Incremental training of Markov mixture models

Andreas Kakoliris and Konstantinos Blekas
Department of Computer Science, University of Ioannina
P.O.Box 1186, Ioannina 45110 - GREECE
E-mail: {akakolir,kblekas}@cs.uoi.gr

Abstract

This paper presents an incremental approach for training a Markov mixture model to a set of sequences of discrete states. Starting from a single Markov model that captures the background information, at each step a new component is added to the mixture in order to improve the data fit. This is done by making at first an exploration of a relevant parametric space to initialize the new component, based on an extension of the k-means algorithm. Then, by performing a two-stage scheme of the EM algorithm, the new component is optimally incorporated to the body of the current mixture. To assess the effectiveness of the proposed method, we have conducted experiments with several data sets and we make a performance comparison with the classical mixture model.

1 Introduction

Sequential data analysis is an important research area with a wide range of applications, such as web log mining, bioinformatics, speech recognition, robotics, natural language processing and many others. Several attempts have been made on the task of clustering sequential data of discrete states [8, 3, 1]. In model-based approaches the most efficient scheme used is through mixture models [6]. Therefore, each cluster is described by a generative model and the aim of clustering is to find an optimal set of such models in order to best fit the data. Markov models [7] provide a reasonable tool for modeling sequential data. Recently, many efforts have been also made to address visualization capabilities of clustering approaches using Markov models [3, 5, 11, 9]. Thus, we can display the behavior of sequences within clusters and provide an explanatory analysis for the dynamics of data. In most of the these approaches, the EM algorithm [4] is used for estimating the parameters of the Markov mixture models. Since the EM algorithm has the drawback to be dependent on the initial values of their parameters, several schemes have been introduced to reduce

this effect. In [3] for example, a noisy-marginal scheme is proposed by perturbing the parameters of a single model to obtain K sets of models. An alternative approach is presented in [8], where an agglomerative clustering technique is applied together with an appropriate distance function for sequences, in order to initialize the K parametric models of the mixture.

In this paper we propose an incremental approach for achieving an efficient training of Markov mixture models. Borrowing strength from recent advances on mixture models [10, 2], our method performs a more systematic exploration of the parameter space and simultaneously tries to eliminate the dependence of the EM algorithm on the initialization. The method starts with a single Markov model that fits all sequences, and sequentially adds new components to the mixture following three major steps. At first we initialize the new inserted Markov model by searching over a parametric likelihood space. The latter is specified by a set of candidate models that have been constructed through the use of an adaptation of the classical k-means algorithm for treating sequential data. This is the initialization step. Then, we perform a partial EM scheme allowing the adjustment of only the new model parameters. Finally, the new component is incorporated to the current mixture and the EM algorithm can be applied to fit the new mixture model with the data. The procedure stops when reaching a number of K components. We have tested our training method on a suite of artificial and real benchmarks taking into account a variety of cases with excellent results. During the experiments we have evaluated the proposed scheme in terms of its capability to fit the data and measure its robustness. Comparative results have been also obtained with the classical Markov mixture model under two schemes for initialization.

In section 2 we give the basic scheme of the Markov mixture models, while section 3 describes the proposed approach for incremental training. In section 4 we present experimental results and finally, in section 5 we give some concluded remarks.

2 Markov mixture models

Consider a dataset $X = \{X_1, \ldots, X_N\}$, where each data point $X_i = (X_{il})_{l=1}^{L_i}$ is a sequence of length L_i observed states. We further assume that each state takes values from a discrete alphabet of M symbols, i.e. $X_{il} \in \{1, \ldots, M\}$. The problem of clustering is to find a grouping of the set X into K clusters, such that each cluster will contain similar in nature sequences. Every cluster is represented by a generative model that fits well the observed data that supports.

Mixture models represent an efficient architecture that is particularly suitable for clustering. It assumes that data have been generated from a mixture model with K components according to the following density function

$$f(X_i|\Theta_K) = \sum_{j=1}^{K} \pi_j p(X_i|\theta^j) \ , \tag{1}$$

where $\Theta_K = \{\pi_j, \theta^j\}$ denotes the set of the mixture parameters. In particular, the parameters π_j determine the prior probabilities of the K components satisfying that $\sum_{j=1}^{K} \pi_j = 1$. Each component captures a data generation mechanism through a probability distribution function $p(X_i|\theta^j)$, whose parameters θ^j are unknown. A natural way to modeling sequential data is through the first-order Markov model. It is defined by the initial states probabilities $\theta^j_{0m} = p(X_{i1} = m)$, as well as the transition probabilities $\theta^j_{nm} = p(X_{i,l+1} = m|X_{il} = n)$ from every state to another. Thus, each model parameter θ^j is a stochastic matrix with a set of $M+1$ multinomial distributions (rows), holding that $\sum_{m=1}^{M} \theta^j_{nm} = 1$, $\forall n = 0, \ldots, M$. The density function for the jth component can be then written as

$$p(X_i|\theta^j) = \theta^j_{0,X_{i1}} \prod_{l=1}^{L_i-1} \theta^j_{X_{il},X_{i,l+1}} == \prod_{m=1}^{M} (\theta^j_{0m})^{\gamma_i(m)} \prod_{n=1}^{M} \prod_{m=1}^{M} (\theta^j_{nm})^{\delta_i(n,m)}, \tag{2}$$

where $\gamma_i(m) = \begin{cases} 1 & \text{if } X_{i1} = m \\ 0 & \text{otherwise} \end{cases}$ and $\delta_i(n,m)$ defines the number of transitions from state n to state m in the sequence X_i. Following the Bayes rule, we can then associate every sequence X_i to the cluster j that has the maximum posterior probability value $p(j|X_i) = \frac{\pi_j p(X_i|\theta^j)}{f(X_i|\Theta_K)}$.

Clustering the set X into K clusters is then equivalent to estimating the mixture model parameters Θ_K. This in turn leads to maximize the log-likelihood function arisen from the model. We can also introduce non-informative Dirichlet priors, which are conjugate for the multinomial distributions $\{\theta^j_{nm}\}_{m=1}^{M}$, of the form

$$p(\theta^j_n|a^j_n) = \frac{\Gamma(\sum_{m=1}^{M}(a^j_{nm}+1))}{\prod_{m=1}^{M}\Gamma(a^j_{nm}+1)} \prod_{m=1}^{M} (\theta^j_{nm})^{a^j_{nm}} \ , \tag{3}$$

where the parameter a^j_n is a M-vector with components $a^j_{nm} > 0$ and $\Gamma(a)$ is the Gamma function. Adding Dirichlet priors in effect introduces pseudo-counts for regularization purposes. Therefore, the derived *maximum a-posteriori* (MAP) log-likelihood function is given by

$$L(X|\Theta_K) = \sum_{i=1}^{N} \log f(X_i|\Theta_K) + \sum_{j=1}^{K} \sum_{n=0}^{M} \log p(\theta^j_n|a^j_n) \ . \tag{4}$$

It must be noted that in our case the Dirichlet parameters a_n^j were common to every component j and set equal to a small proportion (e.g. 10%) of the corresponding maximum likelihood (ML) estimated multinomial parameter values of the single Markov model that fits the data set X (using relative frequencies of states). The latter from now on it will be referred to as "single ML-estimated Markov model".

The EM algorithm [4] is an efficient framework for estimating the mixture model parameters. It requires the computation of the conditional expectation values z_{ij} (posterior probabilities) of the hidden variables during the E-step

$$z_{ij}^{(t)} = \frac{\pi_j^{(t)} p(X_i|\theta^{j\,(t)})}{\sum_{j'=1}^{K} \pi_{j'}^{(t)} p(X_i|\theta^{j'\,(t)})} \ , \tag{5}$$

while at the M-step the maximization of the following log-likelihood function of the complete dataset is performed

$$\begin{aligned} Q(X, \Theta_K|\Theta_K^{(t)}) = \sum_{i=1}^{N}\sum_{j=1}^{K} z_{ij}^{(t)} \{\log \pi_j + \sum_{m=1}^{M} \gamma_i(m) \log \theta_{0m}^j + \\ + \sum_{n=1}^{M}\sum_{m=1}^{M} \delta_i(n,m) \log \theta_{nm}^j\} + \sum_{j=1}^{K}\sum_{n=0}^{M}\sum_{m=1}^{K} a_{nm}^j \log \theta_{nm}^j \ . \end{aligned} \tag{6}$$

This leads to the following updated equations for the mixture model parameters:

$$\pi_j^{(t+1)} = \frac{\sum_{i=1}^{N} z_{ij}^{(t)}}{N} \ , \tag{7}$$

$$\theta_{nm}^{j\ (t+1)} = \begin{cases} \dfrac{\sum_{i=1}^{N} z_{ij}^{(t)} \gamma_i(m) + a_{0m}^j}{\sum_{i=1}^{N} z_{ij}^{(t)} + \sum_{m'=0}^{M} a_{0m'}^j} & n = 0 \\ \dfrac{\sum_{i=1}^{N} z_{ij}^{(t)} \delta_i(n,m) + a_{nm}^j}{\sum_{i=1}^{N} z_{ij}^{(t)} \sum_{m'=1}^{M} \delta_i(n,m') + \sum_{m'=1}^{M} a_{nm'}^j} & n > 0 \end{cases} \tag{8}$$

The EM algorithm guarantees the convergence of the log-likelihood function to a local maximum satisfying all the constraints of the parameters. However, due to its local nature, the enormous dependence on the initial parameter values may drastically effect its performance [6]. In the next section we present a greedy approach for building a Markov mixture models that eliminates this problem of poor initialization.

3 Incremental mixture training

The proposed method starts with a simple model with one component that comes from the single ML-estimated Markov model of the whole dataset X. At each step a new component is added to the mixture by performing a combined scheme of searching for good initial estimators and for fine local tuning its parameters. It must be noted that a same in nature strategy have been also presented in [10] and [2] for Gaussian mixture models and for discovering patterns in biological sequences, correspondingly.

Lets assume that we have already constructed a k-length mixture model with Θ_k parameters. By inserting a new component, the resulting mixture can take the following form

$$f(X_i|\Theta_k, \pi^*, \theta^*) = (1-\pi^*)f(X_i|\Theta_k) + \pi^* p(X_i|\theta^*) \ . \tag{9}$$

where $\pi^* \in (0,1)$ is the prior probability. The above scheme can be viewed as a two-component mixture model, where the first one captures the current mixture with density $f(X_i|\Theta_k)$ and the second one the new Markov model that has a density function $p(X_i|\theta^*)$ with an unknown stochastic matrix θ^*.

If we fix the parameters of the old mixture model Θ_k we can then maximize the resulting log-likelihood function $\mathcal{L}_k$ of the above two-components mixture with respect only to the new model parameters $\{\pi^*, \theta^*\}$

$$\mathcal{L}_k(X|\pi^*, \theta^*) = \sum_{i=1}^{N} \log\{(1-\pi^*)f(X_i|\Theta_k) + \pi^* p(X_i|\theta^*)\} + + \sum_{n=0}^{M} \log p(\theta_n^*|a_n) \ . \tag{10}$$

In this light, we can applied the EM algorithm for estimating only the parameters of the new model, namely as *partial EM*. This results into obtaining the following update equations:
at the E-step

$$\zeta_i^{(t)} = \frac{\pi^{*(t)} p(X_i|\theta^{*(t)})}{(1-\pi^{*(t)})f(X_i|\Theta_k) + \pi^{*(t)} p(X_i|\theta^{*(t)})} \ , \tag{11}$$

and at the M-step

$$\pi^{*(t+1)} = \frac{\sum_{i=1}^{N} \zeta_i^{(t)}}{N}, \tag{12}$$

$$\theta_{nm}^{*\ (t+1)} = \begin{cases} \dfrac{\sum_{i=1}^{N} \zeta_{ij}^{(t)} \gamma_i(m) + a_{0m}}{\sum_{i=1}^{N} \zeta_{ij}^{(t)} + \sum_{m'=0}^{M} a_{0m'}} & n = 0 \\ \dfrac{\sum_{i=1}^{N} \zeta_{ij}^{(t)} \delta_i(n,m) + a_{nm}}{\sum_{i=1}^{N} \zeta_{ij}^{(t)} \sum_{m'=1}^{M} \delta_i(n,m') + \sum_{m'=1}^{M} a_{nm'}} & n > 0 \end{cases} \tag{13}$$

The above partial EM steps offer more flexibility to the general scheme and simplifies the estimation problem during the insertion of a new Markov model to the mixture. At a second stage, the new component can be incorporated to the body of the current mixture and construct a new mixture $f(X_i|\Theta_{k+1})$ with $k+1$ components. The EM algorithm can be then used to maximize the log-likelihood function $L(X|\Theta_{k+1})$ in the new parameter space Θ_{k+1}, following Eqs. 5-8. The mixture parameters are initialized from the solution of the partial EM, i.e. $\pi_{k+1}^{(0)} = \pi^*$, $\pi_j^{(0)} = (1-\pi^*)\pi_j$, $\forall j = 1, \ldots, k$, and $\theta_{k+1}^{(0)} = \theta^*$. This iterative procedure is repeated until the desired order K of the Markov mixture model is reached.

3.1 Initializing new model parameters

As it is clear from the previous analysis, we must provide good initial estimates of the new component parameters in order to conduct the partial EM scheme. This can be accomplished by establishing a parametric search space through a set of K_m candidate Markov models $\{\phi_j\}_{j=1}^{K_m}$. In particular, we perform one step of the partial EM, after initializing the multinomial parameters of the new model with a candidate Markov model ($\theta^{*(0)} = \phi_j$) and the prior probability π^* with the typical value $\pi^{*(0)} = \frac{1}{k+1}$. Finally, we select the solution that corresponds to the maximum value of the log-likelihood function $\mathcal{L}_k$ (Eq. 10) for initializing the parameters $\{\pi^*, \theta^*\}$.

In our study we have used an extension of the known k-means algorithm in order to create such a set of candidate models. In the general case, the k-means algorithm aims at finding a partition of K_m disjoint clusters C_j to a set of N objects, so as the overall sum of distances between cluster centers μ_j and objects X_i is minimized. In order to adopt this framework to sequential data we need to make some modifications. At first, a distance function between two sequences X_i and X_k. must be provided so as to encapsulate an appropriate measure of dissimilarity between data. For this purpose we have used a

symmetrized log-likelihood distance defined as [8]

$$D(i,k) = \frac{1}{2}\{\log p(X_i|\vartheta_k) + \log p(X_k|\vartheta_i)\} \ ,$$

where the parameters ϑ_i denote the single ML-estimated Markov model specified by each sequence X_i. Furthermore, at each step t of the k-means algorithm we re-estimate the new center $\mu_j^{(t+1)}$ of every cluster C_j by finding the *medoid* sequence among the sequences that currently supports, i.e. $\mu_j^{(t+1)} = \arg\min_{X_i \in C_j^{(t)}} \sum_{X_k \in C_j^{(t)}} D(i,k)$. At the end of the algorithm, we assign a Markov model ϕ_j to each cluster C_j found by finding the single (ML-estimated) Markov model that fits well all sequences associated with this cluster. The above scheme allows us to create a pool of K_m candidate models capable for initializing the parameter θ^* during the partial EM steps. As experimental study has shown, the proposed method is not sensitive to the value of K_m. A small proportion of the population size of sequences N (e.g. 5%) is enough for constructing a rich search space that can provide good initial estimators.

3.2 The proposed algorithm

The proposed incremental approach for training a mixture of K Markov models can be summarized in the following algorithmic form.

- Set $\Theta_1 = \{\theta^1, \pi_1 = 1\}$ using the single ML-estimated Markov model from the data set X.

  Use k-means to provide K_m candidate Markov models ϕ_j.

- for $k = 1 : K-1$

  1. $\forall j = 1, \ldots, K_m$ perform one partial EM step (Eqs.11-13) by setting $\pi^{*(0)} = \frac{1}{k+1}$ and $\theta^{*(0)} = \phi_j$. Select the solution that has the maximum log-likelihood value $\mathcal{L}_k$ (Eq. 10).
  2. Perform partial EM (Eqs.11-13) until convergence and estimate new model parameters $\{\pi^*, \theta^*\}$.
  3. Set $\Theta_{k+1} = \Theta_k \cup \{\pi_{k+1}, \theta_{k+1}\}$, where ${\pi_{k+1}}^{(0)} = \pi^*$, ${\pi_j}^{(0)} = (1-\pi^*)\pi_j \ \forall j \leq k$, ${\theta^{k+1}}^{(0)} = \theta^*$.
  4. Perform general EM (Eqs.5-8) to maximize $L(X|\Theta_{k+1})$.

M symbols	*model*	K components			
		5	8	10	15
5	*IMM*	100%	100%	100%	100%
	RMM	80.5%	67.5%	50%	25.5%
	KMM	56%	49.5%	31.5%	7%
8	*IMM*	100%	100%	100%	100%
	RMM	67%	47.5%	35.5%	11.5%
	KMM	50%	32%	19%	7%
10	*IMM*	100%	100%	100%	100%
	RMM	74.5%	45.5%	28.5%	10%
	KMM	47%	28.5%	15.5%	2.5%
12	*IMM*	100%	100%	100%	100%
	RMM	70%	42%	26.5%	9.5%
	KMM	35%	25%	11%	2.5%
15	*IMM*	100%	100%	100%	100%
	RMM	75%	35.5%	21%	6%
	KMM	52%	20.5%	9.5%	1%

Table 1: Percentage of times the correct model was detected by each one of the three methods IMM, KMM and RMM. The results were taken using several values for the alphabet size M and the number of components K.

4 Experimental results

Several experiments have been made in an attempt to evaluate the performance of the proposed incremental training approach, namely as IMM. Comparative results have been also taken using two methods for initializing Markov mixture models:

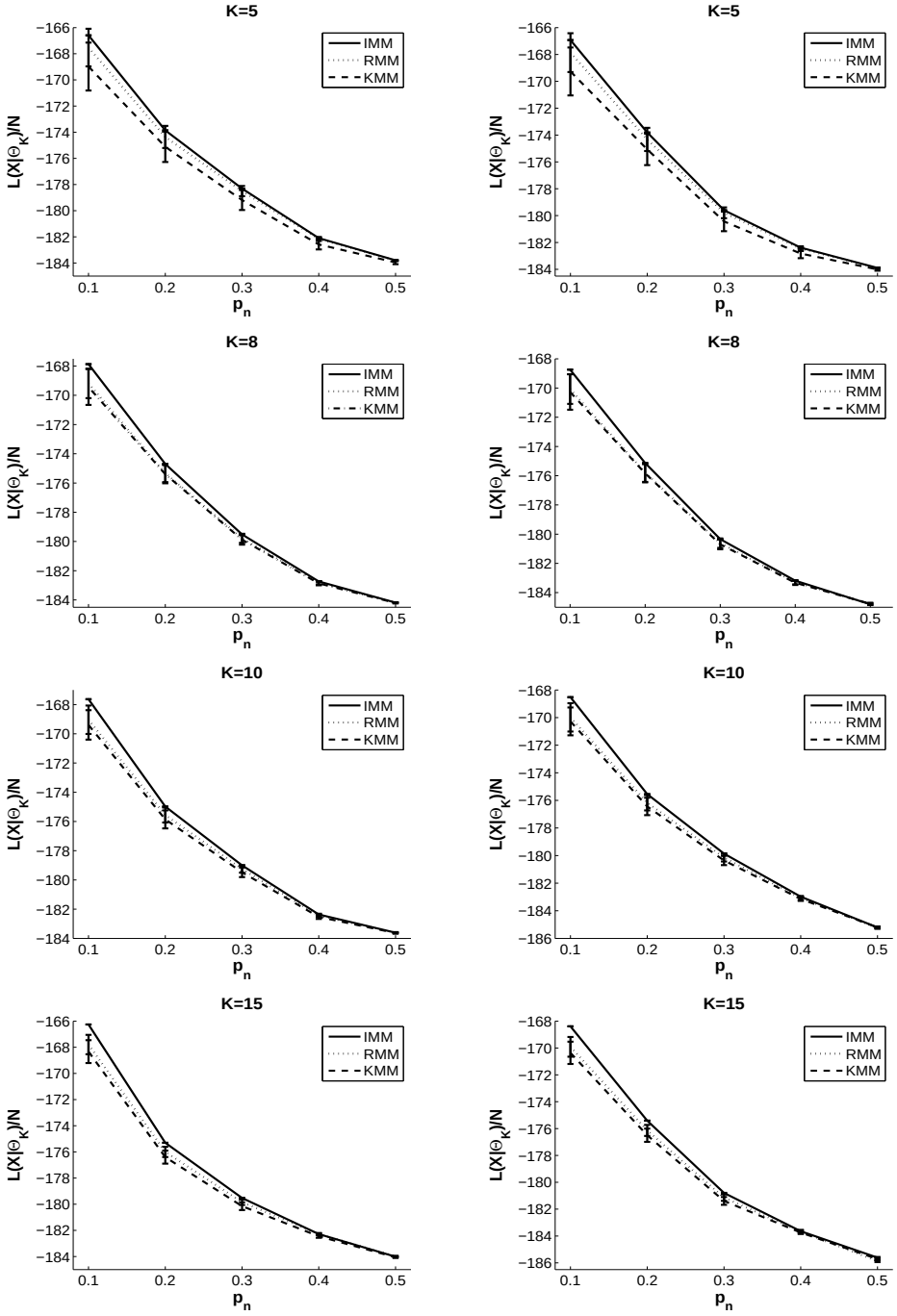
- the RMM that follows the initialization scheme presented in [3], which creates K noisy copies from the single ML-estimated Markov model.
- the KMM that first determines K Markov models using the k-means algorithm as described previously ($K_m = K$), and then uses this information to initialize the EM algorithm.

Since both last methods depends on the initialization, twenty (20) runs of the EM algorithm were performed for each data set. We kept records of the mean value and the standard deviation of the log-likelihood function. On the other hand, the proposed IMM model was executed only once for fitting a Markov mixture model to each data set.

The first series of experiments was carried out using artificial data in order to evaluate the robustness of our method. We created artificial sequences by sampling from a K Markov mixture model using a discrete alphabet of M different symbols. Totally, we generated $N = 1000$ number of sequences of length between 50 and 100 states ($L_i \in [50, 100]$). Since we are aware of the true mixture model that best fit the experimental sequences, we used this information to evaluate each method. In particular, we created ten (10) different datasets for several values of the pair (M, K) and calculated the percentage of times each met hod found the log-likelihood value that corresponds to the true model. Table 4 summarizes the results from the experiments in these artificial datasets. The weakness of both the RMM and KMM approaches in obtaining the global maximum value, especially in higher model order values is obvious. On the other hand, the proposed IMM approach was able to correctly estimate the true model in all cases.

Another series of experiments with artificial sequences has been made using sets of K patterns of length 50 from an alphabet of $M = 10$ symbols. The data generation mechanism follows the next scheme: A pattern is randomly selected at first, and then a noisy copy of it, according to a probability p_n for mutation common to every pattern site, is located at a random position in the sequence. The rest non-pattern sites are filled uniformly from the same alphabet. Following this scheme, two sets of $N = 1000$ sequences of length $L_i \in [50, 100]$ were created: one used for training and another one for testing. As it is clear, this clustering problem is more difficult since the Markov property exists only locally in the sequences and under different levels of noise. Again, for each pattern family we created ten (10) different datasets and we evaluated each method in terms of the log-likelihood value found. Figure 1 illustrates the results obtained with four values for the number of components $K = \{5, 8, 10, 15\}$ and five different levels of noise $p_n = \{0.1, 0.2, 0.3, 0.4, 0.5\}$. In each diagram, the error bars display the standard deviation of the difference between the log-likelihood values of the true model that is known and those reached by each method. Our method was able to achieve a high degree of noise tolerance since always managed to discover the correct model, with no deviation from it, even for extremely noisy datasets.

Moreover, we have tested our method on the msnbc.com web navigation dataset [3]. This is a collection of sequences that corresponds to page-category views of users during twenty-four hour period. Totally there are $M = 17$ such page categories. In our study we have considered only a subset of the whole collection containing 4600 sequences of length between $[40, 100]$. We randomly divided it into a training set (40% of the total size) and a set for testing (the rest of them). Our method was executed only once until reaching



(a) Training set (b) Test set

Figure 1: The log-likelihood values found by the three comparative methods as a function of noise parameter p_n. The experiments were made with artificial datasets created from various K noisy patterns.

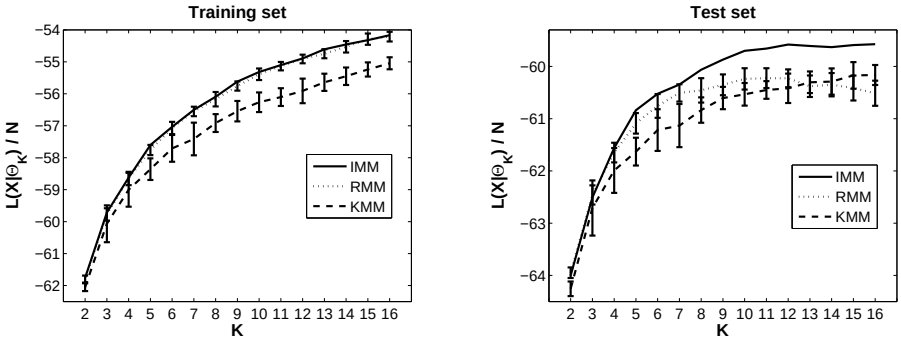


Figure 2: Application of the three methods to the msnbc.com dataset. The log-likelihood values are calculated for several values of K in the training and the test set.

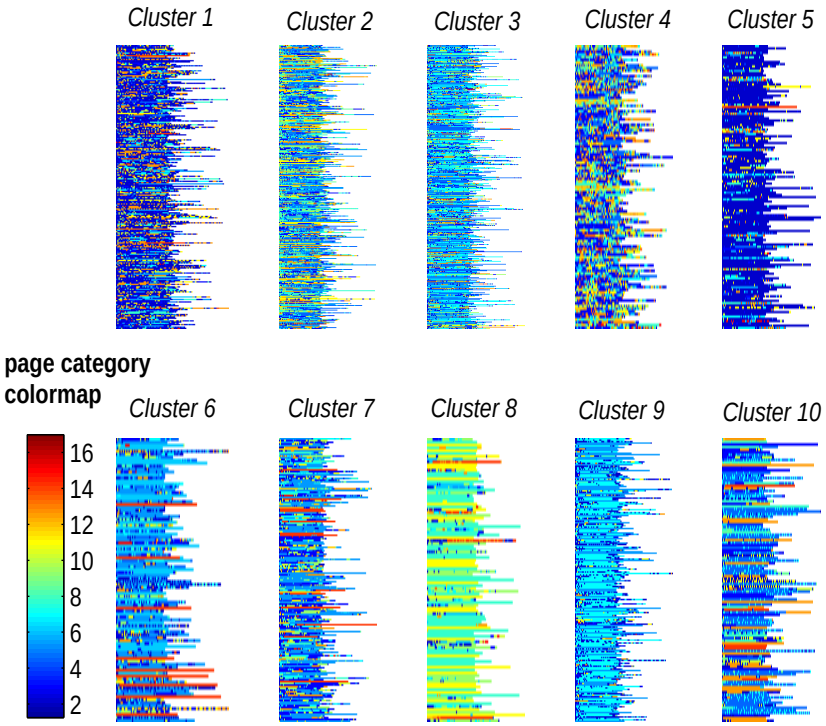


Figure 3: Visualization of the clustering results of our method on the msnbc.com data as generated by $K = 10$ components. Each image represents a cluster of user sessions in a colored raw form.

$K = 16$ components in the training set. For the other two approaches we calculated the mean value and the standard deviation of the log-likelihood function by executing 20 times the EM algorithm per value of K. Figure 2 shows the results depicted by each method in both sets. Our training method showed an improvement performance and achieved better generalization capabilities on the test set in comparison with the other two approaches. Finally, in Figure 3 we give a visualization of the clustering results obtained by our method to the training set in the case of $K = 10$. Each one of the ten images corresponds to a cluster, where we have convert every page category with a unique color. Therefore, each sequence is represented as a raw of colored squares. In the case of applications such as web log mining, the visualization techniques provide a useful mechanism for identifying and interpreting user behavior patterns for the Web data [3, 5].

5 Conclusions

In this paper we have presented an incremental strategy for training Markov mixture models to a set of sequences of discrete states. The approach sequentially adds components to a mixture model by performing a combined scheme of the EM algorithm. In order to initialize properly each new component, we have also set up an efficient parameter search space of Markov models. Experiments on a variety of benchmarks have shown the ability of our method to improve the data fit and also demonstrated its generalization capability. Further research can be made in the light of employing the proposed approach in mixtures of hidden Markov models, since they can be seen as more general probabilistic models for sequential data. Applying also other techniques for creating candidate models and efficient parameter search spaces constitutes one of our future directions.

References

[1] M. Bicego, V. Murino, and M. Figueiredo. Similarity-based classification of sequences using hidden Markov models. *Pattern Recognition*, 37:2281–2291, 2004.

[2] K. Blekas, D.I. Fotiadis, and A. Likas. Greedy mixture learning for multiple motif discovering in biological sequences. *Bioinformatics*, 19(5):607–617, 2003.

[3] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Model-based clustering and visualization of navigation patterns on a web site. *Data Mining and Knowledge Discovery*, 7(4):399–424, 2003.

[4] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Statist. Soc. B*, 39:1–38, 1977.

[5] E. Manavoglu, D. Pavlov, and C.L. Giles. Probabilistic user behavior models. In *IEEE International Conference on Data Mining (ICDM'03)*, pages 203–210, 2003.

[6] G.M. McLachlan and D. Peel. *Finite mixture models*. New York: John Wiley & Sons, Inc., 2001.

[7] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286, 1989.

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

[9] P. Tino, A. Kaban, and Y. Sun. A generative probabilistic approach to visualizing sets of symbolic sequences. In *ACM SIGKDD - International Conference on Knowledge Discovery and Data Mining - KDD-2004*, pages 701–706, 2004.

[10] N. Vlassis and A. Likas. A greedy EM algorithm for Gaussian mixture learning. *Neural Processing Letters*, 15(1):77–87, 2002.

[11] A. Ypma and T.M. Heskes. Automatic categorization of web pages and user clustering with mixtures of hidden Markov models. In *WEBKDD 2002 - Mining web data for discovering usage patterns and profiles*, pages 35–49, Berlin, 2003.