters gets empty, we decrease \( K \) by one, and delete the empty cluster and relabel the existing ones.

4 Application: Human Action Recognition from Videos

We tested our clustering algorithm on KTH Action Database [8]. This dataset contains 600 sequences of 25 people collected in four sessions. There are in total 6 actions: Boxing, hand clapping, hand waving, running, jogging and walking. In each action frame we place a bounding box on the human body. Then, this box is divided into \( I \) blocks, indexed by \( i \) in section 2. We set \( I = 9 \) in our experiments. We characterize each block by the number of spatio-temporal interest points (STIPs). In other words, for any given time \( t \), the number of STIPs in each box are the observations \( x_{1:I,t} \). The box and interest points are automatically extracted by using the off-the-shelf code in [9]. For each class, we use 64 sequences for training and 36 sequences for testing. First, we trained a separate multiple observation HMM per action class using EM and spectral method described in section 2. In test, we assign a test sequence to the highest likelihood action class. We achieved 70% and 68% test accuracies with EM and spectral method, respectively. Confusion matrices are given in Table 1(a) and 1(b). Secondly, we did clustering and trained more than one HMM for each action class using the algorithm suggested in section 3. During testing, we did the class assignments according to the likelihoods averaged over the clusters for each class. Consequently, the performance has increased by 6% to reach 74%, as detailed in the confusion matrix in Table 1(c). This result indicates that doing clustering in the training phase helps the algorithm to distinguish some of the classes more accurately.

5 Conclusions and Future Work

We have proposed a sequence clustering method which makes use of the spectral learning for HMMs. Experimental results in human action recognition indicate that, clustering in the training phase helps us to get better test accuracy as compared to the conventional classification based on EM. The classification accuracies appear modest compared to the state of the art. However, the goal of this work was to provide a proof of the clustering concept. We plan to pursue this work by incorporating the much richer STIP features such as HoG, HoF [10] into our model to improve the accuracy. Also as a methodological advancement, we will try to incorporate full Bayesian learning in spectral methods to exactly compute or at least better approximate the integrals in eqs. (13), (15).
References


