Pergamon

Pattern Recognition, Vol. 30, No. 6, pp. 1041–1049, 1997

0031-3203/97 $17.00+.00

PII: S0031-3203(96)00134-3

FITNESS FUNCTIONS IN EDITING *k*-NN REFERENCE SET BY GENETIC ALGORITHMS

LUDMILA I. KUNCHEVA[1,*]
Department of Electrical and Electronic Engineering, Imperial College, Exhibition Road,
London SW7 2BT, U.K.

(*Received* 10 *August* 1995; *in revised form* 25 *June* 1996; *received for publication* 26 *August* 1996)

Abstract—In a previous paper the use of GAs as an editing technique for the *k*-nearest neighbor (*k-NN*) classification technique has been suggested. Here we are looking at different fitness functions. An experimental study with the IRIS data set and with a medical data set has been carried out. Best results (smallest subsets with highest test classification accuracy) have been obtained by including in the fitness function a penalizing term accounting for the cardinality of the reference set. © 1997 Pattern Recognition Society. Published by Elsevier Science Ltd.

k-Nearest Neighbors (*k-NN*) rule Genetic algorithms Fitness functions Editing strategies

1. INTRODUCTION

It has been recognized that an appropriate editing of the reference set for the *k*-Nearest Neighbors (*k*-NN) rule can lead both to better computational efficiency and to higher classification accuracy (non-error rate).(1) Let $\mathbf{Z} = \{Z_1, \dots, Z_N\}$ be the set of reference objects (also called "design set" or "set of prototypes") for the *k*-NN rule. Each object has a known class label from the set $\Omega = \{\omega_1, \dots, \omega_M\}$. Let $\mathcal{P}(\mathbf{Z})$ denote the power set of **Z**. An editing technique is supposed to select a set $S^* \in \mathcal{P}(\mathbf{Z})$ subject to a criterion function $J(S)$. The challenge stems from the fact that, as a rule, $J(S)$ is not monotonic on nested sets from $\mathcal{P}(\mathbf{Z})$, neither is it monotonically increasing on the cardinality of S.

Among many editing methods, the following two have been proved to yield optimal results:

- A condensing technique that provides the minimal consistent subset has been developed by Dasarathy.(2) The method results in a set $S^c_{\min}$ which guarantees the correct classification of all objects of **Z** using $S^c_{\min}$ as the reference set, and 1-NN classification rule (consistency). Moreover, the cardinality of $S^c_{\min}$ is the smallest one among these of all consistent $S^c \in \mathcal{P}(\mathbf{Z})$.
- An editing algorithm called MULTIEDIT has been developed by Devijver and Kittler.(3,4) The algorithm consists in iterative reduction of the reference set until finding compact clusters. MULTIEDIT has been proven to be asymptotically Bayes-optimal, i.e. when N tends to infinity, and when the number of repetitions of the reduction process tends to infinity too, the 1-NN classification on the so reduced reference set will lead to Bayesian classification decision.

If **Z** contains all possible objects then Dasarathy's condensing algorithm is obviously optimal. Otherwise, it would need an estimation of how the algorithm generalizes. MULTIEDIT stands at the other extreme—its finite-sample performance needs investigation. The rationale of the majority of other editing techniques is some kind of heuristics about searching through $\mathcal{P}(\mathbf{Z})$.

Whatever the algorithm is, however, the criterion function inevitably contains, explicitly or not, some estimate of the classification accuracy (or, alternatively, the error rate) P_c. It turned out that just a wrong assumption of independence in calculating this accuracy invalidated the optimality proof of Wilson's editing technique.(3,5)

In this paper we consider the formulation of the fitness function [criterion $J(S)$] for a genetic algorithm (GA) used as an editing technique.(6,7) In Section 2, the GA is described, and some fitness functions are proposed. Section 3 contains the description of the experiments with the IRIS data set and with a medical data set. The results are discussed in Section 4.

2. FITNESS FUNCTIONS

The problem of editing of the reference set can be formalized as follows:

$$\text{Find } S^*, \text{ such that } J(S^*) = \max_{S \in \mathcal{P}(\mathbf{Z})} J(S). \qquad (1)$$

The two major problems are:

(a) How to construct $J(S)$?
(b) How to search through $\mathcal{P}(\mathbf{Z})$?

Leaving the combinatorial problem (b) to the GA, as described in Section 2.1., the main concern here is the formulation of $J(S)$.

---

[*] Address for correspondence: Central Laboratory of Biomedical Engineering, Bulgarian Academy of Sciences, Acad. G. Bonchev Street, Block 105, 1113 Sofia, Bulgaria. E-mail:l.kuncheva@ic.ac.uk.

[1] On leave from CLBME, Bulgarian Academy of Sciences.

2.1. *GA as editing technique*

GAs are powerful searching techniques in discrete spaces, suitable for large-scale problems with complex multimodal criterion functions.(8) Recently they have been applied to editing of the reference set for the k-NN rule.(6,7)

Every $S \in \mathcal{P}(\mathbf{Z})$ is represented as a binary string called "chromosome" of length card($\mathbf{Z}$). It contains "0" at the ith position if Z_i is not included in S, and "1" otherwise. (We will denote the chromosome and the set it represents by the same letter.)

Briefly, the GA used here consists in the following steps:

1. *Initialization.* A set of "ps" randomly generated chromosomes is constituted, called "population set" $\Pi = \{S_1, \ldots, S_{ps}\}$. Chromosomes are evaluated through the fitness function.
2. *Forming of the mating set* **M**. Classically, the mating set **M** is formed on a "roulette" principle: each chromosome has as many copies in **M**, as proportional to its fitness value. In the current implementation **M** coincides with Π.
3. *Crossover.* Parent couples are randomly selected out of the elements of **M**. Every couple produces two offspring chromosomes by exchanging parts of their codes as depicted below:

$$\begin{array}{lccc|cc c lccccc} \text{parent}_1 & 1 & 0 & 1 & 0 & 0 & \multirow{2}{*}{$\rightarrow$} & \text{child}_1 & 1 & 0 & 1 & 0 & 1 \\ \text{parent}_2 & 1 & 1 & 0 & 0 & 1 & & \text{child}_2 & 1 & 1 & 0 & 0 & 0 \end{array}$$

The cut-off point is selected at random. The offspring chromosomes form the set O.

4. *Mutation.* Each bit of each offspring chromosome alternates (mutates) with a predefined probability (mutation rate). All elements of O are then evaluated through the fitness function.
5. *Combination.* Π and O are pooled together and the best ps individuals survive, i.e. they stand as the new Π (elitist strategy).

The algorithm continues from step (2) until a stop condition is met.

2.2. *The fitness function*

The most natural choice of the fitness function is some measure of the classification performance of the k-NN rule using S as the reference set. Expectedly, best results can be obtained if we denote the true value of the Bayesian probability of correct classification as $J(S)$. Since this is practically impossible, some estimate has to be used. One can put an arbitrarily complex estimate,(9–12) preferably of low bias, small variance, suitable for small-sample-size problems, etc. Along with this, the fitness function must be computationally feasible.

The general form of the fitness function used here is

$$J(S) = \sum_{j=1}^{N} h_S(Z_j), \tag{2}$$

where $h_S(Z_j)$ expresses the amount that Z_j contributes to the overall assessment of the criterion, given S, and using the k-NN classification rule.

The following fitness functions correspond to the most widely used estimators of the probability of correct classification:

- *Counting estimator* (*CE*).(9) By using

$$h_S^{CE}(Z_j) = \begin{cases} 1, & \text{if } Z_j \text{ is correctly classified on} \\ & \quad S - \{Z_j\} \text{ by } k-\text{NN}, \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

equation (2) coincides with the counting estimator of the probability of correct classification up to a multiplying coefficient ($1/N$).

- *Posterior probability estimator* (*PC*) *based on classified data.*(12,13) Let

$$h_S^{PU}(Z_j) = k_i, \quad \omega_i \text{ is the class label of } Z_j, \tag{4}$$

where k_i is the number of neighbors of Z_j which belong to class ω_i, among the k nearest to Z_j elements of $S - \{Z_j\}$. Then equation (2) corresponds to the posterior probability estimator on classified data up to a coefficient $1/(N{\cdot}k)$. The equivalence is easily checked stipulating that (k_i/k) is an estimate of the posterior probability $P(\omega_i/Z_j)$.

- *Smoothing modification.*(9,13) Let $\Omega = \{\omega_1, \omega_2\}$ be the set of classes, and let $\hat{g}(x)$ be certain discrimination function such that x is allocated to class ω_1 if $\hat{g}(x) > 0$, and to class ω_2, otherwise. The smoothing modification is implemented by equation (2) with

$$h_S^{SM}(Z_j) = \begin{cases} 0 & \text{if } \hat{g}(Z_j) \geq a \text{ and } Z_j \in \omega_2, \\ 0 & \text{if } \hat{g}(Z_j) \leq -a \text{ and } Z_j \in \omega_1, \\ \frac{\hat{g}(Z_j)+a}{b} & \text{if } -a < \hat{g}(Z_j) < b-a \text{ and } Z_j \in \omega_1, \\ -\frac{\hat{g}(Z_j)-a}{b} & \text{if } a-b < \hat{g}(Z_j) < a \text{ and } Z_j \in \omega_2, \\ 1 & \text{otherwise.} \end{cases} \tag{5}$$

The corresponding fitness function can be viewed as a "discrete version", e.g. by substituting

$$\hat{g}(Z_j) = \frac{k_1 - k_2}{k}; \quad a = 1; \quad b = 2.$$

Then equation (5) becomes

$$h_S^{SM}(Z_j) = \begin{cases} 0 & \text{if } k_1 = k \text{ and } Z_j \in \omega_2, \\ 0 & \text{if } k_2 = k \text{ and } Z_j \in \omega_1, \\ \frac{k_1}{k} & \text{if } -1 < (k_1 - k_2)/k < 1 \text{ and } Z_j \in \omega_1, \\ \frac{k_2}{k} & \text{if } -1 < (k_1 - k_2)/k < 1 \text{ and } Z_j \in \omega_2, \\ 1 & \text{otherwise.} \end{cases} \tag{6}$$

Clearly, equation (6) is equivalent to the posterior probability estimator calculated through equation (4). It should be mentioned that the same formula fits the setting of the smoothing modification for multiple classes given by Tutz,(14) except for the part that requires continuity.

- A k-*NN estimator of the 1-NN performance.*(15)

$$h_S^{k\text{-NN}}(Z_j) = 1 - \frac{1}{k \cdot (k-1)} \sum_{i=1}^{M} k_i(k - k_i). \qquad (7)$$

The latter estimate has been reported to lead to very good results even in very small sample sets.(15)

In the light of utilization of the data set it has to be mentioned that equation (2) is a combination of the leave-one-out method (for the elements of S) and the hold-out method (for the rest of $\mathbf{Z}$). This holds for any h_S.

It is reasonable to force the algorithm to select smaller subsets. Here we propose to use an additive term in equation (2), thus modifying the fitness function. There are different ways to introduce the penalty term. The following two have been considered here:

$$J(S) = \sum_{j=1}^{N} h_S(Z_j) - \alpha \cdot \text{card}(S), \qquad (8)$$

where card($\cdot$) denotes cardinality, and α is a coefficient; and

$$J(S) = \sum_{j=1}^{N} h_S(Z_j) - \alpha \cdot P(S), \qquad (9)$$

where

$$P(S) = \begin{cases} 0 & \text{if card}(S) < \text{limit}, \\ [\text{card}(S) - \text{limit}]^2 & \text{otherwise.} \end{cases} \qquad (10)$$

In this paper we have carried out experiments with fitness functions equation (2) with h_S^{CE} and $h_S^{k\text{-NN}}$, and equations (8) and (9) with h_S^{CE}.

3. EXPERIMENTAL SETTING

3.1. *Data sets*

The Fisher's IRIS data set has been used. Since there are many experimental studies on this classical benchmark set we will not contrast our results with others' because many other excellent results might be neglected.

The data set contains 150 four-dimensional feature vectors from three classes: Sestosa, Virginica, and Versicolor. Using all four features, most of the studies reported only two or three misclassified samples. Purposefully we have chosen the worst two-dimensional projection of the data set, i.e. that with the highest overlapping of the classes: Sepal Length versus Sepal Width. The two-dimensional scatterplot is shown in Fig. 1. The data set is divided at random in two parts, used in turn for training and for test. In sequel we will refer to the two partitions as IRIS1 and IRIS2. The results from the two hold-out experiments are then averaged.

The second data set is from the database PROBEN1

(ftp : //ftp.ira.uka.de/pub/neuron/proben1.tar.gz).

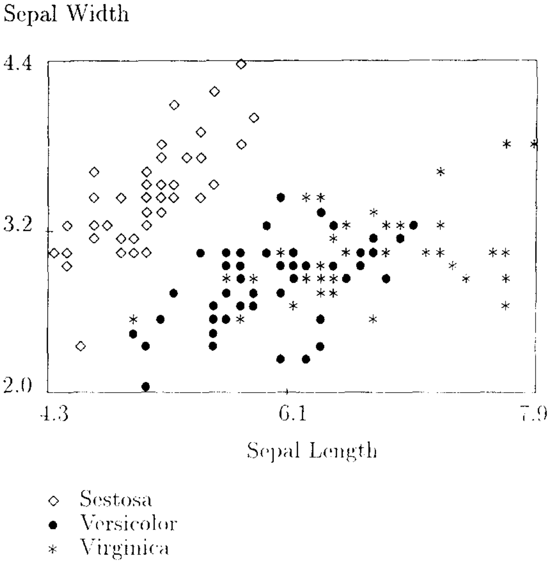

Fig. 1. Two-dimensional projection of IRIS data.

A detailed description can be found in the Technical Report by Prechelt.(16) The data set, called here *heart*, has been supplied by Dr Robert Detrano, V. A. Medical Center, Long Beach and Cleveland Clinic Foundation. The problem is to predict whether or not at least one of four major coronary blood vessels of a patient is reduced in diameter by more than 50%. For the current experiments we used only the following five continuous-valued features:

- age
- resting blood pressure
- cholesterol serum
- maximum heart rate achieved
- ST depression induced by exercise relative to rest

The three partitions of the set into training and test parts are the same as those used by Prechelt.(16) Each one consists of 228 training samples and 75 test ones. The respective partitions will be denoted by heart1, heart2, and heart3. Since we are not using all the features, the recognition accuracy might be inferior to that reported elsewhere.

3.2. *Editing techniques*

The following experiments have been carried out.

1. *Whole sample.* The whole training sample has been used as the reference set for 1-, 3-, 5-, 7- and 9-NN. The results on the test sets, and the averaged results are presented in Tables 1 and 2.

2. *Random selection.* Five reference sets of cardinality l have been generated with $l = 10, 15, 20, \ldots, 70$ for each partition of the IRIS data. Ten sets of each cardinality $10, 20, 30, \ldots, 110, 140, 170, 220$ have been generated for each partition of the *heart* data. The 1-NN rule has been applied with the respective reference sets, and the P_c on the test set has been averaged on the respective five (ten) runs. The piecewise linear functions of P_c versus card(S) are shown in Figs 2–6.

Table 1. *k*-NN results with the whole set as reference (IRIS data)

Partition	Classification accuracy on the test set %				
	k=1	k=3	k=5	k=7	k=9
IRIS1	69.3	72.0	76.0	74.7	80.0
IRIS2	76.0	70.7	78.7	74.4	74.4
Average	72.65	71.35	77.35	74.70	77.35

Table 2. *k*-NN results with the whole set as reference (*heart* data)

Partition	Classification accuracy on the test set %				
	k=1	k=3	k=5	k=7	k=9
heart1	66.7	65.3	68.0	64.0	64.0
heart2	68.0	73.3	73.3	69.3	73.3
heart3	66.7	66.7	70.7	70.7	74.7
Average	67.13	68.43	70.67	68.00	70.67

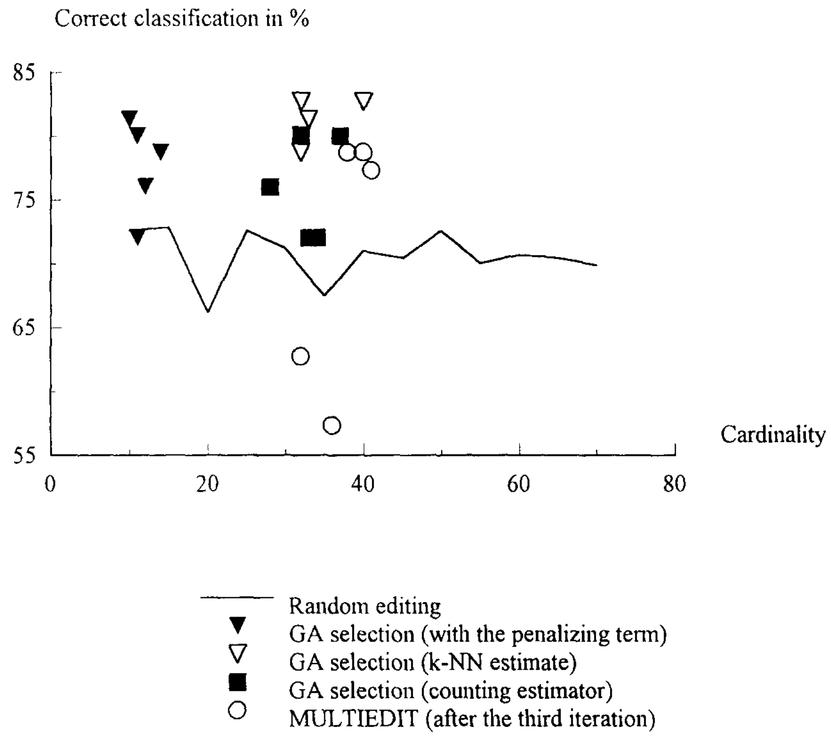

Fig. 2. Experimental results with partition IRIS1.

3. *MULTIEDIT.* It turns out that with the IRIS data, MULTIEDIT algorithm tends to rule out all objects from one or two of the classes, if applied as prescribed. Five independent runs have been performed because of the random component. Table 3 shows the number of the iteration at which the algorithm converged, along with the number of objects retained, and the classes having been ruled out. We tried to stop the algorithm after the third iteration, and to test the performance of the remaining reference sets on the test sets. The results are visualized in Figs 2 and 3 as points with coordinates (card(S), P_c). Averaged results are presented in Table 4.

With the *heart* data, MULTIEDIT converges successfully (holding both classes). The average accuracy, cardinality and number of iterations needed by the algorithm to converge are presented in Table 5.

4. *Wilson's methods.* Wilson's editing technique was also applied. Indeed, the asymptotic optimality of this method has been disproved. Some previous experiments indicated, however, that this technique yields good results in finite-sample-size problems. The results with 1-,

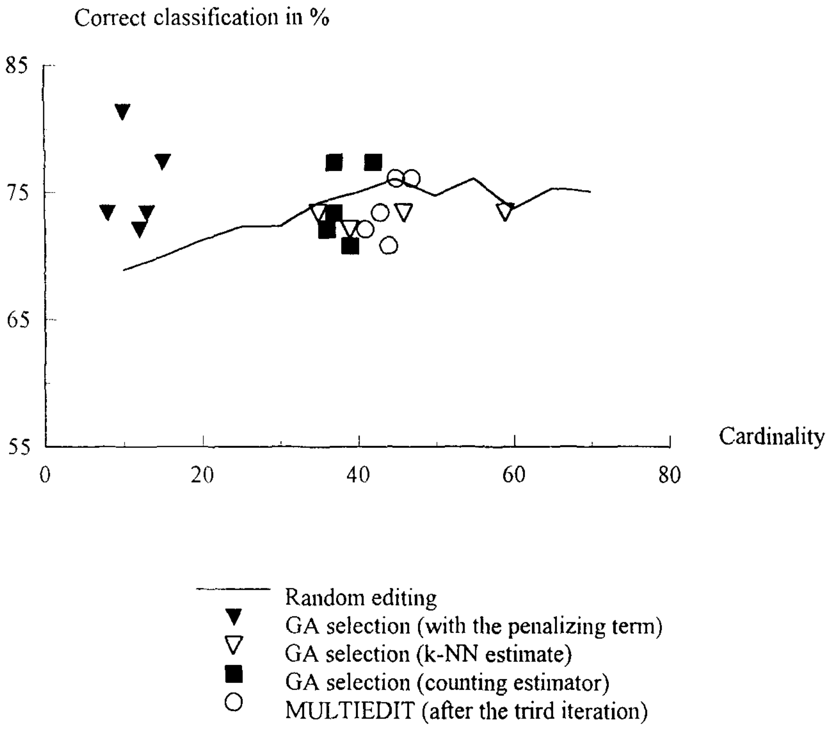

Fig. 3. Experimental results with partition IRIS2.

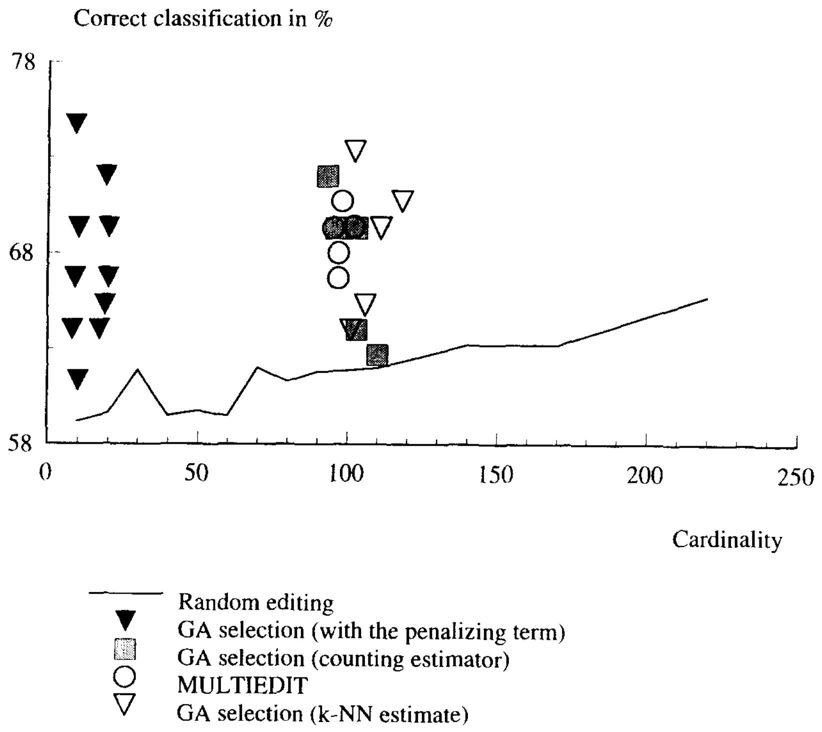

Fig. 4. Experimental results with partition heart1.

3-, 5-, and 7-NN training (and 1-NN test) are shown in Table 6.

5. *GA* [*counting estimator* equation (3)]. Five independent runs with each training-test setting have been performed. It should be emphasized that the number of neighbors is not restricted to a certain value, as it is in other editing techniques. It has been pointed out[1] that the generalization of the editing methods to more than 1-

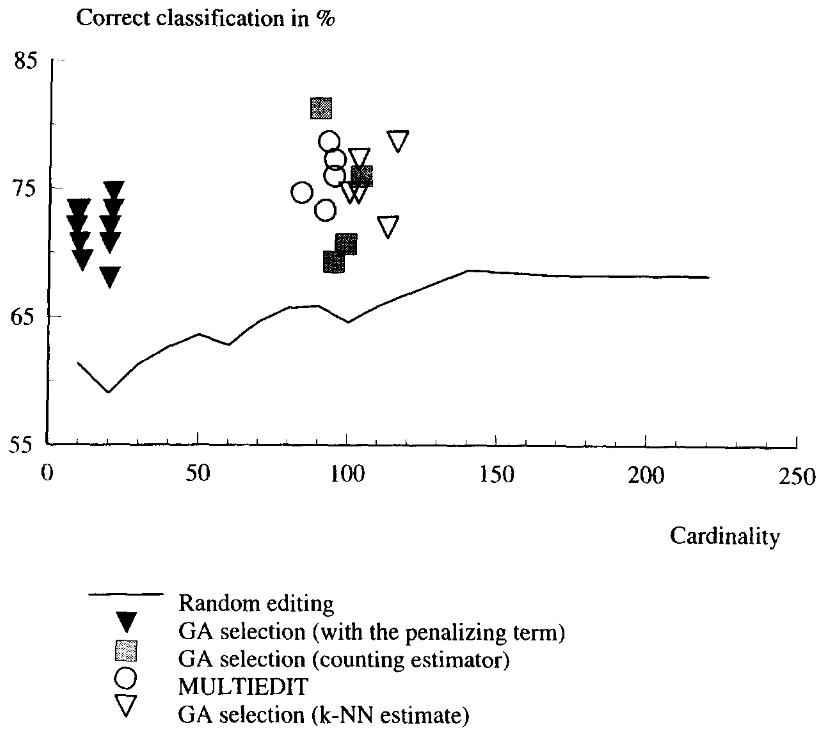

Fig. 5. Experimental results with partition heart2.

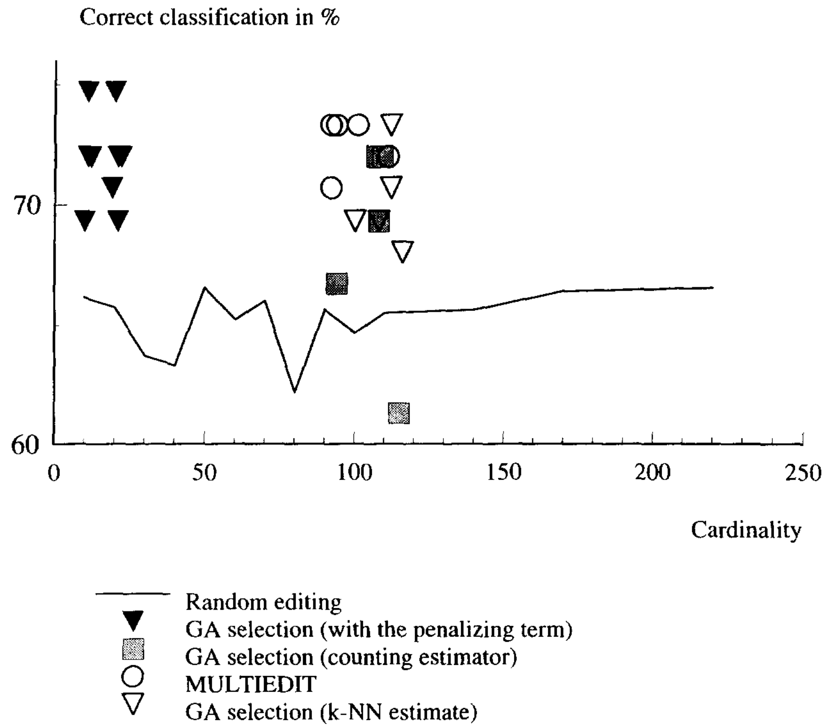

Fig. 6. Experimental results with partition heart3.

NN is not trivial. On the contrary, GA can be trained without any change in the procedure on an arbitrary number of neighbors. The only limitation might be connected to the size of the sample set. We tried 1-NN for training and 1-, 3-, 5-, 7-NN for test. The averaged results are shown in Table 7. In order to estimate the relative performance of the algorithm, points (card(S), P_c) from the five runs (k=3 for training

Table 3. Results from MULTIEDIT editing algorithm (1-Sestosa, 2-Versicolor, 3-Virginica)

Partition	Experiment No.	Iterations to convergence	Cardinality of the resultant set	Classes ruled out		
				1	2	3
IRIS1	1	7	25			■
	2	9	32			■
	3	11	25		■	■
	4	8	25		■	■
	5	5	36			■
IRIS2	6	6	44			
	7	7	43			
	8	13	24		■	■
	9	13	35		■	■
	10	12	35			

Table 4. Averaged results from MULTIEDIT with IRIS data, stopped after the third iteration

Partition	1-NN accuracy in %	Cardinality
IRIS1	70.94	37.4
IRIS2	73.60	44.0
Average	72.27	40.7

Table 5. Averaged results from MULTIEDIT with *heart* data

Partition	Iterations to convergence	1-NN accuracyin %	Cardinality
heart1	11.2	68.80	97.8
heart2	13.6	76.00	91.8
heart3	13.2	72.52	98.0
Average	12.67	72.44	95.87

Table 6. Results from Wilson's method trained with 1-, 3-, 5-, and 7-NN

Partition	1-NN accuracy % (cardinality)			
	k=1	k=3	k=5	k=7
IRIS1	70.7 (61)	74.7 (58)	76.0 (56)	73.3 (60)
IRIS2	84.0 (48)	81.3 (48)	78.7 (48)	78.7 (51)
Average	77.35 (54.5)	78.0 (53)	77.35 (52)	77.35 (52)
heart1	65.3 (137)	70.7 (144)	70.7 (150)	69.3 (150)
heart2	77.3 (136)	77.3 (147)	72.0 (155)	70.7 (152)
heart3	74.7 (139)	72.0 (148)	70.7 (150)	70.7 (159)
Average	71.43 (137.3)	73.33 (146.3)	71.13 (151.7)	70.23 (153.7)

Table 7. Averaged results from GA editing with counting estimator as the fitness function

Partition	Cardinality	Classification accuracy, %			
		k=1	k=3	k=5	k=7
IRIS1	37	74.68	71.20	75.18	75.46
IRIS2	40.2	72.80	72.80	75.54	77.08
Average	38.6	73.74	72.00	75.46	76.27
heart1	101	67.46	66.14	66.92	66.14
heart2	95.4	75.20	73.36	77.34	77.06
heart3	106.6	68.26	70.66	69.08	70.92
Average	101	70.31	70.05	71.11	71.37

and k=1 for test with IRIS data, and k=1 both for training and test with *heart* data) are visualized in Figs 2–6.

6. *GA* [k-NN *estimate* equation (7)]. With this fitness function we performed five independent runs with 3-NN training for each training-test splitting. The averaged results are shown in Table 8. Figures 2–6 contain (card(S), P_c) pixels corresponding to those five runs of the GA with 3-NN estimate and 1-NN for test.

Table 8. Averaged results from GA editing with k-NN estimator as the fitness function

Partition	Cardinality	Classification accuracy, %
IRIS1	33.8	81.08
IRIS2	43.0	72.54
Average	38.4	76.81
heart1	107.6	68.52
heart2	107.0	75.48
heart3	109.6	70.12
Average	108.1	71.37

Table 9. Averaged results from GA editing with counting estimator and a penalizing term in the fitness function (IRIS data)

α	Partition	Cardinality	1-NN accuracy, %
0.03	IRIS1	13.8	78.66
	IRIS2	12.6	74.40
	Average	13.2	76.53
0.05	IRIS1	11.6	77.60
	IRIS2	11.6	75.44
	Average	11.6	76.52
0.08	IRIS1	13.4	74.12
	IRIS2	12.6	69.62
	Average	13.0	71.87

Table 10. Averaged results from GA editing with counting estimator and a penalizing term in the fitness function (*heart* data)

Limit	Partition	Cardinality	Classification accuracy, %
10	heart1	9.2	67.2
	heart2	9.8	71.72
	heart3	10.8	71.46
	Average	9.93	70.13
20	heart1	19	64.47
	heart2	20.4	71.74
	heart3	20.6	71.74
	Average	20.0	70.32

7. *GA (counting estimator with a penalizing term)*. - Here the experiments have been confined to 1-NN only, because of the assumption that GA will ultimately select small-size sets. The averaged results from five independent runs with IRIS data are reported in Table 9 with α=0.03, 0.05, and 0.08. The respective points with α=0.05 are shown in Figs 2 and 3. The scatterplot of one of the sets selected with the penalizing term is shown in Fig. 7. We tried a fitness function with a penalizing term equation (9) on the *heart* data. In the experiments we fixed α=0.0045. The *limits* on the cardinality have been set to 10 and 20. In order to prevent GA from an unnecessary initial search among highly nonfeasible solutions we used "sparse" initialization. We set 10% of the genes to 1 and the rest to 0 with *limit* = 10, and 20% 1-s with *limit* = 20. The results are shown in Table 10, and visualized in Figs 4–6.

It appeared that random breaking of ties led to a minor improvement of about 1%. For the sake of purity, and without losing generality, in all experiments the tied votes were considered as misclassification.

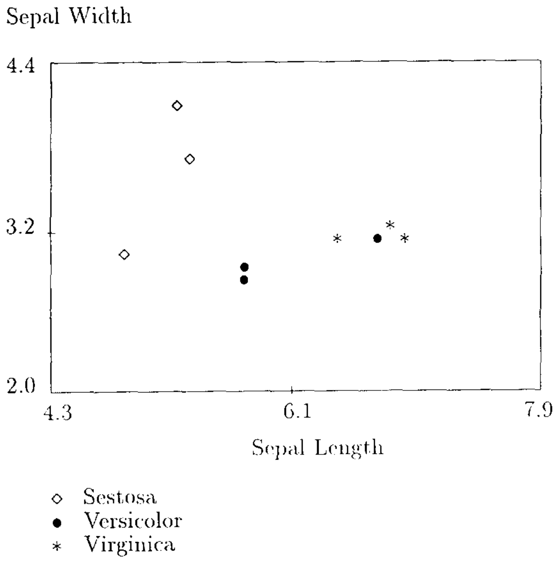

Fig. 7. One of the reference set selected from IRIS data with a penalizing term in the fitness function.

4. ANALYSIS AND CONCLUSIONS

- The most interesting (and somewhat surprising) result is that the best fitness function was that with the penalizing term equation (8) or (9). It led to best reference subsets (small size and high P_c) in all the runs in all training-test settings. This can be explained by the fact that the fitness function forces the GA to select true prototypes of the classes without any redundancy. Obviously, the choice of the coefficient α determines greatly the overall performance. Note that the a priori probabilities of the classes are not explicitly used. The algorithm itself decides the "balance" of the objects so that the performance remains unaffected. The approaches that reduce the reference set by clustering require a procedure to solve the cluster validity problem, or simply requires the number of clusters to be set up in advance. Using a GA, this problem is supposed to be solved empirically, (or even *adaptively*) during the run.
- It turned out that the two fitness functions based on equations (3) and (7) did not differ substantially. Being a more robust estimate, the k-NN fitness function (7) appeared to be slightly better.
- Without limiting the cardinality, all methods selected subsets of cardinality close to half of the original one.

- Generally, Wilson's technique provided high P_c at the expense of larger cardinality. Slightly higher values have been reached with k=3 for training, which coincides with the recommendation of the author of the technique.(5)
- MULTIEDIT in its pure form tended to rule out one or two classes in few steps when the classes are highly overlapping (IRIS data). With the *heart* data, the algorithm yielded very good reference sets but still of a comparatively high cardinality.
- It seems that the controlled initial seeding facilitates the editing by a GA when the fitness function contains a penalizing term.

All this lends support to our claim that using GAs as editing techniques we can significantly reduce the reference set without losing the classification accuracy. It seems promising to combine not only the counting estimator with a penalizing term in the fitness function equations (8) and (9) but other estimates too. The penalizing coefficient can be made to vary along with the generations.

REFERENCES

1. B. V. Dasarathy, *Nearest Neighbor* (NN) *Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, Los Alamitos, California (1990).
2. B. V. Dasarathy, Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design, *IEEE Trans. Systems Man Cybernet.* **24**, 511–517 (1994).
3. P. A. Devijver and J. Kittler, On the edited nearest neighbor rule, *Proc. 5th Int. Conf. on Pattern Recognition*, pp. 72–80 (1980).
4. P. A. Devijver and J. Kittler, *Pattern Recognition. A Statistical Approach*. Prentice-Hall, Englewood Cliffs, New Jersey (1982).
5. D. L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, *IEEE Trans. Systems Man Cybernet.* **SMC-2**, 408–421 (1972).
6. L. I. Kuncheva, Selection of a k-NN reference set by genetic algorithm and index of fuzziness, *Proc. Second European Conf. on Fuzzy and Intelligent Technologies, EUFIT'94*, Aachen, Germany, pp. 640–644 (1994).
7. L. I. Kuncheva, Editing for the k-nearest neighbors rule by a genetic algorithm, *Pattern Recognition Lett.* **16**, 809–814 (1995).
8. D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, Massachusetts (1989).
9. S. J. Raudys and A. K. Jain, Small sample size effects in statistical pattern recognition: Recommendations for practitioners and open problems, *Proc. 10th Int. Conf. on Pattern Recognition*, Atlantic City, New Jersey, pp. 417–423 (1990).
10. G. T. Toussaint, Bibliography on estimation of misclassification, *IEEE Trans. Inform. Theory* **20**, 472–479 (1974).
11. J. Kittler, On the efficiency of data utilization in system error estimation, *Proc. 6th Int. Conf. on Pattern Recognition*, Munich, Germany, pp. 868–871 (1982).
12. D. S. Moore, S. J. Whitsitt and D. A. Landgrebe, Variance comparisons for unbiased estimators of probability of correct classification, *IEEE Trans. Inform. Theory* **22**, 102–105 (1976).
13. N. Glick, Additive estimators for probabilities of correct classification, *Pattern Recognition* **10**, 211–222 (1978).
14. G. E. Tutz, Smoothed additive estimators for non-error rates in multiple discriminant analysis, *Pattern Recognition* **18**, 151–159 (1985).
15. P. A. Devijver, A multiclass k-NN approach to Bayes risk estimation, *Pattern Recognition Lett.* **3**, 1–6 (1985).
16. L. Prechelt, PROBEN1—A set of neural network benchmark problems and benchmarking rules, Technical Report, # 21/94 (1994) (1985).

About the Author—LUDMILA KUNCHEVA graduated in Automatics from the Higher Institute of Mechanical and Electrical Engineering, Sofia, in 1982. She received her Ph.D. degree in Pattern Recognition in 1987 from the Bulgarian Academy of Sciences. Dr Kuncheva is an Associate Professor at the Central Laboratory of Biomedical Engineering, Bulgarian Academy of Sciences. At present she is a visiting research fellow at the Department of Electrical and Electronic Engineering, Imperial College, London. Her main scientific interests include fuzzy pattern recognition, aggregation of multiple classification decisions, neural networks, genetic algorithms and medical applications.