Simulated Annealing Using a Reversible Jump Markov Chain Monte Carlo Algorithm for Fuzzy Clustering

Sanghamitra Bandyopadhyay

Abstract—In this paper, an approach for automatically clustering a data set into a number of fuzzy partitions with a simulated annealing using a Reversible Jump Markov Chain Monte Carlo algorithm is proposed. This is in contrast to the widely used fuzzy clustering scheme, the Fuzzy C-Means (FCM) algorithm, which requires the a priori knowledge of the number of clusters. The said approach performs the clustering by optimizing a cluster validity index, the Xie-Beni index. It makes use of the homogeneous Reversible Jump Markov Chain Monte Carlo (RJMCMC) kernel as the proposal so that the algorithm is able to jump between different dimensions, i.e., number of clusters, until the correct value is obtained. Different moves, like birth, death, split, merge, and update, are used for sampling a candidate state given the current state. The effectiveness of the proposed technique in optimizing the Xie-Beni index and thereby determining the appropriate clustering is demonstrated for both artificial and real-life data sets. In a part of the investigation, the utility of the fuzzy clustering scheme for classifying pixels in an IRS satellite image of Kolkata is studied. A technique for reducing the computation efforts in the case of satellite image data is incorporated.

Index Terms—Pattern recognition, fuzzy clustering, cluster validity index, determining the number of clusters, Reversible Jump Markov Chain Monte Carlo, simulated annealing, remote sensing.

1 Introduction

Clustering [1], also known as unsupervised learning, is a process of assigning labels to a set of n patterns in $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^N$ such that patterns that are similar are assigned the same label, while those that are dissimilar get different labels. A c-partition of X can be conveniently represented by a $c \times n$ matrix, called the partition matrix $U = [u_{ik}]$, where u_{ik} denotes the membership of the pattern x_k, $1 \leq k \leq n$, to cluster i, $1 \leq i \leq c$. Different approaches to partitioning the data into c clusters are available in the literature [1], [2], [3], viz. hard, fuzzy or probabilistic, and possibilistic. For classification of unlabeled data, there are three fundamental issues that must be addressed: 1) whether there is any clustering tendency in the data or not, 2) if yes, then what is a good method to find the clusters, and 3) in what way can one validate the obtained partitions.

Clustering tendency is an important, though little researched, step in exploratory data analysis where it is assessed whether the data is purely random in nature or some inherent structure exists in it. Some approaches to assessing clustering tendency can be found in [1]. Selecting a good and effective method to find clusters in data depends on several factors, such as size of the data, match of the data to the algorithm, choice of the parameters of the algorithm, etc. Once clustering is done, a cluster validity measure is used to indicate the goodness of the obtained partitioning. The cluster validity criterion may be external, internal, or relative. In the external validity index, the cluster structure is matched to a priori information. Internal indices assess the fit between the structures and the data using only the data set. Relative index compares between two structures to indicate the relative goodness of one over the other in terms of stability or fit to the data. Many cluster validity indices proposed in the literature may be found in [1], [4], [5], [6].

In the present article, we deal with fuzzy partitioning of the data. Fuzzy C-Means (FCM) [3] is a widely used technique in this regard. However, it suffers from two major limitations. First of all, FCM requires the a priori specification of the number of clusters. Second, depending on the initial cluster centers selected, the algorithm may get stuck at locally optimal error values. Here, we aim to develop an algorithm that will automatically find the fuzzy partitions and will be able to escape from local optima. Since the number of clusters is variable, the optimizing criterion of the FCM algorithm cannot be used in this scenario. Therefore, the automatic clustering problem is posed as one of optimization of a fuzzy cluster validity index. Some validity measures in the context of fuzzy clustering are the *partition coefficient*, *partition entropy*, *uniform data function*, *Fukuyama-Sugeno index*, *partition index*, and *Xie-Beni index* [6], [7], [8], [9], [10], [11]. In this article, we use the Xie-Beni (XB) cluster validity index as the underlying optimizing criterion since it has been shown to be better able to indicate the correct number of clusters in several experiments [12]. The capability of simulated annealing (SA) is used for searching in the space of fuzzy partitionings (represented by the corresponding cluster centers) for one that is associated with the minimum value the XB index.

Simulated Annealing (SA) [13], [14] belongs to a class of local search algorithms. It utilizes the principles of statistical

• *The author is with the Machine Intelligence Unit, Indian Statistical Institute, 203 B. T. Road, Kolkata 700 108, India.*
E-mail: sanghami@isical.ac.in.

mechanics regarding the behavior of a large number of atoms at low temperature for finding minimal cost solutions to large optimization problems by minimizing the associated energy. In statistical mechanics, investigating the ground states or low energy states of matter is of fundamental importance. These states are achieved at very low temperature. However, it is not sufficient to lower the temperature alone since this results in unstable states. In the annealing process, the temperature is first raised, then decreased gradually to a very low value (T_{min}) while ensuring that one spends sufficient time at each temperature. This process yields stable low energy states.

SA has been used previously for optimization of the K-means clustering model [15] in [16] (with the concept of probabilistic redistribution of data points) and [17], [18] for a fixed number of clusters known a priori. A similar approach is used in [19], where SA is used for the optimization of the K-means squared error criterion. The number of clusters is varied over a fixed range and the SA algorithm is executed for each fixed value of the number of clusters. The correct number of clusters is considered to be one for which the average value of the fraction of feature vectors incorrectly placed in the same cluster reaches a specified quantity. Here, two vectors are said to be incorrectly placed in the same cluster if the distance between them exceeds a threshold, which is computed using reverse engineering experiments. In [20], the total within cluster distance is used as the criterion to be minimized using SA for a fixed number of clusters. Moreover, the need to accurately estimate the number of clusters is eliminated by using Barker's criterion, which incorporates a user specified threshold. The method is found to be sensitive to the choice of the threshold. In [17], [18], [20], the partition assignment of the individual data points is taken as representing a state in SA and the perturbation operator is defined as moving a randomly chosen point from one cluster to another. Using such a representation where the size of the encoded solution is equal to the number of data points to be clustered is likely to result in a slow exploration of the search space. Another approach that has been adopted in this paper could be to use the set of cluster centers as representing the state and define the perturbation and other operators on the cluster centers. This is likely to be beneficial in terms of a faster exploration of the search space and, therefore, better convergence of the algorithm. (This was observed in [21] using genetic algorithms as the underlying search strategy where a comparison, in terms of the number of iterations for attaining a desired objective value, was conducted between such an encoding strategy and the one where the length of the encoded solution is equal to the number of data points to be clustered.) Other applications of simulated annealing or deterministic annealing for clustering may be found in [22], [23], [24], [25]. Some applications of SA in other domains involving incorporation of fuzzy set theoretic concepts are available in [26], [27].

In this paper, we propose an SA-based clustering technique where a configuration or state encodes the centers of a variable number of fuzzy clusters and the XB index of the corresponding partition is used to measure its energy value. Since the number of clusters is variable, the search has to jump between different dimensions (number of clusters). The Reversible Jump Markov Chain Monte Carlo (RJMCMC) [28] perspective is used in the simulated annealing procedure for solving the stochastic optimization of the cluster validity index. This makes use of the homogeneous Reversible Jump MCMC kernel as the proposal, which is able to jump between different dimensions of the state space until it finds the correct value. The Hasting-Metropolis algorithm [29] is used for computing the acceptance probability of a new configuration. The RJMCMC algorithm has been used earlier for segmentation of an image [30]. Another attempt using simulated annealing with the RJMCMC algorithm for model selection in radial basis function networks is available in [31]. Motivated by this work, we use simulated annealing using RJMCMC (SA-RJMCMC) for optimizing a cluster validity index, thereby yielding a clustering algorithm that can automatically determine the correct number of clusters as well as the appropriate clustering. To the best of our knowledge, use of SA-RJMCMC for clustering a data set when the number of clusters is not fixed a priori by optimizing a cluster validity index has not been attempted earlier. The effectiveness of the proposed technique is demonstrated for several artificial and real-life data sets, including a satellite image data, with the number of features ranging from two to nine and the number of clusters ranging from two to 10.

2 Fuzzy Clustering Model

In most of the real-life cases, cluster boundaries are not well defined and unambiguous assignment of patterns to clusters is difficult. In such situations, the principles of fuzzy set theory, which permits an object to belong to a cluster with a grade of membership, are applied to provide fuzzy clustering algorithms [3]. One such widely used algorithm, the Fuzzy C-Means (FCM), is first described in this section. Thereafter, the Xie-Beni cluster validity index is explained.

2.1 FCM Algorithm

Fuzzy C-Means is a widely used technique that uses the principles of fuzzy sets to evolve a fuzzy partition matrix for a given data set. The set of all $c \times n$ (where c and n denote the number of clusters and data points, respectively) nondegenerate constrained fuzzy partition matrices, denoted by M_{fcn}, is defined as

$$M_{fcn} = \left\{ U \in R^{c \times n} \mid \sum_{i=1}^{c} u_{ik} = 1, \ \sum_{k=1}^{n} u_{ik} > 0, \ \forall i \text{ and } u_{ik} \in [0,1]; 1 \le i \le c; 1 \le k \le n \right\}. \tag{1}$$

The minimizing criterion used to define good clusters for Fuzzy C-Means partitions is the FCM function, defined as

$$J_\mu(U,V) = \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^\mu S_{ik}^2. \tag{2}$$

Here, $U \in M_{fcn}$ is a fuzzy partition matrix, $\mu \in [1, \infty]$ is the weighting exponent on each fuzzy membership, $V = [v_1, \ldots, v_c]$ represents c cluster centers, $v_i \in \mathbb{R}^N$, and S_{ik} is

the distance from x_k to the ith cluster center. The Fuzzy C-Means theorem [3] states that if $S_{ik} > 0$ for all i and k, then (U, V) may minimize J_μ only if when $\mu > 1$

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{S_{ik}}{S_{jk}}\right)^{\frac{2}{\mu-1}}}, \quad \text{for } 1 \le \text{i} \le \text{c}; \ \ 1 \le \text{k} \le \text{n} \tag{3}$$

and

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^\mu x_k}{\sum_{k=1}^{n} (u_{ik})^\mu}, \quad 1 \le i \le c. \tag{4}$$

A common strategy for generating the approximate solutions of the minimization problem in (2) is by iterating through (3) and (4) (also known as the Picard iteration technique). A detailed description of the FCM algorithm may be found in [3]. Note that minimization of $J_\mu(U, V)$ is appropriate when the number of clusters is known a priori. However, in case the number of clusters is variable, this cannot be done since, in such cases, the minimum (= 0) will occur when the number of clusters is equal to n, the number of data points.

2.2 Fuzzy Cluster Validity Index

The cluster validity index is often used to indicate the goodness of a partitioning of a data set as obtained by a clustering algorithm. Xie and Beni [11] proposed one such validity measure, which is defined as the ratio of the total variation σ to the minimum separation sep of the clusters, where σ and sep can be written as

$$\sigma(U, V; X) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2, \tag{5}$$

where X is the feature vectors and

$$sep(V) = \min_{i \neq j} \{\|v_i - v_j\|^2\}. \tag{6}$$

The XB index is then written as

$$XB(U, V; X) = \frac{\sigma(U, V; X)}{n \, sep(V)} = \frac{\sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^2 ||x_k - v_i||^2}{n(min_{i \neq j}\{||v_i - v_j||^2\})}. \tag{7}$$

Note that, when the partitioning is compact and good, the value of σ should be low, while sep should be high. Therefore, the Xie-Beni index (XB) should have a low value when the data has been appropriately clustered. In other words, if the XB index of a particular tuple (U_1, V_1) is XB_1 and that of another tuple (U_2, V_2) is XB_2 and if $XB_1 < XB_2$, then the partitioning corresponding to (U_1, V_1) is taken to be better than that of (U_2, V_2).

There are several other fuzzy cluster validity indices that are available in the literature. Some such indices are Bezdek's partition coefficient (PC) [7] and partition entropy (PE) [8], partition index (SC) [10], and the Fukuyama and Sugeno index (FS) [6]. As already mentioned, the XB index has been found to be better able to indicate the correct number of partitions in the data [12] for a wide range of the choice of the number of clusters. Hence, the XB index is used as the criterion that is optimized using a simulated annealing strategy with an RJMCMC algorithm.

3 Simulated Annealing Using the RJMCMC Algorithm for Clustering

In this section, the utility of simulated annealing using an RJMCMC algorithm (SA-RJMCMC) for the clustering problem is described in detail. The SA-RJMCMC is motivated by the approach in [31], where it is used for finding the parameters of a radial basis function network. In the present paper, SA-RJMCMC is used to find the appropriate clustering parameters such that the overall XB value (= $\sum_{i=1}^{n} XB_i$, where XB_i is the contribution of x_i to the overall XB index value) is minimized. In matrix notation, this can be written as

$$\begin{bmatrix} XB_1 \\ XB_2 \\ \vdots \\ XB_n \end{bmatrix} = \begin{bmatrix} \phi(x_1, v_1) \ \phi(x_1, v_2) \ldots \ \phi(x_1, v_c) \\ \phi(x_2, v_1) \ \phi(x_2, v_2) \ldots \ \phi(x_2, v_c) \\ \vdots \\ \phi(x_n, v_1) \ \phi(x_n, v_2) \ldots \ \phi(x_n, v_c) \end{bmatrix} \begin{bmatrix} 1/(n \min_{i \neq j} \|v_i - v_j\|^2) \\ \vdots \\ c \, times \end{bmatrix}, \tag{8}$$

where

$$\phi(x_k, v_i) = u_{ik}^2 ||x_k - v_i||^2. \tag{9}$$

Or, this may be written as

$$\mathbf{XB} = \mathbf{D}(X, V) \, \mathbf{I_c}, \tag{10}$$

where $\mathbf{I_c}$ is a vector of all 1s having length c, $\mathbf{XB}$ is the XB vector, $\mathbf{D}$ is the $n \times c$ matrix that is to be learned for clustering, and X and V are the feature vector and the vector of the cluster centers, respectively.

In the SA method, a nonhomogeneous Markov chain is simulated whose invariant distribution at iteration i, $\pi_i(z)$ (where z is a configuration or state), is

$$\pi_i(z) \ \propto \ \pi^{1/T_i}(z). \tag{11}$$

Here, T_i is a decreasing cooling schedule with $\lim_{i \to \infty} T_i = 0$. With the Hasting-Metropolis [29] method and a proposal distribution $q(z^*|z)$ (which implies that, given the current state z, a candidate next state z^* is drawn according to the distribution $q(z^*|z)$), the acceptance probability of z^* (when the model parameter space has fixed dimensionality) becomes equal to

$$\min\left(1, \frac{\pi^{(1/T_i - 1)}(z^*) q(z|z^*)}{\pi^{(1/T_i - 1)}(z) q(z^*|z)}\right). \tag{12}$$

When the dimensionality of the parameter space is variable, the acceptance probability is written as (following the deduction in [28]):

$$\begin{aligned} &\min(1, (posterior \ distribution \ ratio) \\ &\times (proposal \ ratio) \times (Jacobian)). \end{aligned} \tag{13}$$

```
Begin
    Initialize c_0, V_0 /* c_0 - initial number of clusters, V_0 initial cluster centers */
    T = T_max
    while (T ≥ T_min)
      for i = 1 to k
        r = rand(0,1) /* rand(0,1) returns a number in [0,1] with uniform distribution */
        if (r < 0.2) Birth move
        if (r < 0.4) Death move
        if (r < 0.6) Split move
        if (r < 0.8) Merge move
        else Perturb move
      end
      Accept the new state with probability given by Eqn. 13
      T = αT
    end while
End
```

Fig. 1. Simulated annealing using RJMCMC algorithm with several moves.

If the proposal distribution that ensures reversibility property (explained later) is selected, the acceptance probability can be simplified further. For details, the reader may refer to [31].

The RJMCMC process consists of simulating a discrete Markov chain which performs small jumps between spaces of different dimensions (the number of clusters in this article). The measure of interest occurs as the stationary measure of the chain. This iterative algorithm does not depend on the initial state. At each step, a transition from the current state to a new state is proposed according to a proposal kernel. The transition is accepted with a probability given by (13). This acceptance ratio is computed so that the detailed balance condition is satisfied, under which the algorithm converges to the measure of interest. A characteristic feature of the algorithm is that the proposition kernel can be decomposed into several kernels, each corresponding to a reversible move, as has been proposed in [28]. In order for the underlying sampler to ensure the jump between different dimensions, the various moves used are the *birth move, death move, split move, merge move,* and *perturb move,* each selected with equal probability (= 0.2) [31]. The simulated annealing using RJMCMC algorithm is shown in Fig. 1. Here, α is the cooling rate.

3.1 Birth and Death Moves

The birth and death moves form a pair of reversible perturbations. In the birth move, a new center is selected at random from within the box bounding the data points. If d is the number of features and Max_i and Min_i represent the maximum and minimum value of feature i, $i = 1, \ldots, d$, then a new center, v_{new}, is selected as

$$v_{new}^{i} = rand(Min_i, Max_i) \quad \text{for } i = 1, 2, \ldots, d. \tag{14}$$

Thus, if the current number of centers is c, it now becomes $c+1$ after the birth move. Thereafter, the membership values are computed using (3) and the centers are recomputed using (4). The acceptance ratio for the proposed birth move is deduced from the expression (13) [28]. After some simplifications, this expression reduces to [31]:

$$r_{birth} = \left[\left(\frac{\mathbf{XB}'\, \mathbf{P}_c^*\, \mathbf{XB}}{\mathbf{XB}'\, \mathbf{P}_{c+1}^*\, \mathbf{XB}} \right)^{n/2} \right] \frac{\Im \exp(-\mathcal{C})}{c+1}, \tag{15}$$

where $\mathbf{P}_c^*$ is given by

$$\mathbf{P}_c^* = \mathbf{I_n} - \mathbf{D}(V, X)[\mathbf{D}'(V, X)\mathbf{D}(V, X)]^{-1}\mathbf{D}'(V, X), \tag{16}$$

$\Im$ is the hypervolume of the feature space given by

$$\Im = \Pi_{i=1}^{d}(Max_i - Min_i) \tag{17}$$

and $\mathcal{C} = p + 1$, p being the dimensionality of the output space, which is 1 in this case. This follows from the discussion in [31] for optimization using the AIC criterion

[32]. Hence, the acceptance probability corresponding to the birth move is:

$$A_{birth} = \min(1, r_{birth}). \tag{18}$$

The death move is the reverse of the birth move and consists of simply selecting one of c centers from V at random and deleting it. As a result, the number of clusters will reduce from c to $c-1$. The membership values are then computed using (3) and the centers are recomputed using (4). Similar to the case of the birth move, for the death move [31], we have

$$r_{death} = \left[\left(\frac{\mathbf{XB}'\,\mathbf{P}_{\mathbf{c}}^{*}\,\mathbf{XB}}{\mathbf{XB}'\,\mathbf{P}_{\mathbf{c-1}}^{*}\,\mathbf{XB}}\right)^{n/2}\right]\frac{c\exp(\mathcal{C})}{\Im}. \tag{19}$$

Hence, the acceptance probability corresponding to the death move is:

$$A_{death} = \min(1, r_{death}). \tag{20}$$

3.2 Split and Merge Moves

The split and the merge moves also form a pair of reversible perturbations. The split move involves randomly choosing a center from V. Let v_i be the selected center. It is replaced by two new centers, v_{i1} and v_{i2}, which are obtained as follows:

$$\begin{aligned} v_{i1} &= v_i - rand(0, \zeta^*) \\ v_{i2} &= v_i + rand(0, \zeta^*). \end{aligned} \tag{21}$$

Both v_{i1} and v_{i2} are bound to lie within the feature hypervolume and are adjusted if they happen to cross any feature boundary. Moreover, the distance between v_{i1} and v_{i2} has to be shorter than the distance between them and all other centers. Otherwise, the new centers are not accepted. Here, ζ^* is a user defined parameter and it, along with the criterion that the distance between the new centers should be shorter than the distance between them and all other centers, are important considerations for ensuring reversibility. After splitting, the membership values are computed using (3) and the centers are recomputed using (4). The expression for r_{split} is given by [31]

$$r_{split} = \left[\left(\frac{\mathbf{XB}'\mathbf{P}_{\mathbf{c}}^{*}\mathbf{XB}}{\mathbf{XB}'\mathbf{P}_{\mathbf{c+1}}^{*}\mathbf{XB}}\right)^{n/2}\right]\frac{c\zeta^*\exp(-\mathcal{C})}{c+1} \tag{22}$$

and the acceptance probability is given by

$$A_{split} = \min(1, r_{split}). \tag{23}$$

Similarly, in the merge move, a center is chosen randomly from V. Let this be v_i. Then, the center v_j that is closest to v_i is found. If $\|v_i - v_j\| < 2\zeta^*$, then the two centers v_i and v_j are replaced by a single center v_i^*, where

$$v_i^* = \frac{v_i + v_j}{2}. \tag{24}$$

The condition $\|v_i - v_j\| < 2\zeta^*$ ensures reversibility in the split and merge moves. After merging, the membership values are computed using (3) and the centers are recomputed using (4). Thereafter, r_{merge} is computed as [31]

$$r_{merge} = \left[\left(\frac{\mathbf{XB}'P_c^*\mathbf{XB}}{\mathbf{XB}'P_{c-1}^*\mathbf{XB}}\right)^{n/2}\right]\frac{c\exp(\mathcal{C})}{\zeta^*(c-1)}, \tag{25}$$

and the acceptance probability for this move is:

$$A_{merge} = \min(1, r_{merge}). \tag{26}$$

3.3 Perturb Move

Here, a cluster center, say v_i, is picked at random from the string and its value (on each dimension) is allowed to change (increase or decrease with equal probability) by at most f percent to yield v_i'. Then, v_i is replaced by v_i' and the membership values are computed using (3) and the centers are recomputed using (4). Thereafter, $r_{perturb}$ is computed as [31]

$$r_{perturb} = \left[\left(\frac{\mathbf{XB}'P_c^*\mathbf{XB}}{\mathbf{XB}'P_c^{1*}\mathbf{XB}}\right)^{n/2}\right], \tag{27}$$

where P_c^{1*} is the same as P^*, except that v_i is replaced by v_i'. The acceptance probability for this move is:

$$A_{perturb} = \min(1, r_{perturb}). \tag{28}$$

4 Experimental Results

The effectiveness of SA-RJMCMC in optimizing the Xie-Beni cluster validity index and thereby determining the appropriate number of clusters as also the proper partitioning is demonstrated for three artificial and two real-life data sets. The data sets, along with the implementation parameters, are first described in this section. Subsequently, the XB index values provided by the proposed technique, its comparison with FCM, and the partitionings obtained are presented. Finally, the effectiveness of SA-RJMCMC for automatically classifying the different landcover types in a satellite image of a part of Kolkata (which is a metropolitan city of India) is demonstrated. A technique for reducing the time complexity while clustering the satellite image data is also described.

4.1 Data Sets and Implementation Parameters

The three artificial data sets that have been used in this paper are *AD_10_2*, *AD_9_2*, and *AD_4_3N* and the two real-life data sets are *Cancer* and *Kalaazar*. Table 1 provides a

TABLE 1
Description of the Data Sets

Name	# points	# clusters	# features	Points per cluster
AD_10_2	500	10	2	50 per cluster
AD_9_2	900	9	2	100 per cluster
AD_4_3N	402	4	3	101,100,101,100
Cancer	683	2	9	444,239
Kalaazar	68	2	4	20,48

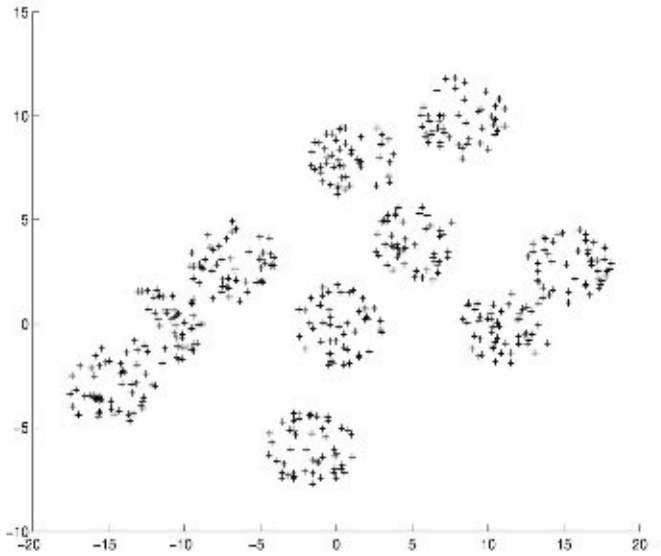

Fig. 2. *AD_10_2*.

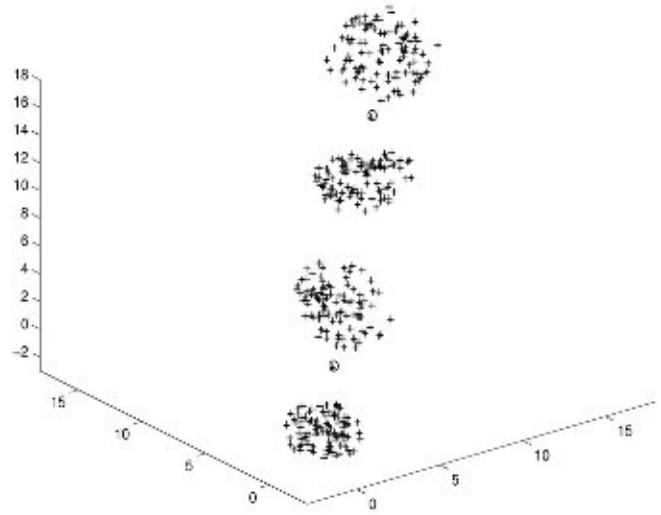

Fig. 3. *AD_4_3N*.

snapshot of the five data sets considered. These are now described.

AD_10_2 is a two-dimensional overlapping data set with 10 clusters having 50 patterns in each cluster. *AD_4_3N* is a three-dimensional data set with four clusters and 402 patterns. Fig. 2 and Fig. 3 show the two data sets, respectively. Note that *AD_4_3N* has two noisy points, each situated between a pair of well-separated clusters.

AD_9_2 is a two-dimensional, nine class data with triangular distribution of data points, each class having 100 patterns. All the classes are assumed to have equal a priori probabilities (= 1/9). The $X - Y$ ranges for the nine classes are as follows:

$$\begin{aligned}
&\text{Class } 1 : [-3.3, -0.7] \times [0.7, 3.3]\\
&\text{Class } 2 : [-1.3, 1.3] \times [0.7, 3.3]\\
&\text{Class } 3 : [0.7, 3.3] \times [0.7, 3.3]\\
&\text{Class } 4 : [-3.3, -0.7] \times [-1.3, 1.3]\\
&\text{Class } 5 : [-1.3, 1.3] \times [-1.3, 1.3]\\
&\text{Class } 6 : [0.7, 3.3] \times [-1.3, 1.3]\\
&\text{Class } 7 : [-3.3, -0.7] \times [-3.3, -0.7]\\
&\text{Class } 8 : [-1.3, 1.3] \times [-3.3, -0.7]\\
&\text{Class } 9 : [0.7, 3.3] \times [-3.3, -0.7].
\end{aligned}$$

The form of the triangular distribution is shown in Fig. 4. We have chosen the classes in such a way that each class has some overlap with each of its adjacent classes. In any direction, the nonoverlapping portion is 1.4 units and the overlapping portion is 1.2 units. The data set is shown in Fig. 5.

The *Cancer* data is the Wisconsin Breast Cancer data set, which is available at http://www.ics.uci.edu/~mlearn/MLRepository.html. Each pattern has nine features, corresponding to *clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli*, and *mitoses*. There are two categories in the data: malignant and benign. The two classes are known to be linearly inseparable. There are a total of 683 points in the data set.

The *Kalaazar* data [33] consists of 68 patterns with four features. There are two classes, diseased and normal/cured, and four input features/symptoms. These symptoms are the measurements of blood urea (mg percent), serum creatinine (mg percent), urinary creatinine (mg percent), and creatinine clearance (ml/min).

The parameters of the SA-based algorithm are as follows: $T_{max} = 100$, $T_{min} = 0.0001$, $\alpha = 0.8$, and number of iterations at each temperature = 30. The value of ζ^* is selected in the range $[1, 10]$ after some experimentations with the data and f is set equal to 10 percent. The number of clusters, c, is varied from 2 to $\sqrt{n}$ (since c is usually assumed to be less than or equal to $\sqrt{n}$), where n is the size of the data set. The average values obtained over five different runs of the proposed algorithm are reported in the tables. The FCM algorithm was sometimes found to get stuck in extremely poor local optima and, therefore, had to be run repeatedly to overcome this condition. The results reported here are the average values over the five best runs of the algorithm.

4.2 Comparative Results with FCM

Tables 2, 3, 4, 5, and 6 provide the comparative results of SA-RJMCMC and FCM for *AD_10_2*, *AD_9_2*, *AD_4_3N*,

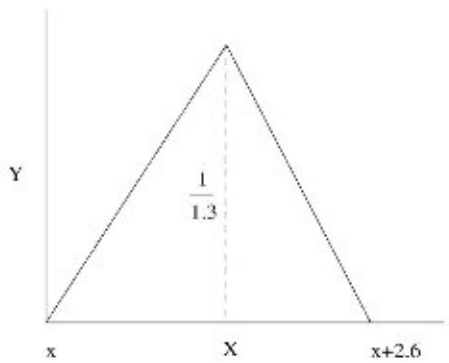

Fig. 4. Triangular distribution of data for *AD_9_2*.

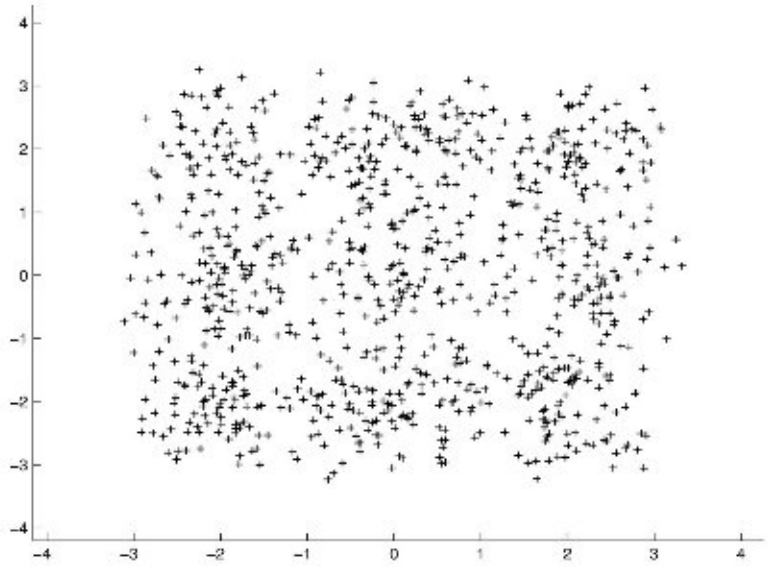

Fig. 5. *AD_9_2*.

Cancer, and *Kalaazar*, respectively. Table 7 provides a summary of the number of clusters provided by the two techniques when the minimum value of XB index is achieved for the five data sets. Recall the μ is the weighting exponent in (2). As can be seen from Tables 2, 3, 4, 5, and 6, the SA-RJMCMC technique generally provides lower values of the XB index for all values of μ for all the data sets, except for AD_10_2 when $\mu = 1.8$ (where FCM performs better) and AD_4_3N when $\mu = 1.6$ (where both are equal). (It may be noted that the standard deviation of the XB values was very small, being negligible for FCM and ranging between [0.00001,0.0005], [0.0001,0.0008], [0.000002,0.00005], [0.00007,0.0001], and [0.0004,0.0008] for *AD_10_2*, *AD_9_2*, *AD_4_3N*, *Cancer*, and *Kalaazar*, respectively.) Moreover, as can be seen from Table 7, the SA-based method provides the correct number of clusters for all the data sets. In contrast, FCM fails to provide this for *AD_10_2* and *Kalaazar*, where it yields nine and five clusters, respectively. Note that the number of clusters provided by FCM for a particular value of μ was obtained by varying c, the number of clusters, from $\max[2, c^* - 2]$ to $c^* + 2$, where c^* is equal to the actual number of clusters present in the data. The value of c for which the minimum value of the XB index was obtained was considered to be the number of clusters provided by FCM (as presented in Tables 2, 3, 4, 5, and 6). Table 8 is provided to illustrate the variation of the value of the XB index with the number of clusters for different values of μ for AD_10_2 when FCM is used for clustering.

It may be noted from Table 7 that, in general, the lowest value of the XB index corresponds to $\mu = 2.0$, except for the cases of AD_10_2 using SA-RJMCMC clustering (where this is obtained for $\mu = 2.2$), and the Kalaazar data set for both SA-RJMCMC clustering and FCM (where this is obtained for $\mu = 1.6$). Fig. 6 and Fig. 7 demonstrate the clustered *AD_10_2* and *AD_9_2* for the purpose of illustration.

This section focused primarily on the comparison of the proposed technique with FCM for the different data sets and for varying values of the weighting exponent μ. Results obtained by a genetic algorithm-based scheme [34] are now reported briefly for the sake of completion, assuming $\mu = 2.0$. Note that the genetic algorithm-based scheme also optimizes the XB index. The values obtained

TABLE 2
Comparative Values of Number of Clusters and XB Index for *AD_10_2*

weighting	FCM		SA-RJMCMC	
exponent μ	# clusters	XB	# clusters	XB
1.6	9	0.099873	9	0.098549
1.8	9	0.085746	10	0.088481
2.0	9	0.082805	10	0.076313
2.2	9	0.090368	10	0.075333
2.4	8	0.105605	10	0.086582

TABLE 3
Comparative Values of Number of Clusters and XB Index for *AD_9_2*

weighting	FCM		SA-RJMCMC	
exponent μ	# clusters	XB	# clusters	XB
1.6	9	0.089271	9	0.088668
1.8	9	0.078834	9	0.076344
2.0	9	0.076506	9	0.073647
2.2	9	0.081750	9	0.079578
2.4	9	0.092139	9	0.090645

TABLE 4
Comparative Values of Number of Clusters and XB Index for *AD_4_3N*

weighting	FCM		SA-RJMCMC	
exponent μ	# clusters	XB	# clusters	XB
1.6	4	0.050721	4	0.050721
1.8	4	0.048224	4	0.048169
2.0	4	0.046901	4	0.046880
2.2	4	0.048526	4	0.048353
2.4	4	0.053905	4	0.053699

TABLE 5
Comparative Values of Number of Clusters and XB Index for *Cancer*

weighting	FCM		SA-RJMCMC	
exponent μ	# clusters	XB	# clusters	XB
1.6	2	0.116757	2	0.115837
1.8	2	0.111636	2	0.110364
2.0	2	0.110321	2	0.108823
2.2	2	0.111721	2	0.110106
2.4	2	0.115064	2	0.113003

TABLE 6
Comparative Values of Number of Clusters and XB Index for *Kalaazar*

weighting	FCM		SA-RJMCMC	
exponent μ	# clusters	XB	# clusters	XB
1.6	5	0.203480	2	0.160129
1.8	5	0.206727	2	0.164318
2.0	5	0.219227	2	0.169988
2.2	5	0.235390	2	0.180719
2.4	5	0.303354	2	0.191887

for *AD_10_2*, *AD_9_2*, *AD_4_3N*, *Cancer*, and *Kalaazar* are 0.079711 for nine clusters, 0.071400 for nine clusters, 0.046895 for four clusters, 0.109099 for two clusters, and 0.165399 for two clusters, respectively. From Tables 2, 3, 4, 5, and 6. it becomes evident that, while for *AD_9_2* and *Kalaazar*, the genetic scheme performs better, for the other three data sets, the proposed scheme performs better. For *AD_10_2*, the genetic scheme is not able to indicate the proper number of clusters. It may be noted that both the proposed, as well as the genetic, schemes are stochastic in nature and require proper tuning of several parameters.

TABLE 7
Number of Clusters, μ and Value of XB Index Obtained by SA-RJMCMC and FCM

Data Set	FCM			SA-RJMCMC		
	# clusters	μ	XB	# clusters	μ	XB
AD_10_2	9	2.0	0.082805	10	2.2	0.075333
AD_9_2	9	2.0	0.076506	9	2.0	0.073647
AD_4_3N	4	2.0	0.046901	4	2.0	0.046880
Cancer	2	2.0	0.110321	2	2.0	0.108823
Kalaazar	5	1.6	0.203480	2	1.6	0.160129

TABLE 8
Variation of XB Index with the Number of Clusters in FCM for *AD_10_2*

μ	XB index for # clusters =				
	8	9	10	11	12
1.6	0.115106	0.099873	0.109384	0.295036	0.363325
1.8	0.101389	0.085746	0.099056	0.184069	0.178549
2.0	0.094680	0.082805	0.098666	0.170353	0.158347
2.2	0.096473	0.090368	0.112757	0.178307	0.161483
2.4	0.105605	0.105959	0.140555	0.388541	0.281217

They demonstrate similar performance for the given data sets, though the exact results may change with a different parameter setting.

The FCM is also dependent on the choice of the initial configuration and sometimes gets stuck at local optima. As mentioned earlier, it was run repeatedly so that it was able to overcome this condition and results were taken for such runs.

4.3 Classification of Satellite Image

In this section, we describe the application of the SA-RJMCMC technique for classifying landcover types from an IRS satellite image of a part of the city of Kolkata. The data used here was acquired from Indian Remote Sensing Satellite (IRS-1A), LISS-II sensor with a resolution of $36.25m \times 36.25m$. Four bands of the image are considered, namely, blue band of wavelength 0.45-$0.52\mu m$, green band of wavelength 0.52-$0.59\mu m$, red band of wavelength 0.62-$0.68\mu m$, and near infrared band of wavelength 0.77-$0.86\mu m$.

Fig. 8a and Fig. 8b show the histogram equalized Kolkata image in the blue and green bands, respectively. Some characteristic regions in the image are the river *Hooghly* cutting across the middle of the image, several fisheries observed toward the lower-right portion, a township, *SaltLake*, to the upper-left-hand side of the fisheries. This township is bounded on the top by a canal. Two parallel lines observed towards the upper right-hand-side of the

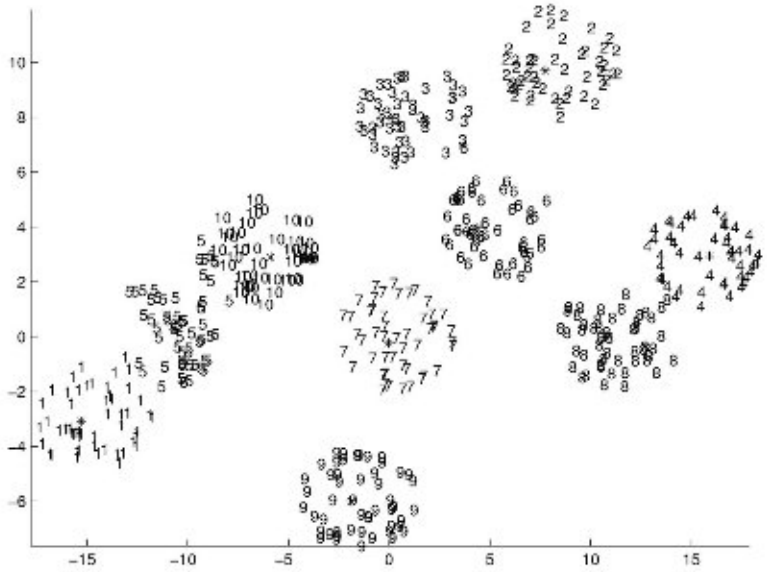

Fig. 6. *AD_10_2* clustered using SA-RJMCMC with the XB index when 10 clusters were obtained labeled 1-10.

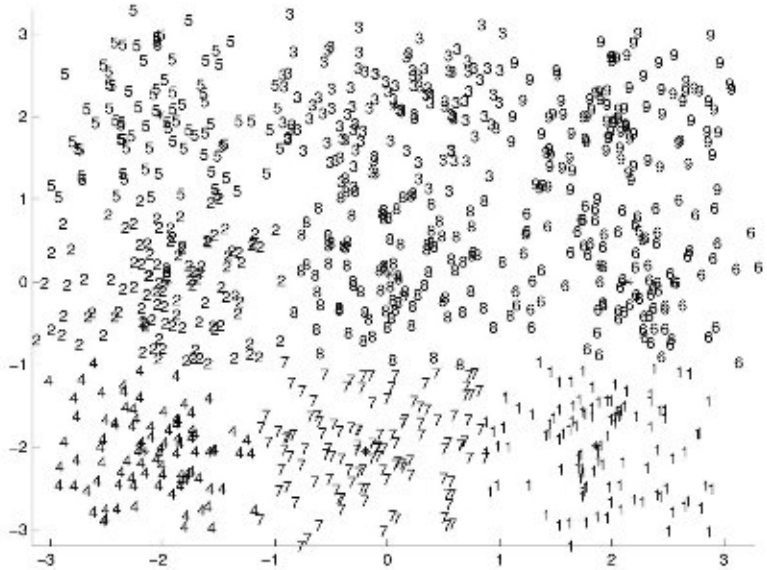

Fig. 7. *AD_9_2* clustered using SA-RJMCMC with the XB index when nine clusters were obtained labeled 1-9.

image correspond to the airstrips in the *Dumdum* airport. Other than these there are several water bodies, roads etc. in the image.

The image considered in this article is of size 512×512, having 262,144 pixels in four bands. Therefore, the data to be clustered, in essence, consists of 262,144 samples in four-dimensional space. Clustering the entire data set using the proposed method requires a large amount of time. However, it is observed that, in general, a large number of samples are often repeated (e.g., pixels that belong to the same region generally share the same spectral property). Clustering using samples that are repeated leads to a wastage of computational resources. In order to alleviate this problem, the approach adopted in this paper is as follows: Let there be n samples, $x_1, x_2, \ldots, x_n$. Of these, let there be m distinct samples, $x_{s1}, x_{s2}, \ldots, x_{sm}$, such that there are m_1 occurrences of x_{s1}, m_2 occurrences of x_{s2}, and so on. Note that $m_j \geq 1$, $j = 1, 2, \ldots, m$, and $\sum_{j=1}^{m} m_j = n$. Let us now rename the m distinct samples as $x_1, x_2, \ldots, x_m$ for the sake of convenience.

For a state of the SA-RJMCMC-based method which is comprised of c cluster centers, the XB index can now be computed using the following equation:

$$XB(U, V; X) = \frac{\sum_{i=1}^{c} \sum_{k=1}^{m} u_{ik}^2 m_k ||x_k - v_i||^2}{n(min_{i \neq j}\{||v_i - v_j||^2\})}. \quad (29)$$

It may be noted that u_{ik}, $i = 1, 2, \ldots, c$, and $k = 1, 2, \ldots, m$, the membership value of the kth sample to the i cluster depends only on the distances of the kth sample from the cluster centers, including the ith cluster center. Hence, samples that are the same will have the same degrees of

(a) (b)

Fig. 8. (a) Band 1 and (b) Band 2 of the IRS image of Kolkata with histogram equalization.

memberships to the different clusters. Equation (9) is now rewritten as

$$\phi(x_k, v_i) = u_{ik}^2 m_k ||x_k - v_i||^2 \tag{30}$$

and $\mathbf{D}$ is now an $m \times c$ matrix. In order to update the cluster centers using repeated patterns, (4) needs to be modified as follows:

$$v_i = \frac{\sum_{k=1}^{m} (u_{ik})^m m_k x_k}{\sum_{k=1}^{m} m_k (u_{ik})^m}, \quad i \leq i \leq c. \tag{31}$$

Using the above equations, the SA-RJMCMC-based clustering algorithm performs the computation on m points instead of n points, where, in general, $m << n$. This is found to greatly reduce the computation time when used in the case of satellite images. For the Kolkata image under consideration, the values of n and m are found to be equal to 262,144 and 20,415, respectively, which is a substantial reduction of about 92.2 percent.

Fig. 9 shows the image classified using the proposed method. It is automatically found to yield five clusters, which, according to our ground knowledge, correspond to two types of water bodies (pond water and turbid water), concrete, habitation, and open space. The river *Hooghly*, *Salt Lake* area, its bounding canal, the fisheries, etc., have all been properly differentiated. It is interesting to note that one bridge across *Hooghly* has been clearly distinguished from the large body of water underneath. A small part of the other bridge is also visible as a few dots.

In order to compare this result with the one obtained using a genetic algorithm-based scheme [34], Fig. 10 is provided. Here, four landcover types were found which

Fig. 9. Kolkata image clustered using the SA-RJMCMC algorithm when five clusters were obtained.

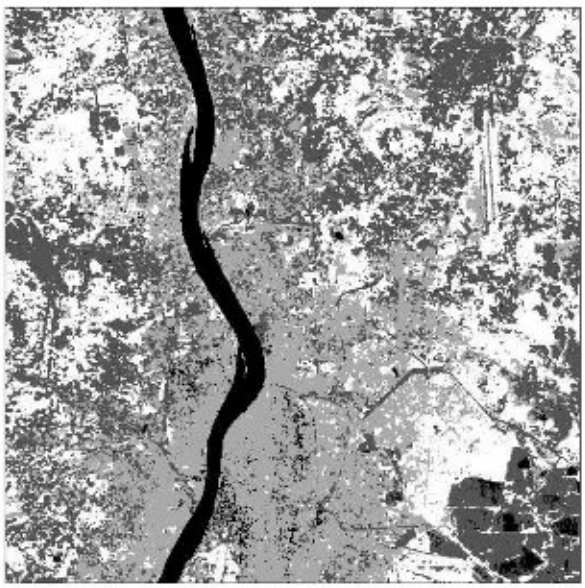

Fig. 10. Kolkata image clustered using the genetic fuzzy clustering scheme [34] when four clusters were obtained.

correspond to the two types of water bodies (pond water and turbid water), concrete, and open space. As can be seen from Fig. 10, none of the bridges over the river *Hooghly* could be differentiated, even partially, from the water body beneath. Detecting the bridges has traditionally been a difficult task for this image with both supervised and unsupervised approaches. Moreover, parts of the bodies of water corresponding to the fisheries (which appear on the bottom right of the figure) have erroneously gone into class turbid water (that corresponds to the river water). The value of the XB index provided by the genetic clustering scheme is 0.236703 [34], while that for the proposed method is 0.210183, which is smaller (and, hence, better) than the former.

5 Discussion and Conclusions

In this paper, an unsupervised fuzzy classification algorithm has been proposed that minimizes the Xie-Beni cluster validity index for automatically determining the number of clusters, as well as the proper partitioning of the data. Operators are defined which can make jumps between different number of clusters. Simulated annealing with an RJMCMC algorithm that uses the Hasting-Metropolis kernel for computing the acceptance probability of a new configuration is utilized as the underlying search technique. The sampler uses reversible moves, such as birth and death, split and merge, and a symmetrical perturbation of the cluster centers for sampling a candidate state given the current state, such that a jump between spaces of different dimensions is made possible. Experimental results demonstrating the effectiveness of the proposed algorithm has been provided for several artificial and real-life data sets.

The FCM algorithm is a standard and popular fuzzy clustering technique when the number of clusters is known a priori. This method is based on the optimization of $J_\mu(U,V)$ (see (2)). However, FCM solves the minimization problem through an iterative process to provide the local minima of $J_\mu(U,V)$. We have attempted to tackle both the problems of FCM by developing a solution strategy that does not require the a priori assumption of the number of clusters, c, while providing near-optimal solutions in practical cases. The performance of the proposed fuzzy clustering technique (that incorporates automatically evolving c) is compared with that of FCM for several artificial and real-life data sets. The number of clusters is varied over a given range in the FCM algorithm and the minimum value of the XB index and the corresponding number of clusters is noted for the different runs. It is found that the SA-RJMCMC technique is not only able to determine the appropriate value of c automatically, but also provides better values of the XB index in almost all the cases.

Results are also provided for a satellite image of Kolkata, where the proposed method is found to be able to distinguish the different landcover types reasonably well. Comparison with a recently proposed genetic clustering scheme also demonstrates this fact. Moreover, a technique for reducing the computation effort for data sets where the same pattern appears more than once is suggested. This is found to be particularly effective for the image under consideration.

There are several directions in which this work may be extended further. Firstly, a detailed comparative analysis can be carried out with other ways of computing the validity indices, e.g., the resampling scheme discussed in [35], as well as other similar search techniques (including SA) in order to justify the use of a particular index and an underlying search tool for a given problem. Next, a detailed time and sensitivity analysis of the developed technique can be performed and the use of other distance metrics may be investigated in the future. Finally, the algorithm developed here may be suitably modified and tailored so that it can be applied to data mining problems where the size of the data to be clustered may be huge.

Acknowledgments

The author gratefully acknowledges the reviewers for their valuable comments and suggestions and the Indian National Science Academy for providing partial support to carry out the work (project no. BS/YSP/36/886, 2002-2005).

References

[1] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, N.J.: Prentice Hall, 1988.
[2] R. Krishnapuram and J. Keller, "A Possibilistic Approach to Clustering," *IEEE Trans. Fuzzy Sets*, vol. 1, pp. 98-110, 1993.
[3] J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum, 1981.
[4] J.C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," *J. Cybernetics*, vol. 3, pp. 32-57, 1973.
[5] D.L. Davies and D.W. Bouldin, "A Cluster Separation Measure," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 1, pp. 224-227, 1979.
[6] R.E. Hammah and J.H. Curran, "Validity Measures for the Fuzzy Cluster Analysis of Orientations," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1467-1472, Dec. 2000.
[7] J.C. Bezdek, "Cluster Validity with Fuzzy Sets," *J. Cybernetics*, vol. 3, pp. 58-72, 1974.
[8] J.C. Bezdek, "Mathematical Models for Systematics and Taxonomy," *Proc. Eighth Int'l Conf. Numerical Taxonomy*, G. Estabrook, ed., pp. 143-166, 1975.
[9] M.P. Windham, "Cluster Validity for the Fuzzy C-Means Clustering Algorithm," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 4, pp. 357-363, 1982.
[10] A.M. Bensaid, L.O. Hall, J.C. Bezdek, L.P. Clarke, M.L. Silbiger, J.A. Arrington, and R.F. Murtagh, "Validity-Guided Reclustering with Applications to Image Segmentation," *IEEE Trans. Fuzzy Systems*, vol. 4, no. 2, pp. 112-123, 1996.
[11] X.L. Xie and G. Beni, "A Validity Measure for Fuzzy Clustering," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, pp. 841-847, July 1991.
[12] N.R. Pal and J.C. Bezdek, "On Cluster Validity for the Fuzzy C-Means Model," *IEEE Trans. Fuzzy Systems*, vol. 3, pp. 370-379, 1995.
[13] S. Kirkpatrik, C. Gelatt, and M. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, pp. 671-680, 1983.
[14] P.J.M. van Laarhoven and E.H.L. Aarts, *Simulated Annealing: Theory and Applications*. Kluwer Academic, 1987
[15] E. Forgy, "Cluster Analysis of Multivariate Data: Efficiency vs. Interpretability of Classifications," *Biometrics*, vol. 21, no. 3, p. 768, 1965.
[16] S. Bandyopadhyay, U. Maulik, and M.K. Pakhira, "Clustering Using Simulated Annealing with Probabilistic Redistribution," *Int'l J. Pattern Recognition and Artificial Intelligence*, vol. 15, no. 2, pp. 269-285, 2001.
[17] R.W. Klein and R.C. Dubes, "Experiments in Projection and Clustering by Simulated Annealing," *Pattern Recognition*, 1989.
[18] S.Z. Selim and K. Alsultan, "A Simulated Annealing Algorithm for the Clustering Problem," *Pattern Recognition*, vol. 24, pp. 1003-1008, 1991.
[19] A.V. Lukashin and R. Fuchs, "Analysis of Temporal Gene Expression Profiles: Clustering by Simulated Annealing and

Determining the Optimal Number of Clusters," *Bioinformatics*, vol. 17, no. 5, pp. 405-419, 2001.
[20] D.E. Brown and C.L. Huntley, "A Practical Application of Simulated Annealing to Clustering," *Pattern Recognition*, vol. 25, pp. 401-412, 1992.
[21] S. Bandyopadhyay and U. Maulik, "An Evolutionary Technique Based on K-Means Algorithm for Optimal Clustering in $\mathbb{R}^N$," *Information Sciences*, vol. 146, pp. 221-237, 2002.
[22] K. Rose, G. Gurewitz, and G.C. Fox, "A Deterministic Annealing Approach to Clustering," *Pattern Recognition Letters*, vol. 11, pp. 589-594, 1990.
[23] T. Hofmann and J. Buhmann, "Pairwise Data Clustering by Deterministic Annealing," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 19, no. 1, pp. 1-14, Jan. 1997.
[24] B. Bakker and T. Heskes, "Model Clustering by Deterministic Annealing," *Proc. European Symp. Artificial Neural Networks*, pp. 87-92, 1999.
[25] L. Hermes and J.M. Buhmann, "Semi-Supervised Image Segmentation by Parametric Distributional Clustering," *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 229-245, 2003.
[26] J.M. Garibaldi and E.C. Ifeachor, "Application of Simulated Annealing Fuzzy Model Tuning to Umbilical Cord Acid-Base Interpretation," *IEEE Trans. Fuzzy Systems*, vol. 7, no. 1, pp. 72-84, 1999.
[27] O. Cordon, F. Herrera, and P. Villar, "Analysis and Guidelines to Obtain a Good Uniform Fuzzy Partition Granularity for Fuzzy Rule-Based Systems Using Simulated Annealing," *Int'l J. Approximate Reasoning*, vol. 25, no. 3 pp. 187-216, 2000.
[28] P.J. Green, "Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination," *Biometrika*, vol. 82, pp. 711-732, 1995.
[29] W.K. Hastings, "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, vol. 57, pp. 97-109, 1970.
[30] S.P. Wilson and J. Zerubia, "Segmentation of Textured Satellite and Aerial Images by Bayesian Inference and Markov Fields," Technical Report 4336, INRIA, Sophia-Antipolis, France, 2001.
[31] A. Andrieu, J.F.G. de Freitas, and A. Doucet, "Robust F··" Bayesian Learning for Radial Basis Networks," *Neural Computation*, vol. 13, pp. 2359-2407, 2001.
[32] H. Akaike, "A New Look at Statistical Model Identification," *IEEE Trans. Automatic Control*, vol. 19, pp. 716-723, 1974.
[33] S. Mitra and S.K. Pal, "Fuzzy Multi-Layer Perceptron, Inferencing and Rule Generation," *IEEE Trans. Neural Networks*, vol. 6, pp. 51-63, 1995.
[34] U. Maulik and S. Bandyopadhyay, "Fuzzy Partitioning Using a Real-Coded Variable-Length Genetic Algorithm for Pixel Classification," *IEEE Trans. Geoscience and Remote Sensing*, vol. 41, no. 5, pp. 1075-1081, 2003.
[35] E. Levine and E. Domany, "Resampling Method for Unsupervised Estimation of Cluster Validity," *Neural Computation*, vol. 13, no. 11, pp. 2573-2593, 2001.

Sanghamitra Bandyopadhyay received BS degrees in physics and computer science in 1988 and 1992, respectively. Subsequently, she received the MS degree in computer science from the Indian Institute of Technology (IIT), Kharagpur, in 1994 and the PhD degree in computer science from the Indian Statistical Institute, Calcutta, in 1998. Currently, she is a faculty member at the Indian Statistical Institute, Calcutta, India. Dr. Bandyopadhyay is the first recipient of the Dr. Shanker Dayal Sharma Gold Medal and Institute Silver Medal for being judged the best all around post-graduate performer at IIT, Kharagpur, in 1994. She worked at the Los Alamos National Laboratory, Los Alamos, New Mexico, in 1997 as a graduate research assistant, at the University of New South Wales, Sydney, Australia, in 1999, as a postdoctoral fellow in the Department of Computer Science and Engineering, the University of Texas at Arlington, in 2001 as a faculty member and researcher, and in the Department of Computer Science and Engineering, University of Maryland Baltimore County, in 2004 as a visiting research faculty member. Dr. Bandyopadhyay received the Indian National Science Academy (INSA) and the Indian Science Congress Association (ISCA) Young Scientist Awards in 2000, as well as the Indian National Academy of Engineering (INAE) Young Engineers' Award in 2002. She has published more than 75 articles in international journals, conference and workshop proceedings, edited books, and journal special issues and served on the committees of several conferences. She is serving as a program cochair of the First International Conference on Pattern Recognition and Machine Intelligence, 2005, to be held in Kolkata, India, and as tutorial cochair, World Congress on Lateral Computing, 2004, to be held in Bangalore, India. She is on the editorial board of the *International Journal on Computational Intelligence*. Her research interests include evolutionary and soft computation, pattern recognition, data mining, bioinformatics, and parallel and distributed systems.