ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 399-404

A General Approach to Construct RBF Net-Based Classifier

Fabien Belloir, Antoine Fache, Alain Billat

Laboratoire d'Automatique et de Microélectronique
Université de Reims Champagne-Ardenne,
BP 1039, 51687 REIMS Cedex 2, France
Email : fabien.belloir@univ-reims.fr

Abstract : This paper describes a global approach to the construction of Radial Basis Function (RBF) neural net classifier. We used a new simple algorithm to completely define the structure of the RBF classifier. This algorithm has the major advantage to require only the training set (no step learning, threshold or other parameters as in other methods). Tests on several benchmark datasets showed, despite its simplicity, that this algorithm provides a robust and efficient classifier. The results of this built RBF classifier are compared to those obtained with three other classifiers : a classic one and two neural ones. The robustness and efficiency of this kind of RBF classifier make the proposed algorithm very attractive.

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 layer. The input layer corresponds to the input vector space and the output layer to the pattern classes. The whole architecture is therefore fixed 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 activation function $\varphi(.)$ is described by a centre $C_l \in R^n$ and 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 a possible bias. The activation function used was a 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.. Thence, the RBF classifier contains m outputs when there are m disjoined classes. These outputs can be directly used to assign the prototype X to a class, by taking the one which gives the largest membership. But other decision rules can also be used. There are several methods for constructing efficient RBF classifiers **[3] [4],** but the algorithms are usually complex. Here, we have used a very simple

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 399-404

algorithm directly drawn from the intrinsic working of the RBF net-based classifier.

2. Algorithm presentation

2.1. Principle

The algorithm is designed to iteratively subdivide each of the *m* basic classes, disjoined but not necessarily convex, into a set of convex regions called clusters. Each cluster in the RBF network is represented by a hidden neuron and an output s_j joins together some of them to form the corresponding class Ω_j. The proposed algorithm can be considered to be a "fully self-organised" one. It determines the minimal number of local units needed to represent all the classes known from the learning set. It also arranges them in such a manner that the receptive field induced by each hidden neuron optimally covers, in some sense, the attributes space. Each of these receptive fields is controlled by a scale factor, the width of the neuron, which is automatically adjusted according to the closest class. The algorithm gives the size and structure of the RBF net from only the learning set after a number of iterations that is proportional to the number of defined neurons. Lastly, a least mean squares technique was used to determine the weights w_{ij}. The RBF classifier is then totally defined and can be used in a decision making test. All this is done without having to set-up any parameters. We assume that we have a learning set of *N* patterns X_i for which we know the class, taken from m disjoined classes Ω_j, j=*1 to m*. During iterations, a cluster *j* is described by its centre C_j and it is spatially limited in the feature space by an hyperball whose radius is proportional to the width σ_j. These clusters are arranged in two ways. When they belong to different classes, they are disjoined; otherwise, they could overlap to cover the maximum space region with the smallest number of hyperballs. The union of the volume delimited by the hyperballs is R_l . The algorithm adds new clusters until each point X_i of the learning set is included in at least one cluster of its respective class. The method necessarily converges, since there will be one cluster for each point in the worst case where data can not be globally partitioned.

2.2. Algorithm description

Step 1 : Initialisation.
We define *m* centres C_j, each of which is defined as the gravity centre of the points $X_i \in \Omega_j$:

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

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 399-404

Step 2 : Width definition.
The width σ_j of the neuron j is defined as half the distance between his centre C_j and the closest centre of another class :

$$\sigma_j = \frac{1}{2} \arg \min_{C_k \notin \Omega_j} \left\| C_j - C_k \right\| .$$

Step 3 : Search for isolated point.
We look for the point $X_i \notin R_l$ that is at the maximum distance from R_l : $X_i = \arg \max_{X_i \in \Omega_l} \left\| X_i - R_l \right\|$ and $\min_{C_j \in \Omega_l} \left\| X_i - C_j \right\| > \sigma_j$. 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 used to fit the position of the centres. Then we go back to step 2.

Step 4 : Learning.
The network weights w_{lj}, which mathematically produce a non-convex union of each class clusters, are calculated by a least square method. The desired output of a point X_i which belongs to Ω_j is set to 1 while the others are set to zero.

2.3. Basic example

We illustrate here the main steps of the algorithm for solving pattern recognition problem of 3 classes from two attributes. So the RBF network has two inputs and three outputs. The learnset shown in figure 1, is composed of 100 points for each class.

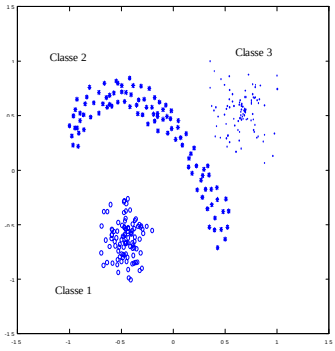

Figure 1

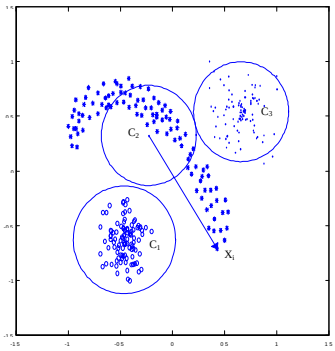

Figure 2

We can see that these classes are disjoined and the second one is non-convex. Figure 2 shows the three initial centres $\{C_1,C_2,C_3\}$ which are the gravity centres of the classes. Each induced cluster is delimited by a circle whose radius is equal to the width of the corresponding neuron. Obviously, the cluster of centre C_2 is not sufficient to represent class 2. This one will be subdivided in several subclasses. At the first algorithm iteration, the point named X_j in figure 2 is the farther from the cluster and does not

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 399-404

belong to it. Adding a new centre and the application of a k-means algorithm gives the new distribution $\{C_1,C'_2,C_3,C_4\}$ shown in figure 3. Centre C_2 has moved to C'_2 due to the minimal distance assignment principle and all the widths have been updated. The three classes have been completely discriminated after nine iterations and the nearly-built neural classifier is composed of eleven neurons, as shown in figure 4. Though class 3 is convex, it can not be represented by one cluster because of the proximity of class 2.

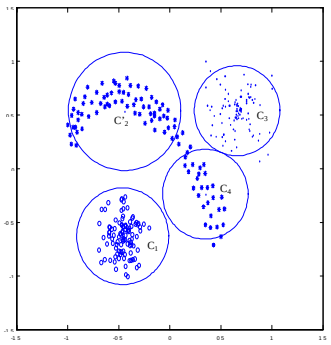

Figure 3

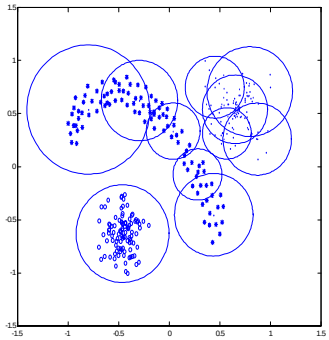

Figure 4

After determining the number and the position of the centres, the weights of the network are computed.

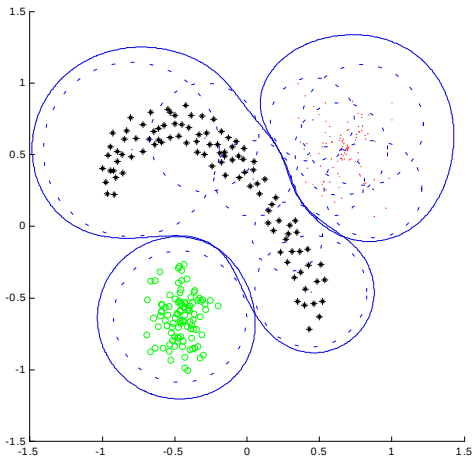

Figure 5

Figure 5 shows, in solid line, the borders computed by the network for an output $s_j=0.5$, $j=1,2,3$.

We can notice that the border of each class does not exactly correspond to the borders of the different clusters, because the weights are not unitary. The small overlap of the classes is due to the points in these areas being at equal distances between two classes. In a classical pattern recognition problem these points would be rejected as ambiguous.

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 399-404

3. Benchmarking studies

Our objective was to develop a general high performance classifier able to be used for many kinds of pattern recognition problems. We therefore tested it on a set of databases and compared the results with them of a European project called ELENA **[5]** (Enhanced Learning for Evolutive Neural Architecture). The four databases used project covered a wide range of domains, two artificial ones (clouds and concentric data), and two real ones (the Iris and phoneme, used in speech recognition data). All the files can be downloaded at ftp.dice.ucl.ac.be/pub/neural-nets/elena/databases.
We compared the results given by the classifier built with our algorithm with those obtained with three other well-known classifiers (KNN, MLP and LVQ). The "K nearest neighbor" classifier is classical and is used as a reference. The MLP, acronym for Multi-Layer Perceptrons, network, combined with the backpropagation algorithm, is widely known within the neural networks community. The Learning Vector Quantization model, proposed by Kohonen, is a simple adaptive method of vector quantization.
The test used to compare the classifier results is performed, using for each classifier the optimal parameters for each particular database. It is the Holdout method averaged over five different partitions of the original database in two independent learnset and testset, each containing half the total available patterns (patterns used in each partition of the original database in a learnset and a testset being always the same for each particular trial from one classifier to another).

Database	This classifier	KNN	MLP	LVQ
Concentric	0.8	1.1	2.1	1.6
	1.8	2.3	3.6	2.4
	1.3	1.8	2.9	2.1
Clouds	12.2	11.6	11.8	12.1
	14.2	13.2	14.1	14.4
	13.6	12.2	12.8	13.1
Phoneme	10.8	12.1	13.8	16.1
	11.1	13.5	17.6	17.9
	10.9	12.9	16.1	17.1
Iris	1.3	2.5	1.1	1.2
	4.1	5.1	6.7	10.3
	2.9	3.5	4.1	6.1

Table 1 : Min, max, mean HO errors in percent of the classifiers for the databases

This kind of test allows several types of analysis. For each classifier the difference between the minimum and maximum value of the error obtained by several Holdout tests may be taken as a good image of the classifier robustness to learnset modifications. On the other hand, it is always possible to compute the 95% confidence interval for the maximum and minimum errors, and then decide if the difference between the performances of classifiers is reliable.

We show that the proposed algorithm gives very good results in term of percent classification error and also in robustness. The performances are at least equal to the best of the other neural classifiers for each database. Its robustness and its efficiency for different kinds of pattern recognition problems are always very good, and that without any parameters to setup to improve its working.

4. Conclusion

Incremental RBF networks have been previously studied **[6]** 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 having to setup any parameters. We show that the classifier built with this algorithm gives impressive results for a variety of different databases. The robustness and efficiency of this RBF classifier are equal, and often better than, those of other neural network classifiers. We have tried with this algorithm to translate the RBF network working in the most simple fashion.

Bibliography

[1] Moody J. and Darken C.J. "Fast Learning in Networks of Locally-Tuned Processing Units" *Neural Computation 1*, pp. 281-294, 1989.

[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] Fritzke B. "Transforming Hard Problems into Linearly Seprable 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.