PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE

IC-AI'2001

Volume II

Editor:
H. R. Arabnia

Associate Editors:
J-Y. Bezieau, B. Blake, S. Buckley, O. Castillo, C. Delrieux, M. Garagnani
P. H. Hanh, R. Joshua, R. A. Liuzzi, Y. Mun, J. A. Olivas
M. Piattini, B. Prasad, J. Tang, M. A. Wani

Las Vegas, Nevada, USA
June 25 - 28, 2001

The ISBN for the volume is 1-892512-79-3.

SESSION: Evlutionary Computing

Genetic-guided Model-based Clustering Algorithms*

Hui-Dong Jin Kwong-Sak Leung
Department of Computer Sci. & Eng.
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong

Man-Leung Wong
Department of Information Systems
Lingnan University
Tuen Mun, Hong Kong

Abstract *Clustering, or the unsupervised classification of data items into clusters, can reveal some intrinsic structures among data. The intrinsic structures, like the number of clusters, are key issues of data mining. In this paper, we propose some genetic-guided model-based clustering techniques to determine the optimal number of clusters and the characteristics of these clusters automatically. The model-based clustering techniques are used to describe the clusters and tune their descriptions, while genetic algorithms lead the search to some promising search space. Several clustering-specific genetic operators are developed to enhance the search procedure. The simulation results, on both synthetic and real-life data sets, demonstrate that our proposed clustering techniques outperform two widely-used model-based clustering algorithms.*

Keywords: Genetic algorithm, data mining, mixture model, the expectation-maximization algorithm, number of clusters

1 Introduction

Clustering is the unsupervised classification of data items into meaningful clusters based on similarity. It can reveal some intrinsic structures, like the number of clusters and 'natural' clusters among the data set, when no prior information is available other than the observed values. Clustering analysis is very useful in exploratory data analysis [2, 3, 5]. This appeals to researchers from many disciplines [2, 5, 11, 14]. They have produced a rich assortment of clustering methods, like model-based [2, 3, 4], genetic-guided [8, 10], and distance-based approaches [13, 12].

Genetic Algorithms (GAs), motivated by natural evolution, maintain a population of solutions and make use of genetic operators to obtain the globally optimal solution [9]. This optimization technique has been successfully combined with other clustering approaches to conduct exploratory data analysis [8, 10]. However, the approaches require users to predefine the number of clusters, and their usefulness is reduced.

In theory, model-based clustering approaches are able to determine the optimal number of clusters automatically. The enumeration strategy is the common procedure used to select the optimal number of clusters [1, 7, 14]. Given the number of clusters, it uses a local search algorithm, say, the Expectation-Maximization (EM) algorithm, to find a good description for the data set, then these resulting models with different numbers of clusters compete with one another based on certain criterion [1, 3, 7, 14]. It is referred to as the *enumeration model-based clustering algorithm* (EnumEM) hereafter. The strategy spend too much time on the mixture models with an inappropriate number of clusters. As an alternative, like in AutoClass [3], some promising numbers of clusters can be driven from the previous clustering results to invoke the local search algorithm. Both of them suffer the local search property of the EM algorithm.

By introducing a global search mechanism into model-based clustering techniques, we can improve the effectiveness in determining the number of clusters. The global search mechanism, say, GAs, can be used to explore the natural clusters among data and determine the optimal number of clusters automatically. This distinguishes our genetic-guided model-based clustering approaches from most of the previous research work on GA-guided clustering algorithms [8, 10]. In the next section, we outline some model-based clustering techniques involved. Our genetic-guided model-based clustering approaches are proposed in section 3, followed by several specific genetic operators. Sim-

*The work was partially supported by RGC Grant CUHK 4161/97E of Hong Kong.

ulation results are given in section 4 and we conclude the paper in section 5.

2 Model-based Clustering Analysis

Given a data set x=(x_1,..., x_N), model-based clustering techniques assume that each data item x_i is drawn from a mixture model Φ with the density:

$$P(x_i|\Phi) = \sum_{k=1}^{K} p_k \phi(x_i|\theta_k). \quad (1)$$

Here K is the number of clusters, p_k is the mixing proportion for the k^{th} cluster ($0 < p_k < 1$ for all k=1,..., K and $\sum_{k=1}^{K} p_k = 1$), $\phi(x_i|\theta_k)$ is the corresponding density for cluster k and θ_k denotes the parameters involved.

This paper concentrates on the case where $\phi(x_i|\theta_k)$ is the multivariate normal distribution, a model that has been used with considerable success [3, 6, 7]. Thus, the parameter θ_k consists of a mean vector μ_k and a covariance matrix Σ_k. The density is of the form $\phi(x_i|\theta_k) = \frac{\exp\{-\frac{1}{2}(x_i-\mu_k)^T\Sigma_k^{-1}(x_i-\mu_k)\}}{(2\pi)^{\frac{D}{2}}|\Sigma_k|^{\frac{1}{2}}}$where D is the dimensionality of the data items.

Given a mixture model Φ, we can get the membership probability $t_{ik} = \frac{p_k\phi(x_i|\theta_k)}{P(x_i|\Phi)}$ for the data item x_i. Then, we can get crisp classification by assigning the data items x_i to cluster k where $k = \arg\max_j\{t_{ij}\}$. Thus, a mixture model Φ can be viewed as a clustering solution. Several techniques to find a good mixture model are outlined below.

2.1 The EM Algorithm

The Expectation-Maximization (EM) algorithm is the most commonly used method to find a mixture model with higher *log-likelihood*, which is calculated by $L_M(\Phi) = \sum_{i=1}^{N}[\log P(x_i|\Phi)]$ [3, 4, 7, 14]. The EM algorithm consists of four steps as follows.

1. **Initialization**: Given the number of clusters K and the current iteration $j = 0$, initialize the parameters in the mixture model: $p_k^{(j)}, \mu_k^{(j)}$ and $\Sigma_k^{(j)}$ ($k = 1, ..., K$).

2. **E-Step**: Given the mixture model parameters, compute the membership $t_{ik}^{(j)}$:

$$t_{ik}^{(j)} = \frac{p_k^{(j)}\phi(x_i|u_k^{(j)}, \Sigma_k^{(j)})}{\sum_{l=1}^{K} p_l^{(j)}\phi(x_i|u_l^{(j)}, \Sigma_l^{(j)})} \quad (2)$$

3. **M-step**: Given $t_{ik}^{(j)}$, update the mixture model parameters for $k = 1, ..., K$:

$$p_k^{(j+1)} = \frac{1}{N}\sum_{i=1}^{N} t_{ik}^{(j)} \quad (3)$$

$$\mu_k^{(j+1)} = \frac{\sum_{i=1}^{N} t_{ik}^{(j)} x_i}{N \cdot p_k^{(j+1)}} \quad (4)$$

$$\Sigma_k^{(j+1)} = \frac{\sum_{i=1}^{N} t_{ik}^{(j)}(x_i-\mu_k^{(j)})(x_i-\mu_k^{(j)})^T}{N \cdot p_k^{(j+1)}} \quad (5)$$

4. **Termination**: If $\left|L_M(\Phi^{(j+1)}) - L_M(\Phi^{(j)})\right|$ is not small enough, set $j = j + 1$ and go to step 2.

As a greedy algorithm, the EM algorithm converges to a near optimal solution, but there is no guarantee to find the optimal one. Furthermore, the convergence rate may be very slow if the clusters are not well separated or the number of clusters is not properly predefined [6].

2.2 Hierarchical Agglomerative Clustering

Hierarchical Agglomerative Clustering (HAC) is an iterative procedure in which "optimal" pairs of clusters are successively merged. In the model-based HAC algorithm, a pair of clusters with the least loss of the *classification log-likelihood* is chosen to merge at each stage [6]. The classification log-likelihood is calculated according to $L_C(\Phi) = \sum_{k=1}^{K}\sum_{x_i \in C_k}\phi(x_i|\theta_k)$where C_k indicates the k^{th} cluster.

The time and the memory complexities of hierarchical agglomerative clustering algorithms depend quadratically on the number of components in the initial partition, which is usually a set of singleton clusters. Thus, it is impractical to process large data sets [6].

2.3 Model Selection

By balancing the model accuracy and the complexity, various criteria have been proposed to

measure a model's suitability [1, 3, 14]. A classic criterion is the *Bayesian Information Criterion* (BIC). It is defined by:

$$BIC(\Phi) = -2 \cdot L_M(\Phi) + v(\Phi) \log N \qquad (6)$$

where $v(\Phi)$ is the number of free parameters in the mixture model Φ. Some simulation results have shown its good performance in practice [1, 3, 7].

Based on this criterion, the widely used approach, EnumEM, works as follows. Given the number of clusters, the best log-likelihood is estimated by invoking the EM algorithm several times. Then, the BIC values for all possible K compete with one another. The model with the minimal BIC value is chosen to determine the number of clusters in the data set [1, 14]. Due to the local search property of the EM algorithm and little communication during running, the EnumEM algorithm does not work well, especially on complicated data sets.

AutoClass selects models using an "Occam Factor", which implies that Bayesian parameter priori can somehow prevent the over fitting [3]. During the running, a new promising number of cluster may be specified heuristically to invoke the EM algorithm. However, this strategy still suffers from the local search property of the EM algorithm.

3 Genetic-guided Model-based Clustering Analysis

For the last decade there has been a growing interest in evolutionary algorithms that are based on Darwin's theory of evolution. One of the implementations is Genetic Algorithms (GAs) [9]. GAs maintain a population of solutions and manipulate them with several genetic operators including mutation, crossover, and selection. These operators model some natural phenomena: genetic inheritance and Darwinian strife for survive. The most significant advantages of GAs are the flexibility and adaptability to the task on hand, the robust and global search characteristics. Thus they are often employed to handle inherently hard problems. GAs have been widely used in clustering analysis [8, 10]. These hybridization algorithms have to specify the number of clusters in advance, which greatly impacts their utility in real world problems.

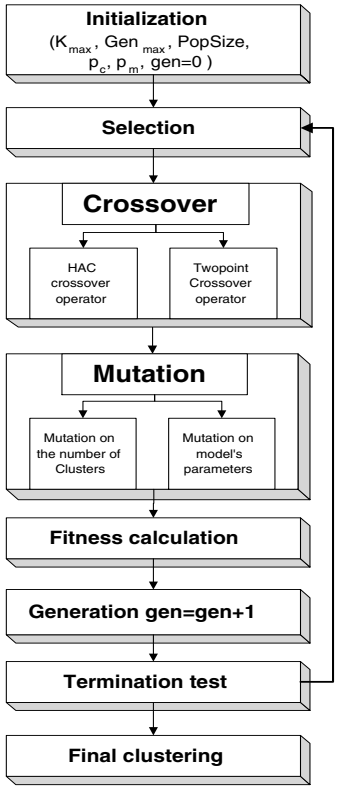

Figure 1: The flow chart of the genetic-guided model-based clustering algorithms.

By combining GAs with several model-based clustering techniques, we can construct the genetic-guided model-based clustering analysis techniques. The global search capability of GAs guides the algorithms to focus on the promising solutions and avoid the search for solutions with an inappropriate number of clusters as early as possible. The algorithms are outlined in Fig.1 in which four different genetic operators are developed to enhance the performance. Different combinations of these operators can lead to different clustering algorithms. Briefly, we call the one with all the genetic operators GAXEM, and the one without the HAC crossover operator GAEM. These genetic operators are described in details below.

3.1 Representation and Fitness

In GAs, each mixture model is coded as a chromosome to represent a clustering solution: the first gene is the number of clusters, followed by genes representing the parameters the clusters. The parameters for a single cluster in-

clude the mixing proportion, the mean vector and the covariance matrix. So, the length of the chromosome is varying during the searching of GAXEM/GAEM.

The fitness of a chromosome is evaluated by invoking an *n-iteration EM algorithm*. Starting from the model indicated by a chromosome, EM runs the E-step and the M-step for n iterations. The updated model then replaces the old one, and its log-likelihood is used to calculate the BIC value according to Eq. (6). We use a selection operator for minimization. In other words, the less the BIC is, the more probably the chromosome is selected. In our implementation, the number of iterations n increases linearly with the generation number.

3.2 Crossover Operators

There are two crossover operators as shown in Fig.1. The first one, often used in GAs, is the two-point crossover operator. It exchanges the parameter values for some clusters between the two random crossover points.

The other one, the HAC crossover operator, is derived from the model-based Hierarchical Agglomerative Clustering (HAC) algorithm [6]. Two parent models firstly merge into one new mixture model by appending one model behind the other followed by adjusting the number of clusters and the mixture proportions. Starting from this initial partition we can alleviate the significant overhead associated with the HAC algorithm [6]. Then a pair of clusters with the least loss of the log-likelihood will agglomerate into a new cluster iteratively. The procedure stops when the BIC value reaches the minimum, which can lead to an optimal number of clusters.

Since the accurate calculation of the loss of the log-likelihood is rather time-consuming, we approximate it based on the assumption that $p_k\phi(x_k|\theta_k) >> p_s\phi(x_k|\theta_s)$ $(s \neq k)$ as x_k belongs to the k^{th} cluster C_k. Then we have

$$\begin{aligned} L_M(\Phi) &\approx N\sum_{k=1}^{K} p_k \log p_k - \frac{1}{2}N\sum_{k=1}^{K} p_k \log|\Sigma_k| \\ &\quad -\frac{ND}{2}\log(2\pi) - \frac{1}{2}ND. \end{aligned} \tag{7}$$

Here, we use $\Sigma_k \approx \sum_{x_i \in C_k} \frac{\{(x_i-\mu_k)(x_i-\mu_k)^T\}}{Np_k}$ and $\sum_{x_i \in C_k} \{(x_i-\mu_k)^T\Sigma_k^{-1}(x_i-\mu_k)\} = Np_kD$. If clusters l and j merge into cluster r, we have

$$\begin{aligned} \Delta L_M(K) &= L_M(\Phi_K) - L_M(\Phi_{K-1}) \\ &\approx N\left(p_l \log p_l + p_j \log p_j - p_r \log p_r\right) \\ &\quad -\frac{N}{2}\left(p_l \log|\Sigma_l| + p_j \log|\Sigma_j| - p_r \log|\Sigma_r|\right). \end{aligned} \tag{8}$$

As the number of free parameters for every cluster is fixed, say F, we can reformulate $BIC(\Phi_K)$ and get

$$\begin{aligned} \Delta BIC(K) &= BIC(\Phi_K) - BIC(\Phi_{k-1}) \\ &\approx -2\Delta L_M(K) + F\log N. \end{aligned}$$

In order to minimize BIC within the HAC agglomeration procedure, we should keep $\Delta BIC(K) \geq 0$. That is

$$\Delta L_M(K) \leq \frac{F}{2}\log N. \tag{9}$$

Thus, we have a simple termination criterion for the HAC crossover operator to terminate with a good number of clusters.

Now let us derive the update formulae for the model parameters during the agglomeration procedure. We assume that, if cluster r agglomerates from clusters j and l, all their membership probabilities are agglomerated. So

$$u_r = \sum_{i=1}^{N} \frac{(t_{ij}+t_{il})x_i}{N(p_j+p_l)} = \frac{p_j\mu_j + p_l\mu_l}{p_j+p_l}$$

and

$$\Sigma_r = \frac{p_j\Sigma_j + p_l\Sigma_l}{p_j+p_l} + \frac{p_jp_l}{(p_j+p_l)^2}(\mu_j-\mu_l)(\mu_j-\mu_l)^T$$

Note that the HAC crossover operator does not need to access the data set again. Thus, it is suitable for processing large scale data sets.

3.3 Mutation Operators

The proposed GAXEM and GAEM have two mutation operators that introduce certain diversity into the population of chromosomes. These operators enable the algorithm to generate a mixture model from the another one. And this favors the global convergence of GAXEM and GAEM.

The first mutation operator modifies the parameters of the model. It randomly selects a cluster, say k, and a data item, say x_i. Then the mean vector μ_k is moved to x_i with a small step.

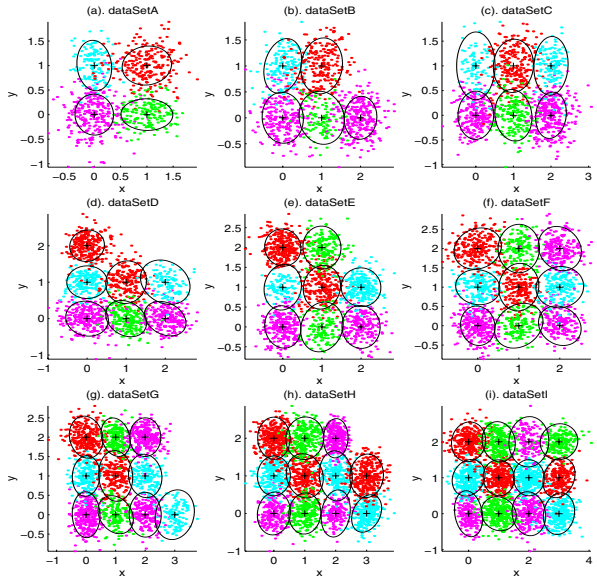

Figure 2: The 9 synthetic data sets and their original mixture models. The data items are indicated by dots. The center '+' and its corresponding solid ellipse indicate the mean and the contour of a component distribution.

The covariance matrix Σ_k is adjusted by the covariance between μ_k and x_i. It is formulated as follows:

$$\begin{aligned} \mu'_k &= (1-\alpha)\mu_k + \alpha x_i \\ \Sigma'_k &= (1-\beta)\Sigma_k + \beta(x_i-\mu_k)(x_i-\mu_k)^T \end{aligned}$$

Here α and β are two small positive numbers.

The second mutation operator can mutate the number of clusters K. It first generates a new number K' around K. If $K' < K$, then it randomly selects K' clusters to form a new chromosome. Otherwise, it selects some data items as the mean vectors and the identity matrices as the covariance matrices to form some additional cluster descriptions.

4 Simulation Results

We illustrate the performance of the proposed GAXEM and GAEM algorithms on various data sets and compare them with EnumEM and AutoClass [3]. All algorithms were implemented with MATLAB except that AutoClass was coded in C programming language (http://ic-www.arc.nasa.gov /ic/projects/bayes-group), and executed on Sun

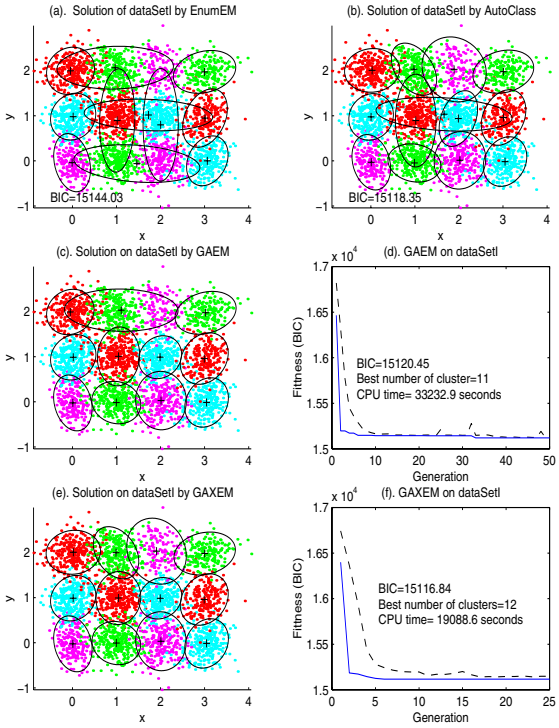

Figure 3: Typical results for data set I obtained by EnumEM, AutoClass, GAEM and GAXEM. In (a), (b), (c) and (e), the center '+' and its corresponding solid ellipse indicate the center and the contour of a component distribution generated. The dotted lines and the solid ones in (d) and (f) shows the mean and the minimal BIC values of the algorithms respectively.

Ultra 5/270. GAXEM and GAEM employed the elitism strategy, and they were terminated when the fitness of the best chromosome did not change for 5 generations. To make it fair to compare with EnumEM and AutoClass, the population size was set as $5 \times N^{\frac{1}{3}}$. Here the number $N^{\frac{1}{3}}$ was the upper bound for the numbers of clusters, and the EM algorithm was invoked 5 times for every mixture model with a given number of clusters in EnumEM [7, 14]. Based on some preliminary experiments, the crossover and the mutation probability, respectively, were set to 0.99 and 0.05. The two mutation and the two crossover operators were invoked with equal probability.

The first set of simulations has been conducted on 9 synthetic data sets. These 9 data sets are depicted in Fig.2. The centers of the component distributions are located on the 2-

Table 1: The simulation results on 9 synthetic data sets of EnumEM, AutoClass, GAEM and GAXEM[1]

Data set		EnumEM			AutoClass			GAEM			GAXEM		
	K	Accu	Suc	Time	Accu	Suc	Time	Accu	Suc	Time	Accu	Suc	Time
A	4	56.8	8	1799	63.5	10	435	58.4	8	1079	63.5	10	1374
B	5	51.4	4	2581	53.7	10	785	52.8	7	2096	53.4	9	3450
C	6	42.6	5	2184	46.4	7	1031	47.9	8	3115	48.3	9	4385
D	7	42.8	2	3135	58.6	8	1252	59.5	8	4803	63.7	10	4585
E	8	62.0	4	3224	63.7	8	1543	64.2	7	4262	64.7	9	7380
F	9	53.9	3	3318	60.2	9	1691	58.8	7	7399	58.6	7	7992
G	10	54.0	3	4369	55.2	4	2158	56.9	5	9806	59.2	8	12921
H	11	45.2	3	8570	47.1	4	2975	44.7	3	21871	52.9	6	26732
I	12	37.5	2	9149	43.6	4	3763	50.3	3	29487	51.4	7	30296
Average		49.6	3.8	4259	54.7	7.1	1737	54.8	6.2	9324	57.6	8.3	12757

dimensional grid and the covariance matrices are generated randomly around $0.1 * I_2$, where I_2 is the 2 by 2 identity matrix. We draw about 200 data items from each cluster for the first 7 data sets and 250 for the last two data sets.

The simulation results are summarized in Table 1 based on 10 independent runs. GAXEM can determine the optimal number of clusters more frequently than the others for all data sets except data set F. For example, GAXEM succeeds 7 times within 10 runs, while EnumEM, AutoClass and GAEM succeed 2, 4 and 3 times within 10 runs, respectively, to detect 12 clusters among data set I. On average, EnumEM, AutoClass, GAEM and GAXEM respectively succeed 3.8, 7.1, 6.3 and 8.3 times within 10 runs. Similar situation can be observed on the other measurement: accuracy. It is defined to measure the match between two classifications C and C' by

$$Accuracy(C, C') = 1 - \frac{E(C, C') + E(C', C)}{2}$$

and $E(C, C') = \sum_{k=1}^{K} p_k \left[-\sum_{i=1}^{K'} q_{ki'} \log q_{ki'} \right]$, where $q_{ki'}$ is the ratio of data items in cluster k of classification C assigned to cluster i' of classification C'. The accuracy value reaches the maximum 1 as two classifications are identical. Due to the ill-separated data sets, the accuracy values in Table 1 with respect to the original classification can not hard to approach 1. GAXEM generates better classifications with higher accuracy values than EnumEM, AutoClass and GAEM for all tested data sets except data set F. GAEM performs better than EnumEM for all data sets except data set H. Fig.3 gives some typical results and the run-time behaviors of GAEM and GAXEM. In Fig.3(e), we can see that GAXEM generates a mixture model similar to the original one as shown in Fig.2(i). In summary, the solutions obtained by GAXEM are better than those obtained by other algorithms. GAXEM performs better than GAEM, which confirms the significance of the proposed HAC crossover operator. On the other hand, although GAXEM and GAEM require longer computation time, their computation time grows in a similar way as that of EnumEM. Their computation time is within 10 times of that of AutoClass, which runs faster partially because it is coded in C programming language [3].

Our second set of simulations has been conducted on several real-life data sets from the UCI machine learning repository (www.sgi.com/Technology/mlc/db). The simulation results are summarized in Table 2. GAXEM can determine the optimal number of clusters in most simulations. On average, EnumEM, AutoClass and GAXEM respectively succeed 6.4, 4.4 and 8.6 times within 10 runs. Normally, GAXEM generates better solutions with lower BIC values than EnumEM and AutoClass for all 5 data sets. Similar to the first set of experiments, AutoClass is apt to generate more clusters than the optimal. Especially, it failed to detect the five clusters for the data set 'sleep' within 10 runs. On the other hand, both EnumEM and GAXEM can determine 5 clusters correctly in 10 runs. This maybe caused by the the "Occam Factor" in AutoClass [3] which fits data sets different from the BIC in GAXEM. In addition, the execution time of GAXEM is longer than that of the other algorithms, but it is not significantly different from the others.

In summary, GAXEM can determine the optimal number of clusters more frequently than EnumEM, AutoClass and GAEM with longer execution time. GAEM, similar to AutoClass, performs better than EnumEM. GAXEM outperforms GAEM which shows the significant role of the proposed genetic operator.

[1] 'K' indicates the optimal number of clusters in the data set, 'Accu' the average accuracy value (%) and 'Suc' the successful trials on finding the optimal number of clusters within 10 runs. 'N' indicates the number of data items, 'Attributes' the attributes used in the simulation. The unit for the average execution time 'Time' is second.

Table 2: The simulation results on 5 real world data sets generated by EnumEM, AutoClass and GAXEM[1]

				EnumEM				AutoClass				GAXEM			
data set	K	N	Attributes	BIC	Accu	Suc	Time	BIC	Accu	Suc	Time	BIC	Accu	Suc	Time
diabetes	3	145	1,2,3	4762	60.4	9	81	4766	60.2	9	68	4770	60.7	10	113
thyroid	3	215	1,2,3,4,5	4948	74.0	6	104	4921	79.7	7	108	4810	83.0	8	203
iris	3	150	1,3,4	514	90.6	7	88	573	62.1	2	87	510	91.2	9	124
liver	2	345	1,2,3,4	1059	38.5	6	485	1223	36.9	4	264	1009	40.8	9	570
sleep	5	2500	5,7,8,10	38932	45.7	4	3045	42233	38.4	0	3481	37994	50.2	7	8793
Average				10043	61.8	6.4	761	10743	55.5	4.4	802	9819	65.2	8.6	1961

5 Conclusion

In this paper, we have proposed the new genetic-guided model-based clustering algorithms, GAXEM and GAEM. Besides finding the good clusters among data sets, they can determine the optimal number of clusters. Based on the model-based clustering techniques, we have established several novel genetic operators to integrate the evolutionary mechanism with problem-specific techniques to improve its performance. The simulation results on both synthetic and real-life data sets have illustrated that GAXEM and GAEM perform better than the widely used model-based clustering algorithm EnumEM, and GAXEM performs better than AutoClass. Simulation results have also substantiated the significance of these proposed genetic operators. In future, we will extend the proposed algorithms to handle the data sets with different kinds of attributes such as categorical and continuous. The scalability of our proposed algorithms is also our future research.

References

[1] C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:719 –725, 2000.

[2] I. Cadez, S. Gaffney, and P. Smyth. A general probabilistic framework for clustering individuals. In *Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, Massachusetts, 2000.

[3] P. Cheeseman and J. Stutz. Bayesian classification (AutoClass): Theory and results. In Fayyad et al. [5], pages 153–180.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 1977.

[5] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.

[6] C. Fraley. Algorithms for model-based Gaussian hierarchical clustering. *SIAM Journal on Scientific Computing*, 20(1):270–281, Jan. 1999.

[7] C. Fraley and A. E. Raftery. How many clusters? Which clustering method? Answers via model-based cluster analysis. *Computer Journal*, 41:578–588, 1998.

[8] P. Franti. Genetic algorithm with deterministic crossover for vector quantization. *Pattern Recognition Letters*, 21(1):61–68, 2000.

[9] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Pub. Co., 1989.

[10] L. O. Hall, I. B. Özyurt, and J. C. Bezdek. Clustering with a genetically optimized approach. *IEEE Transactions on Evolutionary Computation*, 3(2):103–112, 1999.

[11] H.D. Jin. Genetic-guided model-based clustering algorithms and their scalability. In *Proceeding of 2001 Genetic and Evolutionary Computation Conference Workshop program*, July 2001.

[12] Y. Leung, J.S. Zhang, and Z.B.Xu. Clustering by scale-space filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1396–1410, Dec. 2000.

[13] Andrew Webb. *Statistical Pattern Recognition*. Oxford University Press, 1999.

[14] L. Xu. Bayesian Ying-Yang machine, clustering and number of clusters. *Pattern Recognition Letters*, pages 1167–1178, Nov. 1997.