ELSEVIER

Neurocomputing 14 (1997) 273–288

NEUROCOMPUTING

Initializing of an RBF network by a genetic algorithm [1]

Ludmila I. Kuncheva *

Department of Biomedical Engineering (CLBME), Bulgarian Academy of Sciences, Acad. G. Bonchev Street, Block 105, 1113 Sofia, Bulgaria

Received 7 July 1995; accepted 31 March 1996

Abstract

In this paper we use a genetic algorithm (GA) for selecting the initial seed points (prototypes, kernels) for a Radial Basis Function (RBF) classifier. The chromosome is directly mapped onto the training set and represents a subset: it contains 1 at the ith position if the ith element of the set is included, and 0, otherwise. Thus the GA serves a condensing technique that can hopefully lead to a small subset which still retains relevant classification information. We propose to use the set corresponding to the best chromosome from the final population as the seed points of the RBF network. Simulated annealing is used to tune the parameters of the radial function without changing kernels location. Experimental results with IRIS and two-spirals data sets are presented.

Keywords: Pattern recognition; Radial-basis-functions (RBF) networks; Genetic algorithms; K-NN condensing techniques; Nonparametric classifiers; Prototypes selection

1. Introduction

The background of the classifiers implemented as *Radial Basis Functions (RBF) networks* are the nonparametric techniques for probability density approximation. This relates the RBF classifiers to Parzen's windows, potential functions, even to the k-Nearest Neighbors (k-NN) method [18]. Being universal function approximators, RBF networks are asymptotically Bayes-optimal classifiers [15,21], and besides, are considered to be an improved alternative of the classical multilayer neural networks in generalization ability, computational efficiency, and biological plausibility [5,21].

* Email: lucy@bgcict.acad.bg

[1] This work was supported by the contract No. TH-468/94 with the Bulgarian National Scientific Fund.

PII S0925-2312(96)00035-5

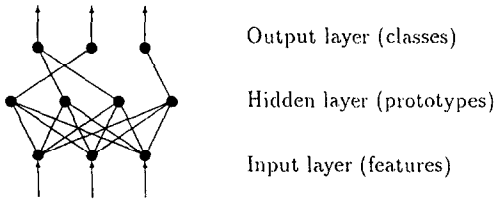

Fig. 1. RBF network configuration.

According to its philosophy, an RBF network consists of an input layer, a hidden layer, and an output layer (Fig. 1). It has been shown that such a network is isomorphic to a classifier that uses Parzen's estimation of the conditional probability density functions [5]. To keep this correspondence the network should be sparsely connected so that each node at the hidden layer is "responsible" for one class only.

Compared to other nonparametric classifiers, RBF networks possess a larger set of tunable parameters:

- *the distance type*;
- *the radial function type and parameters*: Gaussian, exponential, elliptical [20], π-shaped fuzzy [24], cubic [10], B-spline [3], triangular, etc.;
- *the kernels location*;
- *the initial reference set of prototypes.*

Apart from this, various techniques for inferring the final decision of the network can be applied: weighted sum, product, OR-like operators, etc. All this could lead to a better performance in the finite-sample case at the expense of a certain lack of theoretical strictness. In other words, the heuristic schemes and techniques that have appeared useful in the finite-sample applications may defy asymptotic generalization and analysis.

One of the most important problems in setting up an RBF classifier is that of seeding of the initial set of prototypes (kernels, centers). The type of the network determines in a high degree what strategy for prototype selection should be followed. For example, the sparse configuration adopted in this study requires that each kernel comes with its class label. Therefore, separate clustering of the class data can be considered rather than clustering of the entire data set.

Recently a relation of RBF networks to a class of fuzzy systems both for classification and linguistic knowledge extraction has been commented [4,10,19]. The analogy can be directly seen if the RBF value is regarded as a membership degree which the classification rule (the prototype) renders to the given object. What is worth mentioning, is that a fuzzy system with large number of rules does not comply with the main presumption, i.e. it loses its transparency. The task of selecting the most relevant rules is exactly dual to selecting the set of prototypes in RBF initialization. Reducing the reference set of uniformly distributed fuzzy rules by a GA has been extensively discussed by Ishibuchi and coworkers [11,12].

In this paper we use a genetic algorithm (GA) for prototype selection prior to network training. Some preliminary results have been reported in [17]. They contrast the GA selection with the plain random choice. Here we give a more detailed experimental statement and discussion, together with further results obtained by extensive (*heavy*)

random search and clustering. In Section 2 the initialization problem is outlined. The GA for selection of the prototype set is presented in Section 3. The configuration and the training scheme of the RBF network are described in Section 4. Experimental results with the "two-spirals" and with IRIS data set are presented in Section 5.

2. The problem of the RBF network initialization

Unlike some other algorithms for training an RBF network, in the proposed scheme only the parameters of the radial basis functions are tuned without changing or moving the seed points, which puts a special emphasis onto the initial choice of prototypes. Five strategies for determining the set of prototypes can be detailed:

1. *Random selection.* Pawlak and Ng [21] have proven that the random selection of prototypes from the data set does not affect the asymptotic optimality of the RBF classifiers. Although this is apparently a naive choice, sometimes it is more justifiable than the more sophisticated techniques. The reason is that, especially in small sample size problems, intricate algorithms can easily lead to overfitting.
2. *Clustering.* This is perhaps the most widely used way of selecting prototypes. Most of the procedures operate on the whole data set thereby finding as cluster centers the modes of the mixture density distribution rather than the modes of the conditional density distributions. This means that one cluster may correspond to more than one class. In the RBF network considered in this study we need labeled prototypes which can be obtained by separate clustering of the class data.
3. *Sequential growing.* There are lots of heuristic algorithms for sequential growing and/or reducing of the prototype set by using various creating and fusing operations. They usually depend on the concrete implementation of the network since the prototype set is not set up in advance but is evolved in the process of tuning of the whole system.
4. *Systematic seeding.* In some studies the prototypes are uniformly placed onto a grid. Other authors consider prototypes specified by experts, or by analyzing the function to be approximated (e.g., [23,13]).
5. *Editing techniques* (see [6,8]). A few papers describe prototype selection by editing techniques designed originally for reducing the reference data set for the k-Nearest Neighbors classifiers (e.g., [7,14]).

In this paper we propose to use a genetic algorithm as a k-NN editing (condensing) technique for selecting a reference set of prototypes [16]. The algorithm is supposed to end up with a sufficiently small subset with high classification accuracy, as tested with the k-NN rule. We believe that this resultant subset contains relevant information about the whole training set.

Why is an editing technique preferred to all the others listed above?

It is clear that (2), (3), and (4) from the above strategies do not provide prototypes corresponding to real data items from the training set. We would like to keep the ability to map back the reference set to the initial domain, i.e. to be able to point out which set of real data items has proven to be a good reference set. This can be a first step towards a case-based knowledge extraction from the trained RBF network. The seed points

created by one of the other strategies may represent "impossible" objects, especially if the feature space dimensionality is high, and there are complex interdependencies between the features.

Second, since we need labeled prototypes for the RBF network, it is easier to choose them form the preliminary labeled training set than to associate class labels to the artificial reference points afterwards.

Another supportive argument is that a GA could provide a prototype set which is not related to a particular RBF network scheme.

3. GA condensing technique

Genetic algorithms (GAs) are viewed as robust and powerful searching techniques imitating natural reproduction [1,2]. In this study we consider the application of a GA to selecting a reference set for the k-NN rule [16].

Let $Z = \{Z_1, \ldots, Z_N\}$ be the set of equally distributed n-dimensional random vectors, each one coming from one of the predefined classes from the set $\Omega = \{\omega_1, \ldots, \omega_M\}$. We are interested in selecting the "best" subset $S^* \subset Z$ in terms of a certain criterion $\mathcal{F}(S)$, $S \subset Z$. Each chromosome represents a subset of the initial training set Z, e.g., the chromosome $S = [0101100010\ldots]$ stands for the subset $S \subset Z$, $S = \{Z_2, Z_4, Z_5, Z_9, \ldots\}$.

The algorithm operates simultaneously on a set of ps chromosomes (population set) $\pi = \{S_1, \ldots, S_{ps}\}$. The population set is subsequently evolved in order to find chromosomes with as high as possible criterion value.

The most natural criterion (fitness function) $\mathcal{F}(S)$ is a certain measure of the classification accuracy of the k-NN rule when using for reference only the subset corresponding to the chromosome S. Obviously, the criterion is not monotonic, i.e. $S_1 \subset S_2$ does not involve $\mathcal{F}(S_1) < \mathcal{F}(S_2)$, which can be considered a finite-sample effect. Clearly, if the condensing algorithm rules out all objects that do not belong to their own Bayesian classification regions, this will lead to improved finite sample accuracy [6,8]. The non-monotonicity, along with the large discrete searching space (2^N) form the rationale for trying out a GA instead of some other condensing or searching technique.

In this study we use the following criterion

$$\mathcal{F}(S) = \begin{cases} \mathcal{F}_{\mathrm{CE}}(S) - \alpha\,(\mathit{card}(S) - \mathit{limit})^2, \text{ if } \mathit{card}(S) > \mathit{limit}; \\ \mathcal{F}_{\mathrm{CE}}(S), \text{ otherwise}, \end{cases} \tag{1}$$

where $\mathit{card}(\cdot)$ denotes cardinality, α is a coefficient, limit is the maximal desired cardinality of the reference set, and $\mathcal{F}_{\mathrm{CE}}(S)$ is the counting estimate of the probability of correct classification of the k-NN rule:

$$\mathcal{F}_{\mathrm{CE}}(S) = \frac{1}{N} \sum_{j=1}^{N} h_{\mathrm{CE}}(S, Z_j), \tag{2}$$

where

$$h_{\mathrm{CE}}(S,Z_j) = \begin{cases} 1, & \text{if } Z_j \text{ is correctly classified,} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Note that in the calculation of $h_{\mathrm{CE}}(S,Z_j)$ we use a pseudo-leave-one-out technique: if Z_j is not in S, the whole S is used as the reference set, otherwise – the k-Nearest Neighbors are searched in $S - \{Z_j\}$. The second term of Eq. (1) (the penalty term) forces the algorithm to choose prototype sets that are either smaller or slightly exceeding the limit.

The condensing GA is implemented as follows:

1. *Initialization.* Random generation of ps chromosomes. Usually, each bit in a chromosome is set to either 0 or 1 with probability 0.5. In this study we use an external parameter to set up this probability in order to start with chromosomes of lesser cardinality. This prevents the algorithm from wasting initial iterations on processing chromosomes with lower survival perspective because of the heavy violation of the limit condition. Each chromosome is assessed by the fitness function (1).
2. *Constituting the mating set M.* We use the entire population set π as the mating set in order to keep diversity and to avoid premature convergence.
3. *Selecting of parents and crossover.* Parent couples are selected at random from the mating set (with replacement). The probability of crossover is set to 1.0, i.e. each couple produces two children, thus forming the offspring set O. In order to reduce the influence of some eventual epistasis (dependence of the neighboring genes in the chromosome) we chose the uniform crossover operator [1,2].
4. *Mutation.* Each element of O is subjected to mutation. This operation inverts each bit in a chromosome with a predefined probability. The resulting offspring chromosomes are then evaluated by means of the fitness function.
5. *Combination.* The two sets π and O are pooled together and only the "best" ps chromosomes, according to the criterion, survive as the next population π (elitist strategy).

The process is repeated from step 2 to 5 until the condition of a stop criterion is met; in our case it was a limit on the number of generations. The set of parameters of the GA forms the so called *GA template.* The concrete GA template for each set of experiments is presented together with the experimental results in Section 5.

4. Configuration and training of the RBF network

The proposed RBF network consists of three layers, as Fig. 1 shows. The input nodes only transmit the input values (feature values of the object $x \in \Re^n$ to be classified) to all nodes at the hidden layer. The outputs correspond to the classes from Ω. Each node at the hidden layer is connected with the output corresponding to its class label. Each such node N_j, $j = 1,\ldots,L$ with a corresponding prototype vector $c_j \in \Re^n$ is supplied with two kernel functions: $\mu(x,c_j)$ for activation, and $\nu(x,c_j)$ for restraining, respectively, $\mu(x,c_j), \nu(x,c_j) \in [0,1]$. A node at the hidden layer renders a certain degree of

"support" to the newcomer x for the class ω_i which the node is responsible for. This degree is formulated on the basis of $\mu(x,c_j)$ and $\nu(x,c_j)$. Output nodes sum up the support given to x and label it as belonging to the class with the maximal support.

The output of the hidden-layer node N_j is

$$y_j^h(x) = F\big(\mu(x,c_j),\nu(x,c_j)\big), \quad j = 1,\ldots,L, \tag{4}$$

where $F(\cdot,\cdot)$ is a function aggregating the activating and inhibitory potentials of the node. Note that the aggregation can be asymmetric which corresponds to the known behavior of the biological neuron.

The output neurons perform the summation

$$y_k^o(x) = \sum_{j=1}^{L} Ind(k,j)\, y_j^h(x), \quad k = 1,\ldots,M, \tag{5}$$

where $Ind(k,j)$ is an indicator function such that

$$Ind(k,j) = \begin{cases} 1, & \text{if the neuron } N_j \text{ corresponds to class } \omega_k, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

In the current study the following two functions are used

$$\mu(x,c_j) = \exp\big(-\|x - c_j\|_{L_2}^2\big), \tag{7}$$

where $\|x - c_j\|_{L_2}$ is the Euclidean norm in $\Re^n$, and

$$\nu(x,c_j) = 1 - \exp\big(-\|x - c_j\|_{L_1}^2\big), \tag{8}$$

with $\|x - c_j\|_{L_1}$, the L_1 norm. In the proposed network the location of the seed points (centers of the neurons at the hidden layer) is not changed. In order to make the two functions trainable we use a weighted version of the L_p norm:

$$\|x - c_j\|_{L_p} = \left\{ \sum_{k=1}^{n} w_{k,j} |x^k - c_j^k|^p \right\}^{1/p}, \tag{9}$$

where the tunable parameters are $w_{k,j}$.

The following aggregation function has been used:

$$F\big(\mu(x,c_j),\nu(x,c_j)\big) = \begin{cases} \mu(x,c_j), \text{ for } \nu(x,c_j) \le K_\nu \\ \mu(x,c_j)\big(1 - \nu(x,c_j)\big), \text{ otherwise} \end{cases} \tag{10}$$

where K_ν is a parameter. Using the above formula we can obtain radial basis functions of quite exotic shapes (the one used here is shown in Fig. 2). The rationale behind the above equation is that when the inhibitory rate is greater than a certain threshold the resultant activation potential decreases proportionally. Attaching $2n + 1$ adjustable parameters to each n-dimensional prototype allows us to form attraction regions that differ both in size and in shape. This may lead us to approximate decision regions by storing fewer prototypes compared with other RBF networks.

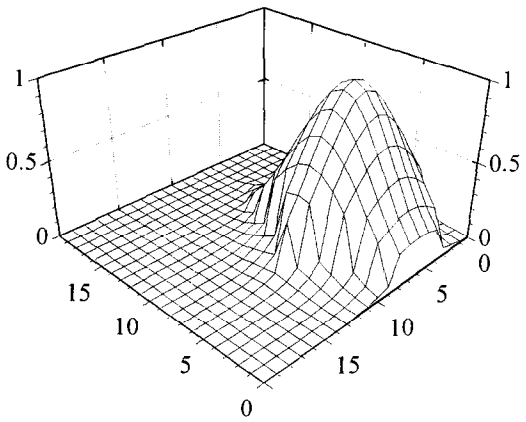

Fig. 2. The function sample: $c = [3,3]^T$, $w_\mu = [0.20,0.35]^T$; $w_\nu = [0.40,0.10]^T$, $K_\nu = 0.67$.

Let $W = \{w_{ij}^{\mu}, w_{ij}^{\nu}, K_\nu\}$, $i = 1,\ldots,n$, $j = 1,\ldots,L$ be the set of trainable parameters. Let $J_{\mathrm{RBF}}(W)$ be the criterion function for training the RBF network. As in the GA selection, it is reasonable to choose as a criterion a certain measure of the classification accuracy, so we chose the counting estimate of the probability of correct classification (analogues to Eq. (2)). The simulated annealing (SA) [9] has been adopted as the training algorithm because: it does not require any derivatives of the criterion function; being a random-type search it is supposed to avoid local extrema; and since a finite range for each parameter is used, the algorithm prevents unlimited growth of the parameter values.

Briefly, the training algorithm operates as follows:

1. Generate randomly a parameter vector W (each parameter takes a value in the respective range). Calculate $J_{\mathrm{RBF}}(W)$. Fix the initial value of temperature $T = T_{\mathrm{ini}} > 0$. Set the iteration counter t to 0.
2. Conduct l_T trials at the current T:
   (a) generate a neighbor state W' of W according to the current temperature spanning limitation;
   (b) calculate $J_{\mathrm{RBF}}(W')$;
   (c) calculate $\delta = J_{\mathrm{RBF}}(W) - J_{\mathrm{RBF}}(W')$;
   (d) if $\delta < 0$ then $W := W'$, else if $random(0,1) < \exp^{(-\delta/T)}$ then $W := W'$.
3. $t := t + 1$; $T := \eta T$.

The procedure stops when a predefined number of iterations t_{MAX} is reached. In some versions of the SA algorithm the maximal value of the criterion is stored along with the parameters of the network with which this value has been reached. The output of the algorithm is set back to this point in the search space instead of the last point (where the system has frozen). Some previous experiments showed that the algorithm converged very slowly in problems like those considered in this study because the criterion surface seems to be quite irregular and multimodal. Therefore, we applied a modification of the algorithm which brings it closer to a kind of a random descent search. At the end of each step 2, the current point is set back to the point with highest criterion value encountered so far.

The set of SA parameters forms its *cooling schedule*. The cooling schedules used in each experimental setting are presented together with the experimental results in Section 5.

5. Experimental results and discussion

It has been found that the plain random choice of prototypes is a weak competitor to the GA [15]. In this study we compare GA selection with a *heavy* random search and with clustering. The *heavy* random search means that the random selection is given the same chance to choose a prototype set as the GA. In the latter case the final set has been subsequently evolved, while in the former one the same number of sets have been assessed (by the fitness value $\mathcal{F}(S)$, Eq. (1)) but seeded purely at random.

As it was mentioned earlier, the RBF network configuration adopted here requires that the nodes at the hidden layer are labeled. Therefore, we clustered the data separately, one class at a time, and pooled together the cluster centers as the prototype set.

The following two data sets have been used:

(A) The famous IRIS data. The set comprises of 150 four-dimensional vectors from three classes: Sestosa, Virginica, and Versicolor. Purposefully we have chosen perhaps the worst projection plane of the IRIS data: using the features x_1 (Sepal length) and x_2 (Sepal width). The data set is shown in Fig. 3. We used 75 vectors for training and the reminder (75) for testing, with crossvalidation. There are so many intriguing experimen-

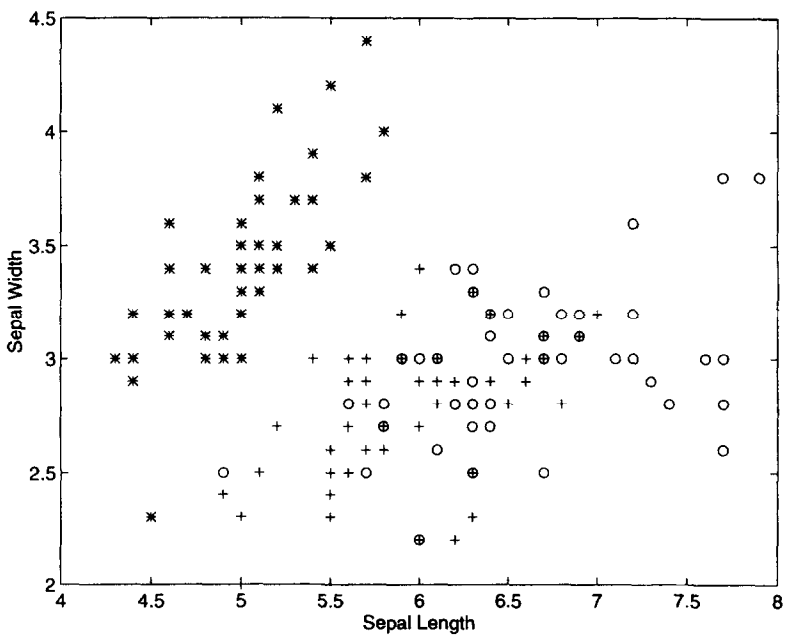

Fig. 3. 2-D projection of the IRIS data set.

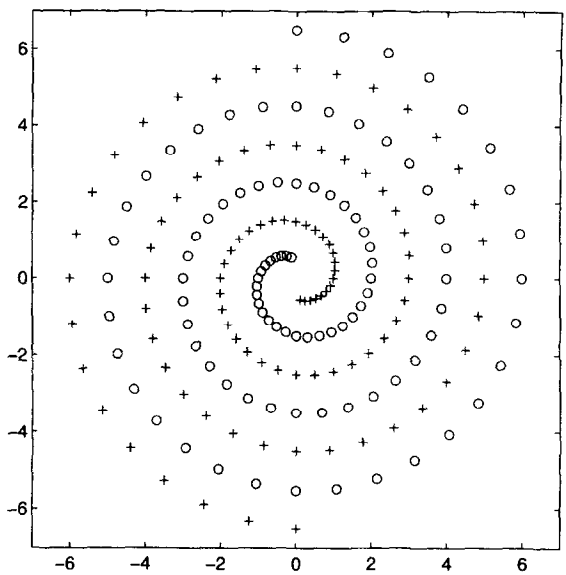

Fig. 4. Two-spirals training set.

tal results reported on this set that, venturing to compare our results with some of them, many others will be inevitably neglected. Yet another reason why the comparison might not be fair is that the splitting of the sample into two parts always bears the risk that those two parts differ significantly in their probabilistic characteristics, thus making the training and the test results biased. This concerns especially the small sample size problems with highly overlapping classes. Therefore, current results are only reported and not contrasted with others on the IRIS data set.

(B) The two-spirals data (Fig. 4). This set consists of two intertwined spirals, each one containing 96 points assigned the same class label. A test set with virtually the same structure is also available. The problem is meant to be a benchmark for neural networks because of the complex highly-nonlinear discrimination boundary [25]. What makes this set appealing for trying out neural networks on it is that the highest classification accuracy is known (100%), which is an advantage of most artificial data sets, and therefore the accuracy of the network can be a measure of its capability on this type of data. Second, knowing exactly the structure we can avoid using beneficial network configurations (e.g., nodes with appropriately designed transition functions) corrupting thereby the comparison.

We carried out two series of experiments with the IRIS data set, setting the limit of the prototype set to 12 and 15, respectively. The GA templates were as follows:

- 12 prototypes:
  - generation number = 45;
  - population size $ps = 10$;
  - initial seeding = 20%;
  - $\alpha = 0.005$;

  - *limit* = 12;
  - mutation rate = 0.03.
- 15 prototypes:
  - generation number = 25;
  - population size $ps = 30$;
  - initial seeding = 20%;
  - $\alpha = 0.005$;
  - *limit* = 15;
  - mutation rate = 0.015.

In both experimental settings, as well as in the experiments with the heavy random search and with clustering, the same cooling schedule has been applied:

- $\eta = 0.7$;
- $l_T = 5$;
- $T_{\text{ini}} = 30°$;
- $t_{MAX} = 15$.

It can be seen that the above cooling scheme does not imply extensive training. This has been done because it appeared that the accuracy on the test set does not show the desired improvement with respect to the training iterations (this fact is commented further on). We used the whole training set both for prototype selection and for training of the RBF network because of the comparatively small cardinality. Should we split further the training set into "training" and "validation" parts (as suggested in [22]), we will run to the peril to overtrain the network even more, or to deprive it of the ability to train at all. This will be caused by the biased estimates both of the training criterion value and the classification accuracy on the validation set.

Five sets for each training/test partition have been selected by the GA and by the heavy random search. The GA has been rerunning five times for each template, and the

Table 1
Averaged classification accuracy and the standard deviations with the IRIS data set (in %)

Prototypes	Test results	
	1-NN	RBF
11.2 (heavy random search)	75.20 (±2.54)	76.610 (±6.36)
12.0 (clustering)	74.00	74.950 (±4.69)
11.5 (GA selection)	76.27 (±5.57)	77.125 (±4.86)
13.9 (heavy random search)	73.81 (±2.55)	73.410 (±5.06)
15.0 (clustering)	75.30	74.120 (±3.34)
14.5 (GA selection)	75.74 (±1.86)	76.005 (±4.90)
The whole sample	68.65	

best chromosome has been nominated as the prototype set. The heavy random search has been done by five runs of 450 and 750 criterion evaluations, respectively, and the winning subset has been taken. On each prototype set two independent runs of the SA training have been performed, starting from different initialization of the parameters. Table 1 shows the average results on the test sets of the experiments with the GA, heavy random search, and clustering. For comparison, the results with the 1-NN classifier are also shown.

The results show that:

- All selected subsets surpass the whole sets with respect to the classification accuracy. This means that both data sets (used in turn for training and for test) contain large numbers of "redundant" objects.
- The large variance of the estimates does not allow us to draw up a firm conclusion but a slight predominance of the GA selection to the other two techniques can be seen.
- It seems that the RBF network does not lead to better accuracy than the 1-NN classifier. The average training and test rates are shown in Fig. 5 with (a) and (b) standing for the two partitionings of the IRIS data set. In fact, the behavior of the test accuracy during the training process is quite random: neither the improvement nor the deterioration can be deemed a stable trend.

There can be several reasons for the latter. First, we can think that the classification information is already "encoded" in some way in the set of prototypes, and any further exploitation of the *same* information would lead to overfitting. Second, the criterion

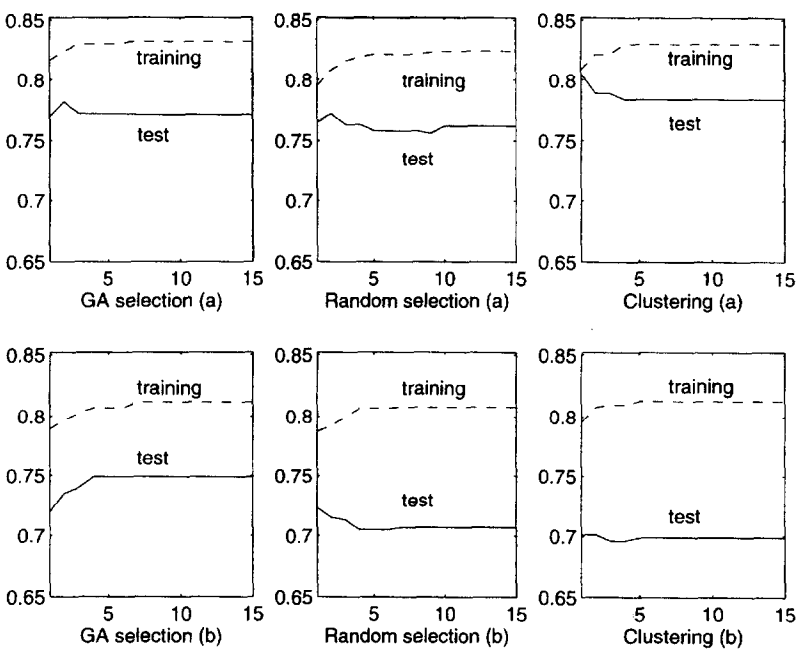

Fig. 5. Averaged classification accuracy with the IRIS data with respect to the iteration number.

$J_{RBF}(W)$ is actually discrete (the number of correctly classified objects divided by N) and when it approaches its maximum on the particular data set it becomes insensitive to small variations. Third, a consequence of the above, it is also likely that the criterion function has multiple extrema of almost equal values. It can be expected that not all of these will generalize well, especially in small sample size problems. And, finally, the high starting values of the criterion on the training set which occurred in most of the experiments does not leave much room for training. This is yet another indication that the sets used as prototypes bear lot of classification power themselves.

The two-spirals data set gives a better basis to check the current hypothesis. Several things are obvious:

1. The 1-NN method will provide excellent results because the training and the test sets are almost identical.
2. Since both classes have a string-like structure the classification rate will abruptly deteriorate with the number of k growing.
3. Any clustering algorithm that looks for compact clusters (ellipsoidal or spherical) will fail to provide a good solution.

Indeed, the test classification accuracy with 1, 3, 5, 7, 9, and 11 NN is as follows (in %): 100, 100, 99, 59.4, 51.0, and 41.7. From the above it can be concluded that the more prototypes we retain, the higher the classification rate will be. Therefore, the matter of interest is not the absolute value of the classification rate but the relative performance of the RBF initialized by means of the investigated techniques, putting the same limit (the choice here was 20) on the cardinality of the reference set. We did not perform cross-validation experiments with exchanging the training and test sets because the results would be practically the same.

The experiments conducted with the two-spirals data sets followed the same scheme as these with the IRIS data. The difference was that, as recommended in [22], we split the training set into training and validation parts, both envisaged to be used in the training phase. The training part was used to tune the RBF parameters, and the validation one, to provide an estimate of the classification accuracy on an independent data set (as a pseudo-test). We considered three stopping criteria: C_1: Stop when the limit number of iterations has been reached, C_2: Stop at the highest validation rate, and C_3: Stop at the highest sum of the training and validation rates. The idea of combining both estimates is rather old. It is known that the assessment on the independent data set is usually pessimistically biased while the assessment on the training set is optimistically biased. Since we monitor both rates we can eventually get advantage on this.

The GA template was as follows:

- generation number = 50;
- population size ps = 10;
- initial seeding = 9%;
- $\alpha = 0.07$;
- $limit = 20$;
- mutation rate = 0.02.

and the cooling schedule:

- $\eta = 0.95$;
- $l_T = 5$;

Table 2
Averaged classification accuracy and the standard deviations with the two-spirals data set (in %)

Prototypes	Test results			
	1-NN	RBF (C_1)	RBF(C_2)	RBF (C_3)
19.8 (heavy random search)	62.92	66.650	66.040	67.240
	(±2.71)	(±3.35)	(±3.92)	(±3.61)
20.0 (clustering)	62.0	63.325	64.975	64.725
		(±1.48)	(±1.45)	(±1.83)
19.4 (GA selection)	65.95	69.650	69.130	69.810
	(±2.50)	(±1.12)	(±1.98)	(±1.00)

- $T_{\text{ini}} = 30°$;
- $t_{MAX} = 70$.

The best five randomly seeded prototype sets have been taken after one run of 500 criterion evaluations.

The average results when applying the above criteria are presented in Table 2.

Fig. 6 illustrates the training and the test rate with respect to the training iterations of the RBF network, and Fig. 7 shows the training, test, and validation rates for one of the randomly selected reference sets where the advantage of the stopping criterion C_3 can be seen. The average of the training and validation rates is also depicted. Not surprisingly, due to the highly coinciding training and test sets, the average rate follows quite closely the test accuracy.

Slightly better results have been obtained with stopping criterion C_3. This criterion can be expected to perform well not only with artificial but with real data sets, as well.

Unlike the results from the experiments with the IRIS data set, here the advantage of selecting the prototypes by a GA is clear. The variance of the estimates is small, so the

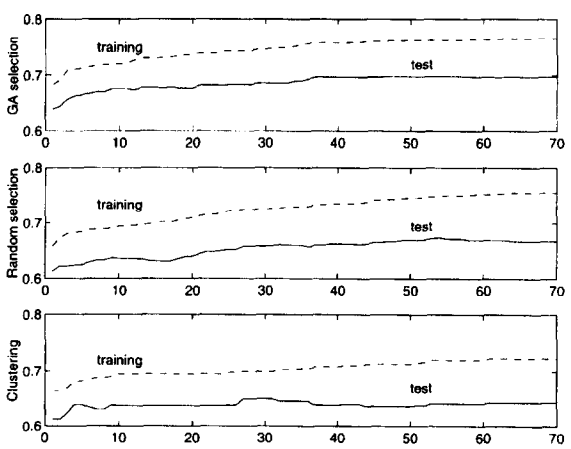

Fig. 6. Averaged classification accuracy with the two-spirals data with respect to the iteration number.

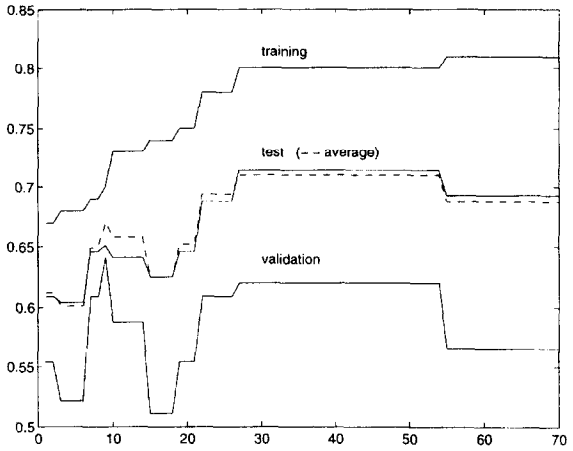

Fig. 7. Classification accuracy with a single prototype subset selected from the two-spiral data by random search.

hypothesis that the GA selection provides better prototype subsets compared to the two other techniques is empirically supported.

6. Conclusions

In this paper an application of a GA is considered for selecting prototypes for an RBF network. The network configuration and training scheme have been chosen so that the location of the seed points (the prototypes) is not changed. This makes the task of initial selection of prototype set very important. Two data sets have been used: the IRIS data and the two-spiral data set. The results from GA selection have been compared with those from a *heavy* random search and with clustering.

It turned out that the prototype sets selected from the IRIS data provided good classification results by simply using 1-NN classifier, and the RBF did not lead to an improvement. Due to the small sample size, the variance of the estimates appeared to be high, so the slight predominance of the GA selection of prototypes could not be deemed experimentally supported.

On the contrary, with the two-spirals data sets, the results clearly showed the improvement of the RBF network classification rate when the prototype set has been selected by a GA. Along with the higher classification accuracy, the variance turned out to be smaller than that with the random search and with clustering. From the three stopping criteria for the RBF network training we would recommend to use the average (or a weighted average) of the training and validation rate (criterion C_3) which can be thought less biased.

It can be expected that if the cardinality of the reference set is to be kept comparatively small and the initial set is small itself, the heavy random search performance will approach that of the GA. This is because the number of the random

trials will be large enough to eventually hit some good solution. Alternatively, the clustering performance may not improve because confining the set to fewer clusters, we may "invent" some centers that will not correspond to the real data structure. This effect may occur with the sets like the two-spirals one.

If we can afford to retain a large reference set (e.g., due to having a large training set), the clustering seems to be a better competitor of the GA than the heavy random search. In this case the GA could appear ineffective and can be run in a cascade way on disjoint subsets.

Acknowledgements

The scholarship kindly provided by the Royal Society and Foreign and Commonwealth Office, which helped in preparing this latter version of the paper, is greatly appreciated by the author.

References

[1] D. Beasley, D.R. Bull and R.R. Martin, An overview of genetic algorithms: Part 1, fundamentals, *University Computing* 15 (1993) 58–69.

[2] D. Beasley, D.R. Bull and R.R. Martin, An overview of genetic algorithms: Part 2, research topics, *University Computing* 15 (1993) 170–181.

[3] M. Brown, K.M. Bossley, D.J. Mills and C.J. Harris, High dimensional neurofuzzy systems: Overcoming the curse of dimensionality, *Proc. FUZZ/IEEE'95, Yokohama, Japan* (1995) 2139–2146.

[4] K.-B. Cho and B.H. Wang, Radial basis function based adaptive fuzzy systems, *Proc. FUZZ/IEEE'95, Yokohama, Japan* (1995) 247–252.

[5] R.L. Coultrip and R.H. Granger, Sparse random networks with LTP learning rules approximate Bayes classifiers via Parzen's method, *Neural Networks* 7 (1994) 463–476.

[6] B.V. Dasarathy, *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques* (IEEE Computer Society Press, Los Alamitos, Calofornia, 1990).

[7] C. Decaestecker, NNP: A neural net classifier using prototype, *Proc. IEEE International Conference on Neural Networks, San Francisco, California* (1993) 822–824.

[8] P.A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach* (Prentice-Hall International, London, 1982).

[9] R.W. Eglese, Simulated annealing: A tool for operational research, *European Journal of Operational Research* 46 (1990) 271–281.

[10] S. Halgamuge, W. Poechmueller and M. Glesner, An alternative approach for generation of membership functions and fuzzy rules based on radial and cubic basis function networks, *International Journal of Approximate Reasoning* (1995), in press.

[11] H. Ishibuchi, K. Nozaki, N. Yamamoto and H.Tanaka, Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms, *Fuzzy Sets and Systems* 65 (1994) 237–253.

[12] H. Ishibuchi, K. Nozaki, N. Yamamoto and H.Tanaka, Selecting fuzzy if-then rules for classification problems using genetic algorithms, *IEEE Transactions on Fuzzy Systems* 3 (1995) 260–270.

[13] A. Katz and P. Thrift, Hybrid neural network classifiers for automatic target detection, *Expert Systems* 10 (1993) 234–250.

[14] A. König, R.J. Rashhofer and M. Glesner, A novel method for the design of radial-basis-function networks and its implication for knowledge extraction, *Proc. IEEE International Conference on Neural Networks, Orlando, Florida* (1994) 1804–1809.

[15] A. Krzyzak, T. Linder and G. Lugosi, Nonparametric classification using radial basis function nets and empirical risk minimization, *Proc. 12th Int. Conf. on Pattern recognition, Jerusalem, Israel* (1994) 72–76.

[16] L. Kuncheva, Editing for the k-nearest neighbors rule by a genetic algorithm, *Pattern Recognition Letters* 16 (1995) 809–814.

[17] L. Kuncheva and L. Todorova, Prototype selection for an RBF network by a genetic algorithm, *International Conference on Soft Computing SOCO'96, Reading, UK*, to appear.

[18] R.P. Lippmann, Pattern classification using neural networks, *IEEE Communication Magazine* (1989) 47–64.

[19] F. Masulli, F. Casalino and F. Vannucci, Bayesian properties and performances of adaptive fuzzy systems in pattern recognition problems, *Proc. ICANN-94 – The European Conference on Artificial Neural Networks, Sorento, Italy* (1994) 189–192.

[20] M.C. Moed and C.-P. Lee, Design of an elliptical neural network with application to degraded character classification, *Proc. IEEE Int. Conf. on Neural Networks, San Diego, CA* (1993) 1576–1582.

[21] M. Pawlak, M.F. Yat and Fung Ng, On kernel and radial basis function techniques for classification and function recovering, *Proc. 12th International Conference on Pattern Recognition, Jerusalem, Israel* (1994) 454–456.

[22] L. Prechelt, PROBEN1 – A set of neural network benchmark problems and benchmarking rules, Technical Report 21/94, University of Karlsruhe, Germany.

[23] D.V.A. Sanchez, On the number and the distribution of RBF centers, *Neurocomputing* 7 (1995) 197–202.

[24] K. Shimojima, T. Fukuda and Y. Hasegawa, RBF-fuzzy system with GA based unsupervised/supervised learning method, *Proc. FUZZ/IEEE'95, Yokohama, Japan* (1995) 253–258.

[25] C.-T. Sun and J.-S. Jang, A neuro-fuzzy classifier and its applications, *Proc. Second IEEE International Conference on Fuzzy Systems, San Francisco, CA* (1993) 94–98.

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 form the Bulgarian Academy of Sciences. Dr. Kuncheva is with the group of Biomedical Informatics at the Central Laboratory of Biomedical Engineering, Bulgarian Academy of Sciences. At present she is a visiting as 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, with medical applications.