A NEW CONSTRUCTION ALGORITHM OF EFFICIENT RADIAL BASIS FUNCTION NEURAL NET CLASSIFIER AND ITS APPLICATION TO CODES IDENTIFICATION

F. Belloir, A. Fache and A. Billat

Laboratoire d'Automatique et de Microélectronique
Université de Reims Champagne-Ardenne, B.P. 1039, 51687 Reims, France
Tel : +33.(0)3.26.91.82.16, Email : fabien.belloir@univ-reims.fr

ABSTRACT

In this paper we present a new simple algorithm to construct Radial Basis Function (RBF) neural net based classifier. This algorithm has the major advantage to require nothing else that the training set to work (no step learning, threshold or other parameters like in other methods). Despite its simplicity, we show, on many benchmark datasets, that this algorithm provides a robust and efficient classifier. These two properties make the proposed algorithm very attractive. We also describe an application of such built RBF classifier on data obtained in a project of buried codes identification. Finally, we compare the results with other new recognition techniques like fuzzy pattern recognition.

1. INTRODUCTION

RBF networks have been extensively studied in the past **[1] [2]**. They consist of three layers, an input, a hidden and an output one. The input layer corresponds to the input vector feature space and the output layer to the pattern classes. So the whole architecture is fixed only by determining the hidden layer and the weights between the middle and the output layers. For an input vector $X=[x_1...x_n]^T \in R^n$, and with N_h middle layer neurons, the l^{th} activation function $\varphi_l(.)$ is characterised by a centre $C_l \in R^n$ and eventually a width σ_l, $l=1,...N_h$. The general equation of an output neuron j is given by : $s_j(X) = \sum_{l=1}^{N_h} w_{lj}\varphi_l(X) + b_j$ where $w_{lj} \in R$ is the weight between the hidden neuron l and the output neuron j, b_j is an eventual bias. As activation function we use here the hypergaussian $\varphi_l(X) = \exp\left\{-\frac{\|X - C_l\|^2}{2\sigma_l^2}\right\}$.

When the RBF network is used as a classifier, X is a vector of attributes to be classified and each output $s_j(X)$ represents the membership of X to the class Ω_j. So, when there are m disjoined classes, the RBF classifier contains m outputs. To assign the prototype X at a class, these outputs can be directly used by taking the one which gives the largest membership. But some other decision rules can also be used.

Different methods to construct classifiers have been presented **[3] [4]** but most of the time the algorithm complexity is important. Here, we present a very simple algorithm directly drawn from the intrinsic working of RBF net based classifier.

2. ALGORITHM PRESENTATION

2.1 Purpose

The algorithm aim is to subdivide iteratively each of the m basic classes, disjoined but not necessarily convex, in a set of convex regions called clusters.

In the RBF network, each cluster is represented by a hidden neuron and each output s_j realises the union of some of them in order to form the corresponding class Ω_j.

The proposed algorithm can be considered as a "fully self-organised one". In fact, it determines a minimal number of local units needed to represent the whole classes known from the learning set. In the same time, it places them in such a manner that the receptive field induced by each hidden neuron covers optimally, in some sense, the attribute space. Each of these receptive fields is controlled by a scale factor, the width σ of the neuron, which is automatically adjusted according to the closest classes. So, from only the learning set and after a number of iterations proportional to the number of defined neurons, the algorithm gives the size and structure of the RBF net.

Finally, it suffices to apply some least mean squares techniques to determinate the weights w_{lj}. The RBF classifier is then totally defined and can be used in a decision making task. And all this procedure is done without having to set-up any parameters. We suppose to have a learning set of N patterns X_k, $k=1$ *to* N, for which we know the class, taken among m disjoined classes Ω_j, $j=1$ *to* m. During iterations, a cluster l is characterised by its centre C_l and it is spatially limited in the feature space by an hyperball of radius proportional to the width σ_l. These clusters are placed in two different ways. In the case where they are owned by different classes, they will be disjoined. In the other case, they could overlapped themselves in order to cover the maximum space region with a small number of hyperballs. The union of the volume delimited by the hyperballs and which represents the class Ω_j is denoted R_l . The algorithm adds new cluster until each point X_k of the learning set is include in at least one cluster of its respective class. The method necessarily converges since, in the worst case where data can not be globally partitioned, there will be a cluster per each point.

2.2 Algorithm Description

Step 1 : (Initialisation) Define m centres C_l, each one is defined like the gravity centre of the points $X_i \in \Omega_j$:

$$C_l^0 = \frac{1}{Card(\Omega_j)} \sum_{X_i \in \Omega_j} X_i \text{ , } l=j=1 \text{ to } m.$$

Step 2 : (Width definition) The width σ_l of the neuron l is defined like the half distance between his centre C_l and the closest centre of an other class : $\sigma_l = \frac{1}{2}\arg\min_{C_i \notin \Omega_j} \|C_l - C_i\|$.

Step 3 : (Search of isolated point) We search the point $X_k \notin R_j$ and that is at the maximum distance from R_j : $X_k = \arg\max_{X_k \in \Omega_j} \|X_i - R_j\|$ and $\min_{C_j \in \Omega_l} \|X_i - C_j\| > \sigma_j$. If there is no such point, we go to step 4, else the point X_i creates a new centre defining the class Ω_j. A K-means clustering algorithm is applied to adjust the centres position. Then we go back to step 2.

Step 4 : (Learning) The calculation of the network weights w_{lj}, which mathematically realises a non convex union of the clusters defining the class, is made by a least square method. For a point X_k which belongs to Ω_j , the desired outputs $s_i(X_k)=\delta(i,j)$ *i=1 to m*.

2.3 Algorithm Illustration

To illustrate the working of the construction algorithm, we describe its application on a example of three classes. As the construction algorithm developed use the centre of gravity to define the first centres in the initialisation step, we chose this example to show that even if the computed centres of gravity don't belong to the good classes, the algorithm working reminds accurate.

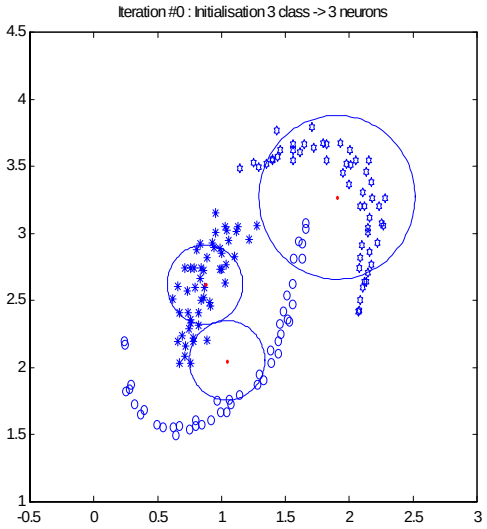

Figure 1. Representation of the three classes and of the first three neurons created with their width.

The figure 1 shows the three classes of the chosen example, the result of the algorithm initialisation step and the creation of the first three corresponding neurons with their respective width as defined in the step 2 of the algorithm.
The result of the first algorithm iteration is presented in the figure 2. The first iteration can be described as below. We search the point which is at the maximum distance of one of the three neurons, then a new neuron corresponding at the belonging class of the found point is created. The K-means clustering algorithm is applied to adjust the position of the two same class neurons centres. Finally, the width of the all four neurons is computed once again.

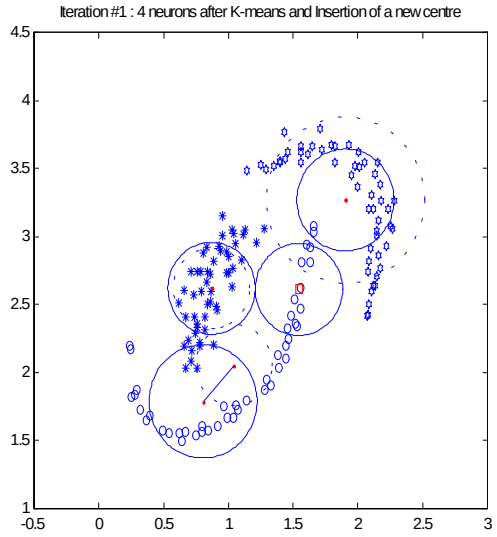

Figure 2. Representation of the algorithm first iteration result.

The final clustering is presented in the figure 3 and it is reached after 11 iterations. The number of necessary neurons is 14 to get the linear separation of the three classes.

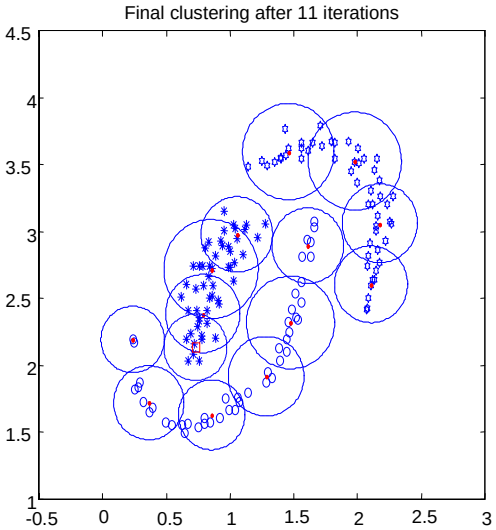

Figure 3. Representation of the final clustering.

The last step of the algorithm which represents the non convex union of each class clusters is shown in the figure 4. The border lines shown correspond to a membership value of 0,5.

For this example we obtain a perfect good classification level of 100%.

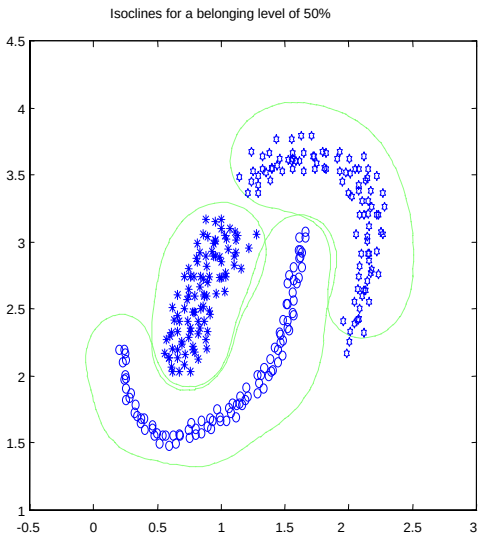

Figure 4. Representation of the border lines for a membership value of 0,5.

3. BENCHMARKS

3.1 Databases

Our objective was not to develop a specialised classifier which only works well on our data but on the contrary we wanted to obtain a general high performance classifier able to be used on many kinds of pattern recognition problems. So before using this algorithm to our application, we test it on a set of databases. The comparison has been made with the results of an European project named ELENA **[5]**. In fact, the databases used in this project cover a large range of domains. There are artificial databases like clouds, concentric and gaussian datasets in several dimensions, and also real databases like the Iris and speech recognition datasets (all the files can be downloaded at ftp.dice.ucl.ac.be/pub/neural-nets/elena/databases).

For the gaussian database we have two classes, with 2500 patterns for each class and 8 attributes, and there is a fully overlapping between them, the centre of gravity is the same for the two classes. The theoretical Bayes error for this dataset is 9%.

The samples distribution for the concentric database which has two classes and two attributes for 2500 patterns for each class, is uniform, there is no overlapping and the boundaries are non linear. Its theoretical error is 0%.

For the clouds database which has the same constitution that the concentric one, the samples distribution is gaussian. The first class is the sum of three different gaussian distributions and the second one is a single gaussian distribution. There is an important overlapping between the two classes and its theoretical error is 9,66%.

The two real databases are the Fisher's "Iris" and the "phoneme" one.

The "Iris" database is a very famous one in pattern recognition and numerous references are available. It is composed of three classes in 4 dimensions and there are 50 patterns per class.

The "phoneme" database was used in the European ROARS Esprit project and presents difficulties for classification. It is composed of two classes in 5 dimensions and there are 5404 patterns, 3818 for the first class and only 1586 for the second one.

For the comparison, it is the Holdout method averaged over five different partitions of the original database which is used. The original database is separated in two independent learnset and testset, each containing half the total available patterns (patterns used in each partition being always the same for each particular trial from one classifier to another).

3.2 Classifiers

We compare the results given by the classifier built with our algorithm with three others neural classifiers (MLP, LVQ, IRVQ) and with a reference one, the KNN.

The "K Nearest Neighbor" classifier is a very classical one . It is used here as a reference for the best estimator of the theoretical Bayes error. It can be shown that the nearest-neighbor rule will give an error rate greater than the minimum possible such as the Bayes error, and with an infinite number of samples, the error rate is never worse than twice the Bayes error.

The first neural classifier compared in our study is the Multi-Layer Perceptrons classifier. This network, combined with the backpropagation algorithm, is the most widely know inside the neural network community.

The second one is the Learning Vector Quantization classifier. It was proposed by Kohonen, it is a simple adaptive method of vector quantization. A finite number of prototypes, each one being labelled with a class identifier, are chosen in the input space.

The last neural classifier we compare to the proposed classifier is the IRVQ one. It has been developed in the framework of the ELENA project. It is a suboptimal Bayesian classifier based on radial Gaussian kernels which uses an iterative unsupervised learning method based on vector quantization to obtain a low memory kernel density estimator, while keeping sufficiently accurate estimations of probability densities.

More information about these classifiers can be obtained in **[5]**.

3.3 Results

The test used to compare the classifier results is performed, using for each classifier the optimal parameters for each particular database, except for our classifier which doesn't need any parameter to set.

The construction algorithm of RBF classifier we propose here is not always the best classifier for all the databases.

As it is shown in the figure 5, for the 5 databases, our RBF classifier obtains the best result three times and the second best result once. The only one disappointed result is for the clouds database, we obtain an error percentage of 13.6%, when the theoretical Bayes error is 9.66%, and the best classifier in this case, the IRVQ one, obtains 11,7%. The two classes of this database are too overlapped to realise an efficient union of the different clusters generated by our algorithm.

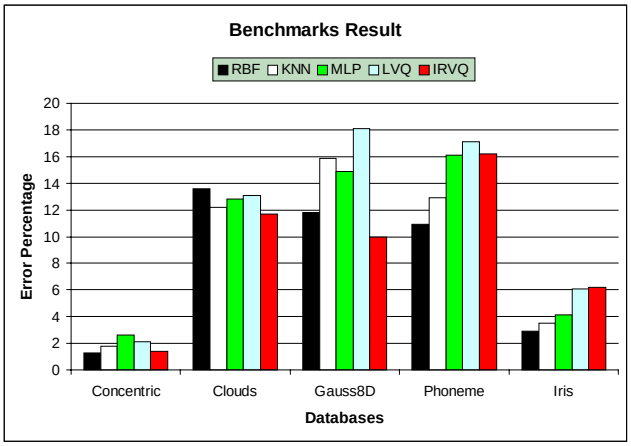

Figure 5. Representation of the classifier results for the 5 chosen databases.

We also notice that the results obtained with the Holdout method averaged over five different experiments show that the developed RBF classifier is particularly robust. As a matter of fact, the difference between the best and the worst level of error percentage is very small, less than 10% of the average value.
So all these results show that our new algorithm of RBF classifier permits to obtain a very high performance and robust general classifier which can be applied on many kinds of pattern recognition problems with very good results.

4. CODE IDENTIFICATION

The general purpose of our application is to detect and identify reliably different buried metallic codes with a smart eddy current sensor **[6]**. The data are collected by a flat coils metal locator based on the induction balance principle. This detector is connected to a mobile measurement system which controls the data sampling.
A code is built from a succession of different metal pieces separated by empty spaces. The different codes are obtained by the combination of different sizes of the metallic parts and empty spaces.
Due to the codes similarity and the non linear locator answer with the burying depth, the classification problem is not very simple. That is why we have developed intelligent methods to well solve it. Our first methods was based on the fuzzy logic theory and the Kohonen SOM. As the SOM algorithm gave disappointing results, we replace it by the new proposed RBF algorithm.
The methods based on the fuzzy logic theory are the well-known Fuzzy Pattern Matching (FPM) **[7]** and the distributed rules (DR) **[8]** developed among others by Ishibuchi.
A comparison is made between these different methods and the proposed RBF classifier.

	RBF	SOM	FPM	DR
Error percentage	6.2	11.3	8.3	7.1

Table 1. Results of code misclassification for the 4 pattern recognition methods implemented.

For a burying depth up to 80cm, we obtain the results given in the table 1. We can notice that the result of the built RBF classifier is better than the others, and always without any specialisation of the construction algorithm.

5. CONCLUSION

The use of incremental RBF networks has been already studied **[9]** but here we have presented a new simple incremental or "self-organised" RBF network algorithm which is able to be used in a lot of domains without any parameters to set. We have tried with this algorithm to translate the most simply the RBF network working.
The results show that the RBF classifier, built simply in the way we have developed, is very robust and particularly efficient in a wide range of pattern recognition problems.

6. REFERENCES

[1] Bishop C.M. "Neural Networks for Pattern Recognition", *Clarendon Press*, Oxford, 1995.

[2] Poggio T. and Girosi F. "Networks for Approximation and Learning" *Proceedings of the IEEE*, Vol. 78, pp. 1481-1497, 1990.

[3] Hwang Y.-S. and Bang S.-Y. "An Efficient Method to Construct a Radial Basis Function Neural Network Classifier" *Neural Networks*, Vol. 10, No. 8, pp. 1495-1503, 1997.

[4] Bianchini M., Frasconi P. and Gori M. "Learning Without Local Minima in Radial Basis Function Networks" *IEEE Transaction On Neural Networks*, Vol. 6, No. 3, pp. 749-756, 1995.

[5] Guerin-Dugue A. and others "Deliverable R3-B4-P Task B4 : Benchmarks" *Technical Report*, ELENA Enhanced Learning for Evolutive Neural Architecture, ESPRIT Basic Research Project Number 6891, 1995.

[6] Belloir F., Klein F. and Billat A. "Pattern Recognition Methods for Identification of Metallic Codes Detected by Eddy Current Sensor" *Signal and Image Processing (SIP'97)*, Proceedings of the IASTED International Conference, pp. 293-297, 1997.

[7] Grabisch M. and Sugeno, "A Comparison of some Methods of Fuzzy Classification on Real Data", *Proc. Of IIZUKA'92*, pp. 659-662, Iizuka, Japan, July 1992.

[8] Ishibuchi H., Nosaki K. and Tanaka H., "Selecting Fuzzy If-Then Rules for Classification Problems Using Genetic Algorithms", *IEEE Tansactions on Fuzzy Systems*, vol. 3, n°3, 1995.

[9] Fritzke B. "Transforming Hard Problems into Linearly Separable one with Incremental Radial Basis Function Networks" In M.J. Vand Der Heyden, J. Mrsic-Flögel and K. Weigel (eds), *HELNET International Workshop on Neural Networks*, Proceedings Volume I/II (1994/1995), VU University Press, 1996.