Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

A Hybrid MGA-BP Algorithm for RBFNs self-generate

Shiwei YU
School of Economic s and Management
China University of Geosciences
Wuhan, P.R.China
ysw81993@sina.com

Kejun ZHU
School of Economic s and Management
China University of Geosciences
Wuhan, P.R.China
zhukejun@cug.edu.cn

***Abstract*—This paper proposes a novel hybrid algorithm to determine the parameters (number of neurons, centers, widths and weights) of radial basis function neural networks automatically. In this work, a hybrid algorithm combines the multi-encoding genetic algorithm (MGA) and the back propagation (BP) algorithm to form a hybrid learning algorithm (MGA-BP) for training Radial Basis Function Networks (RBFNs), which adapts to the network structure and updates its weights by choosing a special fitness function. The proposed method is used to deal with non-linear identification problems, and the results obtained are compared with existent bibliography, showing an improvement over the published methods.**

***Keywords*—genetic algorithm; multi-encoding; Radial basis function neural networks; Self-generate**

I. INTRODUCTION

Radial Basis Function Networks (RBFNs) is a major class of neural network models, where the distance between the input vector and prototype vector determines the activation of a hidden unit. An increasing amount of literature is available on RBFNs methods which has became a popular technique since the 1980s because of their simple structure, well established theoretical basis and fast learning speed, which are all crucial factors in real applications [1-3]. However, a key problem of using the RBF neural network approach is about how to choose the structure and the optimum initial values of the following three parameters: the output weights, the centers and widths of the hidden unit. If they are not appropriately chosen, the RBF neural network may degrade the validity and accuracy of modeling. Researchers have adopted many methodologies to overcome these problems. Different approaches can be categorized as follows: (1) Two-phase approach which merges a hybrid training strategy, using a clustering algorithm, including k-means clustering[4], Fuzzy C-means clustering [5] and Kohonen's self-organizing maps [6] to pick the centers, followed by a supervised one, like back propagation algorithms to obtain the output weights. (2) Online RBFNs self-generate method which the structure of network determines automatically by training process. Most of these methods grow the final net from an initially empty one to which hidden neurons are added until a given condition is reached. One of such methods is orthogonal least squares (OLS) algorithm [7]. Although this algorithm is widely used, Shwrstinsky [8] who has been studied the algorithm from the perspective of the energy compression, found that OLS algorithm might not be able to design a structure with minimal network. (3) Growing strategies [9-10]. They established dynamic structure in which units are successively added to the network until convergence. The network structure is developed without really relating the addition order of the units and the relevance of the category boundaries introduced, so probably generating over-structured networks and the generality is not well. (4) Pruning methods that start with an initial selection of a large number of hidden units which remove are reduced as the algorithm proceeds [11]. The main drawback of pruning methods is that it tends to be very restrictive, and, thus, find suboptimal networks. An improvement over this algorithm is growing and pruning RBF networks called GAP-RBF algorithm [12]; and modified GAP-RBF in which selection of the network centers is based on a fuzzy partition of the input space [13-14]. GAP-RBF algorithm was derived based on the significance of a neuron assuming the input data distribution is uniform. If the input distribution is not uniform, then the performance of GAP-RBF will be degraded [15]. This may be a serious problem for applications.(5) Evolutionary computation techniques, in which the RBFNs parameters are selected simultaneously by employing optimization methods based on genetic algorithms(GA) [16].

This paper is focused on the evolutionary strategies, but it's different from the existing methods. We combined the multi-encoding genetic algorithms with back propagation (BP) algorithm to form a hybrid learning algorithm (MGA-BP) for training RBFNs, which adapts to the network structure and updates its weights according to the performance of the networks. As a powerful optimization tools, GA can be used effectively in the evolution to find a near-optimal set of connection weights globally without computing gradient information. However, its local search capabilities were restricted and often cost more time to achieve the optimal solution. On the other way, the global search capability of BP algorithm is not very efficient and easy to converge on local optimal value, but it GA and the rapid local approximation of BP algorithm, the new algorithm fully shows the ability of nonlinear approach of multilayer feed forward network,

improves the performance of RBFNs, and provides a favorable basis for further on-line application of a comprehensive model.

The hybrid algorithm in which every individual consists of binary and real parts is briefly described as following: Firstly, the hidden nodes, parameters of RBFNs are been optimized by MGA. Secondly, in order to get proper RBF parameters (the centers, widths and weights. etc) of a certain network structure, which corresponds to the real part of particle go to BP algorithm for a fast local gradient guide under a continues GA global optimization search. Finally, set parameters obtained by BP to corresponding particle, and then go to next iteration. Furthermore a special fitness function is introduced to ensure accuracy with a few centers and the optimal number of hidden nodes corresponds to the evolved solution with the best fitness value with respect to each individual run. Approximate a nonlinear function and identify a discrete dynamic system experiments have been done. The results show that fewer hidden nods are used during the modeling and therefore the model prediction error is significantly reduced. The modeling results are compared with other methods and the superiority emerges obviously.

II. BRIEF OF RBFNS AND GA

A. RBFNs network topology and node characteristics

An RBF network is a type of feed forward neural network that learns using a supervised training technique. In 1988, based on Cover's [17] theorem on the separability of patterns, Broomhead et al[18] employed radial basis functions in the design of neural networks. The nonlinearities possess the localization properties [19], which make the network sensitive only to particular regions of the input space. The RBF network is composed of three layers. The network structure is shown in Figure. 1. The input layer transfers input signals to a hidden layer. The hidden layer, that is, the basis function layer, is composed of a basis function, like a Gaussian function. And the output layer is usually a certain simple linear function.

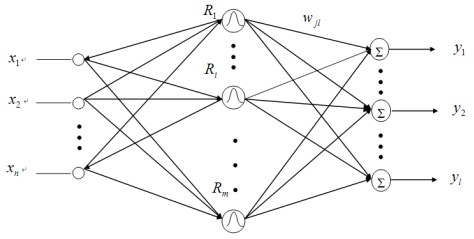


Figure 1. The structure of radial basis function neural networks

An RBF network has the ability of local approximation, so when the input signals are in the middle of the basis function, the hidden layer will have large output. The most common basis function is a Gaussian function, as in (1).

$$R_i(X) = \exp\left[-\frac{\|X - c_i\|^2}{2\sigma_i^2}\right],\ i = 1,2,\cdots,m, \tag{1}$$

where $X = [x_1, x_2, \cdots, x_n]^T$ is a vector of n dimensions; c_i, with the same dimension as X, is the center of the *ith* basis function; σ_i is the *ith* variable, which determines the extent of the basis function surrounding the center; m is the number of hidden neurons; $\|X - c_i\|$ is the norm of the vector $X - c_i$, which means the distance between X and c_i; $R_i(X)$ gets the maximum at c_i, and with the increase of $\|X - c_i\|$, $R_i(X)$ decreases rapidly to zero. Given the input $X \in R^n$, only a small section of inputs near X are activated.

The input layer implements the nonlinear mapping $X \to R_i(X)$, and the output layer implements the linear mapping $R_i(X) \to y_k$, which is shown in (2).

$$y_k = \sum_{i=1}^{m} w_{ik} R_i(X), k = 1,2,\cdots, p \tag{2}$$

Where p is the number of output neurons. Detailed descriptions of the RBF network can be found in [20].

B. Genetic algorithms

GA comprehends a stochastic searching process based on the mechanisms of natural selection and natural genetics [21]. The pseudo-code algorithm depicted in Figure. 2 summarize the general procedure of GA [22]. In the algorithm, $p(t)$ denotes a population of m individuals at generation t. GA does not have many mathematical requirements for optimization problems. In addition, the ergodicity of genetic and evolution operations makes GA more effective at global search. Also, the easy-to-grasp implementation procedure underlying GA provides great opportunity to hybridize them with domain-dependent heuristics towards a more effective strategy cast in accordance with the problem at hand. For these advantages, GA has received considerable attention regarding their potential as a more robust technique.

```
Begin
t ← 0;
Initialize population p(t) ;
Evaluate every individual in p(t) ;
  Do
  Perform crossover in  p(t)  to yield p'(t) ;
  Perform crossover in  p'(t)  to yield p''(t) ;
  Evaluate every individual in p''(t) ;
  Select  p(t+1)  from p''(t) ;
  t ← t+1;
  While (not termination condition)
End
```

Figure 2. Genetic algorithms procedure.

III. DESIGN OF THE MODELTO GA

A. Multi-enconding

In GA one chromosome is corresponding to a solution of a problem to solve. So every individual in this algorithm should include the information of the structure and parameters of the network, namely the number of hidden nod *hid_no*, the center c_i and the width σ_i of each hidden unit as well as the

weight value w_i. Divide the whole chromosome into two parts $chrom = [b\_chrom, r\_chrom]$ which binary part par_1 corresponding to each hidden node and real part par_2 corresponding to, c_i, σ_i and w_i. If the max number of hidden nod is $\max\_nod$, the input vector X is n dimensions, the output number is l, then total length of every particle is $m \cdot (n + l + 1)$. If the value coded in binary system, is "1," the neuron is selected. On the contrary, if the value is "0," the neuron is unselected. For example, let the max hidden nod is 7, the masking binary string encoded as [1 0 1 1 0 0 1] presents that the 1, 3,4and 7 hidden nods should be kept, and the others nods should be removed. The real part codes the weights and bias parameters which may be in values of [-1, 1]. Therefore, the structure of a particle is and an example of 4 hidden nods and corresponding RBF parameters selected are shown in Figure.3.

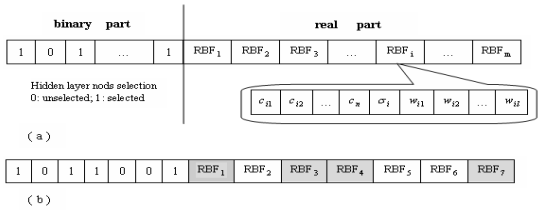


Figure 3. RBFN particle structure; (b). an example of 4 nods and corresponding RBF parameters selected

B. Fitness function

To train the RBFN structure is to make it have the simplest network form at an allowable accuracy. It means that we make the accuracy and complexity of the network attains the minimum. Thus, a fitness value is given by (3).

$$fitness = \frac{1}{E + \alpha \cdot hide\_nod} \tag{3}$$

Where $E = \frac{1}{2}\sum_{j=1}^{n}(y^d(x^{(j)}) - f(x^{(j)}))^2$ means the actual error between the desired output $y^d(x^{(j)})$ and the actual output $f(x^{(j)})$ of the RBFN, which is determined by the individual parameters. It's should point out, in Eq. (1), E may be instead by other network performance function such as recognition rate of a model recognition problem in some applications. The complexity of the network, which is decided by the number of hidden nodes $hide\_nod$, while $\alpha, (0 < \alpha < 0.01)$ is an influential coefficient of the number of hidden layer nods. Furthermore, one hidden layer nod must be selected at least in every particle, namely, $hide\_nod \geq 1$. In this manner, the proposed MGA-BP algorithm can find the best RBFN which approach higher accuracy (i.e. smaller E) with fewer number of radial basis functions with the guidance of the proposed fitness function corresponding to the same input. Therefore, the RBFN with very good performance can be automatically generated by the multi-encoding GA method as compared with the time-consuming, traditional trial-and error methods used to determining appropriate RBFN parameters.

IV. The Algorithm Approaches For RBFN Self-generate

The simultaneous optimization training method of the topology structure and parameters of the network are used, in which the Gaussian function is adopted as the radial basis function. More details are described as follows and illustrated by the flowchart in Figure. 4.

1) Initialize prior parameters of the model: the size of population pop_size, the max generation $\max\_g$ and $\max\_k$, the max number of hidden nod $\max\_nod$, and the epoch of BP learning BP_epoch.

2) Randomly generate initial population $chrom = [b\_chrom, r\_chrom]$ which randomly initialize chromosomes of binary variables in b_chrom and continuous real variables in r_chrom.

3) If $g \leq \max\_g$, evaluate objective function *fitness* for population, else go to step *9)*.

4) Genetic operators are implemented.

a) select and reproduce population according to value of fitness.

b) Calculate the self-adaptive crossover probability P_c and P_m by (4) and (5), respectively, and crossover and mutation operation of different code genes.

$$P_c = \begin{cases} (f_{\max} - f')/(f_{\max} - f_{ave}) & if \quad f' \geq f_{avg} \\ 1.0 & if \quad f' < f_{ave} \end{cases} \tag{4}$$

$$P_m = \begin{cases} 0.5 \cdot (f_{\max} - f)/(f_{\max} - f_{ave}) & if \quad f \geq f_{ave} \\ (f_{ave} - f)/(f_{ave} - f_{\min}) & if \quad f < f_{ave} \end{cases} \tag{5}$$

Here, f is the fitness of an individual, f_{ave} the average fitness value of the population, and $f_{\max}$ and $f_{\min}$ the maximum and minimum fitness value of the population respectively. f' is the larger of the fitness values of the solutions to be crossed.

c) estimate the number of 1s in the b_chrom gene code, if the number is 0, one of the genes set to 1 randomly, else go to 5).

5) In order to get the optimal network parameters c_i, σ_i, w_i based on a fixed structure, optimize b_chrom using continuous GA in further.

a) Let $k = 1$, if $k \leq \max\_k$, evaluate objective function fitness for all individuals, otherwise go to step 6).

b) crossover and mutation operate for r_chrom.

c) $k = k + 1$ and go to step (a).

6) Back propagations algorithms are used to adjust the position of par_2, namely parameters c_i, σ_i, w_i of RBFN by following formulas:

$$\begin{aligned} c_i(t) &= c_i(t-1) + \eta_1 * \Delta c_i \\ &= c_i(t-1) + \eta_1 (-\sum_{j=1}^{n}(y^d(x^{(j)}) - f(x^{(j)})) * w_i * \exp\left[-\frac{\left\|x^{(j)} - c_i\right\|^2}{2\sigma_i^2}\right] * \frac{1}{\sigma_i^2} * (x^{(j)} - c_i)) \end{aligned} \tag{6}$$

$$\sigma_i(t)=\sigma_i(t-1)+\eta_2*\Delta\sigma_i$$
$$=\sigma_i(t-1)+\eta_2(-\frac{1}{\sigma_i^3}\sum_{j=1}^{n}(y^d(x^{(j)})-f(x^{(j)}))*w_i*\exp\left[-\frac{\left\|x^{(j)}-c_i\right\|^2}{2\sigma_i^2}\right]*\left\|x^{(j)}-c_i\right\|^2) \quad (7)$$

$$w_i(t)=w_i(t-1)+\eta_3*\Delta w_i=w_i(t-1)+\eta_3(\sum_{j=1}^{n}(y^d(x^{(j)})-f(x^{(j)}))*\exp\left[-\frac{\left\|x^{(j)}-c_i\right\|^2}{2\sigma_i^2}\right]) \quad (8)$$

7) Let $g=g+1$ and go to step 3).

8) Stop and the global best solution will be selected to generate the final parameter set $S=\{hid\_no,c_{i1},c_{i2},\cdots,c_{in},\sigma_i,w_{i1},w_{i2},\cdots,w_{il}\}$; then, the desired RBFN system can be developed.

From the above descriptions, the only carefully predetermined parameter is max node of RBFN which mainly depends on the complexity of the problem. Usually, the more complexity of the problem, the more max nodes of RBFN is needed. The other parameters such as max generation and pop size of GA related to the search range and precision level, which influence only on the utilization of the computational resources.

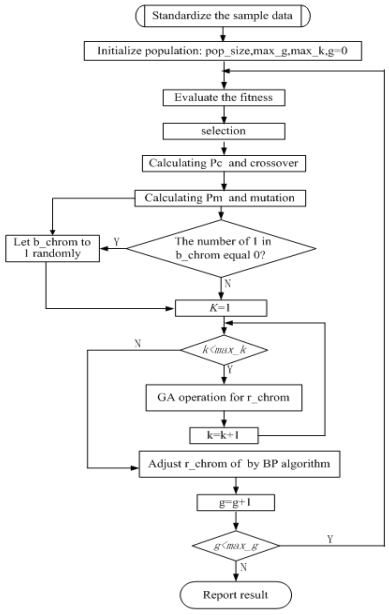


Figure 4. The flowchart of MGA-BP-RBFN.

V. ILLUSTRATED EXAMPLE FOR NON-LINEAR SYSTEM IDENTIFICATION

In order to demonstrate the efficiency of the proposed MGA-BP learning method, different types of nonlinear problems are presented in this section. Programs of all examples has been done using Matlab 6.5, and been run on an Intel core 2 CPU 4400 PC with 1 GB of RAM.

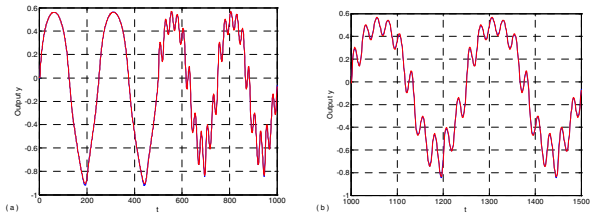
Consider the nonlinear system described by the input–output model:

$$y(t+1)=\frac{y(t-1)y(t-2)y(t-3)y(t-2)(y(t-3)-1)+u(t-1)}{1+y(t-2)^2+y(t-3)^2} \quad (9)$$

Where $u(t)=\begin{cases}\sin\left(\frac{2\pi}{250}\right) & k\le 500\\ 0.8\sin\left(\frac{2\pi}{250}\right)+0.2\sin\left(\frac{2\pi}{25}\right) & k>500\end{cases}$. (10)

This system was introduced by Narendra and Parthasarathy [23] and was used subsequently by Liu et al. [7], Alexandridis et al. [13], as a case study for illustrating an online identification method based on VPBF neural networks and fuzzy subspace RBF, respectively. The input of the RBF model consisted of three previous values of u and three previous values of y : $x(t)=[u(t-1),u(t-2),u(t-3),y(t-1),y(t-2),y(t-3)]$. The training set is $0\le t\le 1000$ and $1001\le t\le 1500$ is the testing set. The maximum number of radial basis functions is 8. The length of every chromosome is 72, in which 64 parameters $\{c_{in},\sigma_i,w_i,1\le i\le 8,1\le n\le 6\}$ and 8 selective genes of hidden nods are required to be efficiently chosen from the solution space. The pop size of GA and BP iterations are 50 and 100, respectively.10 runs have been implemented randomly using the above parameters. Proper RBFNs, with respect to acceptable solutions, are selected and the simulation results are shown in Figure 5. Figure 5(a) show the training desired output (blue line) versus simulated output of the best generated RBFN identifiers (red line). Figure 5(b) and Figure 5(c) plot of the testing real data (blue line) versus predicted data (red line) with no noise and Gaussian noise, and with zero mean and 1 standard deviation was added to the testing input values, respectively. Figure 5(d) shows the preference function fitness at different iteration. On the other hand, Figure 5 (e) shows the selected best, average and worst number of radial basis functions for best individual in 10 runs, respectively. The estimation error is depicted in the Figure 5(e), in which blue line is training data and dark line is testing data. The estimation error is depicted in the Figure 5(f).

Simulation results indicate that only 4 radial basis functions are enough to identify this nonlinear model with small errors. In addition, the generalization performance is more excellent. The performance comparison for the other modeling method with mean AE, MSE and σ of training data in 10 runs is listed in Table 1. The results indicate a much higher accuracy but few hidden units using MGA-BP algorithms compared to other learning methods. Further more, the proposed method is satisfactory in the presence of noisy inputs.



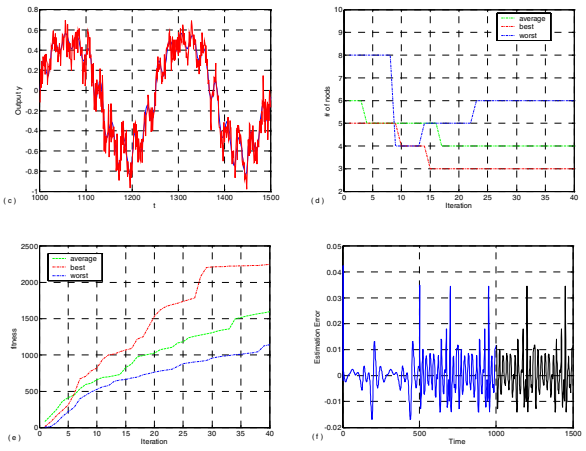


Figure 5. (a) Training data vs best output in 10 runs. (b) Testing data vs output corresponding. (c) Gaussian noise with zero mean and 1 standard deviation was then added to the testing input values. (d) *fitness* values against iteration. (e) selected RBFNs numbers against iteration .and (f) estimation error against data index.

TABLE I. PERFORMANCE COMPARISON ROR ILLUSTRATED EXAMPLE

	hidden nodes	Training AE	Training σ	Training MSE	Testing MSE
MGA-BP	4	34.4368	-9.6593	4.4559e-005	6.0949e-005
BP	18	88.35	432.87	0.0065	0.0072
OLS[7]	15	-6.9281e+003	-44.3064	8.2020e-004	0.0012
Liu [9]	20	-55.29	563.80	0.0015	0.0067
Alexandridis [13]	15	-35.98	378.04	3.3215e-004	3.5572e-004

VI. CONCLUSIONS

A novel algorithm, based on a hybrid of MGA and BP, has been presented and discussed in this paper for training RBFNs to non-linear system identification. The combination of the search capabilities of a global and a fast local optimization method has been explored. Adopting multi-encoding, the hybrid algorithm can optimize the number of RBFNs, centers, widths and weights of RBFNS dynamically and adaptively by choosing a special fitness function. Experiment has been conducted with a modeling example, and it reveals that the proposed RBF network construction method is can be used and in some cases far less RBFNs in comparison with other algorithms. Furthermore, the results indicate that the performance of the adaptive algorithm is satisfactory in the presence of noisy inputs.

ACKNOWLEDGMENT

This research was fully supported by National Natural Science Foundation Grant No. 70573101 of the People's Republic of China and the Research Foundation for Outstanding Young Teachers, China University of Geosciences （Wuhan） No.CUGQNW0901.

REFERENCES

[1] A. Shahsavand and A. Ahmadpour, "Application of optimal RBF neural networks for optimization and characterization of porous materials, "Computers & Chemical Engineering, vol.29, no.10, pp.2134–2143,2005.

[2] J.Park and I.W.Sandberg, "Universal approximation using radial basis functions network," Neural Computation, vol.3, pp.246–257,1991.

[3] T. Mu, K. Asoke and Nandi, "RBF neural networks for solving the inverse problem of backscattering spectra," Neural Computing & Applications, DOI 10.1007/s00521-007-0138-2

[4] C.Darken and J.Moody, "Fast adaptive K-means clustering: some empirical results," IEEE INNS International Joint Conference on Neural Networks; Proceedings, pp.233–238, 1990.

[5] C. C. Wong and C. C. Chen, "A hybrid clustering and gradient descent approach for fuzzy modeling," IEEE Trans. Syst. Man Cybern, vol.29 no.6,pp. 686–693,1999.

[6] S. Haykin, "Neural Networks—A Comprehensive Foundation," IEEE Press, New York, 1994.

[7] S .Chen, C.F.N Cowan and P.M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks,"IEEE Trans Neural Networks, vol.2, no.3, pp.302–309,1991.

[8] A.Sherstinsky and R.W. Picard, "On the efficiency of the orthogonal least squares training method for radial basis function networks," IEEE Trans. Neural Networks, vol.7,no.1,pp. 195-200,1996.

[9] G. P. Liu, V. Kadirkamanathan and S.A.Billings, "Online identification of nonlinear systems using Volterra polynomial basis function neural networks," Neural Networks, vol.11,no.9, pp.1645–1657,1998.

[10] M. Marinaro and S. Scarpetta, "On-line learning in RBF neural networks: a stochastic approach," Neural Networks,vol.13,no 4 ,pp.719–729,2000.

[11] A. Leonardis and H. Bischof, "An efficient MDL-based construction of RBF networks," Neural Networks, vol.11,no.5,pp. 963–973,1998.

[12] R. Zhang,G.B. Huang, P. Saratchandran and N. Sundararajan, "Improved GAP-RBF network for classification problems," Neurocomputing (2006), doi:10.1016/j.neucom.2006.07.016

[13] A. Alexandridis, H. Sarimveis, G. Bafas.,A new algorithm for online structure and parameter adaptation of RBF networks,Neural Networks,16(7)(2003)1003–1017.

[14] A.Guillen, H.Pomares and I.Rojas, "Studying possibility in a clustering algorithm for RBFNN desigen for unction approximation," Neural Computing & Applications ,vol.17,no.1,pp.75-89, 2008.

[15] R. Zhang, N. Sundararajan, G.B. Huang and P. Saratchandran, "An efficient sequential rbf network for bio-medical classification problems," in: Proceedings of the International Joint Conference on Neural Networks, Budapest, pp. 2477–2482. 2004.

[16] Y. Huang, N.Li, Nianping and Y.Shi, "Automated fault detection and diagnosis for an air handling unit based on a GA-trained RBF network," Proceedings of 2006 International Conference on Communications, Circuits and Systems, pp.2038-2041.2006.

[17] T.M.Cover, "Geometrical and statistical properties of systems of linear inequalities with application in pattern recognition," IEEE Trans. Electron. Comput. EC, vol.14, no.2, pp.326–334,1965.

[18] D.S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," Complex Systems, vol.2 ,321–355,1988.

[19] N.B. Karayiannis, "Reformulated radial basis neural networks trained by gradient descent," IEEE Trans. Neural Networks, vol.10, no.3, pp.657–671,1999.

[20] J.Moody and C. Darken, "Faster learning in networks of locally tuned processing units," Neural Computation, vol.1,no. 3, pp. 281–294,1989.

[21] J.H. Holland, "Adaptation of natural and artificaial system," Ann Arbor:The university of Michigan Press,1975.

[22] D.E. Goldberg, "Genetic Algorithms in Search, Optimisation and Machine Learning," Addison-Wesley, Reading, 1998.

[23] K.S.Narendra and K.Parthasarathy, "Identification and control of dynamical systems using neural networks," IEEE Transactions on Neural Networks, vol.2,no.1, pp. 4–27,1990.