Egyptian Informatics Journal (2012) **13**, 103–109

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

Unsupervised learning of mixture models based on swarm intelligence and neural networks with optimal completion using incomplete data

Ahmed R. Abas *

Department of Computer Science, University College in Leith, Umm Al-Qura University, Makka Al-Mukarrama, Leith, Saudi Arabia

Received 7 September 2011; revised 7 March 2012; accepted 26 March 2012
Available online 21 April 2012

KEYWORDS

Particle Swarm Optimization;
Optimal Completion Strategy;
Expectation–Maximization;
Locally-tuned general regression neural networks;
Finite mixture models;
Unsupervised learning;
Clustering;
Incomplete data

Abstract In this paper, a new algorithm is presented for unsupervised learning of finite mixture models (FMMs) using data set with missing values. This algorithm overcomes the local optima problem of the Expectation-Maximization (EM) algorithm via integrating the EM algorithm with Particle Swarm Optimization (PSO). In addition, the proposed algorithm overcomes the problem of biased estimation due to overlapping clusters in estimating missing values in the input data set by integrating locally-tuned general regression neural networks with Optimal Completion Strategy (OCS). A comparison study shows the superiority of the proposed algorithm over other algorithms commonly used in the literature in unsupervised learning of FMM parameters that result in minimum mis-classification errors when used in clustering incomplete data set that is generated from overlapping clusters and these clusters are largely different in their sizes.

1. Introduction

Finite mixture models (FMMs) is a semi-parametric method for density estimation and pattern recognition [1] that is used for fitting complex data distributions. This method has the advantage of the analytic simplicity and the advantage of the flexibility to model complex data distributions [2]. Parameters of FMM are usually estimated by the Expectation Maximization (EM) algorithm [3]. The EM algorithm cannot handle incomplete data sets. Therefore, several algorithms are proposed in the literature to modify the EM algorithm to estimate parameters of FMM using incomplete data set [4–6]. Part of these algorithms is affected by the occurrence of outliers in the data, and all of them are affected by the overlap among classes in the data space and the bias in generating the data

* Address: Department of Computer Science, Faculty of Computers and Informatics, Zagazig University, Zagazig, Egypt.
E-mail addresses: armohamed@uqu.edu.sa, arabas@zu.edu.eg

Peer review under responsibility of Faculty of Computers and Information, Cairo University.
http://dx.doi.org/10.1016/j.eij.2012.03.002

Production and hosting by Elsevier

from its classes [5]. When there is sufficiently large number of observed values, these algorithms outperform the EM algorithm combined with either the unconditional mean imputation or the conditional mean imputation of the missing values according to the classification performance of the resulting FMM [5]. It is shown that better results can be obtained by imputing missing values using the distribution of the input feature vectors rather than using a priori probability distribution function used in the FMM [5,7]. However, these modified EM algorithms have poor performance in learning FMM parameters when clusters of the input data set are largely overlapping and unbalanced in their numbers of feature vectors [5]. This is due to the fact that these algorithms estimate missing values in a certain feature vector only from either parameters or members of one component of the FMM to which this feature vector has the maximum posterior probability. The posterior probability is computed using complete values of each feature vector. This ignores the overlapping among clusters of the data set that is represented by FMM components. This overlapping means that feature vectors of the data set are generated from different components of FMM with different probabilities. Therefore, these algorithms produces inaccurate estimation of missing values which in turn leads to inaccurate estimation of FMM parameters learned from the whole data set.

In this paper, a new algorithm is proposed to overcome problems of the modified EM algorithms for unsupervised learning of the FMM parameters using incomplete data [4,5]. These problems are is the sensitivity to the occurrence of outliers in the data, the overlap among classes in the data space, and the bias in generating the data from its classes i.e., clusters of the input data set are unbalanced in their numbers of feature vectors. The proposed algorithm is less sensitive to the learning problems of the EM algorithm in cases such as the occurrence of outliers in the data set, the overlapping among data classes, and the unbalanced representation of data classes. The rest of this paper is organized as follows. Section 2 presents the proposed algorithm. Section 3 shows results of the comparison study that is carried out to evaluate the performance of the proposed algorithm. These results are discussed in Section 4. Conclusions are presented in Section 5.

2. The proposed algorithm

This section presents a new algorithm that overcomes problems of the modified EM algorithms [4,5] when dealing with a small incomplete data set that may contain outliers, overlapping clusters, or a large difference in the sizes of its clusters. In the rest of this paper, the modified EM algorithm proposed in [4] is referred to as the MEM algorithm while the modified EM algorithm proposed in [5] is referred to as the LGREM algorithm.

2.1. A swarm intelligence based EM algorithm

The EM algorithm [3] is an iterative algorithm that is used in producing maximum likelihood estimation of FMM parameters. However, this algorithm is sensitive to initialization because it stops at the nearest local maximum to the initial point of the likelihood function [8]. To overcome this problem the proposed algorithm uses a new proposed EM algorithm that is based on Particle Swarm Optimization (PSO) [9,10,11] to find the best estimation of FMM parameters that corresponds to the global maximum of the likelihood function. The proposed algorithm uses a swarm of several particles to find the best FMM that fits the input data set. Fig. 1 shows the particle structure.

Each particle in the swarm corresponds to a FMM whose parameters are learnt by the EM algorithm. Its structure contains the mixing weights, the centroid, the covariance matrix of a FMM, and the log-likelihood of this mixture model to represent the input data. After learning, the particle that has the maximum log-likelihood (LOGLH) is selected and its FMM is considered the best model for clustering the input data set. All swarm particles are generated such that the initial values of their mixing weights are equal and sum to one, centroids are feature vectors chosen randomly from the input data set, covariance matrixes are equal to $0.01\mathbf{I}_d$, where $\mathbf{I}_d$ is the identity matrix of order d and d is the number of features of the data set. This removes any outside bias towards certain components during learning of FMM parameters.

2.2. Locally-tuned general regression neural networks combined with Optimal Completion Strategy (OCS)

The OCS [12] allows the fuzzy c-means algorithm (FCM) [13] to be used in clustering incomplete data sets [14,15]. Based on this strategy, missing values in the data set are estimated in every iteration of the FCM algorithm using cluster centroids averaged with fuzzy membership values of feature vectors containing missing values in the data set to different clusters. In the proposed algorithm, locally-tuned general regression neural networks [5] (see, Eq. (2) for description) are modified using the OCS to estimate missing values in the data set (see, Eq. (3) for description). Missing values are estimated in every iteration using non-parametric estimation values obtained from the locally-tuned general regression neural networks [5] averaged by posterior probabilities of feature vectors containing missing values. This agrees with the basic assumption of clustering using FMM that is feature vectors in the data set are generated from different FMM components with different probabilities. The proposed algorithm uses locally-tuned general regression neural networks in estimating missing values [5] because this algorithm produces more accurate estimation of missing values and better learning of FMM parameters than using FMM parameters [4] in estimating missing values in the data set [5].

The proposed algorithm uses the PSO-based EM algorithm and the locally-tuned general regression neural networks combined the OCS to learn FMM parameters from incomplete data. The proposed algorithm is referred to as the POLGREM algorithm in the rest of this paper.

2.3. Description of the POLGREM algorithm

Suppose that the data set $\mathscr{R} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ consists of N feature vectors each of which is a vector in d-feature space such that each feature vector $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]^T$. This data set is

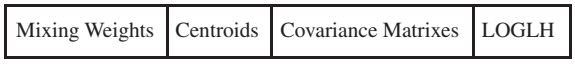

Figure 1 Particle structure.

assumed to be generated from a FMM of K multivariate normal distributions with unknown mixing coefficients $P(c)$, where $\sum_{c=1}^{K} P(c) = 1$, and $0 \leqslant P(c) \leqslant 1$. Let the probability density of the feature vector $\mathbf{x}_i$, which is fully observed, given the kth component in the FMM be $p(\mathbf{x}_i|\boldsymbol{\theta}_k) = N(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are the mean and the covariance matrix of this component. The total density of $\mathbf{x}_i$ from the FMM is then computed as $p(\mathbf{x}_i) = \sum_{c=1}^{K} P(c)p(\mathbf{x}_i|\boldsymbol{\theta}_c)$. In fitting the FMM, there are two types of missing values that have to be considered; the first type is the values of the cluster membership vector for each feature vector $\mathbf{z}_i = [z_{i1}, z_{i2}, \ldots, z_{iK}]^T$; the second type is the missing values in the different features of $\mathscr{R}$. To represent the second type let each feature vector in $\mathscr{R}$ be rewritten as $\mathbf{x}_i = (\mathbf{x}_i^o, \mathbf{x}_i^m)$, where o and m superscripts denote the observed and the missing values in this feature vector, respectively.

Step 1: Linearly scale the values of each feature in the input data set to make them lie in the interval [0,1]. This process removes the effect of different unit scales on different features, and therefore it is essential for finding the true cluster structure of this data set and the contribution of each feature in the creation of this structure. A comparison of several methods of data normalization for cluster analysis has shown the superiority of the linear scaling method over many other normalization methods before applying clustering algorithms in terms of the resulting cluster separation and error condition [16,17]. In addition, determine the optimum smoothing parameter σ for each incomplete feature in the data set using the leave-one-out cross validation method. This parameter is used in the locally-tuned general regression neural networks for estimating missing values from neighboring feature vectors in every cluster [5]. The group of fully observed features used in determining σ for a certain feature consists of both the complete features and the features that have smaller missing rates than this feature. The feature vectors that are used in the leave-one-out cross validation method should be observed in the entire fully observed feature group.

Step 2: As the EM algorithm converges toward the nearest local maximum of the likelihood function to the starting point [8], the EM algorithm is initialized several times using a swarm of particles each of which represents a FMM as explained in Section 2.1. The number of particles in the swarm is chosen arbitrary to be twenty in experiments presented in this paper. The FMM corresponding to the maximum log-likelihood function after convergence is selected as the best model for the input data set.

Step 3: In the *E-step*, compute the following quantities for each model component c in the FMM.

- The posterior probabilities vector $\mathbf{z}_i$ for all feature vectors in the data set $\mathscr{R}$.

$$\hat{z}_{ic} = \frac{\widehat{P}(c)p(\mathbf{x}_i^o|\boldsymbol{\theta}_c)}{\sum_{j=1}^{K} \widehat{P}(j)p(\mathbf{x}_i^o|\boldsymbol{\theta}_j)} \tag{1}$$

- The estimates of the missing values in $\mathscr{R}$ starting with those values in the feature that has the minimum missing rate. Multiple estimated values are computed for each missing value in the data set (see, Eq. (2)). Each estimated value is computed from one component in the FMM using the locally-tuned general regression neural networks [5]. The proposed algorithm, based on the OCS, estimates the missing value as the average of these multiple estimated values weighted by the posterior probabilities of the feature vector containing the missing value to different components of the FMM (see, Eq. (3)).

$$E(\hat{x}_{iq}^c|\mathbf{x}_i^o, \mathbf{R}) = \frac{\sum_{k=1}^{n_o} x_{kq} \exp\left(-\frac{D_k^2}{2\sigma_q^2}\right) r_{kc}}{\sum_{k=1}^{n_o} \exp\left(-\frac{D_k^2}{2\sigma_q^2}\right) r_{kc}} \tag{2}$$

$$E(\hat{x}_{iq}|\mathbf{x}_i^o, \mathbf{R}) = \sum_{c=1}^{K} \hat{z}_{ic} E(\hat{x}_{iq}^c|\mathbf{x}_i^o, \mathbf{R}) \tag{3}$$

where $\mathbf{R} = \{r_{kc}\}$ is the matrix of memberships for each feature vector $\mathbf{x}_k$ to each component c in the FMM such that r_{kc} is either one, if $\widehat{z}_{kc} > \widehat{z}_{kt}$ for all $t \neq c$, or zero otherwise, n_o is the number of feature vectors that are observed on both of the observed subspace for the feature vector $\mathbf{x}_i$ and the qth feature, and D_k^2 is the squared Euclidean distance between feature vectors $\mathbf{x}_k$ and $\mathbf{x}_i$ on the observed subspace of the feature vector $\mathbf{x}_i$.

After estimating its missing values, the qth feature is added to the group of fully observed features and this new group is then used in estimating the missing values in the next feature that has the minimum missing rate.

- The necessary statistics for the *M-step*.

$$E(z_{ic}x_{iq}|\mathbf{x}_i^o, \mathbf{R}) = \begin{cases} \hat{z}_{ic}x_{iq} & x_{iq} \in \mathbf{x}_i^o \\ \hat{z}_{ic}E(x_{iq}^c|\mathbf{x}_i^o, \mathbf{R}) & x_{iq} \in \mathbf{x}_i^m \end{cases} \tag{4}$$

$$E(z_{ic}x_{iq}x_{iq'}|\mathbf{x}_i^o, \mathbf{R}) = \begin{cases} \hat{z}_{ic}x_{iq}x_{iq'} & x_{iq}, x_{iq'} \in \mathbf{x}_i^o \\ \hat{z}_{ic}x_{iq}E(x_{iq'}^c|\mathbf{x}_i^o, \mathbf{R}) & x_{iq'} \in \mathbf{x}_i^m \\ \hat{z}_{ic}E(x_{iq}^c|\mathbf{x}_i^o, \mathbf{R})x_{iq'} & x_{iq} \in \mathbf{x}_i^m \\ \hat{z}_{ic}E(x_{iq}^c|\mathbf{x}_i^o, \mathbf{R})E(x_{iq'}^c|\mathbf{x}_i^o, \mathbf{R}) & x_{iq}, x_{iq'} \in \mathbf{x}_i^m \end{cases} \tag{5}$$

where $i, q, q' = 1, 2, \ldots, d$, E is the expectation operator.

Step 4: In the *M-step*, compute parameters of each component c in the FMM.

$$\widehat{P}(c) = \frac{1}{N}\sum_{j=1}^{N} \hat{z}_{jc} \tag{6}$$

$$\widehat{\boldsymbol{\mu}}_c = \frac{1}{N\widehat{P}(c)} E\left(\sum_{j=1}^{N} z_{jc}\mathbf{x}_j|\mathbf{x}_j^o, \mathbf{R}\right) \tag{7}$$

$$\widehat{\boldsymbol{\Sigma}}_c = \frac{1}{N\widehat{P}(c)} E\left(\sum_{j=1}^{N} z_{jc}\mathbf{x}_j\mathbf{x}_j^T|\mathbf{x}_j^o, \mathbf{R}\right) - \widehat{\boldsymbol{\mu}}_c\widehat{\boldsymbol{\mu}}_c^T \tag{8}$$

Step 5: After convergence, save the resulting FMM parameters and the total log-likelihood of the feature vectors in the data set in the corresponding particle. Since the EM algorithm is a strict gradient algorithm, it converges slowly to the nearest local maximum of the likelihood function to the starting point [18]. Therefore, the convergence criterion of the EM algorithm used in experiments presented in this paper is identical to the one used by Hunt and Jorgensen [19], which is to cease iterating when the difference in the log-likelihood function between iterations (t) and ($t - 10$)

is less than or equal to $10.0E-10$. The use of this criterion does not affect the speed of the algorithm so much as the main interest of this algorithm is to handle small data sets.
Step 6: Repeat *Steps 2–5* for every particle in the swarm and then select the best FMM stored in the particle that has the maximum log-likelihood of the data. This PSO is necessary to overcome the sensitivity to the initialization problem of the EM algorithm.
Step 7: Use the best FMM in estimating missing values in the data set as explained in *Step 3* and in clustering feature vectors in the input data set according to Bayes decision rule such that each fully observed feature vector $\mathbf{x}$ is assigned to a certain component i if $P(i|\mathbf{x}) > P(j|\mathbf{x})$; for all $j \neq i$, where $P(i|\mathbf{x})$ is the probability that $\mathbf{x}$ is generated from the component i. The pseudo code of the POLGREM algorithm is shown in Fig. 2.

The POLGREM algorithm uses local tuning of the general regression (see, Eq. (2)) to produce multiple imputations for each missing value in the data set. Each imputation is obtained via a non-linear multivariate regression using a group of fully observed feature vectors belonging to one of the FMM components. These groups are obtained using Bayes decision rule. Then, the POLGREM estimates the missing value as the average of these multiple imputations weighted by the posterior probabilities of the feature vector containing the missing value to different components of the FMM. This makes the POLGREM algorithm be insensitive to global correlations between features of the input data set. Therefore, the POLGREM does not have the limitation of the general regression neural networks [20] when used for estimating missing values without local tuning with weakly correlated data [5]. In the experiments presented in this paper, the algorithm that uses general regression neural networks with the EM algorithm for learning FMM parameters is referred to as the GREM algorithm. In addition, local tuning of the general regression neural networks used in the POLGREM algorithm makes the algorithm be less dependent of FMM parameters. Therefore, the POLGREM algorithm overcomes the limitation of the MEM algorithm with small data sets which may contain outliers, overlapping clusters, or large differences in the sizes of their clusters [5]. Finally, due to the use of the OCS in estimating missing values the POLGREM algorithm produces accurate estimation of both the missing values in the data set and FMM parameters, especially when clusters of the data set are largely overlapping and different in their sizes. Therefore, the POLGREM algorithm overcomes problems of cluster overlapping and unbalanced cluster sizes that are limitations of the algorithm proposed in [5] and is referred to as the LGREM algorithm in the experiments presented in this paper.

3. Experiments and results

The POLGREM algorithm is evaluated and compared with a number of algorithms proposed in the literature in learning FMM parameters for clustering incomplete data sets. These algorithms are the LGREM algorithm [5], the MEM algorithm [4], the GREM algorithm and the common EM algorithm after

Program model = POLGREM (data)

Step 0. Normalize the values of each input data feature to range from 0 to 1.

Step 1. Determine the optimum smoothing parameter σ for each incomplete feature in the data set.

Step 2. Use a swarm of twenty particles to initialize the EM algorithm twenty times each time using one particle from the swarm, which represents a FMM. After convergence of the EM algorithm the FMM corresponding to the maximum log-likelihood function is selected as the best model for the input data set.

Step 3. In the **E-step**, compute the following quantities for each model component c in the FMM.

- The posterior probabilities vector $\mathbf{z}_i$ for all feature vectors in the data set $\mathcal{R}$ using Equation 1.
- The estimates of the missing values in $\mathcal{R}$ starting with those values in the feature that has the minimum missing rate using Equations 2 and 3.
- The necessary statistics for the **M-step** using Equations 4 and 5.

Step 4. In the **M-step**, compute parameters of each component c in the FMM using Equations 6, 7, and 8.

Step 5. After convergence, save the resulting FMM parameters and the total log-likelihood of the feature vectors in the data set in the corresponding particle.

Step 6. Repeat **Steps 2-5** for every particle in the swarm and then select the best FMM stored in the particle that has the maximum log-likelihood of the data.

Step 7. Use the best FMM in estimating missing values in the data set as explained in **Step 3** and in clustering feature vectors in the input data set according to Bayes decision rule.

Step 8. Stop.

Figure 2 Pseudo code of the POLGREM algorithm.

Table 1 Comparison results of the Student's paired *t*-test statistic with the first data set.

Missing %	*T*(POLGREM, LGREM)		*T*(POLGREM, GREM)		*T*(POLGREM, MEM)		*T*(POLGREM, NNEM)		*T*(POLGREM, MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	1.00	0.00	0.34	−1.00	1.00	0.00	0.05	−2.24	0.02	−2.76
(10%, 20%)	1.00	0.00	0.17	−1.50	1.00	0.00	0.01	−3.21	0.00	−10.83
(15%, 30%)	0.34	1.00	0.08	−1.96	0.34	−1.00	0.00	−4.74	0.00	−27.89
(20%, 40%)	0.34	1.00	0.32	−1.06	0.19	−1.41	0.00	−3.87	0.00	−27.21
(25%, 50%)	0.34	1.00	0.12	−1.71	0.28	−1.15	0.00	−4.33	0.00	−22.12

estimating the missing values in the input data set using the unconditional mean imputation method (MENEM) and the nearest neighbor imputation method (NNEM). In the MENEM algorithm, the missing values in each feature are replaced with the mean of the observed values in that feature. In the NNEM algorithm, each missing value in a certain feature vector is replaced with the observed value in the same feature that is in the nearest feature vector according to the Euclidean distance in the subspace that is composed of the fully observed features. The comparison study in this paper uses three data sets. The missing values are randomly placed with different missing rates in two features of each data set. These features are selected such that the visual separation among data classes is maximum. The mechanism of the occurrence of the missing values in all data sets is missing completely at random (MCAR) [21]. These data sets are described as follows.

3.1. The first data set

This data set is an artificial data set, which contains 150 feature vectors (rows) generated with equal probabilities from three well-separated 4D-Gaussian distributions. The mean vectors of these distributions are $\boldsymbol{\mu}_1 = [2 \;\; 2 \;\; 2 \;\; 2]^T$, $\boldsymbol{\mu}_2 = [4 \;\; 4 \;\; 4 \;\; 4]^T$, and $\boldsymbol{\mu}_3 = [6 \;\; 6 \;\; 6 \;\; 6]^T$. Each one of these Gaussian distributions has a covariance matrix $\boldsymbol{\Sigma} = 0.5\mathbf{I}_4$, where $\mathbf{I}_4$ is the identity matrix of order four. In this data set, all features are strongly correlated.

3.2. The second data set

This data set is similar to the first data set but an outlier feature vector is added to it. The outlier feature vector is $[\max(\mathbf{d}_1), \min(\mathbf{d}_2), 0.5 \max(\mathbf{d}_3), 0.5 \max(\mathbf{d}_4)]^T$, where $\mathbf{d}_i$, $i = 1{:}4$ are the features of the data set.

When each one of these two data sets is used, the missing values are put in the third and in the fourth data features. In addition, each one of the FMMs learnt by all algorithms compared in this study consists of three Gaussian components that have non-restricted covariance matrices.

3.3. The Pima Indians Diabetes data set

The Pima Indians Diabetes data set[1] contains 768 feature vectors each of which is a vector in eight-feature space. These feature vectors represent two classes; the first class has 500 feature vectors belonging to it; the second class has 268 feature vectors belonging to it. These classes are largely overlapping. Correlations between different pairs of features of this data set are too weak. The missing values are put in the second and in the fifth features of the data set. Each one of the FMMs learnt by all algorithms compared in this study consists of two Gaussian components that have non-restricted covariance matrices.

The evaluation criterion used in this study to compare the different algorithms is the Mis-Classification Error (MCE). It is computed by comparing the clustering results, obtained using Bayes decision rule, of the learned FMM with the true classification of the data feature vectors, assuming each class is represented by a component in the FMM. Components of the FMM are allocated to different data classes such that the total number of misclassified feature vectors in the data set is minimum. Let the number of feature vectors belonging to class i be N_i, from which N_i^m feature vectors are not clustered into the component that represents this class in the FMM. Then the MCE for class i is computed as $MCE_{class_i} = N_i^m / N_i$. Assuming the data set is generated from K classes the total MCE is the average of all the class-MCEs and it is computed as $MCE_T = \frac{1}{K}\left(\sum_{i=1}^{K} MCE_{class_i}\right)$.

Tables 1, 3 and 5 show comparisons of different pairs of the algorithms using the Student's paired *t*-test statistic with each one of the data sets. The *P*-value is the significance and the *T*-value is the *t*-statistic. This test examines the statistical significance of the difference in performance of pairs of algorithms using their total MCEs obtained from ten different experiments. In each experiment, a different group of feature vectors is randomly selected to contain missing values. The results of this test are shown for each pair of percentages of missing values in two different features. In the first two data sets the third and the fourth features contain missing values while in the Pima data set the second and the fifth features contain missing values. The shaded cells in each table represent the cases in which the difference in performance of certain pairs of algorithms is statistically significant according to the 5% significance level.

Tables 2, 4 and 6 show comparisons of the algorithms using the mean (MCE) and the standard deviation (STD) of the total Mis-Classification Error obtained from ten different experiments using each one of the data sets. The shaded cells in each table represent the minimum value of the MCE among all algorithms compared.

4. Discussion of results

Tables 1–4 show that the performance of the POLGREM algorithm is not significantly different from the LGREM, the GREM, and the MEM algorithms. On the other hand, it is significantly different ($P \leqslant 0.05$) from the NNEM and the

[1] The Iris and the Pima data sets are available at: http://archive.ics.uci.edu/ml/datasets.html.

Table 2 Comparison results of the mean and the standard deviation of the total Mis-Classification Error with the first data set.

Missing %	POLGREM		GREM		MEM		LGREM		NNEM		MENEM	
	MCE	STD	MCE	STD	MCE	STD	MCE	STD	MCE	STD	MCE	STD
(5%, 10%)	0.001	0.002	0.001	0.003	0.001	0.002	0.001	0.002	0.004	0.005	0.011	0.011
(10%, 20%)	0.001	0.003	0.003	0.003	0.001	0.003	0.001	0.003	0.007	0.004	0.333	0.098
(15%, 30%)	0.002	0.003	0.004	0.005	0.003	0.005	0.001	0.003	0.009	0.004	0.367	0.041
(20%, 40%)	0.003	0.003	0.027	0.075	0.005	0.005	0.002	0.003	0.009	0.006	0.399	0.047
(25%, 50%)	0.004	0.005	0.109	0.194	0.006	0.006	0.003	0.005	0.013	0.006	0.463	0.064

Table 3 Comparison results of the Student's paired *t*-test statistic with the third data set.

Missing %	*T*(POLGREM, LGREM)		*T*(POLGREM, GREM)		*T*(POLGREM, MEM)		*T*(POLGREM, NNEM)		*T*(POLGREM , MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	1.00	0.00	0.34	−1.00	1.00	0.00	0.05	−2.24	0.03	−2.54
(10%, 20%)	1.00	0.00	0.59	−0.56	1.00	0.00	0.02	−2.69	0.00	−7.15
(15%, 30%)	1.00	0.00	0.59	−0.56	0.34	−1.00	0.00	−5.01	0.00	−24.25
(20%, 40%)	0.34	1.00	0.34	−1.00	0.34	−1.00	0.00	−3.86	0.00	−30.36
(25%, 50%)	0.17	1.50	0.04	−2.36	0.59	−0.56	0.00	−3.87	0.00	−16.49

Table 4 Comparison results of the mean and the standard deviation of the total Mis-Classification Error with the third data set.

Missing %	POLGREM		GREM		MEM		LGREM		NNEM		MENEM	
	MCE	STD	MCE	STD	MCE	STD	MCE	STD	MCE	STD	MCE	STD
(5%, 10%)	0.001	0.002	0.001	0.003	0.001	0.002	0.001	0.002	0.004	0.005	0.007	0.008
(10%, 20%)	0.002	0.003	0.003	0.003	0.002	0.003	0.002	0.003	0.007	0.004	0.309	0.135
(15%, 30%)	0.003	0.003	0.003	0.004	0.003	0.005	0.003	0.003	0.009	0.004	0.380	0.049
(20%, 40%)	0.004	0.003	0.027	0.073	0.005	0.004	0.003	0.004	0.010	0.005	0.412	0.042
(25%, 50%)	0.005	0.004	0.130	0.166	0.006	0.006	0.004	0.005	0.012	0.005	0.524	0.100

Table 5 Comparison results of the Student's paired *t*-test statistic with the Pima data set.

Missing %	*T*(POLGREM, LGREM)		*T*(POLGREM, GREM)		*T*(POLGREM, MEM)		*T*(POLGREM, NNEM)		*T*(POLGREM, MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.00	−38.03	0.00	−42.55	0.00	−31.70	0.00	−47.00	0.00	−36.76
(10%, 20%)	0.00	−27.46	0.00	−34.54	0.00	−22.14	0.00	−38.54	0.00	−27.75
(15%, 30%)	0.00	−23.82	0.00	−46.22	0.00	−25.41	0.00	−31.95	0.00	−71.90
(20%, 40%)	0.00	−16.19	0.00	−62.01	0.00	−91.80	0.00	−28.17	0.00	−75.62
(25%, 50%)	0.00	−34.25	0.00	−35.10	0.00	−35.51	0.00	−25.92	0.00	−80.59

MENEM algorithms and outperforms them ($T < 0$ and MCE is the minimum). These results show that the POLGREM algorithm is superior in estimating missing values and learning FMM parameters when the input data set is generated from well-separated clusters and contains features that are strongly correlated.

Tables 5 and 6 show that the performance of the POLGREM algorithm is significantly different from all other algorithms ($P \leqslant 0.05$) and better than them ($T < 0$ and MCE is the minimum). These results show that the performance of the POLGREM algorithm is superior when the input data set is generated from largely overlapping clusters and these clusters are of different sizes. Although the Pima data set contains weakly correlated features and largely overlapping clusters of different sizes the POLGREM has the best performance among all other algorithms compared. This is because clusters of this data set are largely overlapping in the whole feature space and in the subspace that is composed of complete features.

In general, the performance of the POLGREM algorithm proves superiority over the other algorithms compared in this paper when the input data set is generated from overlapping clusters that are of different sizes (see the results with the Pima data set in Tables 5 and 6). In addition, this algorithm is similar to the LGREM, the GREM and the MEM algorithms in their superior performances when the input data set contains outliers and is generated from well-separated clusters that

Table 6 Comparison results of the mean and the standard deviation of the total Mis-Classification Error with the Pima data set.

Missing %	POLGREM		GREM		MEM		LGREM		NNEM		MENEM	
	MCE	STD	MCE	STD	MCE	STD	MCE	STD	MCE	STD	MCE	STD
(5%, 10%)	0.037	0.009	0.377	0.025	0.440	0.045	0.423	0.031	0.432	0.025	0.388	0.031
(10%, 20%)	0.039	0.012	0.372	0.025	0.393	0.047	0.408	0.041	0.425	0.031	0.372	0.031
(15%, 30%)	0.033	0.007	0.378	0.024	0.394	0.044	0.389	0.042	0.424	0.040	0.350	0.011
(20%, 40%)	0.038	0.010	0.368	0.016	0.356	0.005	0.420	0.084	0.414	0.041	0.349	0.007
(25%, 50%)	0.041	0.009	0.378	0.033	0.367	0.023	0.422	0.032	0.405	0.042	0.350	0.010

are of the same sizes (see the results with the first and the third data sets in Tables 1–4). This is due to the use of the OCS and the locally-tuned general regression neural network in the POLGREM algorithm that considers the contribution of different clusters in the data set in estimating the missing values which results in better estimation of both missing values in the data set and FMM parameters.

5. Conclusions

In this paper, the POLGREM algorithm is proposed to overcome the local optima problem of the EM algorithm and the bias problem in estimating missing values when the input data set contains overlapping clusters in learning FMM parameters for clustering using incomplete data sets. A comparison study shows the superiority of the POLGREM algorithm over other algorithms compared when the input data set may contain few outliers, clusters that are largely overlapping, or clusters that have large differences in their sizes. Examples of these algorithms are the LGREM algorithm [5], the MEM algorithm [4], the GREM algorithm and the common EM algorithm after estimating the missing values in the input data set using the unconditional mean imputation method (MENEM) and the nearest neighbor imputation method (NNEM).

References

[1] McLachlan G, Peel D. Finite mixture models. New York: Wiley; 2000.

[2] Tråvén H. A neural network approach to statistical pattern classification by semiparametric estimation of probability density functions. J IEEE Trans Neural Netw 1991;2(3):366–77.

[3] Dempster A, Laird N, Rubin D. Maximum likelihood from incomplete data via the em algorithm (with discussion). J Roy Stat Soc 1977;B(39):1–38.

[4] Ghahramani Z, Jordan M. Supervised learning from incomplete data via an EM approach. In: Cowan J, Tesauro G, Alspector J, editors. Advances in neural information processing systems. San Francisco, CA, USA: Morgan Kaufmann Publishers; 1994. p. 120–7.

[5] Abas AR. Using general regression with local tuning for learning mixture models from incomplete data sets. Egypt Inform J 2010;11(2):49–57.

[6] Abas AR. Using incremental general regression neural network for learning mixture models from incomplete data. Egypt Inform J 2011;12(3):185–96.

[7] Yoon S, Lee S. Training algorithm with incomplete data for feed-forward neural networks. J Neural Process Lett 1999;10:171–9.

[8] Vlassis N, Likas A. A greedy EM algorithm for gaussian mixture learning. J Neural Process Lett 2002;15:77–87.

[9] Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks; 1995. p. 1942–48.

[10] El_Sherbiny MM. Particle swarm inspired optimization algorithm without velocity equation. Egypt Inform 2011;12(1):1–8.

[11] Das S, Abraham A. Pattern clustering using a swarm intelligence approach. In: Maimon O, Rokach L, editors. Data mining and knowledge discovery handbook. Springer Science + Business Media, LLC; 2010.

[12] Hathaway RJ, Bezdek JC. Fuzzy c-means clustering of incomplete data. IEEE Trans Syst Man Cybern Art B: Cybern 2001;31(5):735–44.

[13] Bezdek JC. Pattern recognition with fuzzy objective function algorithms. Norwell: Kluwer Academic Publishers; 1981.

[14] Himmelspach L, Conrad S. Fuzzy clustering of incomplete data based on cluster dispersion. In: Hullermeier E, Kruse R, Hoffmann F, editors. IPMU 2010, LNAI 6178. Berlin, Heidelberg: Springer-Verlag; 2010. p. 59–68.

[15] Park DC. Gradient-based fcm and a neural network for clustering of incomplete data. In: Wang L, Chen K, Ong YS, editors. ICNC 2005, LNCS 3610. Berlin, Heidelberg: Springer-Verlag; 2005. p. 1266–69.

[16] Milligan GW, Cooper MC. A study of standardization of variables in cluster analysis. J Classif 1988;5:181–204.

[17] Schaffer CM, Green PE. An empirical comparison of variable standardization methods in cluster analysis. J Multivar Behav Res 1996;31(2):149–67.

[18] Yin H, Allinson NM. Comparison of a Bayesian SOM with the EM algorithm for gaussian mixtures. In: Proceeding of workshop on self-organising maps (WSOM'97); 1997. p. 118–23.

[19] Hunt L, Jorgensen M. Mixture model clustering for mixed data with missing information. J Comput Stat Data Anal 2003;41:429–40.

[20] Specht D. A general regression neural network. J IEEE Trans Neural Netw 1991;2(6):568–76.

[21] Little R, Rubin D. Statistical analysis with missing data. New York: John Wiley & Sons; 1987.