Radial Basis Function Network Training Using a Fuzzy Clustering Scheme

G. L. Zheng and S. A. Billings
Department of Automatic Control and Systems Engineering,
University of Sheffield, Mappin Street, Sheffield S1 4DU

Training algorithms for radial basis function (**RBF**) networks usually consist of an unsupervised procedure for finding the centres and a supervised learning algorithm for updating the connection weights. Good network performance will often be dependent on the **RBF** centre locations but the k-means clustering or related methods which are often used can be sensitive to the initial conditions and this can result in local minima and a deterioration in overall network performance. In the present study, a fuzzy clustering scheme is implemented to locate the radial basis function centres in a manner which overcomes the sensitivity to initial conditions and improves overall network performance. Artificial and practical data sets are used to demonstrate the properties of the fuzzy clustering scheme.

Research Report No. 505

February 1994

List of Figures

List of Tables

the data, which suggests that clustering techniques may be used to find the centres. The most popular choice of clustering algorithm so far has been the k-means algorithm because of its simplicity. However, it is well known that the clustering results of the k-means algorithm may depend on the sequence in which the data samples are processed and may be sensitive to the initial settings of the algorithm. In the present work, we investigate how the sensitivity problem may be overcome by incorporating a fuzzy clustering scheme into the **RBF** network.

The layout of the paper is organized as follows. Section two describes briefly the radial basis function network, the network structure and a training algorithm. Section three is devoted to a brief disscussion on centre selection and clustering algorithms. A fuzzy clustering scheme is presented in section four. Experimental results are given in section five, which show how the sensitivity problem of the k-means algorithm may be overcome by the new fuzzy clustering scheme. Possible simplifications and a brief disscussion on related algorithms are given in section six. Finally, conclusions are given in section seven.

2 Radial Basis Function Networks

A basic radial basis function (**RBF**) network may be depicted as shown in **Fig 1**. Without loss of generality, in the present study the number of outputs in the network will be assumed to be one, but the architecture can be readily extended to cope with multi-output problems. The architecture consists of an input layer, a hidden layer and an output layer. The input vector to the network is passed to the hidden layer nodes via unit connection weights. The hidden layer consists of a set of radial basis functions. Associated with each hidden layer node is a parameter vector $\mathbf{c_j}$ called a centre. The hidden layer node calculates the Euclidean distance between the centre and the network input vector and then passes the result to a radial basis function. All the radial basis functions in the hidden layer nodes are usually of the same type. Typical choices of the radial basis functions are

i). the thin-plate-spline function:

$$\phi(\mathbf{v}) = \mathbf{v}^2 \times \log(\mathbf{v}) \tag{1}$$

ii). the Gaussian function:

$$\phi(\mathbf{v}) = \epsilon^{-\left(\frac{\mathbf{v}^2}{\beta^2}\right)} \tag{2}$$

iii). the multiquadric function:

$$\phi(\mathbf{v}) = (\mathbf{v}^2 + \beta^2)^{\frac{1}{2}} \tag{3}$$

vi). the inverse multiquadric function:

$$\phi(\mathbf{v}) = \frac{1}{(\mathbf{v}^2 + \beta^2)^{\frac{1}{2}}} \tag{4}$$

where $\mathbf{v}$ is a non-negative number and is the distance from the input vector $\mathbf{x}$ to the radial basis function centre $\mathbf{c}$, and β is the width of the radial basis functions. In radial basis function networks, the thin-plate-spline function has been used by Chen *et. al.* [7], [8], and

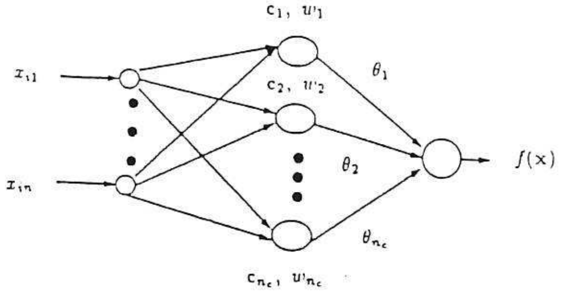

Figure 1: A radial basis function network architucture

the Gaussian and multiquadric functions have been used by Moody [6], Broomhead [9] and Poggio [5].

The response of the output layer node may be considered as a map f: $\mathbf{R}^n \rightarrow \mathbf{R}$, that is

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{N} \theta_i \phi(\| \mathbf{x} - \mathbf{c}_i \|) + \theta_0 \tag{5}$$

where N is the number of training data and $\| \bullet \|$ denotes the Euclidean norm. $\mathbf{c}_i$ is the data sample (i=1, 2, ..., N). $\mathbf{x}$, $\mathbf{c}_i \in \mathbf{R}^n$, θ_i $(i = 1, 2, ..., N)$ are the weights associated with the i^{th} radial basis function centre. θ_0 is a constant term which acts as a shift in the output level. It may be seen that the training of the network is an interpolation problem and the solution may be obtained by solving a set of constrained linear equations. The complexity increases with the number of training data, which may make the implementation of the network above unrealistic. In practical applications, it is often desirable to use a network with a finite number of basis functions. A natural approximated solution would be

$$\mathbf{f}^*(\mathbf{x}) = \sum_{i=1}^{n_c} \theta_i \phi(\| \mathbf{x} - \mathbf{c}_i \|) + \theta_0 \tag{6}$$

where n_c is the number of radial basis function centres. Given a set of data $(\mathbf{x}_i, \mathbf{y}_i)$, $(i = 1, 2, ..., N)$ $\mathbf{x}_i \in \mathbf{R}^n$, $\mathbf{y}_i \in \mathbf{R}$, $\mathbf{x}_i = (x_{i1}, \ x_{i2}, ..., \ x_{in})^T$, the connection weights, centres and widths may be obtained by minimizing the following cost function

$$\mathbf{J} = \sum_{i=1}^{N} (\mathbf{y}_i - \mathbf{f}^*)^T (\mathbf{y}_i - \mathbf{f}^*) \tag{7}$$

The above minimization problem may be solved using a nonlinear optimization or gradient decent algorithm. However, such an algorithm will give similar results as the back propagation algorithm. Thus advantages of the radial basis function networks mentioned above will be lost. Therefore, most learning algorithms developed so far are divided into two stages

i. Learn the centres and widths in the hidden layer;

ii. Learn the connection weights from the hidden layer to the output layer.

The centres and widths are typically obtained by an unsupervised algorithm. For example, Moody *et al* [6] suggested a k-means clustering procedure as an updating rule for the centres and a p-nearest neighbour rule for the widths, and Vogt [10] used a **L**earning **V**ector **Q**uantization (**LVQ**) procedure to locate the centres. The k-means clustering was also used by Chen *et al* [7] in a hybrid training algorithm. Since the cost function $\mathbf{J}$ in (7) is quadratic in the connection weights θ_i ($i = 0, 1, ..., n_c$), after the centres $\mathbf{c}_i$ ($i = 1, 2, ..., n_c$) and the width β have been chosen, the connection weights can be obtained using the least squares algorithm. In the present work, a recursive least squares (**RLS**) method using a Givens transformation will be used to compute the connection weights.

3 Centre Selection and Clustering

Suppose that the centre locations are to be determined using the natural definition of optimiality, then the following condition must be satisfied by the centres $\mathbf{c}_i$ ($i = 1, 2, ..., n_c$)

$$\frac{\partial \mathbf{J}}{\partial \mathbf{c}_i} = 0 \quad i = 1, 2, ..., n_c \tag{8}$$

Using the gradient-decent method, this yields

$$\frac{\partial \mathbf{J}}{\partial \mathbf{c}_i} = 2\theta_i \sum_j^N (y_j - f^\star)\,(\mathbf{x}_j - \mathbf{c}_i)\,\left(\frac{1}{\mathbf{v}}\frac{\partial \phi}{\partial \mathbf{v}}\right)_{\mathbf{v}=||\mathbf{x}_j - \mathbf{c}_i||} \tag{9}$$

Assume that the training errors $(y_j - f^\star)$ are constants, yields

$$\mathbf{c}_i = \frac{\sum_j^N (y_j - f^\star)\left(\frac{1}{\mathbf{v}}\frac{\partial \phi}{\partial \mathbf{v}}\right)_{\mathbf{v}=||\mathbf{x}_j - \mathbf{c}_i||}\mathbf{x}_j}{\sum_j^N (y_j - f^\star)\left(\frac{1}{\mathbf{v}}\frac{\partial \phi}{\partial \mathbf{v}}\right)_{\mathbf{v}=||\mathbf{x}_j - \mathbf{c}_i||}} \tag{10}$$

It may be clear that the above condition means that the centres move towards the majority of the training data. The optimal centres are a weighted sum of the data samples. The weight of the data sample $\mathbf{x}_j$ for a given centre $\mathbf{c}_i$ is proportional to the interpolation error at the data sample, the rate of change of the radial basis function on that centre in the neighbourhood of the data and is inversely proportional to the distance from the data to the centre. This suggests that a clustering algorithm may be used to locate the centres. Many clustering algorithms may serve this purpose. For example, Moody [6] used a sequential version of the k-means algorithm for centre clustering. The k-means clustering is one of the optimization clustering methods, which minimize or maximize a certain clustering criterion. A general expression for the number of distinct partitions of N objects into n_c non-empty groups is given as [11]

$$\mathcal{N}(N, n_c) = \frac{1}{n_c!}\sum_{i=1}^{n_c}(-1)^{n_c - i}\binom{n_c}{i} i^N \tag{11}$$

It may be seen that it is impossible to consider every possible partition of N object into n_c groups when N is large. For example, the number of partitions of 100 objects into 5 groups would be

$$\mathcal{N}(100, 5) = 10^{68}$$

Therefore, nearly all optimization clustering algorithms search for the optimum value of a clustering criterion by arranging existing partitions and keeping the new one only if it provides an improvement in the criterion. The essential steps in these algorithms are

a. Find some initial partition of the objects into the required number of groups.

b. Calculate the change in the clustering criterion produced by moving each object from its own to another cluster.

c. Make the change which leads to the greatest improvement in the value of the clustering criterion.

d. Repeat steps **b** and **c** until no move of a single object causes the clustering criterion to improve.

The initial partition might be given in the following ways

i. Specified on the basis of prior knowledge.

ii. Chosen at random.

iii. n_c points might be selected in some way to act as initial centres.

One of the major disadvantages of the optimization clustering method is that different initial partitions might lead to different local optimum of the clustering criterion since the method is essentially a descent algorithm. In some cases, the results from an optimization method can be largely affected by the choice of the initial partition. This may be appreciated from the experimental examples with the sequential k-means method given later. A similar clustering scheme, the **L**earning **V**ector **Q**uantization (**LVQ**) method was used in reference [10] for locating the **RBF** centres. In this reference, the class membership of the training data was also taken into consideration. When the nearest centre is of the same class as the training data, the centre is moved in the direction of the data, otherwise, it is moved in the opposite direction to the data. It may be correct to say that the **LVQ** scheme is also sensitive to the initial positions of the centres. It is obvious that the output of the radial basis function network depends on the centre locations. If the clustering result is sensitive to the initial locations of the centres, the output of the network will be sensitive to the initial centre locations as well and for some initial settings unsatisfactory results may be obtained. Therefore, clustering algorithms which are more robust to initial settings are required. In the following section, a self-organizing scheme which is insensitive to the initial centre locations will be described in detail.

4 Fuzzy Clustering Scheme

The clustering scheme to be described in this section is a modified version of the Self-**O**rganizing feature **M**ap (**SOM**) of Kohonen [12]. The **SOM** places a number of reference or codebook vectors into a high-dimension input data space to appproximate the data set in an ordered fashion. The algorithm can effectively be used to visualize metric ordering relations

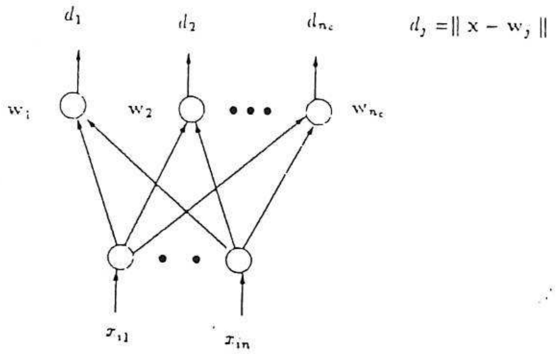

Figure 2: A basic model of the self-organizing map

within the input data. The basic model of the self-organizing map consists of two layers as shown in **Fig 2**. The first layer contains the input nodes and the second layer contains the output nodes. The output nodes are completely connected to the input nodes by unit weights. Associated with each output node is an adjustable weight w_j $(j = 1, 2, ..., n_c)$, which can be viewed as the centre of cluster j and the output d_j is the distance between the centre of cluster j and the data sample $\mathbf{x}$.

A significant feature of the self-organizing map is that the output nodes are not affected independently to each other, but as topologically related subsets. The subset is defined by a topological neighbourhood set N_c around output node c. During the learning process, the subset however, may be composed of different output nodes and the width or radius of N_c can be time-variable. This strategy enhances the lateral interactions among the output nodes and achieves a good global ordering of the data. The algorithm is given as [12]

1. Randomly set the weights w_{ij} $(i = 1, 2, ..., n_c,\ j = 1, 2, ..., n)$. Set the neighbourhood size N_c.

2. For each training data $\mathbf{x}_i$, find the best-match output node c, such that

$$\begin{aligned} \| \mathbf{x}_i - \mathbf{w}_c \| &= \min_j \{ \| \mathbf{x}_i - \mathbf{w}_j \| \} \\ \mathit{where} \qquad \mathbf{x}_i &= [x_{i1}, ..., x_{in}]^T \\ \mathbf{w}_j &= [w_{j1}, ..., w_{jn}]^T \end{aligned} \tag{12}$$

   Update w_{ij} using the following rule

$$\mathbf{w}_j = \begin{cases} \mathbf{w}_j + \alpha(t)[\mathbf{x}_i - \mathbf{w}_j] & if \quad j \in N_c \\ \mathbf{w}_j & if \quad j \notin N_c \end{cases}$$

3. Go to **step 2**, if there are significant changes in weights.

4. If $N_c = \{0\}$, stop. Else, decrease the width of N_c.

For good global ordering, it was suggested in [12] that the width of N_c should be very wide in the beginning and shrink monotonically with time. It is even possible to end the process with $N_c = \{0\}$ as given above. The parameter $\alpha(t)$ is a scalar-valued 'adaptive gain' and

$0 < \alpha(t) < 1$. It is usually related to a similar gain used in the stochastic approximation processes [13]. An alternative is to introduce a 'bell curved' adaptive gain given as [12]

$$\alpha = \alpha_0 \, e^{-\frac{\|\mathbf{r}_j - \mathbf{r}_c\|^2}{\sigma^2}} \tag{13}$$

where $\mathbf{r}_j$ and $\mathbf{r}_c$ denote the coordinates of output nodes j and c respectively. α_0 and σ are suitable decreasing functions of time. Note that the gain is inversly proportional to the topological distance from the j^{th} output node to the best-matching node c.

The self-organizing map is an unsupervised learning process and has been particularly successful in pattern recognition applications. Like other unsupervised classification methods, it may be used to find clusters in the training data. In particular, when the input data has a well-defined density function, the weight vectors of the output nodes tend to imitate this function, no matter how complex it may be. Therefore, the **SOM** algorithm is a strong candidate for clustering in radial basis function networks.

Since it is difficult to express the dynamic properties of the learning process in mathematical theorems, simulation experiments and practical applications are usually used to explain the properties of the algorithm. It was found in [12] that a very wide initial neighbourhood N_c is essential to obtain good global ordering. A wide initial N_c introduces a rough global order in the map, the acquired global order however is not destroyed by using a narrower N_c later.

Note that the SOM algorithm uses the concept of topological neighbourhood. Topological neighbours are not necessarily neighbours in the sense of metric distance. Heuristically, it might be advantageous to introduce metric neighbourhoods for clustering applications. This idea was introduced by Huntsberger *et al* [15] and was also used by Kavuri *at al* [16]. In the **SOM** algorithm, the output nodes within the topological neighbourhood N_c have the same correction towards the training data. In clustering applications, it may make sense to assign extra weight to clusters according to their metric distances to the winning cluster. Such that a cluster centre which is relatively far from the winning cluster has relatively small amount of movement towards the training data. For applications in radial basis function networks, this heuristic may be justified using the condition given in equation (10) in the previous section, where the weight of a data sample for a given centre is inversely proportional to the distance from the data to the centre. The fuzzy membership functions may be used for this weighting purpose, and this was introduced into the **SOM** algorithm by Huntsburger *et al* [15]. The modified algorithm is given as follows. In the following, by neighbourhood we mean the set of neighbours in the sense of metric distance.

1. Randomly initialize the weights w_{ij} $(i = 1, 2, ..., n_c,\ j = 1, 2, ..., n)$. Set the number of neighbour clusters N_c.

2. For each training data $\mathbf{x}_i$, find the output node (or cluster) c, such that

$$\begin{aligned} \| \mathbf{x}_i - \mathbf{w}_c \| \quad &= \min_j \{ \| \mathbf{x}_i - \mathbf{w}_j \| \} \\ \textit{where} \qquad & \mathbf{x}_i = [x_{i1}, ..., x_{in}]^T \\ & \mathbf{w}_j = [w_{j1}, ..., w_{jn}]^T \end{aligned} \tag{14}$$

   Update w_{ij} according to

$$w_{vj} = w_{vj} + \alpha(t)\, \mu_{iv}\, (x_{ij} - w_{vj}) \qquad j = 1, 2, ..., n. \tag{15}$$

where v includes c and its N_c neighbours.

3. Go to **step 2**, if there are significant changes in weights.

4. If $N_c = 0$, stop. Otherwise, set $N_c = N_c - 1$ and go to step 2.

where μ_{iv} is the membership function of the i^{th} pattern (or input data) in the v^{th} cluster.

$$\mu_{iv} = \begin{cases} 1, & if \quad d(\mathbf{x}_i, \mathbf{w}_v) = 0, \\ 0, & if \quad d(\mathbf{x}_i, \mathbf{w}_l) = 0 \quad (l \neq v,\ 1 \leq l, v \leq n_c), \\ \left(\sum_{l=1}^{n_c} \frac{d(\mathbf{x}_i, \mathbf{w}_v)}{d(\mathbf{x}_i, \mathbf{w}_l)}\right)^{-1}, & otherwise. \end{cases}$$

Note that the membership function is in the range [0, 1] and is inversly proportional to the distance of the i^{th} pattern (or input data) from the v^{th} cluster $d(\mathbf{x}_i, \mathbf{w}_v)$. The algorithm updates all the N_c nearest clusters of the data point according to their distances to the data. We refer to this clustering algorithm as a fuzzy clustering scheme in the sence that every data sample is related to N_c clusters. It may be seen that this weighting strategy is similar to the one given in equation (12). In the following section, the scheme will be incorporated into a radial basis function network. The properties of the resulting network are compared with that of the network incorporated with a sequential k-means clustering algorithm.

5 Experimental Results

In this section, the fuzzy clustering scheme described in the previous section will be used for clustering and for finding centre locations in a radial basis function network. It will be shown how the sensitivity problem found in the k-means clustering may be overcome by using the fuzzy clustering scheme.

5.1 Clustering Results

In this section, two data sets are used to test the fuzzy clustering scheme. The first is an artificial data set illustrated in **Fig 3**. There are four clusters with 500 samples each. Each cluster has a Gaussian distribution with a variance of 0.05. The centres of the clusters are (0.5, 0.0), (0.0, 0.5), (-0.5, 0.0) and (0.0, -0.5) respectively. They will be referred as class 1, 2, 3 and 4 respectively in sequence. Note that the four classes are well seperated and good clustering results would be expected with most clustering algorithms. The second data set used is Anderson's Iris data [14]. The iris data consists of four measurements of fifty plants each of three Iris subspecies: Iris setosa, Iris versicolor and Iris virginica. The four measurements are the sepal length, sepal width, petal length and petal width. The data set is plotted in **Fig 4** with sepal width against sepal length and petal width against petal length. It may be seen that the Iris setosas are well seperated from versicolors and virginicas, while versicolors and virginicas slightly overlap. In the following, the three subspecies will be referred as class 1, 2 and 3 respectively.

In the experiments, a constant clustering gain was used in the k-means clustering algorithm and a time varying clustering gain was used in the fuzzy clustering scheme. The k-means clustering used in the experiments was

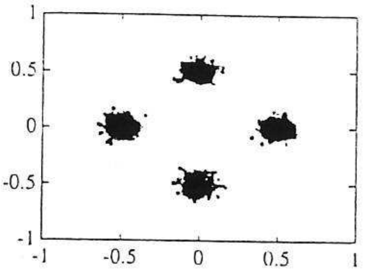

Figure 3: A set of simulated data with four classes

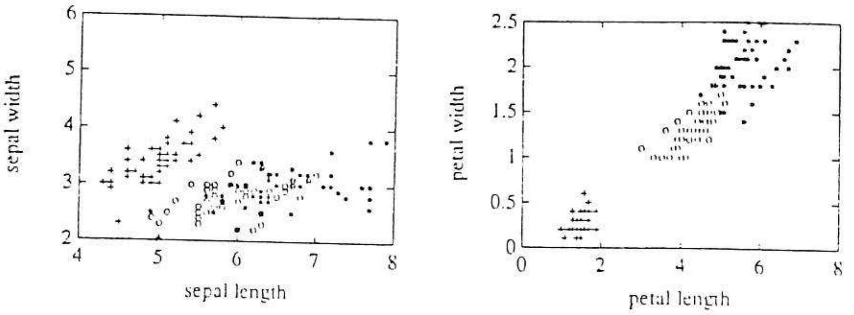

Figure 4: Iris data plotted with sepal width against sepal length, and petal width against petal length, '+': setosas, 'o': versicolors, '*': virginicas.

1. Initialize centres c_{ij}, $(i = 1, 2, ..., n_c,\ j = 1, 2, ..., n)$. Set the clustering gain α_0 and counter $t_0 = 0$. Set the number of iteration T

2. For each sample data $\mathbf{x}_i$, find the nearest centre c, such that

$$\begin{aligned} \| \mathbf{x}_i - \mathbf{c}_c \| &= \min_j \{ \| \mathbf{x}_i - \mathbf{c}_j \| \} \\ where \quad \mathbf{x}_i &= [x_{i1}, ..., x_{in}]^T \\ \mathbf{c}_j &= [c_{j1}, ..., c_{jn}]^T \end{aligned}$$

   Update c_{cj} according to

$$c_{cj} = c_{cj} + \alpha_0 \, (x_{ij} - c_{cj}) \quad j = 1, 2, ..., n.$$

3. If $t_0 < T$, Go to step **2**, otherwise stop.

The fuzzy clustering scheme used in the experiments was given as

1. Initialize centres c_{ij}, $(i = 1, 2, ..., n_c,\ j = 1, 2, ..., n)$. Set the initial clustering gain α_0. Set the number of iteration T_i and counter $t_i = 0$, $t_0 = 0$. Set the initial number of neighbours N_c. Calculate the total number of iterations $T = (N_c + 1) \times T_i$.

2. Set the clustering gain $\alpha = \alpha_0(1 - t_0/T)$.

Table 1: **Final Centres (Fuzzy Clustering) for Different Initial Centres** ($\alpha_0 = 0.1$, $T_i = 4$)

Initial centres	$\mathbf{c}_1$	$\mathbf{c}_2$	$\mathbf{c}_3$	$\mathbf{c}_4$
1IC1,2,3,4,5	0.5004	-0.0041	-0.4966	0.0014
	-0.0020	0.5009	-0.0028	-0.5034

3. For each sample data $\mathbf{x}_i$, find the centre c, such that

$$\begin{aligned} \| \mathbf{x}_i - \mathbf{c}_c \| &= \min_j \{ \| \mathbf{x}_i - \mathbf{c}_j \| \} \\ where \quad & \mathbf{x}_i = [x_{i1}, \ldots. x_{in}]^T \\ & \mathbf{c}_j = [c_{j1}, \ldots. c_{jn}]^T \end{aligned}$$

   Update c_{vj} according to

$$c_{vj} = c_{vj} + \alpha\, \mu_{iv}\, (x_{ij} - c_{vj}) \qquad j = 1, 2, \ldots, n.$$

   where v includes c and its N_c neighbours. The coefficient μ_{iv} is the fuzzy membership function given in section four.

4. Set $t_i = t_i + 1$, $t_0 = t_0 + 1$. If $t_i < T_i$, go to step **3**.

5. If $N_c = 0$, stop. Otherwise, set $N_c = N_c - 1$, $t_i = 0$ and go to step **2**.

Note that, instead of monitoring the changes of the centres, we simply process the data set for a certain number of iterations. For some of the experiments, the final centres may not have converged. Since our aim is to investigate the effect of initialization on the clustering results, the conclusions should not be affected.

Experiment 1: Clustering the four data classes into four clusters. In this experiment, five sets of initial centres were used:

1IC1: The first four samples from class 1.
1IC2: The first four samples from class 2.
1IC3: The first four samples from class 3.
1IC4: The first four samples from class 4.
1IC5: All the four initial centres were placed at (1, 1).

The data samples were processed in the order of class 1, 2, 3 and 4. The final centres obtained using the k-means and the fuzzy clustering algorithms are given in **Table 1** and **Table 2** respectively. The underlined centres are in the wrong locations.

It is obvious that the k-means clustering is very sensitive to the initial centre locations. Although the k-means clustering achieved correct clustering results for initialization 1IC1 and 1IC2, it failed to find the cluster centres for initializations 1IC3 1IC4 and 1IC5. Note that for initialization 1IC5, only one cluster centre was correctly located and the other three centres were left unchanged. These three centres are usually referred to as dead centres.

Table 2: **Final Centres (k-means) for Different Initial Centres**
$(\alpha_0 = 0.1,\ T = 15)$

Initial centres	$\mathbf{c}_1$	$\mathbf{c}_2$	$\mathbf{c}_3$	$\mathbf{c}_4$
1IC1	0.5149	-0.0144	-0.4922	-0.0026
	-0.0032	0.5133	-0.0018	-0.5216
1IC2	0.5149	-0.0144	-0.4922	-0.0026
	-0.0032	0.5133	-0.0018	-0.5216
1IC3	-0.5125	-0.0144	-0.4796	-0.0026
	0.0501	0.5133	-0.0327	-0.5216
1IC4	0.0219	-0.0144	-0.4922	-0.0297
	-0.5462	0.5133	-0.0018	-0.4663
1IC5	1.0000	1.0000	1.0000	-0.0026
	1.0000	1.0000	1.0000	-0.5216

Table 3: **Final Centres (Fuzzy Clustering) for Different Initial Centres**
$(\alpha_0 = 0.1,\ T_i = 15)$

Initial centres	c_1	c_2	c_3
2IC1.2.3.4	5.0039	5.8807	6.8440
	3.4255	2.7403	3.0794
	1.4628	4.4150	5.7102
	0.2472	1.4538	2.0662

Since all four initial centres were placed at the same location and were far away from the data samples, once one of the initial centres was moved towards the data set, it became the nearest centre to all the data samples and was updated at every sample step.

Experiment 2: Clustering the Iris data into three clusters. In this experiment, the fuzzy clustering and the k-means clustering were run on the Iris data. Three cluster centres were chosen and four sets of initial centres were selected.

2IC1: The first three samples of class 1.

2IC2: The first three samples of class 2.

2IC3: The first three samples of class 3.

2IC4: All the three initial centres were placed at (10.0, 10.0, 10.0, 10.0).

The final centres obtained using the k-means and the fuzzy clustering scheme are listed in **Table 3** and **Table 4**.

It may be seen that the k-means algorithm again failed to find the correct cluster centres for most of the initial centre locations. The clustering results are strongly affected by the

Table 4: **Final Centres (k-means) for Different Initial Centres** ($\alpha_0 = 0.1,\ T = 10$)

Initial centres	2IC1	2IC2	2IC3	2IC4
c_1	5.1893	4.9806	4.9578	$\underline{10.0}$
	3.6543	3.4159	3.3915	$\underline{10.0}$
	1.5217	1.4684	1.4656	$\underline{10.0}$
	0.2806	0.2340	0.2534	$\underline{10.0}$
c_2	$\underline{4.7310}$	$\underline{7.2965}$	5.9307	$\underline{10.0}$
	$\underline{3.0248}$	$\underline{3.1789}$	2.7410	$\underline{10.0}$
	$\underline{1.6091}$	$\underline{6.0542}$	4.7814	$\underline{10.0}$
	$\underline{0.2971}$	$\underline{2.1184}$	1.6400	$\underline{10.0}$
c_3	6.4795	6.2458	6.7555	6.4794
	3.0235	3.0422	3.1320	3.0236
	5.3782	5.2211	5.6045	5.3781
	2.0667	2.0475	2.1852	2.0667

initial centres. For the fuzzy clustering scheme, the results are independent of the initial centres.

We also experimented on the fuzzy clustering scheme with different initial clustering gains. This revealed that the algorithm is robust to changes in the clustering gain. The robustiness may be appreciated from the maximum deviations of the final centers from their real locations. For comparision, the actual cluster centres of the iris data are given in the following

$$c_1 = (5.006,\ 3.428,\ 1.462,\ 0.246)$$

$$c_2 = (5.936,\ 2.770,\ 4.260,\ 1.326)$$

$$c_3 = (6.588,\ 2.974,\ 5.552,\ 2.026)$$

The maximum deviation in any one coordinate were 0.256, 0.241 and 0.233 (which were all in the first coordinate of c_3) for $\alpha_0 = 0.1$, 0.3 and 0.5 respectively.

Experiment 3: Clustering the four data classes into eight clusters. In the previous experiments, it was shown that the fuzzy clustering scheme was insensitive to the locations of the initial centres. In both the experiments, the number of centres were set to equal the number of natural clusters within the data set. In practical applications, however, the number of natural clusters is usually unknown and the number of centres chosen is often larger than the number of natural clusters. This is particularly true in radial basis function networks. In radial basis function networks, the exact number of natural clusters may be unimportment. It is required that all the data samples are represented by the centres according to their distribution. Dead centres as found in the k-means clustering and centres lying between clusters (or classes) would deteriate the performance of the network. The former happens because the initial centres are far away from the data samples and

Table 5: **Final Centres (Fuzzy Clustering) for Different Initial Centres**
($\alpha_0 = 0.1$, $T_i = 4$, number of centres = 8)

Initial centres	$\mathbf{c}_{11}, \mathbf{c}_{12}$	$\mathbf{c}_{21}, \mathbf{c}_{22}$	$\mathbf{c}_{31}, \mathbf{c}_{32}$	$\mathbf{c}_{41}, \mathbf{c}_{42}$
3IC1,2,3,4,5	0.5211	0.0327	-0.4586	0.0127
	-0.0339	0.5114	-0.0060	-0.4630
	0.4787	-0.0437	-0.5393	-0.0122
	0.0350	0.4902	0.0013	-0.5458

there exists strong competition between the centres. The later was found in the conscience learning strategy [17] when the number of centres was larger than that of the natural clusters. The conscience learning strategy equalizes the average rates of winning for each cluster by reducing the winning rate of the frequent winning centres. It may be correct to say that the later is due to the lack of competition between the centres. In this experiment, we investigate the performance of the fuzzy clustering scheme when the number of centres is larger than that of the natural clusters. The number of centres was chosen as eight in this experiment. Again five initial centre sets were used, these are

3IC1: The first eight samples from class 1.
3IC2: The first eight samples from class 2.
3IC3: The first eight samples from class 3.
3IC4: The first eight samples from class 4.
3IC5: All the eight initial centres were placed at (1, 1).

For all these initial centres, the final centres converged to the same set of centres and every cluster is represented by two centres. There were not any dead centres or centres lying between classes. The clustering results are given in **Table 5**.

To investigate the effect of the initial clustering gain on the clustering results, the fuzzy clustering scheme was run with different initial values of α_0 (0.3, 0.5, 0.7, 0.9). The final centres are shown in **Fig 5**. In these experiments, all the initial centres were placed at (1, 1). It may be seen that the final centre locations were only slightly affected when different initial clustering gains were selected. These are mainly determined by the number of centres and the properties of the data set.

5.2 Classification Results

As mentioned previously, the **RBF** centres should move towards the majority of the training data. Therefore, network performance would be improved by implementing a better clustering algorithm in the **RBF** network. But k-means clustering is very sensitive to the initial centre locations, such that the final centres are often trapped at local minima. In certain cases, some of the training data are poorly represented. It is obvious that the performance of the network will deteriorate if this sitiutation arises in the **RBF** network. In

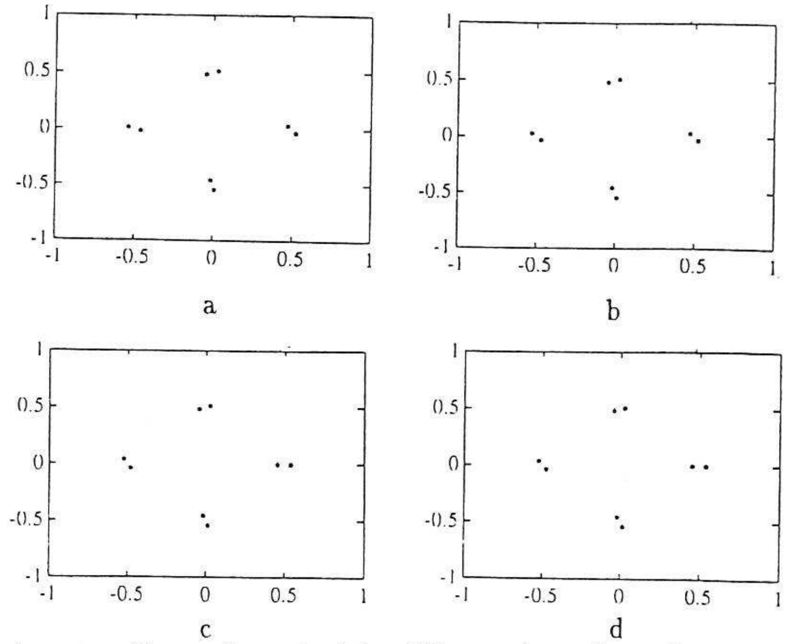

Figure 5: Final centres (fuzzy clustering) for different clustering gain $a: \alpha_0 = 0.3$, $b: \alpha_0 = 0.5$, $c: \alpha_0 = 0.7$, $d: \alpha_0 = 0.9$

network training, it is expected that the final centres should be insensitive to the initial settings of the clustering algorithm and they should not become trapped at local minima. The experiments described in section 5.1 indicate that the fuzzy clustering scheme has the above mentioned properties and it is a strong candidate for centre selection in **RBF** networks. In this section, the fuzzy clustering scheme described in section 5.1 was implemented in a radial basis function network for finding the **RBF** centres. The classification performance of the network was compared with that of the network implemented with the k-means clustering. The Iris data was used to train the network. For fairness of comparision, constant clustering gains were used in both the networks. With the same initial clustering gains, the final centres obtained in this section will be further deviated from their optimal locations than those obtained in the previous section. However, the algorithm converges much faster. We experimented with different clustering gains and different initial centre locations. The results are given in **Table 6** and **Table 7** below. The output of the network is a three dimensional unit vector. For data sample $\mathbf{x}_i$, if $\mathbf{x}_i$ belongs to class j, the desired output of the network is $\mathbf{e}_j$ (with unity on the j^{th} position and zeros on the others). In the experiments, if the j^{th} output of the network in not less than 0.75 and the other two are not larger than 0.25, the pattern $\mathbf{x}_i$ was considered to be correctly classified, otherwise, it was misclassified. This is much more strict than the so-called "winner takes all" criterion, in which the pattern $\mathbf{x}_i$ is correctly classified if the j^{th} output is larger than the others. Ten centres were used for classifying the data set. Eight sets of initial centres were used in the experiments, they are

4IC1: Ten randomly selected data samples.
4IC2: The first ten samples of class 1.
4IC3: The last five samples of class 1 and the first five samples of class 2.
4IC4: The first ten samples of class 2.

Table 6: **Classification Performances** (Initial centres: 4IC1)

	Fuzzy clustering ($T_i = 5$)			K-means ($T = 10$)		
Clustering	Misclassification			Misclassification		
gain	class 1	class 2	class 3	class 1	class 2	class 3
$\alpha_0 = 0.1$	0	7	9	0	9	9
$\alpha_0 = 0.2$	0	6	10	0	6	11
$\alpha_0 = 0.3$	0	6	10	0	8	12
$\alpha_0 = 0.4$	0	6	10	0	4	10
$\alpha_0 = 0.5$	0	6	12	0	6	13
$\alpha_0 = 0.6$	0	6	11	0	9	11
$\alpha_0 = 0.7$	0	9	13	0	7	13
$\alpha_0 = 0.8$	0	9	12	0	9	10
$\alpha_0 = 0.9$	0	10	12	0	9	11

4IC5: The last five samples of class 2 and the first five samples of class 3.
4IC6: The first ten samples of class 3.
4IC7: The last five samples of class 3 and the first five samples of class 1.
4IC8: All the ten centres were placed at (10, 10, 10, 10).

Both the fuzzy clustering scheme and the k-means algorithm were run with different initial clustering gains and the same set of initial centres 4IC1. Since the initial centres are evenly distributed in the region of the data (see **Fig 6**), both the algorithms achieved similar classification accuracy. The average misclassification is 18.2 patterns for the fuzzy clustering scheme and 18.6 patterns for the k-means algorithm. However, the fuzzy clustering scheme showed a clear relation between the number of misclassifications and the clustering gain. The larger the clustering gain, the less accurate the classification. This is because the final centre locations deviate further from their optimal locations when the clustering gain is increased. For the k-means algorithm, there seems no explicit relation between the classification accuracy and the clustering gain. For different clustering gains, the centres become trapped at different local minima and the network exibits very different classification behaviour.

The best clustering gains obtained in the previous experiments were chosen to run the fuzzy clustering and the k-means algorithm respectively with different initial centre locations. The classification performance of the networks are listed in **Table 7**. It may be seen that the network with fuzzy clustering exhibited uniform classification performance for different initial centres. On average, it also achieved much higher classification accuracy than the network with k-means clustering. For initial settings 4IC1 - 4IC7, the average number of misclassification for the network with k-means clustering is 21.3 patterns. For initial setting 4IC8, all the patterns in classes 2 and 3 were misclassified. Note that the network with k-means clustering achieved the best classification performance among the

Table 7: **Classification Performances** (different initial centres)

	Fuzzy clustering ($T_i = 5$)			K-means ($T = 10$)		
Initial	Misclassification			Misclassification		
centres	class 1	class 2	class 3	class 1	class 2	class 3
4IC1	0	7	9	0	4	10
4IC2	0	7	9	0	11	18
4IC3	0	7	9	0	7	10
4IC4	0	7	9	0	13	9
4IC5	0	7	9	0	17	14
4IC6	0	7	9	0	10	10
4IC7	0	7	9	0	9	7
4IC8	0	7	9	0	50	50

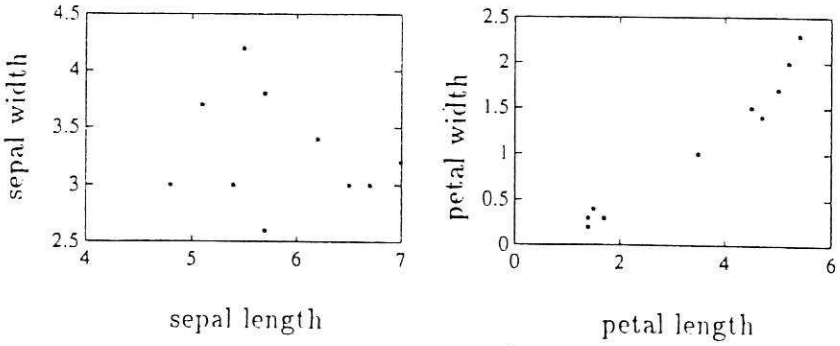

Figure 6: Initial centre set 4IC1

experiments when the initial centres were 4IC1. This may be explained as follows. Since the optimal centres for the radial basis function network depend on the data as well as the properties of the underlying function which is being approximated by the network, the optimal clustering centres are not necessaryly the optimal **RBF** centres. Therefore, for certain centre locations, which are local optimal solutions of the clustering procedure, the **RBF** network may achieve good classification results. However, this is arrived at completely by chance and is unpredictable.

6 Discussions and Related Methods

The fuzzy clustering scheme described in the previous section is a variant of the self-organizing feature map. Several parameters are required such as the clustering gain, number of neighbourhood clusters and a strategy to alter these two parameters during clustering. From the experiments described above, it may be seen that the clustering gain only has a marginal effect on the final clustering results when it is monotonically decreased with time.

When the number of neighbours is $n_c - 1$, there is only one cluster centre. This centre should be the grand mean of the data samples. In practical applications however, this will depend on the sequence in which the data samples are processed. This cluster centre will be split into smaller clusters when the number of neighbours is reduced. As the number of neighbour clusters is reduced from $n_c - 1$ to 0, the cluster centres will undergo a sequence of transitions. For a given number of neighbour centres, the clustering scheme will converge to a local minima. By changing the number of neighbour centres, the scheme will escape from this local minima. The local minima is avoided by a sequence of changes in the number of neighbour centres. The optimal way to alter the number of neighbour centres is probably problem dependent and is unknown. In the experiments above, the number of neighbour clusters was decreased by one from $N_c - 1$ to 0 every T_i iterations. In practical applications, this method may became computationally expensive when the number of clusters is large, although it may be necessary in some cases. While there is no simple rule to indicate how to alter the number of neighbour clusters, we intended to investigate if it is possible to start the clustering procedure with a very large number of neighbour clusters in the first T_i iterations only and reduce this to a much smaller number and then continue to reduce it by smaller steps. We experimented with the Iris data in the hope of reducing the computational cost of the clustering procedure. With ten centres selected, we began the clustering procedure for the first T_i iterations with nine neighbours and reduced it to a smaller number N_{cm} for the next T_i iterations, after which the number of neighbours was reduced by one every T_i iterations. When N_{cm} was 7, 6, 5, 4 and 3, the **RBF** network achieved the same classification results as those listed in **Table 7** (where N_{cm} was 8). Reducing N_{cm} further, produced worse results. We also experimented with different numbers of centres. The results showed that it is possible to start the clustering procedure from a very large number of neighbours and reduce it to roughly half the number of the centres and to get the same clustering results as the procedure given in the previous section. This strategy can reduce the computational cost significantly and still produce the same global cluster centres. As mentioned previously however, it seems essential to start with a very large number of neighbour centres in the first T_i iterations.

The adaptation rule in the fuzzy clustering scheme was determined heuristicly and a cost function which is minimized by the updating formular (11) cannot be specified. Recently, several clustering algorithms which minimize certain objective functions have been developed. The deterministic annealing [18], [19] and the generalized clustering network [20] are such schemes. One common feature of these algorithms is that they introduce a similar adaptation rule as the one given in the fuzzy clustering scheme, this not only updates the 'winning' cluster centre but also affects all cluster centres in a neighbourhood set. The determinstic annealing method is based on statistical physics and minimizes the cost function or free energy

$$E_{da} = -\frac{1}{\beta}\sum_{\mathbf{x}} \ln\left[\sum_{v} e^{-\beta(\mathbf{x}-\mathbf{c}_v)^2}\right] \tag{16}$$

A stochastic gradient descent adaptation rule would be

$$c_{vj} = c_{vj} + \alpha\ \mu_{iv}\ (x_{ij} - c_{vj}) \tag{17}$$

where

$$\mu_{iv} = \frac{e^{-\beta(\mathbf{x}_i-\mathbf{c}_v)^2}}{\sum_{j=1}^{n_c} e^{-\beta(\mathbf{x}_i-\mathbf{c}_j)^2}} \quad v = 1, 2, ..., n_c$$

By anology, β is said to be proportional to the temperature. As β gets larger, the associations between the data samples and the cluster centres become less fuzzy. When β is zero, each data sample is equally associated with all cluster centres, while as β tends to infinity each data sample belongs to exactly one cluster centre with probability one. To avoid local minima of the cost function, the clustering procedure usually begins with a high temperature and this is gradually reduced to zero. At $\beta = 0$ there is only one cluster centre, this cluster will split into smaller clusters at a higher β value. Therefore, the clusters will undergo a sequence of phase transitions during the clustering process. However, to optimize the method, more serious investigations into the phase transition (or annealing schedule) are required. In practical applications, β was usually increased exponentially, this would certainly compromise the clustering results and may result a very slow clustering process. In addition, a strategy to alter the clustering gain is also required.

The generalized clustering network minimizes the cost function

$$E_{gc} = \frac{\sum_{i=1}^{N}\sum_{v=1}^{n_c} g_{cv}(\mathbf{x}_i - \mathbf{c}_v)^2}{N} \tag{18}$$

where

$$g_{cv} = \begin{cases} 1 & if\ \ v = c \\ \frac{1}{\sum_{j=1}^{n_c}(\mathbf{x}-\mathbf{c}_j)^2}, & otherwise \end{cases}$$

The cost function E_{gv} is minimized by local gradient descent search using the sample function

$$L_{\mathbf{x}} = L(\mathbf{x}, \mathbf{c}_1, \ldots, \mathbf{c}_{n_c}) = \sum_{v=1}^{n_c} g_{cv}(\mathbf{x} - \mathbf{c}_v)^2 \tag{19}$$

Note that $L_{\mathbf{x}}$ is a measure of the locally weighted mismatch error of $\mathbf{x}$ with respect to the winning cluster c. The adaptation rules are given as

$$\begin{aligned} c_{cj} &= c_{cj} + \alpha \frac{D^2 - D + (\mathbf{x}_i - \mathbf{c}_c)^2}{D^2}(x_{ij} - c_{cj}) \;\; \textit{for the winning cluster } c && (20) \\ c_{vj} &= c_{vj} + \alpha \frac{(\mathbf{x}_i - \mathbf{c}_c)^2}{D^2}(x_{ij} - c_{cj}) \;\; \textit{for the other } (n_c - 1) \textit{ clusters} && (21) \end{aligned}$$

where

$$D = \sum_{v=1}^{n_c} \| \mathbf{x}_i - \mathbf{c}_v \|^2$$

Note that the updating rules are very similar to those adopted in the fuzzy clustering scheme except that now there is no need to specify a neighbourhood set. The clustering gain may also be altered as similar to the procedure used in the fuzzy clustering scheme. When applied to the Iris data, the generalized clustering network ended up with a maximum deviation of 0.26 for 500 iterations [20]. In our experiment, a slightly smaller deviation was achieved in 45 iterations although different initial cluster centres and clustering gains were used. We also experimented with ten cluster centres, for initial settings 4IC4 and 4IC6 the generalized clustering network failed to converge to the same set of final centres in 2000 iterations. It may be correct to say that the convergence rate of the algorithm is slower than the fuzzy clustering scheme.

7 Conclusions

A fuzzy clustering scheme has been implemented in a radial basis function network. It has been shown by experiments that the fuzzy clustering scheme is insensitive to initial centre locations and is robust with respect to changes in the clustering gain. The fuzzy clustering scheme relates a data sample not only to the nearest cluster centre but also to a set of neighbour cluster centres. By using a sequence of different neighbour sets, the algorithm can avoid the local minima problems found in the k-means or similar clustering algorithms. While there is no simple rule on how to alter the number of neighbour centres, it was shown on a real data set that it is possible to start the clustering scheme with a very large number of neighbour centres in the first T_i iterations and then to reduce this to a much smaller number in the following T_i iterations. The computational cost can thus be significantly reduced and global clustering results can still be obtained.

When the fuzzy clustering scheme was applied in **RBF** networks, all the data samples were well represented by the cluster centres. There were no dead centres or centres lying between classes and the resulting network achieved better classification performance than the network with the k-means algorithm.

Acknowledgements

The authors gratefully acknowledge that this work was supported by the EEC under Brite/Euram contract BREU-G91-0526 (RSJE) with the partners Professor G Tomlinson and Dr K Worden (University of Manchester), and Professor F Brancaleoni and Dr C Valente (University of Pescara, Italy).

References

[1] F. Girosi and T. Poggio, Networks and the Best Approximation Property, Biological Cybernetics, Vol. 63, 1990, pp 169 - 176.

[2] W. A. Light. Some Aspects of Radial Basis Function Approximation, Approximation Theory, Spline Functions and Applications, Vol. 356, 1992, pp 163 - 190.

[3] C. A. Micchelli. Interpolation of Scattered Data: Distance Matrices and Conditionally Positive Definite Functions, Constructive Approximation, 1988, Part 2, pp 11 - 22.

[4] M. J. D. Powell. Radial basis functions in 1990. In "Advances in Numerical Analysis", Vol. II, Oxford University Press, 1992, pp 105 - 210.

[5] T. Poggio, F. Girosi. Network for Approximation and Learning, Proceddings of IEEE, Vol. 78, No. 9, September 1990, 1481 - 1497.

[6] J. Moody, C. Darken. Fast learning in networks of locally-tunned processing units, Neural Computation, 1989, 1, 281 - 294.

[7] S. Chen, S. A. Billings & P. W. Grant. Recursive hybrid algorithm for non-linear system identification using radial basis function network, INT. J. Control, 1992, Vol. 55, No. 5, 1051-1070.

[8] S. Chen, S. A. Billings, C. F. N. Cowan & P. W. Grant. Practical identification of NARMAX models using radial basis functions, INT. J. Control, 1990. Vol. 52, No. 6, 1327-1350.

[9] D. S. Broomhead, D. Lowe. Multivariable Functional Interpolation and Adaptive Networks, Complex Systems, 1988, 2, 321 - 355.

[10] M. Vogt, Combination of Radial Basis Function Neural Networks with Optimaized Vector Quantization, Proceedings of the IEEE International conference on Neural Networks, Vol. 3, 1993, pp1841-1846.

[11] B. Everitt, Cluster Analysis, Third Edition, Edward Arnold, 1993.

[12] T. Kohonen. The Self-Organizing Map, Proceedings of the IEEE, Vol. 78, No. 9, 1990, pp1464-1480.

[13] T. Kohonen, Things You Haven't Heard about the Self-Organizing Map, Proceedings of the IEEE International conference on Neural Networks, Vol. 3, 1993, pp1147-1156.

[14] R. A. Fisher, The Use of Multiple Measurements in Taxonmix Problems, Annals of Eugenics, 7, 1939, pp116-188.

[15] T. L. Huntsberger and P. Ajjimarangsee, Parallel Self-Organizing Feature Maps for Unsupervised Pattern Recognition, Int. J. General Systems, Vol. 16, No. 4, 1990.

[16] S. N. Kaviori and V. Venkata-Subramanian, Using Fuzzy Clustering with Ellipsoidal Units in Neural Networks for Robust Fault Classification, Computers Chem. Engng., Vol. 17, No. 8, 1993.

[17] Lei XU, A. Krzyzak and E. Oja, Rival Penalized Competitive Learning for Clustering Analysis, RBF Net, and Curve Detection, IEEE Trans. on Neural Networks, Vol. 4, No. 4, 1993, pp636-649.

[18] K. Rose, E. Gurewitz and G. C. Fox, Statistical Mechanics and Phase Transitions in Clustering, Physical Review Letters,, Vol. 65, No. 8, 1990, pp945-958.

[19] K. Rose, E. Gurewitz and G. C. Fox, Constrained Clustering as an Optimization Method, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 15, No. 8, 1993, pp785-794.

[20] N. R. Pal, J. C. Bezdek and E. C.-K. Tsao, Generalized Clustering Networks and Kohonen's Self-Organizing Scheme, IEEE Trans. on Neural Networks, Vol. 4, No. 4, 1993, pp549-557.