Feature Weighted Rival Penalized EM for Gaussian Mixture Clustering: Automatic Feature and Model Selections in a Single Paradigm*

Yiu-ming Cheung and Hong Zeng

Department of Computer Science, Hong Kong Baptist University, Hong Kong, SAR, China
ymc@comp.hkbu.edu.hk, hzeng@comp.hkbu.edu.hk

Abstract. Recently, the Rival Penalized Expectation-Maximization (RPEM) algorithm (Cheung 2004 & 2005) has demonstrated its outstanding capability to perform the model selection automatically in the context of density mixture models. Nevertheless, the RPEM is unable to exclude the irrelevant variables (also called *features*) from the clustering process, which may degrade the algorithm's performance. In this paper, we adopt the concept of feature salience (Law et al. 2004) as the feature weight to measure the relevance of features to the cluster structure in the subspace, and integrate it into the RPEM algorithm. The proposed algorithm identifies the irrelevant features and estimates the number of clusters automatically and simultaneously in a single learning paradigm. Experiments show the efficacy of the proposed algorithm on both synthetic and benchmark real data sets.

1 Introduction

Density mixture clustering has been widely applied to a variety of scientific fields such as neural networks, image processing, pattern recognition, and so forth. In such a clustering, each component of a density mixture represents the density distribution of a corresponding cluster of data, and thus clustering can be viewed as identifying the dense regions of the input densities. In general, Expectation Maximization (EM) algorithm [1] has provided a general solution for the parameter estimate in a density mixture model. However, the EM algorithm needs to pre-specify an appropriate number of components in a mixture, which, unfortunately, is difficult or even impossible from the practical viewpoint.

More recently, the *Rival Penalized Expectation-Maximization* (RPEM) algorithm has been developed from the learning framework, namely *Maximum Weighted Likelihood* [6, 7]. This algorithm makes the components in a density mixture compete with each other as given an input (also called *an obervation* interchangeably). Not only are the associated parameters of the winner (i.e. the winning mixture component) updated to adapt to the input, but also all rivals' parameters are penalized with the strength proportional to the corresponding posterior density probabilities. Compared to the EM,

* This work was fully supported by the Research Grant Council of Hong Kong SAR under Projects: HKBU 2156/04E and HKBU 210306.

Y. Wang, Y. Cheung, and H. Liu (Eds.): CIS 2006, LNAI 4456, pp. 1018–1028, 2007.

such a rival penalization mechanism enables the RPEM to gradually fade out the redundant densities in a density mixture. The experiments in [6,7] have shown its outstanding performance on model selection, i.e., determining the number of mixture components (also called *the number of clusters* hereinafter). Nevertheless, analogous to the EM, the RPEM performs the clustering by using all variables (also called *features* interchangeably) within the whole input space without a mechanism to exclude the irrelevant variables, i.e., some variables without contribution to the cluster structure, from the clustering process. Subsequently, the performance of RPEM may deteriorate if some irrelevant variables exist.

Earlier methods for feature selection in clustering can be roughly fallen into two categories: the *feature filter* approaches and the *wrapper* approaches. The feature filter approaches, e.g. principal component analysis (PCA) [2, 3, 4], try to pick out the most influential subset of features, which reflects the characteristics of the original data set. Such an approach may significantly reduce the dimensionality, but the clustering algorithm is not involved in the feature extraction. Consequently, the extracted features may not be well suited to the follow-up clustering algorithm. In contrast, the wrapper approaches utilize a clustering algorithm to evaluate the qualities of each candidate feature subsets [3, 5] generated via a combinatorial search. The classification accuracy of such an approach may be improved in comparison to the filter approaches, but its computation is rather laborious. Essentially, these two kinds of feature selection methods are prone to find a sub-optimal solution because they perform the feature and model selections, which are closely related each other, in two separate steps. Actually, a better solution can be achieved provided that the feature and model selections are performed in a single learning paradigm. In the literature, some works have been done along this promising way. For example, Huang et al. [8] present a k-means type algorithm that weights the importance of each feature in the clustering process. The numerical results have shown that this algorithm can successfully identify noisy variables with comparatively small weights. Nevertheless, this method may be sensitive to the initial cluster centroids and the initial weights. Furthermore, its performance depends on the choice of parameter β, whose value is, however, determined by trial and error. Furthermore, Law et al. [9] adopt a definition of feature salience with respect to the independence of its distribution to a given cluster, and integrate the Minimum Message Length (MML) criterion to the log-like likelihood. Eventually, an EM-like algorithm has been developed to automatically determinate the number of clusters and the feature weights. In addition, Constantinopoulos et al. [10] utilize the same model proposed by [9], but present a variational Bayesian learning for estimating the feature weights and cluster parameters. Paper [10] has shown its superiority in the presence of sparse data by adopting the Bayesian framework other than the statistical MML criterion.

In this paper, we adopt the concept of feature salience [9] to measure the relevance of each feature to cluster structure. Subsequently, we utilize a general probability distribution model for the Gaussian mixture proposed by [9], and integrate it into the Maximum Weighted Likelihood (MWL) framework, through which we develop a variant of the RPEM, namely Feature Weighted RPEM (FW-RPEM) algorithm. Not only is this new algorithm able to make a model selection analogous to the RPEM, but also weights the features based on their relevance to the cluster structure so that the irrelevant features

can be gradually excluded from the clustering process. As a result, an appropriate cluster structure in the subspace of inputs can be found. Experimental results have shown the efficacy of the proposed algorithm in comparison to the existing methods.

2 Overview of the RPEM Algorithm

Suppose an observation comes from a mixture of k^* probability density functions (pdf):

$$p(\mathbf{x}|\Theta^*) = \sum_{j=1}^{k^*} \alpha_j^* p(\mathbf{x}|\theta_j^*), \quad \sum_{j=1}^{k^*} \alpha_j^* = 1, \quad \text{and} \quad \forall 1 \le j \le k^*, \alpha_j^* > 0, \tag{1}$$

where the pdf $p(\mathbf{x}|\theta_j^*)$ is the j^{th} component of the mixture, $\Theta^* = \{\alpha_j^*, \theta_j^*\}_{j=1}^{k^*}$ denotes the set of the true parameters in the mixture model, and k^* is the true number of components. The main learning purpose is to estimate the parameters Θ^* from N i.i.d. observations, denoted as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, where each observation $\mathbf{x}_t$ is a column vector of d features, written as $x_{1t}, x_{2t}, \ldots, x_{dt}$.

The Rival Penalized EM (RPEM) algorithm [7] has been developed from the MWL framework via maximizing the following weighted likelihood:

$$Q(\Theta; \mathbf{X}_N) = \frac{1}{N} \sum_{t=1}^{N} \mathcal{M}(\Theta; \mathbf{x}_t), \quad \mathbf{X}_N = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \tag{2}$$

with

$$\begin{aligned} \mathcal{M}(\Theta; \mathbf{x}_t) &= \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) \ln p(\mathbf{x}_t|\Theta) \\ &= \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) \ln[\alpha_j p(\mathbf{x}_t|\theta_j)] \\ &\quad - \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) \ln h(j|\mathbf{x}_t, \Theta), \end{aligned} \tag{3}$$

where $\Theta = \{\alpha_j, \theta_j\}_{j=1}^{k}$ and k are an estimate of Θ^* and k^*, respectively. Furthermore, we have

$$p(\mathbf{x}_t|\Theta) = \sum_{j=1}^{k} \alpha_j p(\mathbf{x}_t|\theta_j), \tag{4}$$

$$p(\mathbf{x}_t|\theta_j) = p(x_{1t}, \ldots, x_{lt}, \ldots, x_{dt}|\theta_j), \tag{5}$$

and $\forall 1 \le j \le k$ $(k \ge k^*)$, $\alpha_j \ge 0$, $\sum_{j=1}^{k} \alpha_j = 1$. Also,

$$h(j|\mathbf{x}_t, \Theta) = \frac{\alpha_j p(\mathbf{x}_t|\theta_j)}{p(\mathbf{x}_t|\Theta)} \tag{6}$$

is the posterior probability that $\mathbf{x}$ belongs to the jth component in the mixture. In (3), $g(j|\mathbf{x}_t, \Theta)$ is a designable weight with:

$$\sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) = \zeta, \tag{7}$$

and

$$\forall j, \lim_{h(j|\mathbf{x}_t,\Theta)\to 0} g(j|\mathbf{x}_t, \Theta) \ln h(j|\mathbf{x}_t, \Theta) = 0, \tag{8}$$

where ζ is a positive constant. In [7], they are constructed by:

$$g(j|\mathbf{x}_t, \Theta) = (1 + \varepsilon_t) I(j|\mathbf{x}_t, \Theta) - \varepsilon_t h(j|\mathbf{x}_t, \Theta) \tag{9}$$

with

$$I(j|\mathbf{x}, \Theta) = \begin{cases} 1 \text{ if } j = c \equiv \arg\max_{1\le i\le k} h(i|\mathbf{x}, \Theta); \\ 0 \; j = r \neq c. \end{cases} \tag{10}$$

where the ε_t is a small positive quantity. Paper [7] learns Θ towards maximizing (2) via the following alternating steps:

- **E-step:** Given an input $\mathbf{x}_t$ and Θ^{old}, compute $h(j|\mathbf{x}_t, \Theta^{old})$ and $g(j|\mathbf{x}_t, \Theta^{old})$ through (6) and (9), respectively.
- **M-step:** Fixing $h(j|\mathbf{x}_t, \Theta^{\text{old}})$ and $g(j|\mathbf{x}_t, \Theta^{\text{old}})$, update Θ along the direction of maximizing (2) by the gradient ascent approach, i.e.

$$\Theta^{new} = \Theta^{old} + \eta \left. \frac{\partial \mathcal{M}(\Theta; \mathbf{x}_t)}{\partial \Theta} \right|_{\Theta^{old}}. \tag{11}$$

It has been shown in [7] that the RPEM can automatically select the number of components by fading out the redundant densities from a density mixture. Nevertheless, analogous to the most existing clustering algorithms, the RPEM assumes that each feature has the same importance to the intrinsic cluster structure, which, however, may not be always true from the practical viewpoint. In the next section, we therefore present the FW-RPEM algorithm that is able to identify the cluster structure by estimating the feature weights and perform the model selection simultaneously.

3 The Feature Weighted Rival Penalized EM Algorithm

Without loss of generality, we suppose that the features in each observation are independent each other, and the contribution of each dimension is invariant among all the clusters. Considering not all the features of an observation are important, we therefore adopt the measure in [9] to weight the relevancy of these features. That is, the weight is denoted as $W = [w_1, \ldots, w_d]^T$ with $0 \le w_l \le 1, \forall 1 \le l \le d$, where w_l represents

the probability that the lth feature is relevant to all the clusters. The irrelevant features have little contribution to a given cluster in the subspace, thus their distributions may be common to all these clusters in this case. Then, the probability density function of a general Gaussian mixture model can be written below as in [9]:

$$p(\mathbf{x}|\Theta) = \sum_{j=1}^{k} \alpha_j \prod_{l=1}^{d} [w_l p(x_l|\theta_{lj}) + (1 - w_l) q(x_l|\lambda_l)] \tag{12}$$

where $p(x_l|\theta_{lj}) = G(m_{lj}; S_{lj}^2)$ denotes a Gaussian density function of x_l with the mean m_{lj}, and standard deviation S_{lj}. $q(x_l|\lambda_l)$ is the common density of the lth feature with the parameter λ_l if it is irrelevant. The prior knowledge about the density distribution of the irrelevant feature can be Gaussian distribution, uniform distribution, and so forth. In this paper, we let it be a Gaussian for a general purpose, i.e., $q(x_l|\lambda_l) = G(cM_l, cS_l^2)$. Subsequently, we define the full parameter set of the general Gaussian mixture model as $\Theta = \{\{\alpha_j\}_{j=1}^k, \Phi\}$ and $\Phi = \{\{\theta_{lj}\}_{l=1,j=1}^{d,k}, \{w_l\}_{l=1}^d, \{\lambda_l\}_{l=1}^d\}$. Note that

$$p(x_{lt}|\Phi) = w_l p(x_{lt}|\theta_{lj}) + (1 - w_l) q(x_{lt}|\lambda_l) \tag{13}$$

is a coupling form with two possible density models for each feature, where the feature weight w_l acts as a regulator to determine which distribution is more appropriate to describe the feature. By putting (13) into (3), we then obtain:

$$\begin{aligned}
\mathcal{M}(\mathbf{x}_t; \Theta) &= \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) \ln[\alpha_j p(\mathbf{x}_t|\Phi)] \\
&\quad - \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) \ln h(j|\mathbf{x}_t, \Theta) \\
&= \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) \ln\{\alpha_j \prod_{l=1}^{d} [w_l p(x_{lt}|\theta_{lj}) \\
&\quad + (1 - w_l) q(x_{lt}|\lambda_l)]\} \\
&\quad - \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta) \ln h(j|\mathbf{x}_t, \Theta),
\end{aligned} \tag{14}$$

where we let the weight function $g(j|\mathbf{x}_t, \Theta)$ be:

$$g(j|\mathbf{x}_t, \Theta) = I(j|\mathbf{x}_t, \Theta) + h(j|\mathbf{x}_t, \Theta). \tag{15}$$

which satisfies the conditions in (7) and (8).

Consequently, we can estimate the parameter set Θ towards maximizing $\mathcal{M}(\mathbf{x}_t; \Theta)$ of (14) via the adaptive learning algorithm, namely Feature weighted RPEM (FW-RPEM) algorithm, whose learning mechanism is analogous to the RPEM algorithm. In the implementation of FW-RPEM, we have noticed that $\{\alpha_j\}_{j=1}^k$ must satisfy the

constraint of $\sum_{j=1}^{k} \alpha_j = 1$. To circumvent the complicated constraint optimization, we alternatively let

$$\alpha_j = \frac{\exp(\beta_j)}{\sum_{i=1}^{k} \exp(\beta_i)}, \quad \text{for} \quad \forall 1 \le j \le k. \tag{16}$$

As a result, we update $\{\beta_j\}_{j=1}^{k}$s instead of $\{\alpha_j\}_{j=1}^{k}$ like the RPEM in [7].

In summary, the FW-RPEM algorithm is implemented in the following steps after initializing Θ:

– **Step 1:** Calculate $h(j|\mathbf{x}_t, \Theta^{\text{old}})$ and $g(j|\mathbf{x}_t, \Theta^{\text{old}})$:

$$h(j|\mathbf{x}_t, \Theta^{\text{old}}) = \frac{\alpha_j^{\text{old}} \prod_{l=1}^{d} [w_l^{\text{old}} p(x_{lt}|\theta_{lj}^{\text{old}}) + (1 - w_l^{\text{old}}) q(x_{lt}|\lambda_l^{\text{old}})]}{\sum_{j=1}^{k} \alpha_j^{\text{old}} \prod_{l=1}^{d} [w_l^{\text{old}} p(x_{lt}|\theta_{lj}^{\text{old}}) + (1 - w_l^{\text{old}}) q(x_{lt}|\lambda_l^{\text{old}})]} \tag{17}$$

$$g(j|\mathbf{x}_t, \Theta^{\text{old}}) = I(j|\mathbf{x}_t, \Theta^{\text{old}}) + h(j|\mathbf{x}_t, \Theta^{\text{old}}). \tag{18}$$

– **Step 2:** Update the parameter set $\{\{\theta_{lj}\}_{l=1,j=1}^{d,k}, \{\lambda_l\}_{l=1}^{d}, \{\alpha_j\}_{j=1}^{k}, \{w_l\}_{l=1}^{d}\}$ along the direction of maximizing $\mathcal{M}(\mathbf{x}_t; \Theta)$ by fixing $h(.|.)$ and $g(.|.)$ obtained in **Step 1** for each observation:

$$\begin{aligned}
\beta_j^{new} &= \beta_j^{old} + \eta_\beta \left.\frac{\partial \mathcal{M}(\mathbf{x}_t; \Theta)}{\partial \beta_j}\right|_{\Theta^{old}} \\
&= \beta_j^{old} + \eta_\beta [g(j|\mathbf{x}_t, \Theta^{\text{old}}) - \alpha_j^{old}], \\
m_{lj}^{new} &= m_{lj}^{old} + \eta \left.\frac{\partial \mathcal{M}(\mathbf{x}_t; \Theta)}{\partial m_{lj}}\right|_{\Theta^{old}} \\
&= m_{lj}^{old} + \eta g(j|\mathbf{x}_t, \Theta^{\text{old}}) h'(1|x_{lt}, \Phi^{\text{old}}) \frac{x_{tl} - m_{lj}^{\text{old}}}{(S_{lj}^{\text{old}})^2}, \\
S_{lj}^{new} &= S_{lj}^{old} + \eta \left.\frac{\partial \mathcal{M}(\mathbf{x}_t; \Theta)}{\partial S_{lj}}\right|_{\Theta^{old}} \\
&= S_{lj}^{old} + \eta g(j|\mathbf{x}_t, \Theta^{\text{old}}) h'(1|x_{lt}, \Phi^{\text{old}}) \frac{(x_{tl} - m_{lj}^{\text{old}})^2 - (S_{lj}^{\text{old}})^2}{(S_{lj}^{\text{old}})^3}, \\
cM_l^{new} &= cM_l^{old} + \eta \left.\frac{\partial \mathcal{M}(\mathbf{x}_t; \Theta)}{\partial cM_l}\right|_{\Theta^{old}} \\
&= cM_l^{old} + \eta \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta^{\text{old}}) h'(2|x_{lt}, \Phi^{\text{old}}) \frac{x_{lt} - cM_l^{old}}{(cS_l^{old})^2}, \\
cS_l^{new} &= cS_l^{old} + \eta \left.\frac{\partial \mathcal{M}(\mathbf{x}_t; \Theta)}{\partial cS_l}\right|_{\Theta^{old}}
\end{aligned}$$

$$= cS_l^{old} + \eta \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta^{\text{old}}) h'(2|x_{lt}, \Phi^{\text{old}}) \frac{(x_{lt} - cM_l^{old})^2 - (cS_l^{old})^2}{(cS_l^{old})^3},$$

$$w_l^{new} = w_l^{old} + \eta \left. \frac{\partial \mathcal{M}(\mathbf{x}_t; \Theta)}{\partial w_l} \right|_{\Theta^{old}}$$

$$= w_l^{old} + \eta \sum_{j=1}^{k} g(j|\mathbf{x}_t, \Theta^{\text{old}}) \left[\frac{h'(1|x_{lt}, \Phi^{\text{old}})}{w_l^{old}} - \frac{h'(2|x_{lt}, \Phi^{\text{old}})}{1 - w_l^{old}} \right],$$

where

$$h'(1|x_{lt}, \Phi^{\text{old}}) = \frac{w_l^{old} p(x_{lt}|\theta_{lj}^{old})}{w_l^{old} p(x_{lt}|\theta_{lj}^{old}) + (1 - w_l^{old}) q(x_{lt}|\lambda_l^{old})},$$

$$h'(2|x_{lt}, \Phi^{\text{old}}) = \frac{(1 - w_l^{old}) q(x_{lt}|\lambda_l^{old})}{w_l^{old} p(x_{lt}|\theta_{lj}^{old}) + (1 - w_l^{old}) q(x_{lt}|\lambda_l^{old})}.$$

Note that the learning rate of β_js should be chosen as $\eta_\beta < \eta$ to alleviate the sensitivity of α_js to the small fluctuation of β_js (we suggest $\eta_\beta = 0.1\eta$). Furthermore, the values of w_ls should be essentially controlled within the range of $[0, 1]$, but the update of w_ls in Step 2 may not. To avoid this awkward situation, we can use a soft-max function (e.g. see (16)) to transform w_ls to the new variables, say ϱ_ls, analogous to the case of α_js and β_js, whereby the constraints of w_ls can be automatically satisfied. Here, we alternatively adopt a simple procedure. That is, we set w_l at 0.001 when $w_l < 0.001$, and set it at 0.999 when $w_l > 0.999$ during the learning process.

Step 1 and Step 2 are repeated for each observation until Θ converges. As a result, one can identify those features that are more relevance to cluster structure than the others, and the corresponding component parameters can be picked out from $\{m_{lj}, S_{lj}\}_{l=1,j=1}^{d,k}$ with $\{w_l\}_{l=1}^{d}$ and $\{\alpha_j\}_{j=1}^{k}$ as guides.

4 Experimental Results

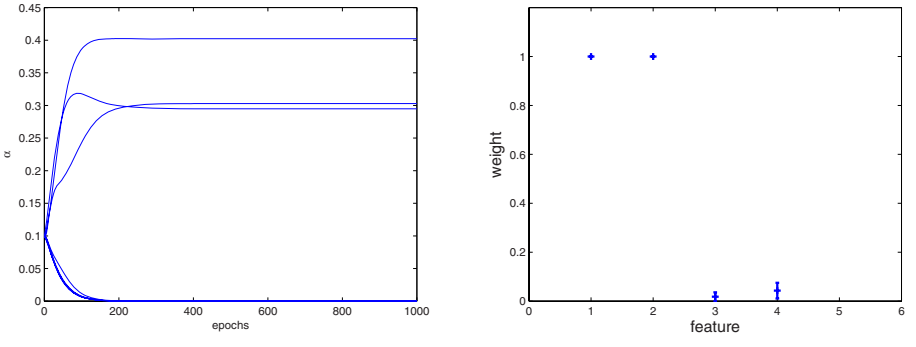
4.1 Experiment 1

This experiment was to investigate the performance of the FW-RPEM on identifying the cluster structure in the presence of irrelevant features. We first generated 2-dimension $1,000$ synthetic data from a mixture of 3 Gaussian components: $G\left[\mathbf{x}|\begin{pmatrix}1\\1\end{pmatrix}; \begin{pmatrix}0.1 & 0.0\\0.0 & 0.1\end{pmatrix}\right]$, $G\left[\mathbf{x}|\begin{pmatrix}1\\5\end{pmatrix}; \begin{pmatrix}0.1 & 0.0\\0.0 & 0.1\end{pmatrix}\right]$, $G\left[\mathbf{x}|\begin{pmatrix}5\\5\end{pmatrix}; \begin{pmatrix}0.1 & 0.0\\0.0 & 0.1\end{pmatrix}\right]$,with 0.3, 0.4, 0.3 being their mixture proportions, respectively. Then we drew 2, 48 and 98 features from the Gaussian noise $G(2, 5^2)$ and appended them to the bivariate Gaussian mixture, yielding a 4-dimension (low-dimension), a 50-dimension (medium-dimension), and a 100-dimension (high-dimension) data sets, respectively. Further, we initialized k at 10, and all α_js and w_ls

were set at $\frac{1}{k}$ and 0.5, respectively. The remaining parameters were initialized as following (in MATLAB form):

$$
\begin{aligned}
&[dim, N] = size(\mathbf{x});\\
&index = randperm(N);\\
&\mathbf{m} = \mathbf{x}(:, index(1:k));\\
&\mathbf{s} = repmat(sqrt(var(\mathbf{x}'))', 1, k);\\
&\mathbf{cM} = mean(\mathbf{x}, 2);\\
&\mathbf{cS} = sqrt(var(\mathbf{x}'))';
\end{aligned}
$$

The learning rates were $\eta = 10^{-3}$, $\eta_\beta = 10^{-4}$. The algorithm was performed on these three data sets 10 times each, and the numerical results are depicted in Fig. 1-3, respectively. Fig. 1(a)-3(a) show that the three out of 10 α_js has converged to give a good estimate of the true values, meanwhile all the remaining α_js have converged towards zero. That is, the FW-RPEM has successfully identified three components in all cases we have tried so far. Furthermore, as expected, the feature weights of the first two dimensions were close to 1, while the rest dimensions were assigned close to 0 as shown in Fig. 1(b)-3(b). That is, the proposed algorithm has correctly identified a large number of noisy features in the input space.



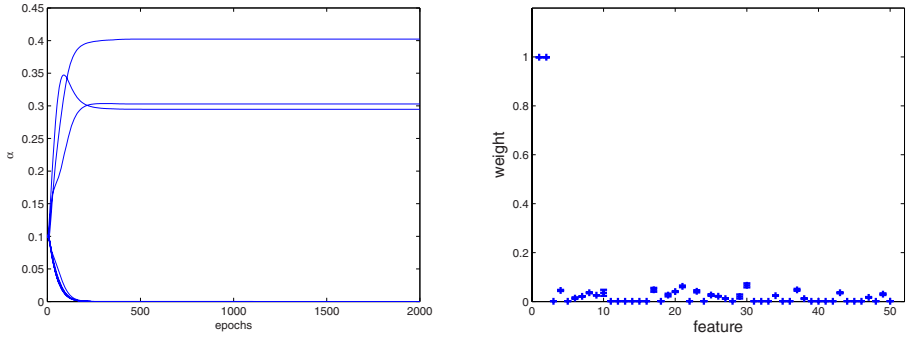
(a) (b)

Fig. 1. Results of the experiment on low-dimension data set. (a) The learning curve of α_js of a typical run. (b) The feature weights, where the average values are marked with "+", and the standard deviations over ten runs are presented by the error bars around the mean values.

4.2 Experiment 2

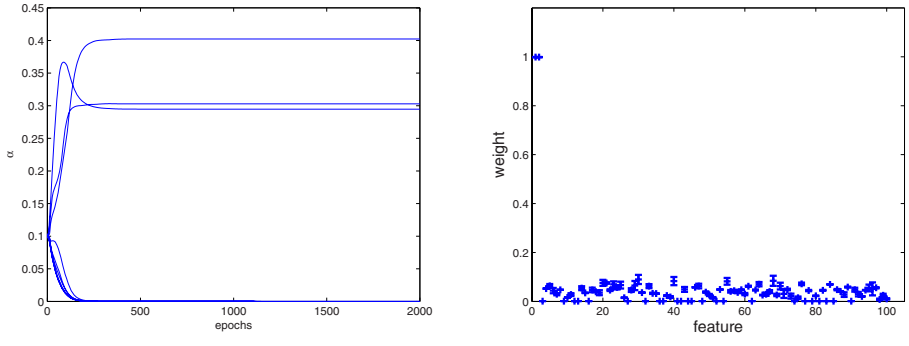
Besides the synthetic data, we also conducted a number of experiments on three well-known databases from UCI Machine Learning Repository [11]:

– *Wine*. There are 178 data points. The analysis is to determine the quantities of 13 constituents found in each of the three types of wines.



(a) (b)

Fig. 2. The result of the experiment on medium-dimension data set. (a) The learning curve of α_js of a typical run. (b)The feature weights, where the average values are marked with "+", and the standard deviations over ten runs are presented by the error bars around the mean values.



(a) (b)

Fig. 3. Results of the experiment on high-dimension data set. (a) The learning curve of α_js of a typical run. (b) The feature weights, where the average values are marked with "+", and the standard deviations over ten runs are presented by the error bars around the mean values.

– *Australian credit card*. This data set consists of 653 credit card applications, and is classified to two classes: *approved* and *rejected* according to the first 14 features.
– *Ionosphere*. There are 351 instances and 34 attributes. The task is to classify the collections from radar into 2 classes denoting obstruction existing or not in the ionosphere.

We utilized a 50% Jackknifing procedure to separate the original data set into the training and testing sets. The training set was formed by randomly picking data from the original data set to its half size, and the remaining points were reserved for testing. The process was repeated 20 times, yielding 20 pairs of different training and testing

Table 1. Comparison on the performance for each algorithm on the real data

Data set	*FW-RPEM*		*RPEM*		*method in [9]*	
	$\%error \pm std$	$Avg.No. \pm std$	$\%error \pm std$	$Avg.No. \pm std$	$\%error \pm std$	$Avg.No. \pm std$
Wine	6.18 ± 1.04	**fixed at 3**	9.06 ± 2.61	2.5 ± 0.71	6.73 ± 2.86	3.28 ± 1.44
Australian	20.63 ± 1.68	**fixed at 2**	45.15 ± 3.37	3.2 ± 0.447	45.94 ± 12.11	3.9 ± 1.21
Ionosphere	23.15 ± 6.20	2.8 ± 0.78	44.7 ± 6.39	1.5 ± 0.527	27.44 ± 10.38	4.7 ± 0.48

sets. After abandoning the class labels in the set, the proposed algorithm was conducted 20 times on each training set. We then utilized the trained model to classify the testing data and evaluated the accuracy by comparing the obtained labels with the ground-truth class labels. For comparison, we also performed the RPEM and Law's algorithm [9] individually on the same pairs of data sets with the same initializations. Their performances over 20 runs are all reported in Table 1. Also, Table 2 lists the average weighting results of the 20 runs on the *Wine* and *Australian credit card* sets, where the feature weights for *ionosphere* are excluded because the number of their features is too large to be listed in Table 2.

Table 2. The average weighting results of *FW-RPEM* on the real data

Features	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wine	0.999	0.652	0.331	0.750	0.102	0.999	0.999	0.279	0.999	0.999	0.288	0.999	0.999	
Australian	0.001	0.001	0.455	0.999	0.208	0.001	0.001	0.999	0.999	0.001	0.001	0.999	0.001	0.001

From Table 1, we can see that the error obtained from the FW-RPEM has been significantly reduced compared to the RPEM because the FW-RPEM is able to identify the features that have unequal contribution to the cluster structure (see Table 2), whereby an appropriate cluster structure can be found in a sub-space. Furthermore, the FW-RPEM also outperforms the algorithm in [9] with the smaller error, particularly on the data of Australian credit card and Ionosphere as listed in Table 1. Further, the algorithm in [9] is prone to use more "components" for the mixture. In contrast, the proposed algorithm not only produces a lower mismatch error, but also gives a better estimation to the number of mixture components.

5 Conclusion

In this paper, we have presented the FW-RPEM algorithm, which extends the RPEM algorithm to deal with the case when some irrelevant features exist in the input space. We have adopted the concept of feature salience as the feature weight to measure the relevance of features to the cluster structure in the subspace, and integrate it into the RPEM algorithm. Consequently, the FW-RPEM can identify the irrelevant features and perform model selection automatically and simultaneously in a single learning paradigm. The promising performance of the algorithm has been shown on both the synthetic data and real benchmark data sets.

References

1. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society (B) 39(1), 1–38 (1977)
2. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. John Wiley & Sons, Chichester (1973)
3. Dy, J.G., Brodley, C.E.: Visualization and Interactive Feature Selection for Unsupervised Data. In: Proceedings of ACM International Conference on Knowledge Discovery and Data Mining, pp. 360–364 (2000)
4. Fisher, D.H.: Knowledge Acquisition via Incremental Conceptual Clustering. Machine Learning, pp. 139–172 (1987)
5. Talavera, L.: Feature Selection and Incremental Learning of Probabilistic Concept Hierarchies. In: Proceedings of International Conference on Machine Learning, pp. 951–958 (2000)
6. Cheung, Y.M.: A Rival Penalized EM Algorithm towards Maximizing Weighted Likelihood for Density Mixture Clustering with Automatic Model Selection. In: Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04), vol. 4, pp. 633–636, Cambridge, United Kingdom (2004)
7. Cheung, Y.M.: Maximum Weighted Likelihood via Rival Penalized EM for Density Mixture Clustering with Automatic Model Selection. IEEE Transactions on Knowledge and Data Engineering 17(6), 750–761 (2005)
8. Huang, J.Z., Ng, M.K., Rong, H., Li, Z.: Automated Variable Weighting in k-means Type Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(5), 657–668 (2005)
9. Law, M.H.C., Figueiredo, M.A.T., Jain, A.K.: Simultaneous Feature Selection and Clustering Using Mixture Models. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(9), 1154–1166 (2004)
10. Constantinopoulos, C., Titsias, M.K., Likas, A.: Bayesian Feature and Model Selection for Gaussian Mixture Models. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(6), 1013–1018 (2006)
11. Blake, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases (1998), `http://www.ics.uci.edu/mlearn/MLRepository.html`.