International Journal of Neural Systems, Vol. 18, No. 6 (2008) 1–9

World Scientific
www.worldscientific.com

EVOLVING EDITED k-NEAREST NEIGHBOR CLASSIFIERS

ROBERTO GIL-PITA
Signal Theory and Communications Department
University of Alcalá, Alcalá de Henares
Madrid 28805, Spain
roberto.gil@uah.es

XIN YAO
The Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA)
School of Computer Sciences, University of Birmingham
Birmingham B15 2TT, United Kingdom
X.Yao@cs.bham.ac.uk

and

Nature Inspired Computation and Applications Laboratory (NICAL)
Department of Computer Science and Technology
University of Science and Technology of China
Hefei, Anhui 230027, China

The k-nearest neighbor method is a classifier based on the evaluation of the distances to each pattern in the training set. The edited version of this method consists of the application of this classifier with a subset of the complete training set in which some of the training patterns are excluded, in order to reduce the classification error rate. In recent works, genetic algorithms have been successfully applied to determine which patterns must be included in the edited subset. In this paper we propose a novel implementation of a genetic algorithm for designing edited k-nearest neighbor classifiers. It includes the definition of a novel mean square error based fitness function, a novel clustered crossover technique, and the proposal of a fast smart mutation scheme. In order to evaluate the performance of the proposed method, results using the breast cancer database, the diabetes database and the letter recognition database from the UCI machine learning benchmark repository have been included. Both error rate and computational cost have been considered in the analysis. Obtained results show the improvement achieved by the proposed editing method.

Keywords:

1. Introduction

The nearest neighbor method is one of the most habitual classification methods. Proposed in 1967 by Cover *et al.*[1] the key point of its success lays in its conceptual simplicity. Considering a training set as a set of pre-classified patterns, this method assigns each input pattern to the class of the less distanced pattern in the training set. The k-nearest neighbor (kNN) classifier is a method inspired in the nearest neighbor rule. For each input pattern, it considers not only the nearest pattern, but the k nearest patterns in the training set. So, it uses the classes of these k patterns to tackle the decision. The particular case of the kNN with $k = 1$ is an implementation of the nearest neighbor rule, and it is denominated as 1NN. Using values of k greater than 1, the performance may be improved in terms of error rate, with a considerably low increase in the computational cost.

Editing a kNN classifier consists of the application of the kNN classifier with an edited training set

in order to improve the performance of the classifier in terms of error rate.[2] This edited training set is a subset of the complete training set in which some of the training patterns are excluded. So, depending on the characteristics of the database,[3] and due to the exclusion of these patterns, the kNN may render better results using the edited set, in terms of both error rate and computational cost.

Genetic algorithms (GA) have been successfully applied to select the training patterns included in the edited training set and in finding the optimal non-Euclidean distance metric in the kNN algorithm.[4] In Ref. 5, a study of editing kNN classifiers using GAs with different objective functions is presented. Several databases like the Iris database or the Heart database are used in the experiments. The paper concludes that, from the analyzed objective functions, the best results are obtained when the Counting Estimator with Penalizing Term (CEPT) is selected as objective function. Other interesting article is Ref. 7, in which a GA with a novel crossover method is applied. When two parents are crossed, a high number of possible offsprings are evaluated, and the best two individuals are selected. The work presented in Ref. 8 is other interesting paper that studies the kNN edited with other heuristic techniques, in which the authors study the use of tabu search for solving the problem of editing a 1NN classifier. They use the CEPT objective function, and they evaluate the results with the Iris database and a synthetic two-dimensional database. At last, the use of a multi-objective evolutionary algorithm to simultaneously edit and select the features of a 1NN classifier is evaluated in Ref. 9.

In this paper we propose a novel application of GAs for editing kNN classifiers. We describe the application of a novel genetic algorithm to the problem of the design of edited training sets for kNN classifiers. The originality of the proposed GA lays on the establishment of the fitness function, and on the design of application-specific mutation and crossover methods. First, we focus on the establishment of a novel objective function, inspired of the well-known mean square error (MSE) criterion, which has been applied in the design of many different classifiers like, for example, Multi-Layer Perceptrons (MLP). The proposed fitness function is more sensitive to changes in the composition in the edited set, which makes easier the optimization of the edited kNN. Concerning the crossover of the GA, an application-specific crossover based on the use of clustering techniques is also proposed. It allows to improve the efficiency of the crossover stage in the optimization process. At last, a novel mutation stage is also included in the proposed GA. It consists of estimating the fitness value of the mutated individuals, which allows to increase the effectiveness of the mutations.

In order to study the performance of the proposed GA, this paper also includes the results of a set of experiments, carried out using the breast cancer database, the diabetes database and the letter recognition database from the UCI machine learning benchmark repository. The use of these well-known standard databases allow to establish comparisons with results of other different classifiers. It is important to highlight that, in order to make easier the comparisons of different kind of classifiers, both the computational cost and the classification error rate are considered in this study.

2. Materials and Methods

In this section we carry out a brief description of the main classification method this paper deals with: the kNN method. After studying the statistical basis of the kNN method, we mathematically analyze the editing process of a kNN method, and how the genetic algorithms can be used for editing training sets.

2.1. *kNN statistical analysis*

The kNN classifier is statistically inspired in the estimation of the posterior probability $p(H_i|\mathbf{x})$ of the hypothesis H_i, conditioned to the observation point $\mathbf{x}$.[10] Let's assume N is the number of available training patterns. Considering a volume V around the observation point $\mathbf{x}$ that encompasses k patterns of the training set, it is possible to approximate the probability density function in $\mathbf{x}$ using Eq. (1).

$$p(\mathbf{x}) \simeq \frac{k}{NV} \tag{1}$$

In a similar way, the probability density function $p(\mathbf{x}|\mathbf{H_i})$ of the observation $\mathbf{x}$ conditioned to the hypothesis H_i can be approximated. Let's now assume N_i is the number of patterns associated to

hypothesis H_i, $i = 1, \ldots, C$, so that $N_1 + \cdots + N_C = N$. Considering that the volume V around the observation point encompasses $k[i]$ patterns of hypothesis H_i ($k[1] + \cdots + k[C] = k$), then Eq. (2) gives an approximation of the probability density function in $\mathbf{x}$, conditioned to the hypothesis H_i.

$$p(\mathbf{x}|H_i) \simeq \frac{k[i]}{N_i V} \tag{2}$$

Taking into account that the prior probability $p(H_i)$ can be estimated by $p(H_i) \simeq N_i/N$, then applying Bayes theorem and using Eqs. (1) and (2), we obtain Eq. (3).

$$p(H_i|\mathbf{x}) = \frac{p(\mathbf{x}|H_i)p(H_i)}{p(\mathbf{x})} \simeq \frac{k[i]}{k} \tag{3}$$

The Maximum A Posteriori criterion establishes that, for a given observation $\mathbf{x}$, the decision that maximizes the associated posterior probability must be taken. The kNN method consists of fixing the value of k, the number of patterns included in the volume, being these patterns the k less distanced patterns from the observation point. The decision is taken by evaluating the values of $k[i]$, $i = 1, \ldots, C$, and selecting the class which obtains a highest $k[i]$ value, and, therefore, maximizes the approximation of the posterior probability $p(H_i|\mathbf{x})$ given by Eq. (3). Concerning the distance metric, there are many choices that achieve good results in specific applications. In this paper we use the Euclidean distance, in which the volume V is a hyper-sphere around the observation point.

The parameter k of the kNN method is a user-specific parameter. In many articles it is automatically selected in order to minimize the classification error over the validation set. In this work we have selected it in a first moment, making use of the validation set, and it has remained fixed for the rest of the experiments.

2.2. *Editing a training set*

The edited nearest neighbor rule was proposed by P.E. Hart in 1968.[11] This rule consists in the application of the k-nearest neighbor method with an edited training set. This set, also known as pruned set, is a subset of the complete training set in which not all the training patterns are included. So, some of the training patterns are excluded, outliers are removed and the implemented solution is smoothed improving, in some cases, the generalization ability of the kNN method and, therefore, causing a reduction of the error performance of the classifier.

It is also important to highlight that the use of an editing technique implies a very high increase in the time needed to design the classifier, since it may require hours to define which patterns are included in the edited training set. Fortunately, taking into account that the design process uses to be carried out offline, once the edited training set is generated the number of operations needed to classify each pattern is lower than those needed by the simple kNN, since the number of distances needed to be evaluated with the edited set is lower than with the original set. So, the reduction in the number of training pattern causes a reduction in the required computational cost after training.

The edited training set is defined by the indexes of the patterns included in the subset. In order to apply optimization processes, each subset is associated to a binary vector $\mathbf{b}$, with N bits, being N the total number of patterns in the original training set, so that if the nth bit is activated $b[n] = 1$, then the corresponding nth training pattern is included in the subset. So, being S the number of patterns included in the reduced set so that $S \leq N$, then $\sum_n b[n] = S$. The bit-stream $\mathbf{b}$ is determined by the minimization of a given objective function. In this task, different optimization algorithms can be applied to obtain the values of $\mathbf{b}$.

One key point in the optimization process is the definition of the fitness function. Most of the papers use the classification error as objective function (Eq. (4)), adding in some cases a penalizing term, to consider the number of patterns in the edited training set.

$$F = \frac{1}{N}\sum_{n=1}^{N} h(\mathbf{x}_n) + \alpha b[n] \tag{4}$$

where $h(\mathbf{x}_n)$ is 1 if the nth pattern is wrongly classified, and 0 in other cases. This objective function is also known as the Counting Estimator with Penalizing Term (CEPT) objective function, and it obtains best results among the objective functions evaluated in Ref. 5.

2.3. *Genetic algorithm for editing kNN classifiers*

Genetic algorithms are optimization processes inspired in natural evolution laws. In recent

years, evolutionary computation and genetic algorithms have been successfully applied for designing and training classifiers in many different pattern recognition problems. The basis of evolutionary computation techniques is the modeling of mathematical problems in an evolving way, so the same rules that are applied in natural evolution (selection, mutation and crossover) can be applied to optimize the mathematical problem.[6]

In this paper GAs are applied to optimize the bit-stream **b**, in order to implement an edited kNN rule. So, the process carried out for editing the kNN method using a GA is described as follows:

1. The objective function over the validation set is evaluated for several values of k, and the value that renders better results is selected.
2. A GA is applied in order to obtain a subset of the training patterns that minimizes the objective function. For this purpose a population of $P = 100$ individuals is generated.
3. After evaluating the performance of each individual in the population, a selection process is applied. It consists in selecting the best 10 individuals of the population, and the remaining individuals are removed.
4. The remaining 90 individuals of the new generation are then generated by crossover of the 10 best individuals.
5. Then, binary mutations are applied to the whole population, changing a bit with a probability of 0.8%. This process iterates in step 2 until 100 generations are reached.

These values of the parameters of the GA (population size, crossover rate, mutation probability and number of generations) have been found to obtain a quite good trade off between training time and achieved error rate for the experiments carried out in this paper. During the training process, the objective function over the validation set is calculated for the best individual of each population. The individual that achieves the lowest objective value over the validation set is selected as the final individual.

3. Description of the Proposals

This section includes the description of the three proposals included in the paper. The first proposal deals with the definition of a novel objective function, inspired in the mean square error (MSE) function, used in many classification systems. The second proposal defines a novel crossover strategy for the GA, denominated clustered crossover, designed using *a priori* knowledge of the problem. The third and last proposal establishes a new mutation scheme for the GA, that allows to lead the mutations in the population, improving the performance of the GA in the search of local minima of the objective function.

3.1. *MSE-based objective function*

In this paper we propose the use of a novel objective function, based on the MSE function. Let's consider the kNN as a system with C outputs, one per class, so that each output is calculated using Eq. (3). Therefore, the C outputs of the kNN system are approximations of the posterior probabilities of the data. So, $y_n[i]$, described in Eq. (5), is the ith output of the kNN system, obtained for the nth training pattern $\mathbf{x}_n$.

$$y_n[i] = \frac{k_n[i]}{k} \tag{5}$$

where $k_n[i]$ is the number of nearest patterns of class i for the nth training pattern. Considering the approximation described in Eq. (3), the outputs of this kNN system are estimations of the posterior probabilities of the classes. So, the objective function to minimize is designed using all the outputs of the system, by minimizing the mean square error, defined by Eq. (6).

$$F = \frac{1}{NC} \sum_{n=1}^{N} \sum_{i=1}^{C} \left(y_n[i] - d_n[i] \right)^2 \tag{6}$$

where $y_n[i]$ is the ith output of the system for the nth training pattern, and $d_n[i]$ is the desired ith output for the nth training pattern, so that $d_n[i]$ is 1 for the patterns of the ith class and 0 for the rest. In function of the Kronecker delta, $d_n[i] = \delta[i - c_n]$, being c_n the index of the class for the nth training pattern. Replacing (5) in (6), we obtain (7).

$$F = \frac{1}{NC} \sum_{n=1}^{N} \sum_{i=1}^{C} \left(\frac{k_n[i]}{k} - \delta[i - c_n] \right)^2 \tag{7}$$

The error surface defined by this function is smoother than those obtained using counting estimator based functions, making easier the obtaining of its local minima.

3.2. *Clustered crossover*

Single point crossover (SPC) and random crossover (RC) are two of the crossover methods more used in the literature. Single point crossover generates the offspring by combining the first part of the bit stream of one parent with the last part of the bit stream of the other parent. On the other hand, random crossover generates the offsprings randomly selecting each bit-gene from one of the parents. These two schemes do not consider possible relationship between the genes.

In this paper we propose a novel crossover scheme for the GA, denominated clustered crossover (CC), in order to improve the determination of the best subset of the training set, that minimizes the selected objective function. In the problem of the selection of the edited training set, it is possible to determine the relationship between the different bits of the bit stream. Each gene is related to the inclusion in the reduced subset of a training pattern. So, the value of a given gene is related to the performance of the classifier in the region of the space around the associated training pattern. Therefore, genes can be grouped into clusters considering the spatial position of their training patterns, using a non supervised clustering technique. In this paper we apply the k-means clustering algorithm[12] to the training data, so that these patterns are grouped in some sets of near patterns (clusters). Each group of patterns defines a cluster of genes, that is considered as a transfer unit in the crossover process. So, the bit-stream of the offsprings are obtained by mixing the clusters of genes of two parents, so that a random crossover using clusters of genes instead of single genes is applied. The clustering process is carried out every generation, and the number of clusters has been selected at random. So, every generation the crossover is carried out with different gene clusters.

3.3. *Fast smart mutation scheme*

In this section we describe the application of a mutation scheme that allows to select the best gene or group of genes to be changed, taking into account the variations of the objective function with respect to each gene for a given edited set. We design a fast method for evaluating the error variation when each gene is changed, and we propose a mutation strategy based on these variations of the objective function. We denominate this mutation scheme as "fast smart mutation" (FSM), as it allows to increase the effectiveness of the mutation stage in the genetic algorithm.

The evaluation of the objective function for all the possible bit mutations of a pattern is implemented taking into account prior knowledge of the objective function. Let's consider the bit stream $\mathbf{b}$, then the goal is to find the bit or bits which changes produce the highest reduction in the performance associated to $\mathbf{b}$.

The change of one bit of $\mathbf{b}$ produces the addition or the removal of a training pattern from the edited subset. It causes changes in the values of $k_n[i]$, with a consequent change in the value of the objective function, that might be considered. Let's consider $B_n(k)$ is the distance from the nth training pattern to its kth nearest pattern, then:

- If the mth bit changes from 0 to 1, then the pattern $\mathbf{x}_m$ must now be considered in the subset. If the distance from this pattern to a training pattern $\mathbf{x}_n$ is lower than $B_n(k)$, then this new pattern replaces the kth nearest neighbor of the training pattern $\mathbf{x}_n$. Due to the addition of this new pattern of class c_m, the value of $k_n[c_m]$ is incremented in 1, and due to the removal of the kth training pattern, the value of $k_n[c_n^k]$ is decremented in 1, where c_n^k is the class of the kth nearest neighbor of the training pattern $\mathbf{x}_n$.
- If the mth bit changes from 1 to 0, then the pattern $\mathbf{x}_m$ is removed from the subset. If the distance from this pattern to a training pattern $\mathbf{x}_n$ is lower than $B_n(k)$, then this pattern will cause changes in the values of $k_n[i]$. The pattern $\mathbf{x}_m$ will not continue in the group of the k nearest neighbors of the pattern $\mathbf{x}_n$, and there will be a new pattern in this group. Due to the removal of this pattern of class c_m, the value of $k_n[c_m]$ is decremented in 1, and due to the inclusion of the $k+1$th training pattern in the group, the value of $k_n[c_n^{k+1}]$ is incremented in 1.

Equation (8) represents the function $f_{mn}[i]$, the variations in the values of $k_n[i]$ due to a change in the mth.

$$f_{mn}[i] = \begin{cases} D_{mn} \cdot (\delta[i - c_m] - \delta[i - c_n^k]), \\ \qquad\qquad \text{if } b[m] = 0 \\ D_{mn} \cdot (\delta[i - c_n^{k+1}] - \delta[i - c_m]), \\ \qquad\qquad \text{if } b[m] = 1 \end{cases} \tag{8}$$

where D_{mn} is 1 if the distance from the pattern $\mathbf{x}_m$ to the pattern $\mathbf{x}_n$ is lower than $B_n(k)$, and 0 in other case. So, the MSE-based objective function F_m obtained after changing the mth gene is represented in Eq. (9).

$$F_m = \frac{1}{CN} \sum_{n=1}^{N} \sum_{i=1}^{C} \left(\frac{k_n[i] + f_{mn}[i]}{k} - \delta[i - c_n] \right)^2 \tag{9}$$

The variation in the objective function $\Delta_m = F_m - F$ due to a change in the mth bit can be expressed using Eq. (10).

$$\Delta_m = \frac{1}{CNk^2} \sum_{n=1}^{N} -2k f_{mn}[c_n] + \sum_{i=1}^{C} f_{mn}[i]^2 + 2k_n[i] f_{mn}[i] \tag{10}$$

Using (8) in (10), we obtain (11).

$$\Delta_m = \frac{2}{k^2 CN} \sum_{n=1}^{N} D_{mn}(1 + g_{mn}) \tag{11}$$

where g_{mn} is defined by Eq. (12).

$$g_{mn} = \begin{cases} -1, & \text{if } b[m] = 0 \text{ and } c_n^k = c_m \\ h_{0mn}, & \text{if } b[m] = 0 \text{ and } c_n^k \neq c_m \\ -1, & \text{if } b[m] = 1 \text{ and } c_n^{k+1} = c_m \\ h_{1mn}, & \text{if } b[m] = 1 \text{ and } c_n^{k+1} \neq c_m \end{cases} \tag{12}$$

being h_{0mn} and h_{1mn} defined by Eqs. (13) and (14), respectively.

$$h_{0mn} = \begin{cases} k_n[c_m] - k_n[c_n^k] + k, & \text{if } c_n = c_n^k \\ k_n[c_m] - k_n[c_n^k] - k, & \text{if } c_n = c_m \\ k_n[c_m] - k_n[c_n^k], & \text{other case} \end{cases} \tag{13}$$

$$h_{1mn} = \begin{cases} -k_n[c_m] + k_n[c_n^{k+1}] - k, & \text{if } c_n = c_n^{k+1} \\ -k_n[c_m] + k_n[c_n^{k+1}] + k, & \text{if } c_n = c_m \\ -k_n[c_m] + k_n[c_n^{k+1}], & \text{other case} \end{cases} \tag{14}$$

The value of Δ_m (Eq. (11)) is evaluated for all the possible values of m, in each generation and for every individual. The proposed algorithm allows to quickly evaluate the variation of the objective function with a unique bit change. So, the change in the value of m that efforts the lowest Δ_m will cause the highest reduction of the objective function.

The GA can be speeded up changing more than one bit in every mutation. In many classification environments, the large size of the training set makes this method quite slow, in so only a gene is changed for each individual every mutation stage. On the other hand, using the clustering process proposed in Subsec. 3.2, it is possible to establish groups of "independent" genes. A change in a bit that belongs to a cluster affects to the performance of the classifier in the region of the space nearer to the corresponding training pattern. So, we propose to use the gene clusters to select a group of genes to be mutated. For each cluster, the value of m that efforts the lowest Δ_m is changed, which allows to mutate as many genes as clusters.

The implementation of the algorithm requires the previous calculation of the values of $k_n[i]$, c_n^k and c_n^{k+1}. The process of the genetic algorithm with fast smart mutation is described as follows:

1. The initial population with 100 individuals is generated, all the variables are initialized.
2. The MSE-based objective function is evaluated for every individual of the population. The values of Δ_m are obtained.
3. The k-means algorithm is applied to the training set. The number of clusters is selected at random.
4. For each cluster and each individual, the gene with the value of m that efforts the lowest value of Δ_m is mutated. This process is applied to each individual, so that the optimization process is speeded up.
5. Every 10 generations, clustered crossover is applied to the data. 10 best individuals are chosen as parents, and remaining 90 individuals are generated by clustered crossover of the parents.
6. The process is iterated in step 2, until 100 generations are reached.

During the training process, the objective function over the validation set is calculated for the best individual of each population. The individual that achieves the lowest objective value over the validation set is selected as the final individual. It is important to highlight that the mutation process is applied to each individual every generation, and crossover is not applied every generation, but every 10 generations. This means every individual mutates 10 genes between crossovers, which speeds up the local minima search time.

4. Results

This section includes the results obtained by the methods proposed in the paper. The databases used in the experiments of the paper have been the breast cancer database, the diabetes database and the letter recognition database, collected from the UCI machine learning benchmark repository. Choosing these three databases we try to be able to compare the performance of the different methods in two different environments, allowing to extract more general conclusions.

In order to carry out the experiments, each database has been divided in three subsets: the training set, the validation set and the test set. The training set has been used to generate the edited subsets. The validation set has been used to select the best classifier, and to determine the values of k. The test set has been used to evaluate the final error rate for each classifier. This third set has not been used during the design of the classifier. These three databases and the data preparation techniques are identical to those used in other papers,[13,14] allowing to make comparisons of the obtained results with other different type of classifiers. Table 1 shows a summary of the main characteristics of the used two databases.

The parameter k of the kNN method is a user-specific parameter. In this work we have selected it in a first stage, making use of the validation set, and it has remained fixed for the rest of the experiments. For each database, different kNN classifiers with values of k ranging from 1 to 50 have been implemented, and the value of k that efforts the lowest classification error rate over the validation set has been selected. This value has been $k = 3$ for the breast cancer database, $k = 27$ for the diabetes database, and $k = 4$ for the letter recognition database.

Table 1. Characteristics of the databases.

	Breast cancer	Diabetes	Letter recognition
Classes C	2	2	26
Inputs L	9	8	16
Patterns	699	768	20000
Training size N	349	384	16000
Validation size	175	192	2000
Test size	175	192	2000
Optimum k	3	27	4

In order to assess the performance of the classification methods, the error rate over the test set is measured. Due to the small size of the test sets for the breast cancer and for the diabetes databases, the precision in the estimation of the error rate is considerably low, and some statistical analysis of the results must be applied. So, each experiment has been repeated 30 times, measuring the error rate for each experiment. Results are represented in function of the mean, the standard deviation, the maximum and the minimum of the error rate over these 30 runs. Tables 2, 3 and 4 show the results, in terms of error rate, obtained by the kNN applied with different editing methods, for the breast cancer database, the diabetes database and the letter recognition database, respectively. All the GA algorithms have been implemented following the process described in Sec. 2.3, except the last one, which process is described in Sec. 3.3.

It is important to analyze the computational complexity of the classifiers after training. In most of the cases, the training stage is carried out offline,

Table 2. Error rate (%) obtained with the breast cancer database for the kNN applied with the different editing methods studied in the paper.

Editing technique	Mean	Std	Max	Min
None	1.14	0.00	1.14	1.14
Wilson[2]	1.71	0.00	1.71	1.71
$GA_{\text{CEPT SPC}}$[5]	1.96	1.06	4.57	0.00
$GA_{\text{MSE SPC}}$	1.43	0.76	2.86	0.00
$GA_{\text{MSE RC}}$	1.68	0.78	4.00	0.57
$GA_{\text{MSE CC}}$	1.22	0.65	3.43	0.00
$GA_{\text{MSE CC FSM}}$	0.72	0.54	2.29	0.00

Table 3. Error rate (%) obtained with the diabetes problem for the kNN applied with the different editing methods studied in the paper.

Editing technique	Mean	Std	Max	Min
None	21.88	0.00	21.88	21.88
Wilson[2]	27.08	0.00	27.08	27.08
$GA_{\text{CEPT SPC}}$[5]	22.76	2.00	26.04	18.75
$GA_{\text{MSE SPC}}$	19.84	1.27	21.88	16.67
$GA_{\text{MSE RC}}$	19.62	1.18	21.88	17.19
$GA_{\text{MSE CC}}$	19.60	1.00	22.40	18.23
$GA_{\text{MSE CC FSM}}$	19.39	1.63	22.92	16.67

Table 4. Error rate (%) obtained with the letter recognition problem for the kNN applied with the different editing methods studied in the paper.

Editing technique	Mean	Std	Max	Min
None	4.34	0.00	4.34	4.34
Wilson[2]	5.55	0.00	5.55	5.55
$GA_{CEPT\ SPC}$[5]	6.72	0.30	7.45	5.90
$GA_{MSE\ SPC}$	5.94	0.36	6.60	5.20
$GA_{MSE\ RC}$	5.68	0.33	6.30	5.15
$GA_{MSE\ CC}$	5.78	0.34	6.30	5.00
$GA_{MSE\ CC\ FSM}$	4.98	0.30	5.55	4.60

Table 5. Average size of the edited sets (S) for the different methods studied in the paper.

Editing technique	Breast cancer	Diabetes	Letter rec.
None	349	384	16000
Wilson[2]	323	262	15307
$GA_{CEPT\ SPC}$[5]	101	173	12170
$GA_{MSE\ SPC}$	157	192	12768
$GA_{MSE\ RC}$	163	193	12670
$GA_{MSE\ CC}$	186	191	12786
$GA_{MSE\ CC\ FSM}$	174	195	12567

so the training time only implies a problem with large training sets. On the other hand, the number of operations needed to classify each pattern is in many cases one of the important issues when selecting a classifier. So, in this paper we analyze the computational cost after training of the studied classifiers. The sizes of the edited set (S) are shown in Table 5, for the different editing techniques studied in this paper.

From the obtained results, we can derive the next conclusions:

- The use of the proposed MSE-based objective function has an associated relative reduction of the mean error rate greater than 12%, when it is compared to the use of the CEPT objective function[5] for all the problems considered in this paper.
- The use of the proposed clustered crossover doest not significantly improve the performance in the case of the diabetes and the letter recognition databases, but it achieves a relative reduction of 15% in the mean error rate in the case of the breast cancer database.
- The results obtained by the joint use of the three proposals has an associated relative reduction of the mean error rate greater than 10%, compared to the use of a kNN classifier without editing technique for the diabetes and the breast cancer databases. In the case of the letter recognition database, none of the editing techniques outperforms the kNN classical approach, but the number of evaluated distances needed to classify each pattern is drastically reduced.

Concerning the training time of the different methods (time required to generate the edited set by the GA proposed in the paper), Table 6 shows the average training time in three different cases: a GA with the MSE-based fitting function and single point crossover, a GA with the MSE-based fitting function and the proposed clustered crossover, and a GA with the MSE-based fitting function, clustered crossover and the proposed mutation scheme. Results are shown as average optimization times obtained by a PC Intel Xeon 3.0 GHz with 2 GB of memory. The analysis of the results shows that the training time related to the Diabetes database is more than twice the training time of the Breast Cancer database. The number of training patterns and the dimension of the input vector for both databases is quite similar, and, therefore, this increase might be caused by the increase in the value of k selected by validation. In the case of the Letter Recognition database, the training times are about sixty times the training times of Breast Cancer database. Both databases have similar values of k, and this difference is caused by the increase in the size of the training set, in the dimension of the input vector, and in the number of different classes.

Comparing the training times of the different classification methods, the clustered crossover

Table 6. Average training times (s) for the GA-based editing techniques obtained by a PC Intel Xeon 3.0 GHz with 2 GB of memory, for the different methods and the different training sets considered in this paper.

Editing technique	Breast cancer	Diabetes	Letter rec.
$GA_{MSE\ SPC}$	1 m 9 s	2 m 50 s	73 m 20 s
$GA_{MSE\ CC}$	1 m 9 s	3 m 19 s	236 m 49 s
$GA_{MSE\ CCFSM}$	7 m 41 s	18 m 11 s	396 m 47 s

supposes a high increase in the training time for the largest database, for which the clustering process increases the computational complexity in a factor of 3. In the case of the smart mutation scheme, it implies a high increment in the training time, mostly for high values of k and high values of N.

5. Conclusions

In this paper genetic algorithms have been successfully applied to select the training patterns included in an edited set of a kNN classifier. We have proposed three improvements of the editing process using genetic algorithms. Considering the statistical properties of the kNN classifier, we have proposed a novel mean square error based objective function, which performs better than the counting estimator based objective function. The second proposal presents an analysis of the relationship of the genes in the GA, which is used to propose a clustered crossover. At last, a new fast smart mutation scheme that allows to quickly evaluate the variations in the MSE-based objective function for a change in one bit is described.

Results achieved using the breast cancer database, the diabetes database and the letter recognition from the UCI machine learning benchmark repository have been included. Comparing these results with the best one obtained using kNN without editing, with Wilson's editing, and with GA-based editing using CEPT and SPC, the proposed method achieves an average reduction of 36% for the breast cancer database and 12% for the diabetes database. In the case of the letter recognition database, none of the editing techniques outperforms the simple kNN, but the number of evaluated distances needed to classify each pattern is drastically reduced. These results make the joint use of the three proposed methods quite interesting in the case of not very large training sets.

Acknowledgements

This work has been both partially funded by the Comunidad de Madrid/Universidad de Alcalá (CCG07-UAH/TIC-1572) and the Spanish Ministry of Education and Science (TEC2006-13883-C04-04/TCM) and the Fund for Foreign Scholars in University Research and Teaching Programs (Grant no. B07033) in China.

References

1. T. M. Cover and P. E. Hart, Nearest neighbor pattern classification, *IEEE Transactions on Information Theory* **IT-13** (1967) 21–27.
2. D. L. Wilson, Asymptotic properties of nearest neighbor rules using edited datasets, *IEEE Transactions on Systems, Man and Cybernetics* **2** (1972) 408–421.
3. R. A. Mollineda, J. S. Sánchez and J. M. Sotoca, Data characterization for effective prototype selection, Lecture Notes in Computer Sciences, Vol. 3523, (2005) 27–34.
4. H. He, S. Hawkins, W. Graco and X. Yao, Application of genetic algorithm and k-nearest neighbour method in real world medical fraud detection problem, *Journal of Advanced Computational Intelligence and Intelligent Informatics* **4**(2) (2000) 130–137.
5. L. I. Kuncheva, Fitness functions in editing k-NN reference set by genetic algorithms *Pattern Recognition* **30**(6) (1997) 1041–1049.
6. R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms* (2nd edn.). John Wiley and Sons, Inc. (2004) New Jersey.
7. S. Y. Ho, C. C. Liu and S. Liu, Design of an optimal nearest neigbor classifier using an intelligent genetic algorithm, *Pattern Recognition Letters* **23** (2002) 1495–1503.
8. V. Cerverón and F. J. Ferri, Another move toward the minimum subset: a tabu search approach to the condensed nearest neighbor rule, *IEEE Transactions on System, Man, and Cybernetics-Part B: Cybernetics* **31**(3) (2001) 408–413.
9. J.-H. Chen, H.-M. Chen and S.-Y Ho, Design of nearest neighbor classifiers using an intelligent multi-objective evolutionary algorithm, *Lecture Notes in Artificial Intelligence* **3157** (2004) 262–271.
10. C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press Inc. (1995) New York.
11. P. E. Hart, The condensed nearest neighbor rule, *IEEE Transactions of Information Theory (corresp.)* **IT-14** (1968) 515–516.
12. J. A. Hartigan and M. A. Wong, Algorithm AS 163: A k-means clustering algorithm, *Applied Statistics* **28**(1) (1979) 100–108.
13. X. Yao and Y. Liu, A new evolutionary system for evolving artificial neural networks, *IEEE Transactions on Neural Networks* **8**(3) (1997) 694–713.
14. M. M. Islam, X. Yao and K. Murase, A constructive algorithm for training cooperative neural network ensembles, *IEEE Transactions on Neural Networks* **14**(4) (2003) 820–834.