A Novel Algorithm to Configure RBF Networks

Insoo Sohn* and Nirwan Ansari
Center for Communications and Signal Processing Research
Department of Electrical and Computer Engineering
New Jersey Institute of Technology
University Heights
Newark, New Jersey 07102
U.S.A.

Abstract

The most important factor in configuring an optimum radial basis function (RBF) network is the appropriate selection of the number of neural units in the hidden layer. This paper proposes a novel algorithm called the scattering-based clustering (SBC) algorithm, in which the FSCL algorithm is first applied to let the neural units converge. Scatter matrices of the clustered data are then used to compute the sphericity *for each k, where k is the number of clusters. The optimum number of neural units to be used in the hidden layer is then obtained. A comparative study is done between the SBC algorithm and RPCL algorithm, and the result shows that the SBC algorithm outperforms other algorithms such as CL, FSCL, and RPCL.*

1. Introduction

The most important consideration in configuring an RBF network is the determination of the number and centers of the hidden units. An obvious, trivial choice is to have each of the data correspond to a center, but this is not practical for a large amount of data. Much research has been done on the training of RBFs. Broomhead and Lowe [2] were among the first, using the k-means algorithm to minimize the number of centers. Other learning methods proposed include the genetic algorithm [8], the orthogonal least squares algorithm [6], and the competitive learning (CL) [3], which is an adaptive version of the k-means algorithm.

CL suffers from producing "dead-units," and an improvement over CL, frequency sensitive competitive learning (FSCL) [5], alleviates this problem, but it is ineffective when the number of clusters is not known *a priori*. Rival penalized competitive learning (RPCL) [3] improves FSCL by introducing a "rival penalizing force," but it is ineffective when the number of initial center units is less than the actual number of clusters with other minor problems. To determine the optimum number of neural units, a novel algorithm, scattering-based clustering (SBC), is proposed in this paper. FSCL is adopted to address the problem of under-utilized units and the characteristics of scatter matrices are utilized to adaptively determine the optimal number of neural units.

2. The Radial Basis Function network

An RBF network can be considered as a mapping $F : R^n \rightarrow R$ according to

$$F(\mathbf{x}) = w_0 + \sum_{i=1}^{k} w_i \varphi(||\mathbf{x} - \mathbf{c}_i||), \tag{1}$$

where k is the total number of RBFs, w_i are the weights of the output layer, $\varphi(\cdot)$ is the basis function, and c_i's are the centers of RBFs.

The weights of the output layer can be easily obtained by using either the pseudo-inverse method or the least mean square (LMS) algorithm if the training set of input $\mathbf{x}$ and the corresponding desired output $\mathbf{d}$ are provided. Different basis functions $\varphi(\cdot)$ can be adopted [7]. The most frequently used basis is the Gaussian function

$$\varphi(x) = \exp(-x^2/2\sigma^2). \tag{2}$$

*I. Sohn is now with the Department of Electrical Engineering, Southern Methodist University, Dallas, TX 75275.

0-7803-4122-8/97 $10.00

3. Rival Penalized Competitive Learning

The essential idea behind the RPCL algorithm [3] is to equalize the average rate of winning for each region, and it is implemented by letting the second winner of the competition respond to the input vectors in addition to the first winner. The second winner is "unlearned" by a smaller learning rate, creating a rival penalizing force. The RPCL algorithm can be summarized as follows:

Step 1: Randomly choose a sample input vector $\mathbf{x}$ among input data points, and for $i = 1, \ldots, k$, where k is the number of clusters. Determine the winner:

$$h_i = \begin{cases} 1 & \text{if } \alpha_i \|\mathbf{x} - \mathbf{c}_i(n)\|^2 \leq \alpha_j \|\mathbf{x} - \mathbf{c}_j(n)\|^2 \ \forall j \neq i \\ -1 & \text{if } \alpha_l \|\mathbf{x} - \mathbf{c}_l(n)\|^2 \leq \alpha_j \|\mathbf{x} - \mathbf{c}_j(n)\|^2 \ \forall j \neq i, l \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where α_i is the total number of times that the current first winner $\mathbf{c}_i(n)$ has been the first winner, and α_l is the total number of times the current second winner $\mathbf{c}_l(n)$ has been the first winner.

Step 2: The first winner center vector $\mathbf{c}_i(n)$ and the second winner center vector $\mathbf{c}_l(n)$ are updated according to

$$\mathbf{c}_i(n+1) = \mathbf{c}_i(n) + \varepsilon(\mathbf{x} - \mathbf{c}_i(n))h_i \tag{4}$$

$$\mathbf{c}_l(n+1) = \mathbf{c}_l(n) + r(\mathbf{x} - \mathbf{c}_l(n))h_i, \tag{5}$$

where $0 \leq \varepsilon, r \leq 1$ are the learning and "unlearning" rates, which can also be dynamically reduced to zero.

By unlearning the second winner, a rival penalizing force is created, which pushes away the second winner, thus guaranteeing the first winner's convergence. In order to demonstrate the inadequacy of RPCL, two cases have been evaluated: 1) the initial number of neural units is larger than the actual number of clusters, and 2) the initial number of neural units is smaller than the actual number of clusters. To investigate the performance of RPCL, five clusters of data shown in Fig. 1 (a) are used in these simulations. Fig. 1 (b), (c), and (d) illustrate case 1, where initial $k = 6$. According to the original paper [3], $r \ll \varepsilon$ is suggested. Thus, initially, r is set to 0.0001 and ε=0.05. The simulation result shown in Fig. 1 (b) shows a disturbing unit at (1.3, 1.3); from r=0.0001 being too small, there is almost no rival penalizing force. Thus, r is increased to 0.005 with ε remaining the same. Fig. 1 (c) shows two center units that are pushed away by the rival penalizing force. Although extra units being pushed away is desired, the desired number of extra units pushed away is one, since the number of clusters is five. This simulation result reveals that r being too big creates too much of a rival penalizing force, as opposed to too little in the previous case. By solely observing results obtained in Fig. 1 (c), one would probably think that the optimal number of hidden units is four, since ideally the rival penalizing force only pushes away the extra center units. Thus, using this invalid result would produce a sub-optimum RBF network. After numerous trials and errors, a viable learning rate of r=0.001 is obtained. Fig. 1 (d) shows that the extra center unit initialized at (3.1, 3.3) is pushed away from data patterns converging around (2.0, 2.1). All other initial center units converge toward the desired centers of data clusters. Thus RPCL does work with a number of initial centers larger than the actual number of clusters, but it is too sensitive to the value of the "unlearning" rate r, resulting in the wrong optimum number of hidden units for the RBF network. Also, some type of post-processing is needed to determine the extra units from the actual centers. Fig. 1 (e) illustrates case 2 where initial $k = 4$, showing that the center units initialized at (3.1, 3.5), (3.3, 3.5), and (3.5, 3.5) do converge to the cluster centers, but the center that was initialized at (3.1, 3.1) becomes a disturbing unit oscillating between the two clusters because there are not enough center units to represent all five data clusters.

As illustrated in the simulations, the RPCL algorithm does work when the number of initial center units is larger than the actual number of data clusters by pushing away extra center units with a rival penalizing force, but it is very sensitive to the value of r. The key problem of obtaining an optimum number of center units to train RBF networks still remains, and thus the SBC algorithm is introduced.

4. Scattering-Based Clustering algorithm

Different clustering criteria functions such as a *squared error* criterion, a *related minimum variance* criterion, and a *scattering* criterion [1], have been used for clustering data. The scattering criterion seems to possess the intrinsic properties for characterizing clusters. The following notations, definitions, and analysis are provided to develop the algorithm.

Definition 1 - The j th d-dimensional pattern vector in the K th cluster:

$$\mathbf{x}_j^{(k)} = [x_{j1}^{(k)} \ldots x_{jd}^{(k)}]^T. \tag{6}$$

Definition 2 - The d-dimensional mean vector in the K th cluster:

$$\mathbf{m}^{(k)} = [m_1^{(k)} \ldots m_d^{(k)}]^T, \tag{7}$$

where

$$m_i^{(k)} = \frac{1}{n_K} \sum_{j=1}^{n_K} x_{ji}^{(k)}, \tag{8}$$

and n_K = the number of patterns in the K th cluster.

Definition 3 - Total mean vector:

$$\mathbf{M} = (\frac{1}{n}) \sum_{k=1}^{K} \sum_{j=1}^{n_K} \mathbf{x}_j^{(k)}, \tag{9}$$

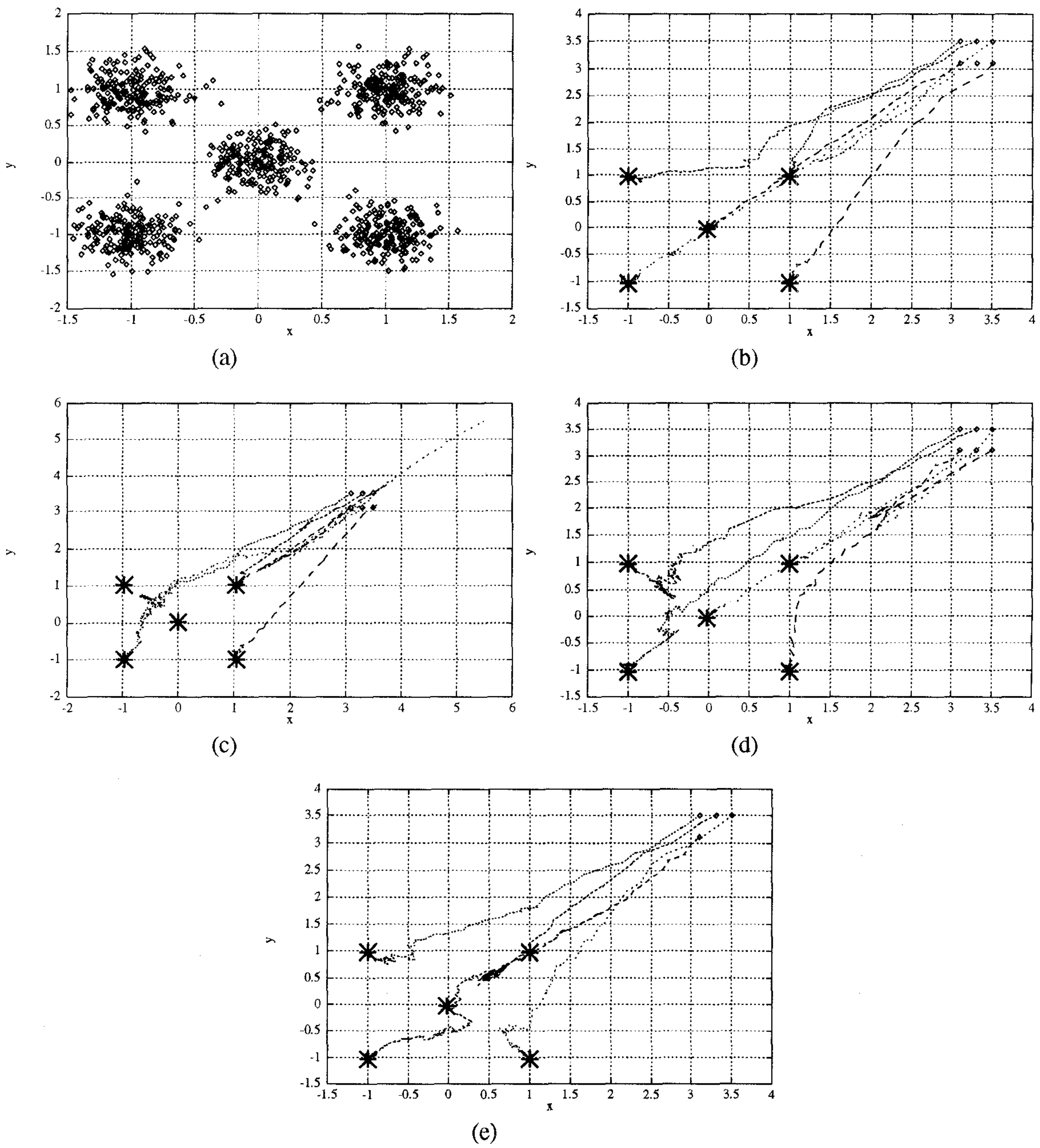

Figure 1. Results obtained by using RPCL on five clusters of data: (a) Five clusters of data used in this simulation centered at (-1.0,-1.0), (1.0,-1.0), (-1.0,1.0), (1.0,1.0), and (0.0,0.0); (b) The learning trace obtained by using RPCL with six initial centers at (3.1,3.5), (3.3,3.5), (3.5,3.5), (3.1,3.1), (3.3,3.1), and (3.5,3.1), and r = 0.0001; (c) The learning trace obtained by using RPCL with the same six initial centers and r = 0.005; (d) The learning trace obtained by using RPCL with the same six initial centers and r = 0.001; (e) The learning trace obtained by using RPCL with four initial centers at (3.1,3.5), (3.3,3.5), (3.5,3.5), and (3.1,3.1).

where

$$n = \sum_{k=1}^{K} n_K. \quad (10)$$

Definition 4 - Total scatter matrix and its trace:

$$S = \sum_{k=1}^{K} \sum_{j=1}^{n_K} (\mathbf{x}_j^{(k)} - \mathbf{M})(\mathbf{x}_j^{(k)} - \mathbf{M})^T \quad (11)$$

$$Tr(S) = \sum_{k=1}^{K} \sum_{j=1}^{n_K} (\mathbf{x}_j^{(k)} - \mathbf{M})^T (\mathbf{x}_j^{(k)} - \mathbf{M}). \quad (12)$$

Definition 5 - Total within scatter matrix and its trace:

$$S_W = \sum_{k=1}^{K} \sum_{j=1}^{n_K} (\mathbf{x}_j^{(k)} - \mathbf{m}^{(k)})(\mathbf{x}_j^{(k)} - \mathbf{m}^{(k)})^T \quad (13)$$

$$Tr(S_W) = \sum_{k=1}^{K} \sum_{j=1}^{n_K} (\mathbf{x}_j^{(k)} - \mathbf{m}^{(k)})^T (\mathbf{x}_j^{(k)} - \mathbf{m}^{(k)}) = \sum_{k=1}^{K} e_k^2, \quad (14)$$

where e_k^2 is the mean squared error at each k.

Definition 6 - Total between scatter matrix and its trace:

$$S_B = \sum_{k=1}^{K} n_K (\mathbf{m}^{(k)} - \mathbf{M})(\mathbf{m}^{(k)} - \mathbf{M})^T \quad (15)$$

$$Tr(S_B) = \sum_{k=1}^{K} n_K (\mathbf{m}^{(k)} - \mathbf{M})^T (\mathbf{m}^{(k)} - \mathbf{M}). \quad (16)$$

Note that $S = S_W + S_B$ and thus $Tr(S) = Tr(S_W) + Tr(S_B)$.

Using the above definitions, a new criterion called *sphericity*, similar to a parameter used for measuring shape [4], is introduced below.

Definition 7 - Sphericity:

$$\gamma[S_W, S_B] = \frac{Tr(S_W)Tr(S_B)}{Tr(S_W) + Tr(S_B)} = \frac{Tr(S_W)Tr(S_B)}{Tr(S)}. \quad (17)$$

The above definitions can be used to quantify how well data are clustered, and various properties will be shown.

Proposition 1. $Tr(S_W)$ monotonically decreases with k.

Proposition 2. $Tr(S_B)$ monotonically increases with k.

Proposition 3. $\gamma[S_W, S_B]$ monotonically decreases for $k \geq 2$ when $\frac{Tr(S_B)}{Tr(S_W)} > 1$.

Using the characteristics of scatter matrices, the SBC algorithm could be summarized as follows:

Step 1: Compute $c_i(\mathbf{n})$ using the FSCL algorithm, where $i = 1 \ldots k$.

Step 2: Assign input patterns to their appropriate centers according to,

$$\mathbf{x} \in c_i \qquad \text{if} \qquad ||\mathbf{x} - c_i(n)||^2 \leq ||\mathbf{x} - c_j(n)||^2 \quad \forall j \neq i. \quad (18)$$

Step 4: Locate the "knee" of the plot of $\gamma[S_W, S_B]$ versus k, as shown in Fig. 2 (b).

Note that one does not have to guess the initial number of center units. Each k is tested incrementally and the optimum number of center units corresponds to the k that yields the minimum angle of the plot of sphericity versus k, as shown in Fig. 2 (b). To investigate the performance of SBC, it is applied to five clusters of data patterns used for the RPCL simulation shown in Fig. 1 (a). Fig. 2 (b) shows that $\gamma[S_W, S_B]$ stablizes at $k = 5$. These simulation results show that the SBC algorithm is more robust as compared to RPCL, always producing the optimum number of center units; whereas RPCL heavily depends on learning rate r and fails when the number of initial center units are smaller than the actual number of clusters.

5. Supervised classification through RBF networks

To illustrate the functionality of the RBF network trained by the RPCL and SBC, the RBF network is utilized to classify the patterns in a "noisy" XOR problem. The data patterns used for this XOR classification are centered at (-1.0, 0.0), (1.0, 0.0), (0.0, 1.0), and (0.0, -1.0). The deviation is the same as before, which is 0.2 with 100 patterns in each cluster. The two clusters centered at (0.0, 1.0) and (0.0, -1.0) form the first class and the other two clusters centered at (-1.0, 0.0) and (1.0, 0.0) form the second class. The basis function of the RBF network is the Gaussian function shown previously, in equation (2), with $\sigma = 0.2$.

To examine the RPCL algorithm on training the hidden layer of the RBF network, where one normally does not know the actual number of data clusters, the number of initial center units is set as five, ε=0.05, and $r = 0.005$, which is a reasonable value between 0 and 1. The result of the RBF network with these parameters shows a recognition rate around 52%. Therefore, the RBF network classifies the patterns correctly only half of the time. The reason for such a low rate of recognition is because the rival penalizing force is too strong, which is similar to the simulation results shown in Fig. 1 (c), where too much of the rival penalizing force pushes away more center units than needed. Thus, after decreasing r several times, the optimum value of $r = 0.002$ is obtained. The results with this optimum r are shown in Table 2, with a recognition rate around 98%. The sensitivity of r is a major hindrance toward obtaining the optimal RBF network. Table 3 shows the result with initial center units less than the actual number of clusters simulated with $k = 3$

and $r = 0.002$. The recognition rate is around 73%. The obvious reason for the low recognition rate is that there are not enough RBFs to represent the four data clusters. Table 4 shows the result of using the SBC algorithm. The optimum number of neural units for the hidden layer was found by searching for the k that has the minimum angle of sphericity, as shown in the previous simulation results, and the value of center units corresponding to that optimal k were used. The recognition rate is around 98%.

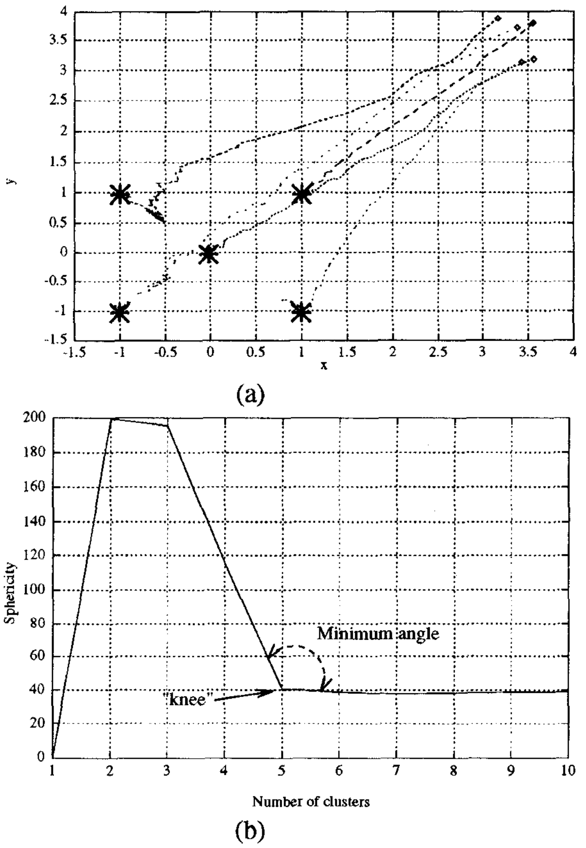

Figure 2. Results obtained by using SBC on five clusters of data: (a) The learning trace for k = 5, with initial centers at (3.6,3.2), (3.2,3.9), (3.4,3.1), (3.5,3.8), and (3.4,3.7); (b) Sphericity obtained for five clusters by using SBC for k = 1...10.

Table 1. Classification using RPCL with r = 0.005 and k = 5.

	classified as		marginally
	class 1	class 2	classified as
class 1	176	0	24
class 2	16	31	153

Table 2. Classification using RPCL with r = 0.002 and k= 5.

	classified as		marginally
	class 1	class 2	classified as
class 1	194	6	0
class 2	0	196	4

Table 3. Classification using RPCL with r = 0.005 and k = 3.

	classified as		marginally
	class 1	class 2	classified as
class 1	96	102	2
class 2	0	197	3

Table 4. Classification using SBC.

	classified as		marginally classified as
	class 1	class 2	
class 1	197	0	3
class 2	0	194	6

6. Conclusion

Although it has been shown and proved that the RBF network is faster and more flexible compared to classical multi-layered neural networks, the major problem toward using the RBF network is the appropriate selection of radial basis function centers. To address and solve this problem, a new learning method based on scatter matrices and sphericity is developed for the construction of the optimal RBF network.

From the simulation results, it has been shown that the RPCL algorithm is inadequate in training the hidden layer of the RBF network even though it is superior among competitive learning algorithms. This inadequacy is caused by sensitivity to the learning rate r, and it failed to work when the number of center units chosen was smaller than the actual one. As for the SBC algorithm, it was able to choose the optimal number of center units by selecting k with the minimum angle of the sphericity plot and the optimal value for the center units. By using the characteristics of scatter matrices, the SBC algorithm is rather robust.

References

[1] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[2] D. S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Syst.*, 2:321--355, 1988.

[3] L. Xu, A. Krzyzak, and E. Oja. Rival penalized competitive learning for clustering analysis, RBF net, and curve detection. *IEEE Trans. Neural Networks*, 4(4):636--649, 1993.

[4] N. Ansari and E. J. Delp. On the distribution of a deforming triangle. *Pattern Recognition*, 23(12):1333--1341, 1990.

[5] S. C. Ahalt, A. K. Krishnamurty, P. Chen, and D. E. Melton. Competitive learning algorithms for vector quantization. *Neural Networks*, 3:277--291, 1990.

[6] S. Chen, C. F. N. Cowan, and P. M. Grant. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Trans. Neural Networks*, 2(2):302--309, 1991.

[7] S. Haykin. *Neural Networks, A Comprehensive Foundation.* Macmillan, New York, 1994.

[8] X. Yao. A review of evolutionary artificial neural networks. *International Journal of Intelligent Systems*, 8(4):539--567, 1993.