A Niching Memetic Algorithm for Simultaneous Clustering and Feature Selection

Weiguo Sheng, Xiaohui Liu, and Michael Fairhurst

Abstract—Clustering is inherently a difficult task and is made even more difficult when the selection of relevant features is also an issue. In this paper, we propose an approach for simultaneous clustering and feature selection using a niching memetic algorithm. Our approach (which we call NMA_CFS) makes feature selection an integral part of the global clustering search procedure and attempts to overcome the problem of identifying less promising locally optimal solutions in both clustering and feature selection, without making any a priori assumption about the number of clusters. Within the NMA_CFS procedure, a variable composite representation is devised to encode both feature selection and cluster centers with different numbers of clusters. Furthermore, local search operations are introduced to refine feature selection and cluster centers encoded in the chromosomes. Finally, a niching method is integrated to preserve the population diversity and prevent premature convergence. In an experimental evaluation, we demonstrate the effectiveness of the proposed approach by using both synthetic and real data.

Index Terms—Clustering, feature selection, genetic algorithm, local search, memetic algorithm, niching method.

1 Introduction

Clustering or cluster analysis is an important but challenging task in unsupervised learning. The essence of the clustering problem is to partition a set of objects into an a priori unknown number of clusters while minimizing the within-cluster variability and maximizing the between-cluster variability. Data clustering is a common technique for statistical data analysis and has been used in a variety of engineering and scientific disciplines such as biology (e.g., to study genome data [3], [48], [53]) and computer vision (e.g., to segment images [15], [25], [47]).

Many clustering algorithms have been proposed in the literature. Generally, they can be divided into two main categories, namely, hierarchical and partitional [24]. Hierarchical clustering constructs a hierarchy of partitionings, represented as a *dendrogram* in which each partitioning is nested within the partitioning at the next level in the hierarchy. In hierarchical clustering, problems due to initialization and local optima do not arise. However, this approach considers only local neighbors in each step and ignores the global shape and size of clusters. Moreover, hierarchical clustering is static; that is, data objects committed to a given cluster in the early stages cannot move to a different cluster. In the work reported here, we concentrate on partitional clustering, which is dynamic and considers the global shape and size of clusters.

In partitional clustering, each data object is represented by a vector of features. Most partitional algorithms assume all features to be equally important for clustering in the sense that they do not distinguish among different features, but this approach to clustering can create significant limitations in an unsupervised learning context. The problem is that not all features are equally important; indeed, some of the features may be redundant, some may be irrelevant, and some can even mislead the clustering process. This is one of the reasons that many clustering algorithms do not perform well in the face of high-dimensional data, and the task of selecting the best feature subset, the process known as feature selection, is therefore important. In addition, feature selection may lead to more economical clustering algorithms (in terms of both storage and computational effort) and contribute to the interpretability of the models generated. Generally, for a data set of nontrivial size, finding the optimal clustering solution is a challenging problem [17] and becomes even more challenging if an appropriate feature set also needs to be selected.

One way of approaching this challenge is to use stochastic optimization schemes, prominent among which is an approach based on genetic algorithms (GAs). The GA, first developed by Holland [23], is biologically inspired and embodies many mechanisms mimicking natural evolution. It has a great deal of potential in scientific and engineering optimization or search problems. Recently, hybrid methods [2], [34], [52], which incorporate local searches with traditional GAs, have been proposed and applied successfully to solve a wide variety of optimization problems. These studies show that pure GAs are not well suited to fine-tuning structures in complex search spaces and that hybridization with other techniques can greatly improve their efficiency. GAs that have been hybridized with local searches are also known as memetic algorithms (MAs) [35], [36]. Since we are concerned here with a GA where local searches play a significant role, this term will be adopted in this paper.

- *W. Sheng annd M. Fairhurst are with the Department of Electronics, University of Kent, Canterbury, Kent, CT2 7NT, United Kingdom. E-mail: {w.sheng, m.c.fairhurst}@kent.ac.uk.*
- *X. Liu is with the School of Information Systems, Computing, and Mathematics, Brunel University, Uxbridge, Middlesex, UB8 3PH, United Kingdom. E-mail: xiaohui.liu@brunel.ac.uk.*

Manuscript received 24 Nov. 2005; revised 1 June 2007; accepted 17 Jan. 2008; published online 25 Jan. 2008.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0522-1105.
Digital Object Identifier no. 10.1109/TKDE.2008.33.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

Traditional GAs and MAs are generally suitable for locating the optimal solution of an optimization problem with a small number of local optima. Complex problems such as clustering, however, often involve a significant number of locally optimal solutions. In such cases, traditional GAs and MAs cannot maintain controlled competitions among the individual solutions and can cause the population to converge prematurely [44]. To improve the situation, various methods [9], [19], [32], [41], [46] (usually called niching methods [44]) have been proposed. The research reported shows that one of the key elements in finding the optimal solution to a difficult problem with a GA approach is to preserve the population diversity during the search, since this permits the GA to investigate many peaks in parallel and helps in preventing it from being trapped in local optima.

In this paper, we first suggest a unified criterion for simultaneous clustering and feature selection based on a well-known scatter separability index. A GA-based evolutionary procedure is then proposed to optimize the criterion. In order to allow simultaneous clustering and feature selection without the number of clusters being known a priori, a composite representation is devised to encode both feature selection and cluster centers with a variable number of clusters. As a consequence, the crossover and mutation operators are suitably modified to tackle the concept of composite chromosomes with variable lengths. Additionally, we hybridize the proposed procedure with local search operations, which are introduced to refine the feature selection and cluster centers, respectively. These local searches move solutions toward local optima and allow a significant improvement in the computational efficiency. Finally, a niching method is integrated with the resulting hybrid GA to preserve the population diversity and prevent premature convergence. To evaluate the proposed algorithm, we have conducted a series of experiments on both synthetic and real data and compared it with related work. The results show that our algorithm is generally able to select relevant features and locate appropriate clustering with the correct number of clusters and that it outperforms other methods implemented for comparison.

The remainder of this paper is organized as follows: After reviewing related work in Section 2, we suggest a unified criterion for simultaneous clustering and feature selection in Section 3. Then, in Section 4, we present a niching MA for optimizing the criterion. Section 5 describes six data sets employed in the experimental evaluation, and this is followed by a discussion of the parameter settings of the algorithm. In the experiments reported in Section 6, the performance of the proposed algorithm is assessed. We complete this paper with some concluding remarks and suggestions for future directions for this work in Section 7.

2 Related Work

GAs are naturally applicable to problems with exponential search spaces and have consequently been a significant source of interest for clustering [21], [30], [33], [51]. For example, Hall et al. [21] and Maulik and Bandyopadhyay [33] proposed the use of traditional GAs for partitional clustering. These methods can be very expensive and susceptible to becoming trapped in locally optimal solutions for clustering large data sets. Krishana and Murty [30] and Tsai et al. [51] introduced hybrid GAs by incorporating clustering-specified local searches into traditional GAs. In contrast to the methods proposed in [21] and [33], clustering based on hybrid GAs can be more efficient, but these techniques can still, however, suffer from premature convergence. Furthermore, all of the above methods may exhibit limited performance, since they perform clustering on all features without selection. GAs have also been proposed for feature selection [42], [54]. However, they are usually developed in the supervised learning context, where class labels of the data are available, and the main purpose is to reduce the number of features used in classification while maintaining acceptable classification accuracies.

The second (and related) theme of this paper is feature selection for clustering, and feature selection research has a long history, as reported in the literature. Feature selection in the context of supervised learning [1], [6], [16], [28], adopts methods that are usually divided into two classes [5], [28]—filters and wrappers—based on whether or not feature selection is implemented independently of the learning algorithm. To maintain the filter/wrapper distinction used in supervised feature selection, we also classify feature selection methods for clustering into these two categories based on whether or not the process is carried out independently of the clustering algorithm. The filters in clustering basically preselect the features and then apply a clustering algorithm to the selected feature subset. The principle is that any feature carrying little or no additional information beyond that subsumed by the remaining features is redundant and should be eliminated. Various measures such as correlation coefficients [20], statistical redundancy [22], and linear dependence [7], [50] have been used in this context. Recently, the Relief Algorithm [26] and its extensions [29], which identify statistically relevant features, have also been reported.

The wrappers in clustering, on the other hand, incorporate the clustering algorithm in the feature subset searching. Methods used in this category involve a clustering algorithm (e.g., EM [10] and K Means [31]) running on a feature subset, with the feature subset being assessed by the clustering performance, as quantified by some appropriate index. These include a sequential unsupervised feature selection algorithm [8], feature selection based on the expectation-maximization (EM) [12], [13] and maximum entropy [4]. The wrappers can be superior in performance when compared with the filters, which ignore the properties of the clustering task at hand [28]. They can be used to identify the clustering solutions as well. However, the performance of both clustering and feature selection is dependent on the incorporated clustering algorithms, which may be sensitive to their initializations and suffer from locally optimal solutions. Furthermore, when the number of clusters is unknown beforehand, the wrappers have to be applied in such a way as to search through a range of possible cluster numbers. In contrast, our proposed method makes feature selection an integral part of the global clustering search procedure and attempts to identify high-quality solutions for clustering and feature selection while automatically evolving the correct number of clusters.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

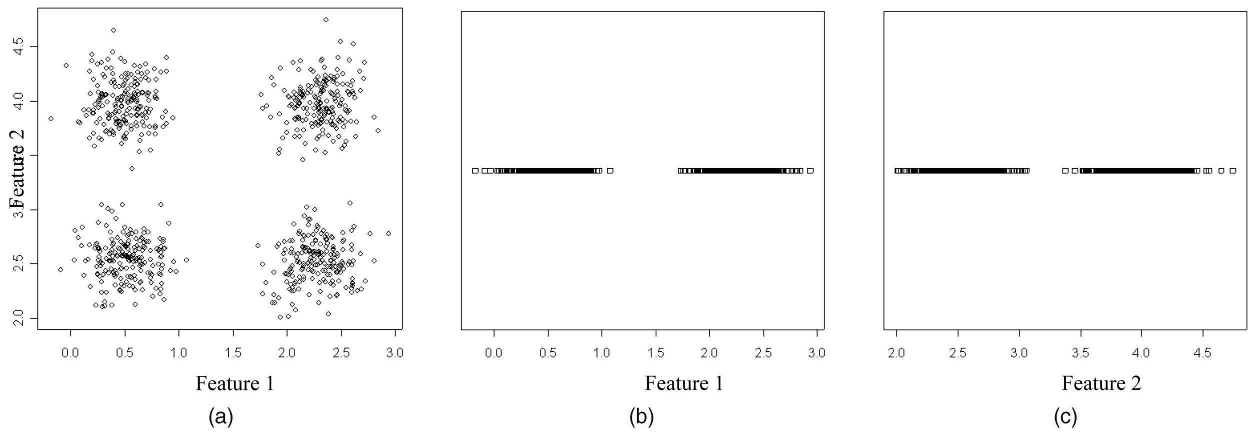


Fig. 1. The number of clusters varies with the dimension.

3 Unified Criterion

A number of implementation criteria have been proposed in the clustering literature. However, they are usually designed for either clustering or feature selection alone. In this section, we suggest a unified criterion for simultaneous clustering and feature selection by investigating a well-known scatter separability index.

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of n data objects in a D-dimensional feature space. A partitioning of the set X is defined as a set of nonempty clusters of X such that data object x_i in X is in exactly one of these clusters. The partitioning is typically achieved by optimizing a specified criterion. In the literature, various criteria have been reported [11], [24], [49]. Some popular criteria are based on the within-cluster and between-cluster scatter matrices. One criterion is the $trace(S_w^{-1}S_b)$, in which the within-cluster variation S_w is defined as

$$S_w = \frac{1}{n}\sum_{j=1}^{k}\sum_{i=1}^{n} z_{ji}(x_i - m_j)(x_i - m_j)^T. \qquad (1)$$

S_w measures how scattered the objects are from their cluster means (compactness), where $z_{ji} = 1$ if $x_i \in \text{cluster } j$; otherwise, this is 0. $m_j = \frac{1}{n_j}\sum_{i=1}^{n} z_{ji}x_i$ is the mean of cluster j, and $n_j = \sum_{i=1}^{n} z_{ji}$ is the number of objects in cluster j. The between-cluster variation S_b is defined as

$$S_b = \sum_{j=1}^{k}\frac{n_j}{n}(m_j - m)(m_j - m)^T \qquad (2)$$

and measures how scattered the cluster means are from the sample mean (separability). Here, $m = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the sample mean. In the $trace(S_w^{-1}S_b)$ criterion, the between-cluster variation S_b is normalized by the within-cluster variation S_w. Hence, large values of the criterion correspond to high-quality clustering solutions. This criterion is invariant under any nonsingular linear transformation and has been widely used for clustering, where issues such as feature selection and the number of clusters do not arise. Here, we investigate these issues and suggest a unified criterion based on the above approach.

Issue 1: feature selection. Typically, for a particular application, as much information as possible is gathered without considering the significance of each feature to the underlying clusters. Thus, it is essential to remove irrelevant or redundant features while performing clustering. Let us denote the total set of collected features as $U = \{1, 2, \ldots, D\}$. Feature selection for clustering can then be defined as the problem of selecting d features from U, which can optimize a specified clustering criterion.

As such a criterion, the $trace(S_w^{-1}S_b)$ is biased toward higher dimensions. The value of this criterion monotonically increases as features are added, assuming equal clustering assignments. This is not desirable, because we would like to retain the minimum number of features consistent with an appropriate level of performance. In order to compare feature subsets of different dimensionalities, we normalize the $trace(S_w^{-1}S_b)$ by a penalty term $(D-d)/(D-1)$. The resulting criterion, denoted by J_1, can be written as

$$J_1 = trace(S_w^{-1}S_b) * (D-d)/(D-1). \qquad (3)$$

Since $trace(S_w^{-1}S_b)$ remains relatively unchanged for any addition of features with little discrimination [8], by minimizing J_1, we aim at attaining a feature subset with good discrimination.

Issue 2: number of clusters. In most real-world situations, the number of clusters k in a data set is usually unknown beforehand. Furthermore, when searching for the best feature subset, we encounter a new situation where the value of k depends on the feature subset. For example, as illustrated in Fig. 1, in two dimensions, the data set has four clusters (as shown in Fig. 1a), whereas in one dimension, it has only two clusters (as shown in Figs. 1b and 1c). Hence, using a fixed number of clusters for all feature subsets may not model the data in each respective subset correctly. Thus, we need to search for the correct number of clusters while performing clustering with each candidate feature subset. For this purpose, we add another penalty term $(k_{\max} - k)/(k-1)$ to the criterion J_1. Our new criterion, denoted by J_2, becomes

$$J_2 = trace(S_w^{-1}S_b) * (D-d)/(D-1) * (k_{\max} - k)/(k-1). \qquad (4)$$

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

This penalty term is needed, because the criterion J_1 is biased toward increasing the number of clusters. For example, S_w is equal to 0 in the extreme case when the number of clusters is equal to the number of data objects. It should be noted, however, that both penalty terms in the J_2 formulation are determined empirically and there may be other more effective penalty terms that could result in better performance.

4 Niching Memetic Algorithm

In this section, we propose a niching MA for simultaneous clustering and feature selection (NMA_CFS) by optimizing the unified criterion J_2. The proposed algorithm works with variable composite chromosomes, which are used to represent solutions. The operation of the algorithm consists of using a niching selection method for selecting pairing parents for reproduction, performing different genetic operators on different parts (i.e., feature selection vector and cluster centers) of the paired parents, applying local search operations (i.e., feature *add* and *remove* procedures and one step of K Means) to each offspring, and carrying out a niching competition replacement. The evolution is terminated when the fitness value of the best solution in the population has not changed for g generations. The output of the algorithm is the best solution encountered during the evolution. The flow of the algorithm (Algorithm 1) is given as follows:

Algorithm 1: NMA_CFS.

```
Step 1. Randomly initialize $p$ sets of solutions, which encode
        both feature selection and cluster centers with
        different numbers of clusters, by using a variable
        composite representation (see Section 4.1).
Step 2. Calculate $J_2$ according to (4) for each solution in the
        initial population and set its fitness value as $f = J_2$.
Step 3. Repeat the following steps until the stopping
        criterion is met:
    a) Select pairing parents based on a niching selection
       method (see Section 4.4). This procedure is repeated
       until $p/2$ parent pairs are selected.
    b) Generate intermediate offspring by applying
       different genetic operators (see Section 4.2) on the
       different parts (i.e., feature selection vector and
       cluster centers) of the paired parents.
    c) Apply feature add and remove procedures
       (see Section 4.3) to the offspring.
    d) Run one step of K Means (see Section 4.3) on the
       offspring.
    e) Pair the offspring with the most similar solution
       found during a restricted competition replacement
       (see Section 4.4).
    f) Calculate $J_2$ according to (4) for each of the offspring.
       If the fitness of the offspring is better than its paired
       solution, then the latter is replaced.
Step 4. Provide the feature subset and cluster centers of the
        solution from the terminal population with the best
        fitness.
```

In the following sections, we describe in more detail how the solutions are initially created and how they evolve during the optimization process, and we briefly analyze the time complexity of the proposed algorithm.

4.1 Representation and Initialization

In the NMA_CFS procedure, we devise a variable composite chromosome, which can encode both feature selection and cluster centers with a variable number of clusters. The feature selection vector in the chromosome is a string with D binary digits (D is the total number of available features in the data to be clustered), and each binary digit represents an individual feature, with values 1 and 0 denoting *selected* and *ignored*, respectively. The cluster centers in the chromosome consist of $D \times k_i$ real numbers, where k_i is the number of clusters. The first D positions represent the D dimensions of the first cluster center, the next D positions represent those of the second cluster center, and so on. For example, in five-dimensional data, the chromosome

$$< 11001 \quad 0.5_1 0.1_1 0.7_0 0.5_0 0.6_1$$
$$0.2_1 0.9_1 0.8_0 0.7_0 0.3_1 \quad 0.8_1 0.9_1 0.5_0 0.6_0 0.2_1 >$$

encodes centers of three clusters (i.e., (0.5, 0.1, 0.6), (0.2, 0.9, 0.3), and (0.8, 0.9, 0.2)), with the first, second, and fifth features being selected. It should be noted that only values of the selected features (values with subscript "1") are used to form the cluster centers and the others (values with subscript "0") are ignored.

Each solution in the population is constructed using the variable composite chromosome. The values are initialized by random assignment of binary digits and real numbers to the feature selection vector and the k_i cluster centers, respectively. The initial values of the cluster centers are constrained to be in the range (determined from the data set) of the feature to which they are assigned but are otherwise random. The initial number of clusters k_i is calculated according to $RandInt(2, k_{max})$. Here, $RandInt()$ is a function returning a natural number in the range from 2 to k_{max} (inclusive), and k_{max} is the upper bound of the number of clusters and is taken to be $\sqrt{n}$ (n is the number of objects in the data set to be clustered), which is a rule of thumb used by many investigators in the clustering literature [38]. The number of clusters for the solutions in the population will therefore range from 2 to k_{max}.

4.2 Crossover and Mutation

In the composite representation, feature selection and cluster centers are encoded in a single solution. Accordingly, we have applied different genetic operators, which are sensitive to the corresponding context, on feature selection vector and cluster center parts of the paired parents. For the feature selection vector part, the m-point crossover and flip mutation [18] are applied. The m-point crossover, which is performed on each set of paired parents, chooses m cutting points at random and alternately copies each segment from the two parents. For example, given a parent pair

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

$$Parent_1 : < 1|10|0|1 \quad 0.50.10.70.50.6 \quad 0.20.90.80.70.3 \quad 0.80.90.50.60.2 >,$$
$$Parent_2 : < 1|01|1|0 \quad 0.40.20.80.40.5 \quad 0.90.70.30.50.1 \quad 0.10.80.60.90.2 \quad 0.40.50.30.40.1 >,$$

suppose that three cutting points are chosen at positions 1, 3, and 4 (denoted by "|") in the feature selection vector. After the m-point crossover, the two intermediate offspring generated would be

$$Offspring_1 : < 10100 \quad 0.50.10.70.50.6 \quad 0.20.90.80.70.3 \quad 0.80.90.50.60.2 >,$$
$$Offspring_2 : < 11011 \quad 0.40.20.80.40.5 \quad 0.10.80.60.90.2 \quad 0.90.70.30.50.1 \quad 0.40.50.30.40.1 > .$$

After crossover, each bit of the offspring is considered for mutation. Mutation consists of flipping the value of the chosen bit from 1 to 0, or vice versa. Both crossover and mutation operations are likely to generate an offspring, with no features being selected. When such an offspring emerges, we repeat the operations until a proper offspring is produced or until a limit on the number of trials is reached.

For the cluster center part, we use a crossover operation analogous to the traditional two-point crossover [18]. During crossover, the cluster centers are considered to be indivisible (i.e., the crossover points can only lie in between two clusters' centers). For this purpose, the crossover operation is defined as follows: Let paired parents P_1 and P_2 encode k_1 and k_2 cluster centers ($k_1 \leq k_2$), respectively. Then, x_1 and x_2, the crossover points in P_1, are generated according to $RandInt(0, k_1 - 1)$. If x_1 is greater than x_2, then swap the value of x_1 and x_2 to make sure that $x_2 > x_1$. The cross points x_3 and x_4 in P_2 are then generated as $x_3 = RandInt(0, k_2 - |x_2 - x_1| - 1)$ and $x_4 = x_3 + |x_2 - x_1|$, where $|x_2 - x_1|$ is the length of segment between cross points of x_2 and x_1. After that, the segment information between x_1 and x_2 in P_1 exchanges with the segment information between x_3 and x_4 in P_2. Continuing with the above example, given the two intermediate offspring after the m-point crossover

$$Offspring_1 : < 10100 \quad 0.50.10.70.50.6 \mid 0.20.90.80.70.3 \mid 0.80.90.50.60.2 >,$$
$$Offspring_2 : < 11011 \quad 0.40.20.80.40.5 \quad 0.90.70.30.50.1 \mid 0.10.80.60.90.2 \mid 0.40.50.30.40.1 >,$$

suppose that the crossover points x_1, x_2, x_3, and x_4 are generated at positions 1, 2, 2, and 3, respectively (denoted by "|"). After the crossover, the two offspring would become

$$Offspring_1 : < 10100 \quad 0.50.10.70.50.6 \quad 0.10.80.60.90.2 \quad 0.80.90.50.60.2 >,$$
$$Offspring_2 : < 11011 \quad 0.40.20.80.40.5 \quad 0.90.70.30.50.1 \quad 0.20.90.80.70.3 \quad 0.40.50.30.40.1 > .$$

It can be seen that according to the above rules, the number of clusters of the offspring will be equal to either k_1 or k_2. The crossover is performed on each set of paired parents.

After crossover, a low probability of Gaussian mutation is applied on the offspring. Gaussian mutation adds a unit Gaussian distributed random value to the chosen feature. The new feature value is clipped if it falls outside the lower or upper bounds of that feature.

4.3 Local Searches

Pure GAs are not well suited to fine-tuning solutions that are close to optima [19], and this results in their having a long runtime. To improve the time efficiency, incorporation of local searches into the regeneration step of GAs, creating the so-called MAs, is essential if competitive GAs are to be used [2]. In this section, we present several local search operations to effectively design an MA for simultaneous clustering and feature selection.

Feature add and remove operations. Sequential forward selection (SFS) and sequential backward selection (SBS) [14] are two classical heuristic feature selection algorithms developed for supervised learning. SFS starts with an empty set of features, and at each iteration, the algorithm tentatively adds each available feature and selects the feature that results in the highest estimated performance. The search terminates when the accuracy of the current subset cannot be improved by adding any other feature. SBS works in an analogous way but starts from the full set of available features and tentatively deletes each feature not deleted previously. SFS and SBS are simple and fast. However, they are prone to being trapped in locally optimal solutions.

Here, we introduce two basic operations—*add* (based on the SFS) and *remove* (based on the SBS)—as noted in the following and incorporate them into the GA to fine-tune the feature selection encoded in the solution for clustering:

- *Add.* Choose a feature from the unselected feature subset that, when combined with the currently selected features, yields the largest value of the criterion J_1 and changes its status to "selected."
- *Remove.* Choose a feature from the selected feature subset that makes the least contribution to the criterion J_1 and changes its status to "ignored."

These operations generate local improvements by adding the most significant feature or removing the least significant feature and aim at speeding up the search for the best feature subset. Each of the operations is applied once (the *add* operation followed by the *remove* operation) to all new offspring after the crossover and mutation operations.

K Means operation. K Means [31] is an iterative scheme attempting to minimize the within-cluster sum of squares errors (SSE):

$$SSE = \sum_{i=1}^{n} \sum_{j=1}^{k} z_{ji} \|x_i - m_j\|^2. \quad (5)$$

Starting from an initial distribution of cluster centers in the data space, each data object is assigned to a cluster with the closest center, after which each center itself is updated as the center of data objects belonging to that particular cluster. This procedure is repeated until there is no reassignment of any data object from one cluster to another or the SSE value ceases to decrease significantly. This iterative scheme is known to converge fast. However, it depends highly on the initialization of cluster centers.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

In order to improve the computational efficiency, one step of K Means is applied to the cluster centers encoded in all the new offspring during each generation after the feature *add* and *remove* operations. This is done by assigning each data object to one of the clusters with the nearest center encoded in the solution. After that, the cluster centers encoded in the solution are replaced by the means of the respective clusters.

4.4 Niching Method

One of the key elements in overcoming less promising locally optimal solutions of a difficult optimization problem with a GA approach is to preserve the population diversity during the search [44]. In this section, we introduce a modification of the niching method proposed in [46] and integrate it into our GA to preserve the population diversity during the simultaneous search for clustering and feature selection.

The niching method presented in [46] was designed for clustering where no feature selection is required and the number of clusters is known beforehand. In this method, a niching selection with a restricted competition replacement was developed to encourage mating among similar solutions while allowing for some competitions among dissimilar solutions. During the niching selection, one parent p_1 is selected randomly from the population, and its mate is selected from a group of solutions called the *selection group*, picked randomly from the population. The one most similar (determined by the euclidean distance based on a *phenotypic metric*) to p_1 is chosen as its mate p_2. During the restricted competition replacement, each offspring is compared with a group of solutions called the *replacement group*, picked randomly from the population, and is then paired with the most similar one. If the fitness of the offspring is better than its paired solution, then the latter is replaced.

With appropriate sizes of the selection and replacement groups, this method can maintain the population diversity with respect to the cluster centers with a fixed number of clusters encoded in the solutions. However, for the problem considered here, it is more important to preserve the population diversity with respect to the number of clusters, since the solutions with different numbers of clusters have rather different feature selection and cluster centers. For this purpose, we modify the niching selection to encourage mating among solutions with similar numbers of clusters and extend the restricted competition replacement to encourage replacement among solutions with the same number of clusters while allowing for some competitions among the solutions with different numbers of clusters. The modified niching method is implemented as follows: During the niching selection, one parent p_1 is still randomly selected from the population. Its mate p_2 is now chosen from the selection group with the most similar number of clusters as for p_1. If this results in a group with more than one candidate solution, the similarity of feature selection and cluster centers is further used to select the most similar one. During the restricted competition replacement, we now compare the offspring with each solution that has the same number of clusters as the offspring in the competition group, and we pair it with the one with the most similar feature selection and cluster centers if this exists; otherwise, we pair it with a solution with the lowest fitness. If the fitness of the offspring is better than its paired solution, then the latter is replaced.

Crossover among solutions with a large difference in cluster numbers often produces low-performance offspring. The modified niching selection tries to promote mating among solutions with similar numbers of clusters. When the size of the selection group is equal to one, it is basically a random selection. As the size increases, there is a greater possibility of selecting parent pairs with the same number of clusters. However, the size should be small enough to allow mating among solutions with different numbers of clusters. The extended restricted competition replacement is mostly used to balance competitions during replacement among solutions with different numbers of clusters. A large replacement group size will restrict the replacement among solutions with the same number of clusters. Decreasing the size will promote more competitions among the solutions with different numbers of clusters. An appropriate value should be set to allow both thorough exploration of the search space with the same number of clusters and competitions among solutions with different numbers of clusters. By measuring the similarity of solutions based on their feature selection and cluster centers during replacement, we are also attempting to preserve the diversity among the solutions of the same number of clusters with respect to feature selection and cluster centers.

4.5 Complexity

The major computational load during each generation of the proposed algorithm is in the feature *add* and *remove* procedures, one step of the K Means operator, and the fitness evaluation. The feature *add* procedure takes $O(nk_{max}D^2)$ time. Similarly, the feature *remove* procedure has $O(nk_{max}D^2)$ time complexity in the worst case. The one-step K Means operator and the fitness evaluation of a given solution take $O(nk_{max}D)$ and $O(nD)$ time, respectively. Therefore, the overall complexity of the proposed algorithm is $O(nk_{max}D^2pg)$, where p is the population size, and g is the number of generations.

5 Data Sets and Implementation Parameters

This section provides a description of the data sets used for experimentation and the parameter settings of the proposed algorithm. Several data sets, both real and synthetic, have been used in our experiments. The synthetic data sets were generated with different numbers of clusters, and each comprises 2,000 data objects. These data sets contain both "relevant" and "irrelevant" features, where "relevant" means that we create the clusters using these features. "Irrelevant" features are generated as Gaussian normal random variables. The first data set *Synthetic*3_7 (as shown in Fig. 2a) consists of three equiprobable Gaussian clusters, with means $\mu_1 = (0.40, 0.20)$, $\mu_2 = (0.20, 0.20)$, and $\mu_3 = (0.30, 0.32)$, respectively. Five irrelevant features are added, yielding a set of seven-dimensional data. The second data set *Synthetic*4_10 (as shown in Fig. 2b) consists of four clusters, with means at (0.2, 0.2), (0.2, 0.35), (0.3, 0.45), and (0.3, 0.3), respectively. We add eight Gaussian normal random irrelevant features. There is some overlap among the four clusters, and the eight additional irrelevant features increase the difficulty of the problem.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

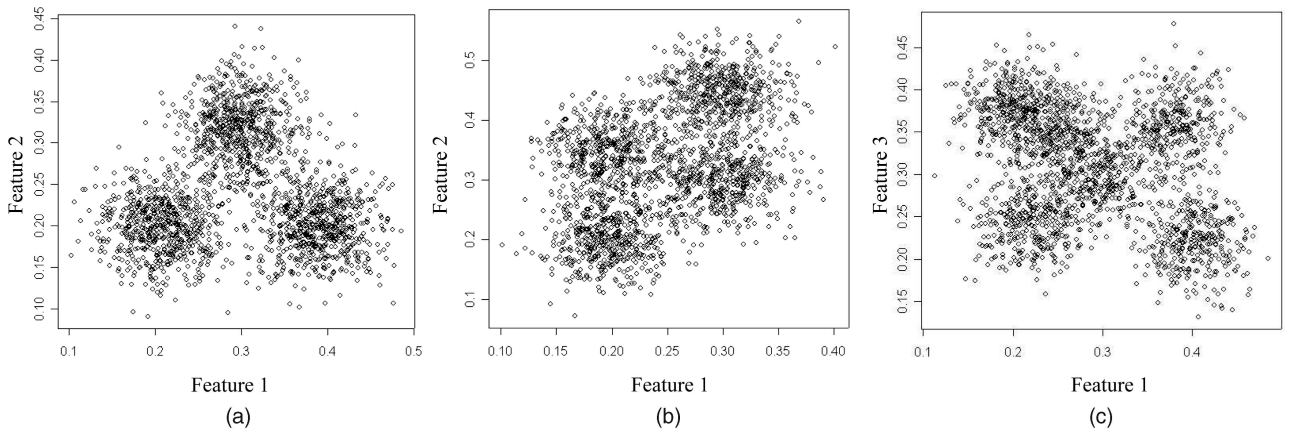


Fig. 2. (a) *Synthetic3_7*, (b) *Synthetic4_10*, and (c) *Synthetic6_20* data sets by projecting the objects onto two of the relevant features.

The third data set *Synthetic*6_20 (as shown in Fig. 2c) has 20 features, only five of which are relevant (features {1, 3, 6, 9, 15}). There are six Gaussian clusters across the five relevant features. The means are sampled from a uniform distribution on the interval [0.1, 0.5]. In each of the data sets, the clusters are generated with equal sizes. Note that the synthetic data sets are generated in such a way that they present different degrees of difficulty for simultaneous clustering and feature selection.

Three real data sets are also considered, and these are Iris, the Wisconsin diagnostic breast cancer (WDBC), and image segmentation. They are taken from the UCI Machine Learning Repository [37]. The Iris data consists of 150 objects belonging to three species of Iris, with 50 objects in each species. This data is described by four real-valued features. The WDBC data set has 576 data objects, with 30 features extracted from cell nuclei presented in an image. The image segmentation data set contains 2,310 objects with 19 features from seven clusters, where each object consists of features extracted from a 3×3 region taken from seven types of outdoor images: brickface, sky, foliage, cement, window, path, and grass. Normalization to zero mean and unit variance is performed on all the three real data sets so as to make the contribution of different features roughly equal a priori. Since we are concerned with unsupervised learning, the class labels in these data sets are used only for evaluation of the clustering results.

All parameter values of the NMA_CFS procedure were determined experimentally on the above data sets. Both mutation rates (flip and Gaussian mutation) are set to be 0.01. To establish these values, all other parameters are held constant, with only the mutation rate changing. Ten runs are completed for a wide range of values of the mutation rate. The best solutions (in terms of the fitness values) from the 10 runs are averaged, and the best average is selected. The sizes of selection and replacement groups are determined in a similar way. First, the size of the selection group is varied as all other parameters are held constant, and then the size of replacement group is determined using the established selection group size. The selection group sizes of 6, 8, 12, 4, 8, and 16 with the replacement group sizes of 15, 20, 30, 10, 20, and 40 have been established on the above six data sets, respectively. The population sizes are set to be 50, 70, 100, 40, 70, and 120, respectively. The number of generations g, which is used to terminate the evolution, is set to be 20. A larger value of either g or population sizes may lead to a longer runtime but with no significant improvement in performance.

6 Experiments

In this section, after defining several performance measures in Section 6.1, we report a series of experiments in Section 6.2 performed over the synthetic and real data sets described above. We first evaluate the performance of the proposed algorithm and compare it with related work. Additionally, we assess the significance of the niching method and local search operations within the proposed algorithm. Finally, the effectiveness of the simultaneous global clustering and feature subset search mechanism for optimizing the unified criterion is examined. All results were obtained on a PC with AMD Athlon 1800 running the Windows 2000 operation system.

6.1 Performance Measures

Let us suppose that we have obtained a clustering solution with feature selection. Some natural questions are given as follows: How good are the clusters? Is the number of clusters correctly identified? Is the selected feature subset relevant? In what follows, we describe several performance measures relevant to answer these questions.

Since the quality of clusters depends on the particular application, there is no standard criterion for evaluating clustering solutions in the literature [24]. To answer the first question, here, we compute classification errors, since we know the "true" clusters of the synthetic data and the class labels of the real data. This is done by first running the algorithm to be tested on each data set. Next, each cluster of the clustering results is assigned to a class based on examining the class labels of the data objects in that cluster and choosing the majority class. After that, the classification errors are computed by counting the number of misclassified data objects. To answer the second question, we report the number of clusters found. We stress that the class labels are not used during the generation of the clustering results, and they are intended only to provide independent verification of the clusters.

To answer the third question, the feature recall and precision are reported on synthetic data, since the relevant features are known a priori. Recall and precision are

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

TABLE 1
Percent Classification Error, Number of Clusters, Feature Recall, and Precision of the Four Algorithms (NMA_CFS, FS-K Means_BIC, GGA, and K Means) Applied to the Three Synthetic Data Sets

Data Set	Method	Evaluation Method			
		% Classification Error	Number of Clusters	Feature Recall	Feature Precision
Synthetic3_7	NMA_CFS	3.2 (±0.3)	3.0 (±0.0)	1.00 (±0.00)	1.00 (±0.00)
	FS-*k*-means_BIC	5.1 (±0.9)	3.0 (±0.0)	1.00 (±0.00)	0.78 (±0.18)
	GGA	9.0 (±2.5)	fixed at 3	N/A	N/A
	k-means	18.4 (±7.6)	fixed at 3	N/A	N/A
Synthetic4_10	NMA_CFS	3.9 (±0.5)	4.0 (±0.0)	1.00 (±0.00)	0.87 (±0.09)
	FS-*k*-means_BIC	5.3 (±1.1)	4.0 (±0.0)	1.00 (±0.00)	0.65 (±0.24)
	GGA	9.4 (±3.1)	fixed at 4	N/A	N/A
	k-means	22.3 (±8.3)	fixed at 4	N/A	N/A
Synthetic6_20	NMA_CFS	2.8 (±0.7)	6.0 (±0.0)	0.88 (±0.10)	0.95 (±0.04)
	FS-*k*-means_BIC	6.9 (±2.3)	5.2 (±0.9)	0.73 (±0.15)	0.75 (±0.12)
	GGA	23.1(±4.2)	fixed at 6	N/A	N/A
	k-means	35.2 (±9.1)	fixed at 6	N/A	N/A

The entries in the table (averaged over 10 runs) give the means in the form mean (± 95 percent confidence interval).

concepts from text retrieval [43]. Feature recall is the number of relevant features in the selected subset divided by the total number of relevant features. Feature precision is the number of relevant features in the selected subset divided by the total number of features selected. These indices give us an indication of the quality of the features selected. High values of feature recall and precision are desired. Note that, with respect to the real data, we report only the number of feature selected, since the relevant features are unknown.

6.2 Results

We first conducted a set of experiments on both synthetic and real data to evaluate the proposed algorithm, comparing this with a feature selection wrapped around the K Means algorithm (denoted by FS-K Means_BIC). To investigate whether feature selection can help in finding better clusters, we also report the results of the genetically guided algorithm (GGA) [21] and K Means [31] (clustering using all features) on the data described above.

Before discussing the results, we first briefly describe the algorithms to be compared and their implementation details. The FS-K Means_BIC uses SFS [14] to search for feature subsets. The criterion used in this algorithm is the $trace(S_W^{-1}S_B)$ normalized using a cross projection scheme [12]. To find the number of clusters, it is applied to search through a range of possible cluster numbers and evaluate the "goodness" of the results in each case by adding the Bayesian information criterion (BIC) [39], [45] penalty term to the normalized criterion. The GGA to be compared employs a traditional GA for clustering. To make the comparison between the two GA-based methods (GGA and NMA_CFS) more meaningful, the same terminal criterion (i.e., that the fitness value of the best solution in the population has not changed for g generations) is used in both algorithms. Since the GGA converges much more slowly than the NMA_CFS, the g value in the GGA is set to be relatively large, with $g = 40$. The population size of GGA is taken to be identical to the NMA_CFS for experiments on each of the data sets. Other parameters of the GGA are specified according to the original values reported with the best performance. The K Means algorithm has been described earlier in Section 4.3. This algorithm is initialized by dividing the data set into a partitioning of k clusters at random and then uses the k cluster centers as the initial centers [40]. The stopping criterion for the K Means algorithm is an iteration for which no data object changes clusters. The number of clusters k, which is assumed to be fixed in both GGA and K Means, is set to be equal to the correct number of clusters for experiments on each data set. All four algorithms were independently run 10 times on each of the data sets, and their results were then averaged.

Table 1 lists the results of classification errors, number of clusters, feature recall, and precision found with respect to the synthetic data. It clearly shows that NMA_CFS is able to select relevant features and locate appropriate clusters with the correct number of clusters. In comparison with the FS-K Means_BIC, NMA_CFS achieves lower classification errors on all three synthetic data sets. For the FS-K Means_BIC, the classification errors turn out to be 5.1 percent, 5.3 percent, and 6.9 percent on the three data sets, respectively. In comparison, in the NMA_CFS, the classification errors are around 3.2 percent, 3.9 percent, and 2.8 percent, respectively. Furthermore, our proposed algorithm offers more precise feature selection. For example, on *Synthetic*4_10, the feature precision of the FS-K Means_BIC is 0.65, while our method shows 0.87. The performance improvement of our proposed algorithm over the FS-K Means_BIC is mainly due to the use of the simultaneous global clustering and feature selection optimization mechanism, which can overcome less promising locally optimal solutions. In comparison with the GGA and K Means, both NMA_CFS and FS-K Means_BIC show significantly better performance in terms of classification errors. The poor performance of both GGA

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

TABLE 2
Percent Classification Error, Number of Clusters, and Number of Features Selected by the Four Algorithms (NMA_CFS, FS-K Means_BIC GGA, and K Means) Applied to the Three Real Data Sets

Data Set	Method	Evaluation Method		
		% Classification Error	Number of Clusters	Number of Features Selected
Iris	NMA_CFS	3.7 (±1.7)	3.0 (±0.0)	1.9 (±0.2)
	FS-*k*-means_BIC	10.8 (±4.8)	4.5 (±0.5)	2.7 (±0.4)
	GGA	14.1 (±5.6)	fixed at 3	fixed at 4
	k-means	17.8 (±6.9)	fixed at 3	fixed at 4
WDBC	NMA_CFS	9.2 (±0.4)	2.0 (±0.0)	14.8 (±0.9)
	FS-*k*-means_BIC	14.2 (±1.6)	2.0 (±0.0)	15.9 (±1.8)
	GGA	15.6 (±0.0)	fixed at 2	fixed at 30
	k-means	15.6 (±0.0)	fixed at 2	fixed at 30
Image segmentation	NMA_CFS	35.2 (±1.8)	6.6 (±1.4)	2.4 (±0.5)
	FS-*k*-means_BIC	36.5 (±2.6)	8.7 (±1.5)	3.4 (±0.7)
	GGA	36.4 (±2.4)	fixed at 7	fixed at 19
	k-means	38.6 (±3.8)	fixed at 7	fixed at 19

The entries in the table (averaged over 10 runs) give the means in the form mean (± 95 percent confidence interval).

and K Means is mainly because they retain the irrelevant features for clustering. This result may indicate that feature selection helps in finding better clusters.

Table 2 shows the results on the three real data sets. Looking first at the Iris data, NMA_CFS achieves the best classification error, followed by FS-K Means_BIC, GGA, and K Means. NMA_CFS also finds the correct number of clusters, while FS-K Means_BIC tends to overestimate the number of clusters. Both NMA_CFS and FS-K Means_BIC consistently choose *features* 3 (petal length) and 4 (petal width). In fact, we learn from this experiment that these are the most important features for clustering the Iris data. Two typical clustering solutions identified by NMA_CFS and FS-K Means_BIC are shown in Figs. 3a and 3b, respectively, as a scatterplot on the two important features. Looking next at the other two data sets, our algorithm still exhibits the best performance in terms of the classification errors. In the case of image segmentation data, feature selection does not significantly improve the classification errors of FS-K Means_BIC and NMA_CFS. However, they produce comparable results with significantly fewer features. FS-K Means_BIC selects about 3.4 features (typically {4, 10, 12, 18}) out of 19, while our proposed algorithm picks 2.4 features (typically {4, 11}) on the average. *Feature* 4 stands for "short-line-density-5." These results imply that many features in the image segmentation data are redundant or irrelevant for clustering.

Next, we carried out experiments to assess the significance of the niching method and local search operations within the NMA_CFS. For this purpose, we assess and compare the NMA_CFS with three variants: NMA_CFS without local search operations (NMA_CFS_1), NMA_CFS without the niching method (NMA_CFS_2), and NMA_CFS without either the niching method or local search operations (NMA_CFS_3). In the cases of NMA_CFS_2 and NMA_CFS_3, parent pairs are selected using roulette wheel selection, and the population of the next generation is generated by replacing the worst solutions of previous population with the offspring. These algorithms are compared using the same parameter settings. In order to investigate the convergence properties, a relatively large parameter value $g = 50$ is used in the terminal condition for all four algorithms.

Fig. 4 shows the average classification errors of the best solutions over the runtime corresponding to the four algorithms on the *Synthetic*6_20. It is observed that NMA_CFS_3 performs the worst. The niching method and local search operations improve the performance of the algorithm in different ways. NMA_CFS_1 shows that the niching method helps prevent the algorithm from converging to less promising solutions. However, the convergence of the algorithm is slow. NMA_CFS_2 shows that as compared with NMA_CFS_1, the local search operations significantly speed up the convergence during the evolution. However, it is susceptible to less promising optimal solutions with higher classification errors. By incorporating both the niching method and local search operations, NMA_CFS can efficiently recover solutions with low classification errors. In fact, this is the main reason for using the niching method and local search operations in NMA_CFS.

Finally, we report experiments to examine the effectiveness of the simultaneous global clustering and feature subset search mechanism by comparing it with the traditional SFS [14] wrapped around the K Means procedure [31] to optimize the unified criterion J_2 (denoted by FS-K Means_J_2). To search for the number of clusters, we run FS-K Means_J_2, with a range of numbers of clusters from 2 to k_{max}, and evaluate J_2 of the results in each case. k_{max} is taken to be $\sqrt{n}$ for each data set. The results are presented in Table 3.

The results show that the NMA_CFS with the simultaneously applied global clustering and feature subset search

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

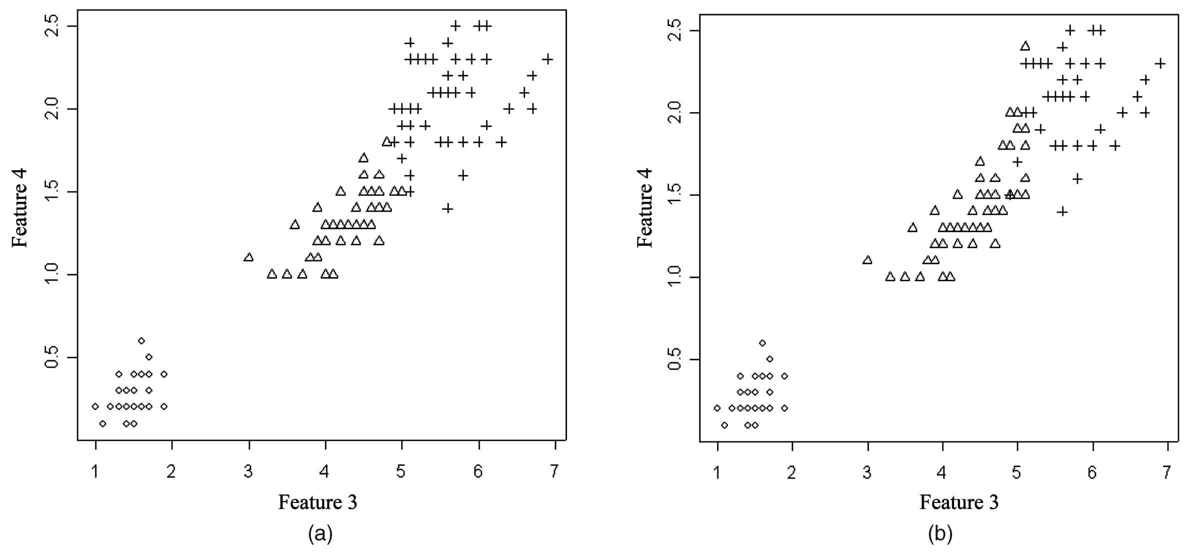


Fig. 3. Typical partitioning found on the Iris data set by (a) NMA_CFS and (b) FS-K Means_BIC shown on the two important features (different clusters represented by different symbols).

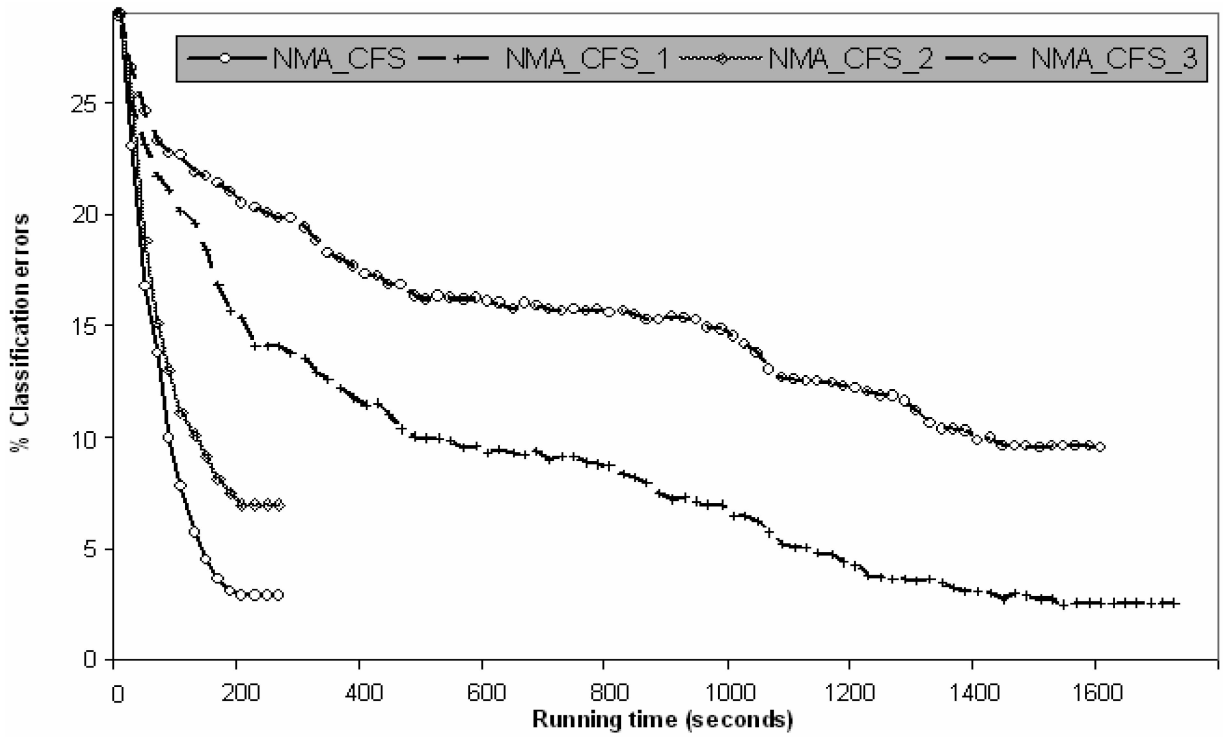


Fig. 4. Percent classification errors of the best solutions (averaged over five runs) plotted against the runtime corresponding to the four algorithms on the *Synthetic*6_20 data set.

mechanism outperforms FS-K Means_J_2. The difference between the two algorithms lies in the search procedure used for the optimization of J_2. Consequently, the results reveal that the proposed mechanism can overcome less promising locally optimal solutions and are therefore able to more accurately select relevant features and identify appropriate partitionings. The traditional SFS wrapped around the K Means procedure, however, usually converges to less promising locally optimal solutions. This is mainly because SFS has difficulty in anticipating the complex interactions among features.

7 Conclusions

We have designed and implemented NMA_CFS by optimizing the suggested unified criterion J_2. The significance of the niching method and local search operations within the proposed algorithm has been clearly shown in the experimental results, which also confirm that the simultaneous global clustering and feature subset optimization mechanism is effective in approaching the problem. The resulting algorithm is generally able to select relevant features and locate appropriate partitionings with the correct number of clusters and outperforms other methods implemented for comparison.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

TABLE 3
Comparing the Results of the Two Algorithms (NMA_CFS and $\mathrm{FS\text{-}K\ Means\_}J_2$) Applied to the Six Data Sets

Data Set	Method	Evaluation Method: % Classification Error	Number of Clusters	Number of Features Selected	Feature Recall	Feature Precision
Synthetic3_7	NMA_CFS	3.2 (±0.3)	3.0 (±0.0)	N/A	1.00 (±0.00)	1.00 (±0.00)
	FS-*k*-means_J_2	5.0 (±1.2)	3.0 (±0.0)	N/A	1.00 (±0.00)	0.82 (±0.16)
Synthetic4_10	NMA_CFS	3.9 (±0.5)	4.0 (±0.0)	N/A	1.00 (±0.00)	0.87 (±0.09)
	FS-*k*-means_J_2	5.9 (±1.3)	4.0 (±0.0)	N/A	1.00 (±0.00)	0.60 (±0.26)
Synthetic6_20	NMA_CFS	2.8 (±0.7)	6.0 (±0.0)	N/A	0.88 (±0.10)	0.95 (±0.04)
	FS-*k*-means_J_2	8.4 (±2.9)	5.5 (±0.6)	N/A	0.64 (±0.19)	0.65 (±0.21)
Iris	NMA_CFS	3.7 (±1.7)	3.0 (±0.0)	1.9 (±0.2)	N/A	N/A
	FS-*k*-means_J_2	9.4 (±3.9)	4.0 (±0.7)	2.5 (±0.6)	N/A	N/A
WDBC	NMA_CFS	9.2 (±0.4)	2.0 (±0.0)	14.8 (±0.9)	N/A	N/A
	FS-*k*-means_J_2	13.7 (±2.1)	2.0 (±0.0)	15.2 (±2.1)	N/A	N/A
Image segmentation	NMA_CFS	35.2 (±1.8)	6.6 (±1.4)	2.4 (±0.5)	N/A	N/A
	FS-*k*-means_J_2	36.9 (±2.8)	8.5 (±1.8)	3.7 (±0.6)	N/A	N/A

The entries in the table (averaged over 10 runs) give the means in the form mean (± 95 percent confidence interval).

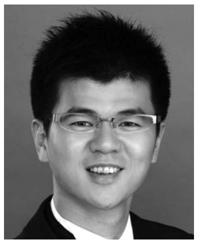
For future work, it will be very interesting to apply the NMA_CFS procedure to real data sets with an abundance of irrelevant or redundant features. An example of such an application is to cluster gene expression data, in which the goal is to identify the informative gene subset for cluster discovery from a large data set that is contaminated with very-high-dimensional irrelevant features. In this case, identifying a relevant subset that adequately captures the underlying structure in the data can be particularly useful. Additionally, as a general optimization framework, the proposed algorithm can be applied for text mining [27]. In such a case, an unbiased clustering criterion in some sense can be produced by computing the mutual information between clusters, thus enabling a better verification of the properties of the proposed optimization scheme.

REFERENCES

[1] H. Almuallim and T. Dietterich, "Learning with Many Irrelevant Features," *Proc. Ninth Nat'l Conf. Artificial Intelligence (AAAI '91),* pp. 547-552, 1991.
[2] S. Areibi and Z. Yang, "Effective Memetic Algorithms for VLSI Design Automation = Genetic Algorithms + Local Search + Multi-Level Clustering," *Evolutionary Computation,* vol. 12, no. 3, pp. 327-353, 2004.
[3] P. Baldi and G.W. Hatfield, *DNA Microarrays and Gene Expression.* Cambridge Univ. Press, 2002.
[4] S. Basu, C.A. Micchelli, and P. Olsen, "Maximum Entropy and Maximum Likelihood Criteria for Feature Selection from Multivariate Data," *Proc. IEEE Int'l Symp. Circuits and Systems (ISCAS '00),* pp. 267-270, 2000.
[5] A. Blum and P. Langley, "Selection of Relevant Features and Examples in Machine Learning," *Artificial Intelligence,* vol. 97, no. 1/2, pp. 245-271, 1997.
[6] C. Cardie, "Using Decision Trees to Improve Case-Based Learning," *Proc. 10th Int'l Conf. Machine Learning (ICML '93),* pp. 25-32, 1993.
[7] S.K. Das, "Feature Selection with a Linear Dependence Measure," *IEEE Trans. Computers,* pp. 1106-1109, 1971.
[8] M. Dash and H. Liu, "Unsupervised Feature Selection," *Proc. Fourth Pacific Asia Conf. Knowledge Discovery and Data Mining (PAKDD '00),* pp. 110-121, 2000.
[9] K.A. DeJong, "An Analysis of the Behavior of a Class of Genetic Adaptative Systems," PhD dissertation, Univ. of Michigan, Ann Arbor, 1975.
[10] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Royal Statistical Soc. B,* vol. 39, no. 1, pp. 1-38, 1977.
[11] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification.* Wiley, 2001.
[12] J. Dy and C. Brodley, "Feature Subset Selection and Order Identification for Unsupervised Learning," *Proc. 17th Int'l Conf. Machine Learning (ICML),* 2000.
[13] J. Dy and C. Brodley, "Feature Selection for Unsupervised Learning," *J. Machine Learning Research,* pp. 845-889, 2004.
[14] I. Foroutan and J. Sklasky, "Feature Selection for Automatic Classification of Non-Gaussian Data," *IEEE Trans. Systems, Man and Cybernetics,* vol. 17, pp. 187-198, 1987.
[15] H. Frigui and R. Krishnapuram, "A Robust Competitive Clustering Algorithm with Applications in Computer Vision," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 5, pp. 450-465, May 1999.
[16] K. Fukunaga, *Statistical Pattern Recognition.* Academic Press, 1990.
[17] M. Garey and D. Johnson, *Computers and Intractability-A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.
[18] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.
[19] D.E. Goldberg and J. Richardson, "Genetic Algorithms with Sharing for Multimodal Function Optimization," *Proc. Second Int'l Conf. Genetic Algorithms (ICGA '87),* pp. 41-49, 1987.
[20] M.A. Hall, "Correlation Based Feature Selection for Discrete and Numeric Class Machine Learning," *Proc. 17th Int'l Conf. Machine Learning (ICML),* 2000.
[21] L.O. Hall, I.B. Ozyurt, and J.C. Bezdek, "Clustering with a Genetically Optimized Approach," *IEEE Trans. Evolutionary Computation,* vol. 3, no. 2, pp. 103-112, 1999.
[22] R.P. Heydorn, "Redundancy in Feature Extraction," *IEEE Trans. Computers,* pp. 1051-1054, 1971.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.

[23] J.H. Holland, *Adaptation in Natural and Artificial Systems.* Univ. of Michigan, Ann Arbor, 1975.
[24] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data.* Prentice Hall, 1988.
[25] A.K. Jain and P. Flynn, "Image Segmentation Using Clustering," *Advances in Image Understanding,* pp. 65-83, 1996.
[26] K. Kira and L. Rendell, "A Practical Approach to Feature Selection," *Proc. Ninth Int'l Conf. Machine Learning (ICML '92),* pp. 249-256, 1992.
[27] J. Kogan, C. Nicholas, and V. Volkovich, "Text Mining with Information-Theoretic Clustering," *IEEE Computational Science and Eng.,* pp. 52-59, 2003.
[28] R. Kohavi and G.H. John, "Wrappers for Feature Subset Selection," *Artificial Intelligence,* vol. 97, no. 1/2, pp. 273-324, 1997.
[29] I. Kononenko, "Estimating Attributes: Analysis and Extension of Relief," *Proc. Seventh European Machine Learning Conf. (ECML '94),* pp. 171-182, 1994.
[30] K. Krishna and M.N. Murty, "Genetic K-Means Algorithm," *IEEE Trans. Systems, Man and Cybernetics Part B,* vol. 29, no. 3, 1999.
[31] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," *Proc. Fifth Berkeley Symp. Math. Statistics and Probability,* pp. 281-297, 1967.
[32] S.W. Mahfoud, "Niching Methods for Genetic Algorithms," PhD dissertation, Univ. of Illinois, Urbana-Champaign, 1995.
[33] U. Maulik and S. Bandyopadhyay, "Genetic-Algorithm-Based Clustering Technique," *Pattern Recognition,* vol. 33, pp. 1455-1465, 2000.
[34] P. Merz and B. Freisleben, "Memetic Algorithms and the Fitness Landscape of the Graph Bipartitioning Problem," *LNCS,* pp. 765-774, 1998.
[35] P. Moscato, "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Toward Memetic Algorithms," technical report, California Inst. Technology, 1989.
[36] P. Moscato, "Memetic Algorithms: A Short Introduction," *New Ideas in Optimization,* D. Corne, M. Dorigo, and F. Glover, eds., McGraw-Hill, pp. 219-234, 1999.
[37] P.M. Murphy and D.W. Aha, "UCI Repository for Machine Learning Databases," technical report, Dept. Information and Computer Science, Univ. of California, Irvine, http://www.ics.uci.edu/mlearn/MLRepository.html, 1994.
[38] N.R. Pal and J.C. Bezdek, "On Cluster Validity for the Fuzzy C-Means Model," *IEEE Trans. Fuzzy Systems,* vol. 3, no. 3, pp. 370-379, 1995.
[39] D. Pelleg and A. Moore, "X-Means: Extending K-Means with Efficient Estimation of the Number of Clusters," *Proc. 17th Int'l Conf. Machine Learning (ICML '00),* pp. 727-734, 2000.
[40] J.M. Pena, J.A. Lozano, and P. Larranaga, "An Empirical Comparison of Four Initialization Methods for the K-Means Algorithms," *Pattern Recognition Letters,* vol. 20, pp. 1027-1040, 1999.
[41] A. Petrowski, "A Clearing Procedure as a Niching Method for Genetic Algorithms," *Proc. IEEE Int'l Conf. Evolutionary Computation (ICEC '96),* pp. 798-803, 1996.
[42] M.L. Raymer, W.F. Punch, E.D. Goodman, L.A. Kuhn, and A.K. Jain, "Dimensionality Reduction Using Genetic Algorithms," *IEEE Trans. Evolutionary Computation,* vol. 4, no. 2, pp. 164-171, 2000.
[43] G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.
[44] B. Sareni and L. Krähenbühl, "Fitness Sharing and Niching Methods Revisited," *IEEE Trans. Evolutionary Computation,* vol. 2, pp. 97-106, 1998.
[45] G. Schwarz, "Estimating the Dimension of a Model," *The Annals of Statistics,* vol. 6, no. 2, pp. 461-464, 1978.
[46] W. Sheng, A. Tucker, and X. Liu, "Clustering with Niching Genetic K-Means Algorithm," *Proc. Genetic and Evolutionary Computation Conf. (GECCO '04),* pp. 162-173, 2004.
[47] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 888-905, Aug. 2000.
[48] S. Tavazoie, D. Hughes, M.J. Campbell, R.J. Cho, and G.M. Church, "Systematic Determination of Genetic Network Architecture," *Nature Genetic,* vol. 22, pp. 281-285, 1999.
[49] S. Theodoridis and K. Koutroumbas, *Pattern Recognition.* Academic, 1999.
[50] G.T. Toussaint and T.R. Vilmansen, "Comments on Feature Selection with a Linear Dependence Measure," *IEEE Trans. Computers,* 1972.
[51] H.K. Tsai, J.M. Yang, Y.F. Tsai, and C.Y. Kao, "An Evolutionary Approach for Gene Expression Patterns," *IEEE Trans. Information Technology in Biomedicine,* vol. 8, no. 2, pp. 69-78, 2004.
[52] D. Whitley, "Modeling Hybrid Genetic Algorithms," *Genetic Algorithms in Eng. and Computer Science,* G. Winter, J. Periaux, M. Galan, and P. Cuesta, eds., John Wiley, pp. 191-201, 1995.
[53] S. Wu, A.W.C. Liew, H. Yan, and M. Yang, "Cluster Analysis of Gene Expression Database on Self-Splitting and Merging Competitive Learning," *IEEE Trans. Information Technology in Biomedicine,* vol. 8, no. 1, 2004.
[54] J.H. Yang and V. Honavar, "Feature Subset Selection Using a Genetic Algorithm," *IEEE Intelligent Systems,* vol. 13, no. 2, pp. 44-49, 1998.



Weiguo Sheng received the MSc degree in information technology from the University of Nottingham, Nottingham, United Kingdom, in 2002 and the PhD degree in computer science from Brunel University, London, in 2005. He is currently a research associate in the Department of Electronics, University of Kent, Canterbury, United Kingdom. His research interests include evolutionary computation, heuristic search, data mining/clustering, and their applications to bioinformatics, biometrics, and security.

Xiaohui Liu is a professor of computing at Brunel University, where he conducts interdisciplinary research concerned with the effective analysis of data, particularly in biomedical areas. He founded a biennial international conference series on intelligent data analysis in 1995 and gave numerous invited and keynote talks. He has more than 100 journal publications in biomedical informatics, data mining, and intelligent systems. He is a charted engineer, a life member of the Association for the Advancement of Artificial Intelligence, and a fellow of the Royal Statistical Society and the British Computer Society.

Michael Fairhurst is the head of the Department of Electronics, University of Kent, Canterbury, United Kingdom. His research interests include computational architectures and algorithms for image analysis and classification, and their applications to handwritten text reading, document processing, and medical image analysis, in particular security and biometrics. His current projects include work on multimodal biometrics, on verification engines for biometrics, on assessing the vulnerability of biometric identification techniques, and on the analysis of handwriting for identification purposes and to improve the effectiveness of automated processing for forensic applications. Biometric processing also underpins work that investigates document encryption linked to biometric data. In related work, he is further developing work that he pioneered at Kent, which has established novel techniques for the assessment and monitoring of neurological conditions through the analysis of patients' writing and drawing abilities. He sits on numerous conference, workshop, and government committees and is on the editorial boards of several international journals. He has published more than 350 papers in the scientific literature. He is a fellow of the International Association for Pattern Recognition (IAPR).

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 24, 2009 at 07:51 from IEEE Xplore. Restrictions apply.