An evolutionary rough partitive clustering

Sushmita Mitra *

Machine Intelligence Unit, Indian Statistical Institute, 203, Barrackpore Trunk Road, Kolkata 700108, India

Received 30 September 2003; received in revised form 16 April 2004
Available online 8 June 2004

Abstract

An evolutionary rough c-means clustering algorithm is proposed. Genetic algorithms are employed to tune the threshold, and relative importance of upper and lower approximations of the rough sets modeling the clusters. The Davies–Bouldin clustering validity index is used as the fitness function, that is minimized while arriving at an optimal partitioning. A comparative study of its performance is made with related partitive algorithms. The effectiveness of the algorithm is demonstrated on real and synthetic datasets, including microarray gene expression data from Bioinformatics.

Keywords: Partitive clustering; Genetic algorithms; Rough sets; Clustering validity index; Gene expression; Microarray data

1. Introduction

A cluster is a collection of data objects which are similar to one another within the same cluster but dissimilar to the objects in other clusters. The problem is to group N patterns into c desired clusters with high *intra-class* similarity and low *inter-class* similarity by optimizing an objective function. Clustering of data is broadly based on two approaches, viz., hierarchical and partitive. The hierarchical approach proceeds by constructing a *dendrogram*, in a top–down or bottom–up manner, and has been found to be computationally expensive. In partitive algorithms, the goal is to find a partition for a given value of c. In this article we restrict ourselves to partitive clustering.

In the c-Means algorithm (Tou and Gonzalez, 1974), each cluster is represented by the center of gravity of the cluster. This need not essentially correspond to an object of the given pattern set. In the c-medoids algorithm (Kaufman and Rousseeuw, 1990), on the other hand, each cluster is represented by one of the representative objects in the cluster located near the center. Partitioning around medoids (PAM) (Kaufman and Rousseeuw, 1990) starts from an initial set of medoids, and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering. Although PAM works effectively for small data, it does not scale well for large datasets. Clustering large applications based on randomized search (CLARANS) (Ng and Han,

* Tel.: +91-33-577-8085; fax: +91-33-577-6680.
E-mail address: sushmita@isical.ac.in (S. Mitra).

1994), using randomized sampling, is capable of dealing with the associated scalability issue.

Soft computing is a consortium of methodologies that works synergistically and provides flexible information processing capability for handling real life ambiguous situations (Zadeh, 1994). Its aim is to exploit the tolerance for imprecision, uncertainty, approximate reasoning, and partial truth in order to achieve tractability, robustness, and low-cost solutions. Recently various soft computing methodologies have been applied to handle the different challenges posed by data mining (Mitra and Acharya, 2003), involving large heterogeneous datasets. The main constituents of soft computing, at this juncture, include fuzzy logic, neural networks, genetic algorithms and rough sets.

In this article we present a partitive clustering algorithm in the soft computing framework. Rough sets (Pawlak, 1991) are used to model the clusters in terms of upper and lower approximations. Genetic algorithms (GAs) (Goldberg, 1989) are used to tune the threshold, and relative importance of upper and lower approximation parameters of the sets. The Davies–Bouldin clustering validity index is used as the fitness function of the GA, that is minimized. This results in optimal generation of clusters for each value of c. The performance of the algorithm, along with results for related clustering algorithms, are provided on real and synthetic datasets including microarray gene expression data from Bioinformatics.

Section 2 describes the different partitive clustering algorithms compared. These include PAM, CLARANS, fuzzy c-means and fuzzy c-medoids, followed by a note on the clustering validity index used to determine optimal clustering. The evolutionary rough c-means algorithm is presented in Section 3. The effectiveness of the method is demonstrated on different datasets in Section 4. Finally, Section 5 concludes the article.

2. Clustering algorithms

In this section we describe the different partitive algorithms used for clustering. Some of the popular methods include c-means and c-medoids (PAM). Scalable algorithms like CLARANS are suitable for handling large datasets. Incorporation of the fuzzy membership concept, in fuzzy c-means and fuzzy c-medoids, enables appropriate modeling of real life overlapping data.

2.1. c-Means algorithm

The algorithm proceeds by partitioning N objects into c non-empty subsets. During each partition, the centroids or means of the clusters are computed. The main steps of the c-means algorithm (Tou and Gonzalez, 1974) are as follows:

- Assign initial means $\boldsymbol{m}_i$ (also called centroids).
- Assign each data object (pattern point) $\boldsymbol{X}_k$ to the cluster U_i for the closest mean.
- Compute new mean for each cluster using
$$\boldsymbol{m}_i = \frac{\sum_{\boldsymbol{X}_k \in U_i} \boldsymbol{X}_k}{|c_i|}, \tag{1}$$
where $|c_i|$ is the number of objects in cluster U_i.
- Iterate until criterion function converges, i.e., there are no more new assignments.

2.2. Partitioning around medoids (PAM)

The algorithm uses the most centrally located object in a cluster, the medoid, instead of the mean. Note that a medoid, unlike a mean, is essentially an existing data object from the cluster. It is closest to the corresponding mean. The basic steps are outlined as follows:

- Arbitrarily choose c objects as the initial medoids or seed points.
- Assign each remaining data object (pattern) to the cluster for the closest medoid.
- Replace each of the medoids by one of all the non-medoids (causing the greatest reduction in square error), as long as the quality of clustering improves.
- Iterate until the criterion function converges.

For large N and c, the c-medoids (Kaufman and Rousseeuw, 1990) algorithm is computationally more costly than the conventional c-means. However, in the presence of noise and outliers, c-medoids is found to be more robust. This is because of the inherent robustness of medoids, as compared to means, with respect to noise.

2.3. Clustering large applications based on randomized search (CLARANS)

Large datasets require the application of scalable algorithms. CLARANS (Ng and Han, 1994) draws a sample of the large data, with some randomness, at each stage of the search. Each cluster is represented by a medoid. Multiple scans of the database are required by the algorithm. Here the clustering process searches through a graph, where each node is represented by a set of c medoids. Two nodes are termed as neighbors if they only differ by one medoid. Hence each node has $c*\ (N-c)$ neighbors.

The main steps are as follows:

- Initially, a node of c medoids is chosen randomly.
- Replace one of the c medoids at random, by selecting a neighbor node randomly.
- Assign data objects (pattern points) to the cluster with the closest medoid, by calculating average distance for this node; this requires one scan of the database.
- **If** the criterion function does not improve **then** revert back to the old medoid (node); **else** set the current node to be the neighbor node.
- Repeat for a fixed number of times.

CLARANS has been experimentally shown to be more effective than PAM. It enables the detection of outliers.

2.4. Fuzzy c-means (FCM)

This is a fuzzification of the c-means algorithm, proposed by Bezdek (1981). It partitions a set of N patterns $\{\boldsymbol{X}_k\}$ into c clusters by minimizing the objective function

$$J=\sum_{k=1}^{N}\sum_{i=1}^{c}(\mu_{ik})^{m'}\|\boldsymbol{X}_k-\boldsymbol{m}_i\|^2, \tag{2}$$

where $1\leqslant m'<\infty$ is the fuzzifier, $\boldsymbol{m}_i$ is the ith cluster center, $\mu_{ik}\in[0,1]$ is the membership of the kth pattern to it, and $\|\cdot\|$ is the distance norm, such that

$$\boldsymbol{m}_i=\frac{\sum_{k=1}^{N}(\mu_{ik})^{m'}\boldsymbol{X}_k}{\sum_{k=1}^{N}(\mu_{ik})^{m'}}, \tag{3}$$

and

$$\mu_{ik}=\frac{1}{\sum_{j=1}^{c}\left(\frac{d_{ik}}{d_{jk}}\right)^{\frac{2}{m'-1}}}, \tag{4}$$

$\forall i$, with $d_{ik}=||\boldsymbol{X}_k-\boldsymbol{m}_i||^2$, subject to $\sum_{i=1}^{c}\mu_{ik}=1$, $\forall k$, and $0<\sum_{k=1}^{N}\mu_{ik}<N$, $\forall i$. The algorithm proceeds as follows.

(i) Pick the initial means $\boldsymbol{m}_i$, $i=1,\ldots,c$. Choose values for fuzzifier m' and threshold ϵ. Set the iteration counter $t=1$.
(ii) Repeat Steps (iii)–(iv), by incrementing t, until $|\mu_{ik}(t)-\mu_{ik}(t-1)|>\epsilon$.
(iii) Compute μ_{ik} by Eq. (4) for c clusters and N data objects.
(iv) Update means $\boldsymbol{m}_i$ by Eq. (3).

Note that for $\mu_{ik}\in[0,1]$ the objective function of Eq. (2) boils down to the hard c-means case, whereby a *winner-take-all* strategy is applied in place of membership values in Eq. (3).

2.5. Fuzzy c-medoids

This is a fuzzification of the c-medoids algorithm and is outlined as follows (Krishnapuram et al., 2001):

(i) Pick the initial medoids $\boldsymbol{m}_i$, $i=1,\ldots,c$.
(ii) Repeat Steps (iii)–(iv) until convergence.
(iii) Compute μ_{ik} for $i=1,\ldots,c$ and $k=1,\ldots,N$.
(iv) Compute new medoids

$$\boldsymbol{m}_i=\boldsymbol{X}_q,$$

where

$$q=\arg\min_{1\leqslant j\leqslant N}\sum_{k=1}^{N}(\mu_{ik})^{m'}\|\boldsymbol{X}_j-\boldsymbol{X}_k\|^2 \tag{5}$$

refers to that j for which the minimum value of the expression is obtained.

Note that this boils down to the hard c-medoids with $\mu_{ik} = 1$, if $i = q$, and to $\mu_{ik} = 0$ otherwise.

2.6. Clustering validity index

The clustering algorithms described in Sections 2.1–2.5 are partitive, requiring prespecification of the number of clusters. The results are dependent on the choice of c. There exist validity indices to evaluate the goodness of clustering, corresponding to a given value of c. In this article we compute the optimal number of clusters c_0 in terms of the Davies–Bouldin cluster validity index (Bezdek and Pal, 1998).

The Davies–Bouldin index is a function of the ratio of the sum of within-cluster distance to between-cluster separation. The optimal clustering, for $c = c_0$, minimizes

$$\frac{1}{c}\sum_{k=1}^{c} \max_{l\neq k}\left\{\frac{S(U_k)+S(U_l)}{d(U_k,U_l)}\right\}, \tag{6}$$

for $1 \leqslant k$, $l \leqslant c$. In this process, the within-cluster distance $S(U_k)$ is minimized and the between-cluster separation $d(U_k, U_l)$ is maximized. The distance can be chosen as the traditional Euclidean metric for numeric features.

3. Evolutionary rough c-means

Here rough sets are used to represent clusters in terms of upper and lower approximations. However, the relative importance of these approximation parameters, as well as a threshold parameter need to be tuned for good partitioning. The evolutionary rough c-means algorithm employs GAs to optimally tune these parameters. The Davies–Bouldin index is used as the fitness function to be minimized. Various values of c are used to generate different sets of clusters, and GA is employed to generate the optimal partitioning.

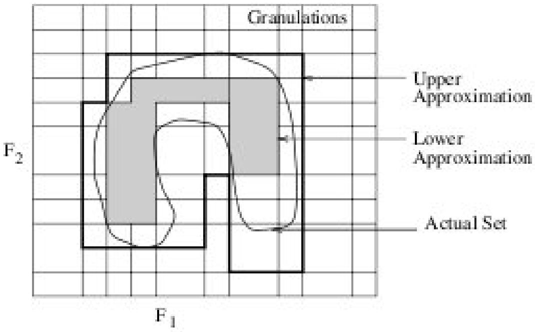

Fig. 1. Lower and upper approximations in a rough set.

3.1. Rough set preliminaries

The theory of *rough sets* (Pawlak, 1991) has recently emerged as another major mathematical tool for managing uncertainty that arises from granularity in the domain of discourse––that is, from the indiscernibility between objects in a set. The intention is to approximate a *rough* (imprecise) concept in the domain of discourse by a pair of *exact* concepts, called the lower and upper approximations. These exact concepts are determined by an *indiscernibility* relation on the domain, which, in turn, may be induced by a given set of *attributes* ascribed to the objects of the domain. The lower approximation is the set of objects definitely belonging to the vague concept, whereas the upper approximation is the set of objects possibly belonging to the same. Fig. 1 provides a schematic diagram of a rough set.

3.2. Rough c-means

In the rough c-means algorithm, the concept of c-means is extended by viewing each cluster as an interval or rough set (Lingras and West, 2002). A rough set Y is characterized by its lower and upper approximations $\underline{B}Y$ and $\overline{B}Y$ respectively. This permits overlaps between clusters. Here an object $\boldsymbol{X}_k$ can be part of at most *one* lower approximation. If $\boldsymbol{X}_k \in \underline{B}Y$ of cluster Y, then simultaneously $\boldsymbol{X}_k \in \overline{B}Y$. If $\boldsymbol{X}_k$ is not a part of any lower approximation, then it belongs to two or more upper approximations.

Adapting Eq. (1), the centroid $\boldsymbol{m}_i$ of cluster U_i is computed as

$$\boldsymbol{m}_i = \begin{cases} w_{\text{low}} \frac{\sum_{X_k \in \underline{B}U_i} X_k}{|\underline{B}U_i|} + w_{\text{up}} \frac{\sum_{X_k \in (\overline{B}U_i - \underline{B}U_i)} X_k}{|\overline{B}U_i - \underline{B}U_i|} & \text{if } \overline{B}U_i - \underline{B}U_i \neq \emptyset, \\ w_{\text{low}} \frac{\sum_{X_k \in \underline{B}U_i} X_k}{|\underline{B}U_i|} & \text{otherwise,} \end{cases} \tag{7}$$

where the parameters w_{low} and w_{up} correspond to the relative importance of the lower and upper approximations respectively. Here $|\underline{B}U_i|$ indicates the number of pattern points in the lower approximation of cluster U_i, while $|\overline{B}U_i - \underline{B}U_i|$ is the number of elements in the rough boundary lying between the two approximations.

The algorithm is outlined as follows.

- Assign initial means $\boldsymbol{m}_i$ for the c clusters.
- Assign each data object (pattern point) $\boldsymbol{X}_k$ to the lower approximation $|\underline{B}U_i|$ or upper approximation $|\overline{B}U_i|$ of cluster U_i, by computing the difference in its distance $d(\boldsymbol{X}_k, \boldsymbol{m}_i) - d(\boldsymbol{X}_k, \boldsymbol{m}_j)$ from cluster centroid pairs $\boldsymbol{m}_i$ and $\boldsymbol{m}_j$.
- **If** $d(\boldsymbol{X}_k, \boldsymbol{m}_i) - d(\boldsymbol{X}_k, \boldsymbol{m}_j)$ is less than some threshold **then** $\boldsymbol{X}_k \in \overline{B}U_i$ and $\boldsymbol{X}_k \in \overline{B}U_j$ and $\boldsymbol{X}_k$ cannot be a member of any lower approximation, **else** $\boldsymbol{X}_k \in \underline{B}U_i$ such that distance $d(\boldsymbol{X}_k, \boldsymbol{m}_i)$ is minimum over the c clusters.
- Compute new mean for each cluster U_i using Eq. (7).
- Iterate until convergence, i.e., there are no more new assignments.

The expression in Eq. (7) boils down to Eq. (1) when the lower approximation is equal to the upper approximation, implying an empty boundary region. It is to be noted that a major disadvantage of this algorithm is the involvement of too many user-defined parameters.

3.3. Evolutionary optimization

In this article we employed an evolutionary approach to compute the optimal values of the parameters involved. It is observed that the performance of the algorithm is dependent on the choice of w_{low}, w_{up} and threshold. We allowed $w_{\text{up}} = 1 - w_{\text{low}}$, $0.5 < w_{\text{low}} < 1$ and $0 <$ threshold < 0.5.

It is to be noted that the parameter threshold measures the relative distance of an object $\boldsymbol{X}_k$ from a pair of clusters having centroids $\boldsymbol{m}_i$ and $\boldsymbol{m}_j$. The smaller the value of threshold, the more likely is $\boldsymbol{X}_k$ to lie within the rough boundary (between upper and lower approximations) of a cluster. This implies that only those points which definitely belong to a cluster (lie close to the centroid) occur within the lower approximation. A large value of threshold implies a relaxation of this criterion, such that more patterns are allowed to belong to any of the lower approximations.

The parameter w_{low} controls the importance of the objects lying within the lower approximation of a cluster in determining its centroid. A lower w_{low} implies a higher w_{up}, and hence an increased importance of patterns located in the rough boundary of a cluster towards the positioning of its centroid.

GAs are used to determine the optimal values of the parameters w_{low} and threshold for each c (number of clusters). Each parameter is encoded using ten bits in a chromosome. The value of the corresponding Davies–Bouldin index is chosen as the fitness function to be minimized. Crossover and mutation probabilities of $p_c = 0.8$ and $p_m = 0.02$ were selected for a population size of 20 chromosomes.

The main steps of the algorithm are provided below.

(i) Choose the initial means $\boldsymbol{m}_i$ for the c clusters.
(ii) Initialize the population of chromosomes encoding parameters threshold and w_{low}.

(iii) Tune the parameters by minimizing the Davies–Bouldin index [expression (6)] as the fitness function for the GA, considering objects lying within the lower approximation of each cluster.

(iv) Assign each data object (pattern point) $\boldsymbol{X}_k$ to the lower approximation $|\underline{B}U_i|$ or upper approximation $|\overline{B}U_i|$ of cluster U_i, by computing the difference in its distance $d(\boldsymbol{X}_k, \boldsymbol{m}_i) - d(\boldsymbol{X}_k, \boldsymbol{m}_j)$ from cluster centroid pairs $\boldsymbol{m}_i$ and $\boldsymbol{m}_j$.

(v) **If** $d(\boldsymbol{X}_k, \boldsymbol{m}_i) - d(\boldsymbol{X}_k, \boldsymbol{m}_j)$ is less than some threshold **then** $\boldsymbol{X}_k \in \overline{B}U_i$ and $\boldsymbol{X}_k \in \overline{B}U_j$ and $\boldsymbol{X}_k$ cannot be a member of any lower approximation, **else** $\boldsymbol{X}_k \in \underline{B}U_i$ such that distance $d(\boldsymbol{X}_k, \boldsymbol{m}_i)$ is minimum over the c clusters.

(vi) Compute new mean for each cluster U_i using Eq. (7).

(vii) Repeat Steps (iii)–(vi) until convergence.

4. Results

The different clustering algorithms were implemented on four benchmark datasets, viz., synthetic data *Pat*, speech data *Vowel*, large data *Forest Cover*, and gene expression data *Colon Cancer*. Table 1 provides the results obtained with the different algorithms. The optimal number of clusters and the corresponding minimum value of the Davies–Bouldin index are given in the last two columns respectively.

4.1. Synthetic data

The synthetic data *Pat* consists of 417 pattern points in the two-dimensional space $F_1 - F_2$ as depicted in Fig. 2(a). There are two linearly nonseparable pattern classes. The figure is marked with classes 1 (C_1) and 2 (C_2).

The original data consists of two classes, *viz.*, C_1 and C_2, as shown in Fig. 2(a). Davies–Bouldin

Table 1
Comparative performance of clustering algorithms

Data	Clustering algorithm	No. of clusters	Davies–Bouldin index
PAT	*c*-Means	10	0.409
	PAM	10	0.413
	CLARANS	10	0.410
	Fuzzy *c*-means	12	0.420
	Fuzzy *c*-medoids	11	0.240
	Evolutionary rough *c*-means	10	0.419
VOWEL	*c*-Means	4	0.757
	PAM	5	0.536
	CLARANS	5	0.547
	Fuzzy *c*-means	5	0.709
	Fuzzy *c*-medoids	4	0.691
	Evolutionary rough *c*-means	7	0.517
FOREST COVER	*c*-Means	5	0.539
	PAM	7	0.531
	CLARANS	5	0.532
	Fuzzy *c*-means	5	0.539
	Fuzzy *c*-medoids	5	0.502
	Evolutionary rough *c*-means	6	0.560
COLON CANCER	*c*-Means	2	0.646
	PAM	2	0.780
	CLARANS	2	0.700
	Fuzzy *c*-means	2	0.742
	Fuzzy *c*-medoids	2	0.732
	Evolutionary rough *c*-means	2	0.646

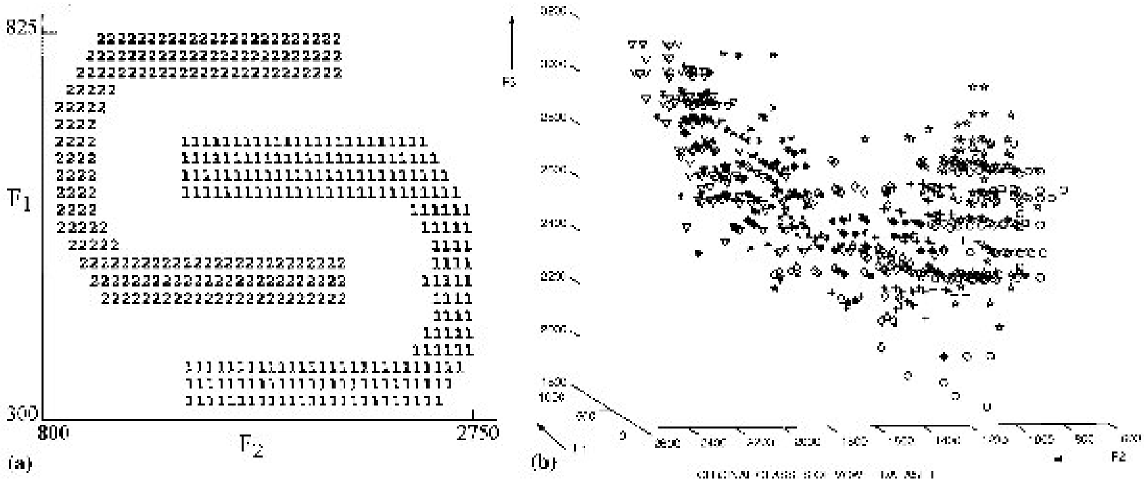

Fig. 2. Datasets (a) *Pat* and (b) *Vowel*.

validity index was used to determine the optimal number of clusters in each case. The results are provided in the first row of Table 1. The corresponding map of the patterns, along with the centroids (mean or medoid, marked by a rectangle), are illustrated in Fig. 3(a)–(e) for different clustering algorithms.

Fig. 3(a)–(c) depict the results with algorithms *c*-means, PAM and CLARANS, forming ten clusters (five from each class). Fuzzy *c*-means in Fig. 3(d) splits class C_1 into seven partitions, resulting in a total of twelve clusters. Fuzzy *c*-medoids in Fig. 3(e) leads to six partitions from class C_1, for a total of eleven clusters. Note that, unlike means, the medoids in Fig. 3(b), (c) and (e) correspond to patterns from the original dataset.

Fig. 3(f) illustrates the generation of ten clusters (five from each class) by the evolutionary rough *c*-means algorithm, with w_{low} and threshold evolved to be 0.97 and 0.34 respectively. The results are comparable to that obtained by the other algorithms in Fig. 3(a)–(c). It is therefore able to efficiently model the highly non-linear decision regions with a lower number of clusters.

4.2. Speech data

The *Vowel* data consists of a set of 871 Indian Telugu vowel sounds (Pal and Mitra, 1999), uttered by three male speakers in the age group of 30–35 years, in a Consonant–Vowel–Consonant context. The three features F_1, F_2 and F_3 correspond to the first, second and third vowel format frequencies obtained through spectrum analysis of the speech data. Fig. 2(b) shows the six vowel classes ∂, *a*, *i*, *u*, *e*, *o*, marked with symbols "*diamond*", "*plus*", "*upper triangle*", "*circle*", "*star*", "*pentagon*", respectively. The boundaries of the classes in the given data set are highly fuzzy.

The second row of Table 1 provides the results with the different algorithms for *Vowel* data. Fig. 4(a)–(e) illustrate the corresponding 3D maps for these algorithms. The centroids are marked by rectangles in each figure. Fig. 4(f) depicts the output map generated by the evolutionary rough *c*-means algorithm, with optimum values of $w_{\mathrm{low}} = 0.97$ and threshold $= 0.41$. The optimized Davies–Bouldin index is found to be the minimum with the evolutionary rough *c*-means algorithm for this fuzzy (overlapping) data. The inherent *roughness* in this clustering mechanism handles the uncertainty (ambiguity) among the six overlapping classes in an appropriate manner. All the other algorithms result in less than six clusters, leading to a clubbing of one or more classes.

4.3. Large data

The *Forest Cover* data (*http://kdd.ics.uci.edu/*) corresponds to the forest cover type for 30×30 m

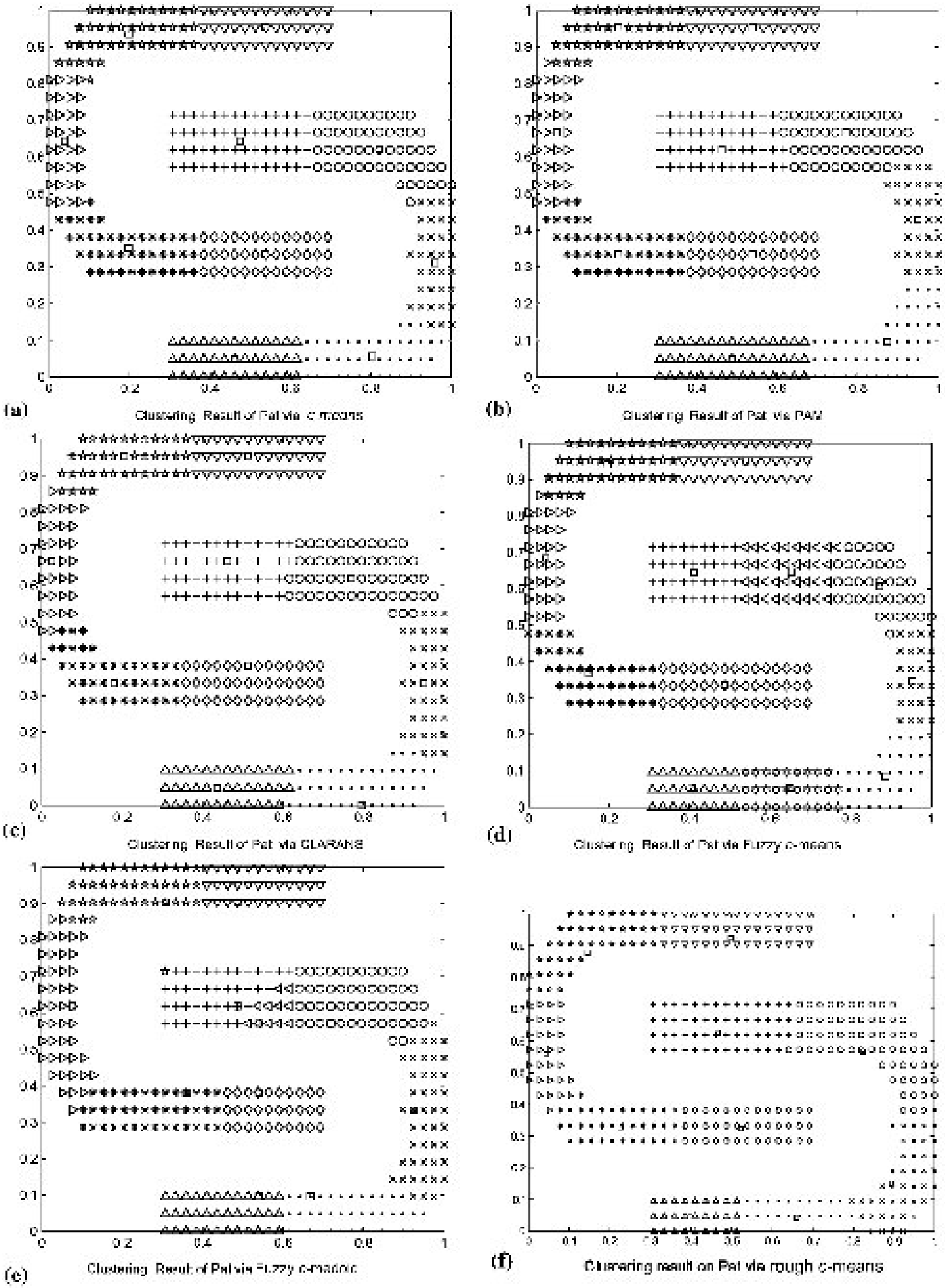

Fig. 3. Synthetic pattern *Pat* (a) *c*-means, (b) PAM, (c) CLARANS, (d) fuzzy *c*-means, (e) fuzzy *c*-medoids, and (f) evolutionary rough *c*-means.

cells obtained from the US Forest Service (USFS) Region 2 Resource Information System (RIS) data. There exist 5,81,012 observations (pattern points) with 54 attributes, of which there are ten quantitative variables (Elevation, Aspect, Slope, Horizontal-Distance-To-Hydrology, Vertical-Dis-

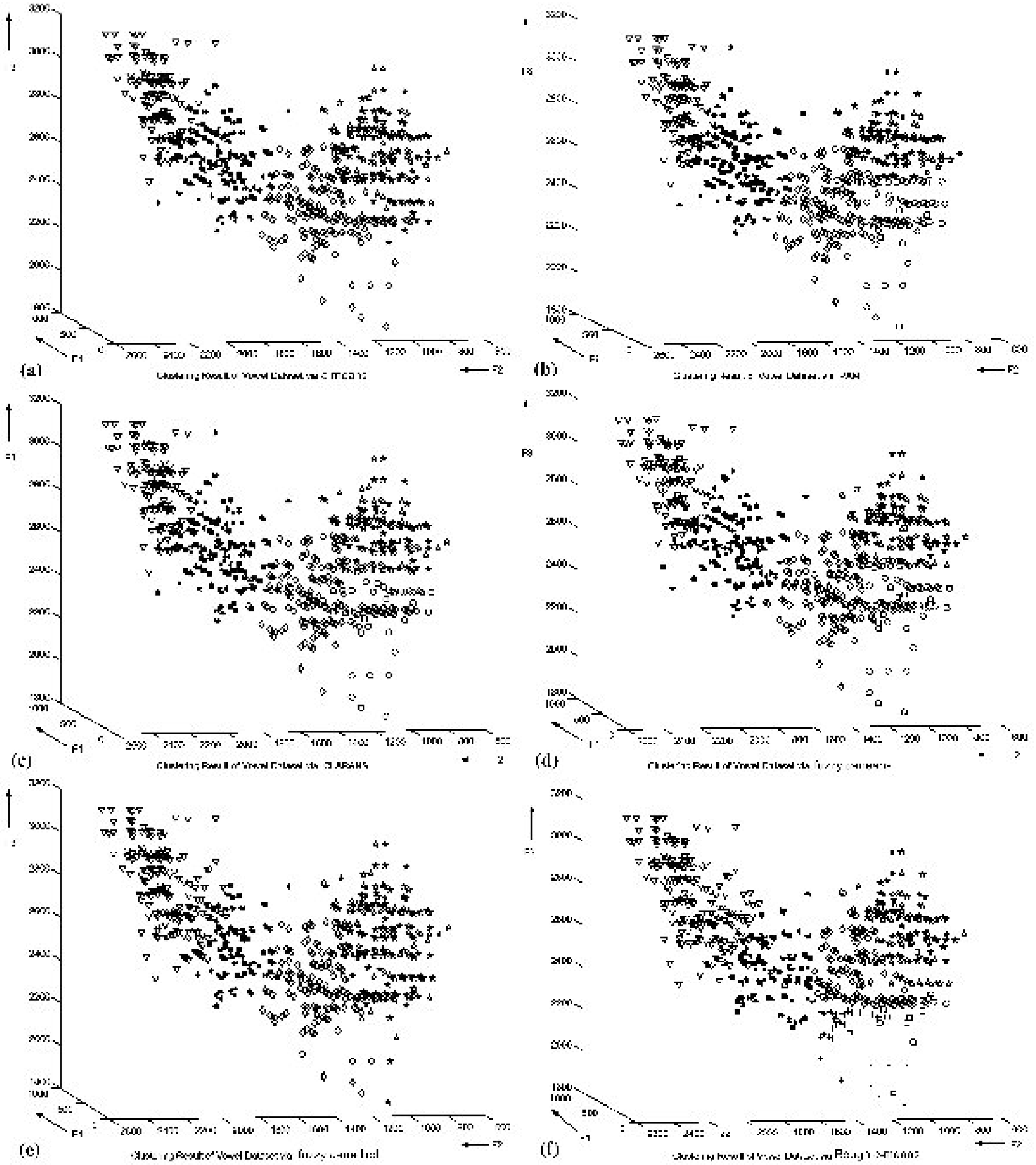

Fig. 4. *Vowel* diagram (a) *c*-means, (b) PAM, (c) CLARANS, (d) fuzzy *c*-means, (e) fuzzy *c*-medoids, and (f) evolutionary rough *c*-means.

tance-To-Hydrology, Horizontal-Distance-To-Roadways, Hillshade-9am, Hillshade-Noon, Hillshade-3pm, Horizontal-Distance-To-Fire-Points, Wilderness-Area, Soil-Type, Cover-Type), four binary wilderness areas (Rawah, Neota, Comanche Peak, Cache la Poudre) and 40 binary soil type variables. There are seven forest cover types corresponding to Spruce/Fir, Lodgepole Pine, Ponderosa Pine, Cottonwood/Willow, Aspen, Douglas-fir and Krummholz.

The clustering results for the different algorithms on this data are provided in the third row of Table 1. PAM generates exactly seven clusters. The optimum values of w_{low} and threshold were evolved to be 0.99 and 0.01 respectively, with the evolutionary rough c-means algorithm. This provided the second closest approximation to the seven forest cover classes.

4.4. Gene expression data

Microarray analysis (Special Issue on Bioinformatics, 2002) consists of (i) extraction of messenger RNAs from a biological sample, (ii) conversion into DNA, (iii) labeling with fluorescent dyes, and (iv) washing over a glass slide bearing a grid spotted with DNA sequences from known genes. The labeled sequences bind to spots representing the genes from which the messenger RNAs were transcribed. By analyzing the location and intensity of the fluorescent signals, one can determine the level of activity for each gene.

This can be used to generate the gene expression matrix, where the rows represent individual genes (features) and the columns are the individual samples. Each cell contains a measure of the gene's activity in the sample. Often genes with similar profiles of activity (coexpressed) have related functions or are regulated by common mechanisms. Hence clustering of microarray data has assumed great importance. Most of the literature, in this direction, use hierarchical clustering. However, it has been established that this technique is not very suitable for handling large data. Hence we investigated the applicability of partitive clustering on gene expression data.

The *Colon Cancer* data (*http://microaaray.princeton.edu/oncology*) is a collection of 62 gene expression measurements from colon biopsy samples. There are 22 normal and 40 colon cancer samples, having 2000 genes (features).

Gene expression data typically consists of a small number of samples with very large number of features, of which many are redundant. We first did some initial clustering on the expression values, to detect those genes that were highly coexpressed (or correlated) in either of the two output cases. In this manner, we selected 29 out of the existing 2000 genes for further processing. This was followed by clustering on the samples using the different related algorithms. Results are provided in the last row of Table 1. The optimum values of parameters generated by the evolutionary rough c-means algorithm was $w_{\text{low}} = 0.92$ and threshold $= 0.39$. In this case, the optimized Davies–Bouldin index for the proposed algorithm was also found to tie (with c-means) at the minimum value.

5. Conclusions and discussion

We have described the formulation of an evolutionary rough c-means clustering algorithm. The relative importance of the upper and lower approximations, and the threshold of the rough clusters are optimized using GAs. The Davies–Bouldin clustering validity index is chosen as the fitness function being minimized. Results are provided on real and synthetic datasets, including microarray gene expression data.

The clusters are modeled as c rough sets, expressed in terms of upper and lower approximations. However the optimal partitioning depends upon the suitable choice of these regions. The cluster center of Eq. (7) is reasonably affected by the user-defined parameter values. These are effectively tuned here, using GAs.

The results provided on microarray gene expression data in the last row of Table 1 serve as an interesting study from the point of view of clustering in Bioinformatics as well. There exist references in literature to the use of hierarchical clustering in this domain (Special Issue on Bioinformatics, 2002). we have tried to present some insight into the use of different partitive clustering algorithms in microarray data. It is found that the evolutionary rough clustering algorithm performs consistently over different benchmark datasets, and particularly well over the *Colon Cancer* gene expression data.

Acknowledgements

The author is thankful to the anonymous referees for their valuable suggestions. The programming support of Ms. S.A. Siddiqui is gratefully acknowledged.

References

Bezdek, J.C., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

Bezdek, J.C., Pal, N.R., 1998. Some new indexes for cluster validity. IEEE Trans. Systems Man Cybernet., Part-B 28, 301–315.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA.

Kaufman, L., Rousseeuw, P.J., 1990. Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley & Sons, New York.

Krishnapuram, R., Joshi, A., Nasraoui, O., Yi, L., 2001. Low complexity fuzzy relational clustering algorithms for web mining. IEEE Trans. Fuzzy Syst. 9, 595–607.

Lingras, P., West, C., 2002. Interval set clustering of Web users with rough *k*-means. Technical Report No. 2002-002, Department of Mathematics and Computer Science, St. Mary's University, Halifax, Canada, 2002.

Mitra, S., Acharya, T., 2003. Data Mining: Multimedia, Soft Computing, and Bioinformatics. John Wiley, New York.

Ng, R., Han, J., 1994. Efficient and effective clustering method for spatial data mining. In: Proc. 1994 Internat. Conf. Very Large Data Bases (VLDB'94). Santiago, Chile, September 1994, pp. 144–155.

Pal, S.K., Mitra, S., 1999. Neuro-fuzzy Pattern Recognition: Methods in Soft Computing. John Wiley, New York.

Pawlak, Z., 1991. Rough Sets, Theoretical Aspects of Reasoning about Data. Kluwer Academic, Dordrecht.

Special issue on bioinformatics. Part I: Advances and challenges. Proc. IEEE, 90 (11), November 2002.

Tou, J.T., Gonzalez, R.C., 1974. Pattern Recognition Principles. Addison-Wesley, London.

Zadeh, L.A., 1994. Fuzzy logic, neural networks, and soft computing. Communications of the ACM 37, 77–84.