GAUSS-CHEBYSHEV NEURAL NETWORKS

HONG-JIE XING[1, 2, 3], BAO-GANG HU[1, 2]

[1]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[2]Beijing Graduate School, Chinese Academy of Sciences P.O. Box 2728, Beijing 100080, China
[3]Faculty of Mathematics and Computer Science, Hebei University, Baoding 071002, Hebei, China
E-MAIL: {hjxing, hubg}@nlpr.ia.ac.cn

Abstract:

This paper presents a novel neural network integrating both Gauss neural network and Chebyshev neural network. The Gauss-Chebyshev neural networks take advantages of the Gauss kernel for local approximation ability, but the Chebyshev one for global generalization ability. Numerical experiments confirm the new strategy on the better performance in comparison with Gauss neural networks. Furthermore, under the same initialization conditions, Gauss-Chebyshev neural network is more efficient than Gauss-Sigmoid neural network for regression application. All eight functions tested from the experiments show the improvements of the proposed neural networks.

Keywords:

Generalization ability; Gauss-Chebyshev; approximation ability; back propagation; learning rate; momentum constant

1. Introduction

Since having the merits of simple structure, rapid convergence rate and excellent approximation ability, Gauss neural network is widely used in many fields, such as time series analysis, pattern recognition, nonlinearity control and etc. However, the generalization and anti-noise abilities of Gauss neural network are not so good in some cases [1] [2]. Furthermore, when approximates a function including both linear and nonlinear segments, Gauss neural network may be inefficient to provide satisfying outcomes [2].

To improve the generalization ability of Gauss neural network, Shibata and Ito constructed a Gauss-Sigmoid neural network model and achieved better simulation results [3]. However, the convergence speed of Gauss-Sigmoid neural network is quite slow.

Chebyshev neural network is first introduced by Namatame and Ueda in 1992 and applied to pattern recognition [4]. Comparing with multiple layer perceptron, Chebyshev neural network is better for its high learning speed and approximation accuracy [5]. Furthermore, our previous study shows that Chebyshev neural network provides the more varieties of nonlinearities than using sigmoidal and Gaussian kernel functions [6].

To posses the advantage of Chebyshev neural network and overcome the shortcomings of Gauss and Gauss-Sigmoid neural networks, a Gauss-Chebyshev neural network is constructed in the paper. The rest of the paper are as follows. Section 2 reviews Gauss, Chebyshev and Gauss-Sigmoid neural networks. In Section 3, the topological structure and learning algorithm of Gauss-Chebyshev neural network are discussed in detail. The results concerning experiments are reported in Section 4 and some concluding remarks are given in Section 5.

2. Gauss, Chebyshev, and Gauss-Sigmoid neural networks

Gauss neural network is a three layers feedforward neural network with one hidden layer, its parametric model can be expressed as:

$$\begin{aligned} y(\mathbf{x}) &= \sum\nolimits_{k=1}^{K} w_k g_k(\mathbf{x}) \\ &= \sum\nolimits_{k=1}^{K} w_k \exp(-\sum\nolimits_{m=1}^{M} \frac{1}{2}(\frac{x_m - c_{km}}{\sigma_{km}})^2) \end{aligned}, \quad (1)$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_M)^T \in \mathbb{R}^M$ is the input vector, $(c_{k1}, c_{k2}, \ldots, c_{kM})^T$ is the center of the k th unit of the hidden layer, while $(\sigma_{k1}, \sigma_{k2}, \ldots, \sigma_{kM})^T$ the width parameter. The training algorithm for Gauss neural network usually uses gradient descent techniques.

Similar to Gauss neural network, Chebyshev neural network is a three layers feedforward neural network with one hidden layer too. Its parametric model is as follows:

$$y(\mathbf{x}) = \sum\nolimits_{k=0}^{K} w_k T_k(\mathbf{x}), \quad (2)$$

where $T_k(\mathbf{x})$ ($k = 0, 1, \ldots, K$) satisfies:

$$\begin{cases} T_0(\mathbf{x}) = 1 \\ T_1(\mathbf{x}) = \mathbf{x} \\ T_{n+1}(\mathbf{x}) = 2\mathbf{x}T_n(\mathbf{x}) - T_{n-1}(\mathbf{x}) \end{cases}, \tag{3}$$

and $n = 1, 2, \ldots, K-1$. Chebyshev neural network usually uses back propagation algorithm for training its weights.

Gauss-Sigmoid neural network is a four layers feedforward neural network with two hidden layers. The network model can be expressed as:

$$y(\mathbf{x}) = f_2(\sum_{n=1}^{N} v_n f_1(\sum_{k=1}^{K} w_{nk} g_k(\mathbf{x}) - \theta_n)), \tag{4}$$

where f_1 and f_2 are both sigmoidal functions which represent the outputs of the first hidden layer and the second hidden layer respectively. g_k ($k = 1, 2, \ldots, K$) are Gaussian functions, thus the forms of their expressions are same with the g_k in equation (1). Training algorithm for Gauss-Sigmoid neural network is back propagation algorithm too.

3. Gauss-Chebyshev neural network

To improve the generalization and approximation abilities especially approximating a function including both linear and nonlinear segments and enhance the anti-noise performance of Gauss neural network, a Gauss-Chebyshev neural network is constructed. At the same time, the aims of constructing this network are enhancing the convergence speed and improving approximation accuracy of Gauss-Sigmoid neural network.

3.1. Topological Structure

Similar to Gauss-Sigmoid neural network, Gauss-Chebyshev neural network consists of one input layer, two hidden layers and one output layer too. It is shown a single output Gauss-Chebyshev neural network in Figure 1. Let the input vector be $\mathbf{x} = (x_1, x_2, \ldots, x_M)^T$, then input $\mathbf{x}$ into the first hidden layer directly. The first hidden layer is composed of Gaussian functions g_k ($k = 1, 2, \ldots, K$), so the k th unit in the first hidden layer is:

$$g_k(\mathbf{x}) = \exp(-\sum_{m=1}^{M} \frac{1}{2}(\frac{x_m - c_{km}}{\sigma_{km}})^2), \tag{5}$$

The second hidden layer of Gauss-Chebyshev neural network consists of Chebyshev polynomials $T_n(\bullet)$ ($n = 0, 1, \ldots, N$), then the n th unit of the second hidden layer can be expressed as: $T_n(\sum_{k=1}^{K} w_{nk} g_k(\mathbf{x}) - \theta_n)$ whose form of expression is same with the formula (3), where w_{nk} is the weight connecting the n th unit of the second hidden layer and the k th unit of the first hidden layer and θ_n is the threshold.

Finally, the output of the network is:

$$\mathbf{y}(\mathbf{x}) = \sum_{n=0}^{N} v_n T_n(\sum_{k=1}^{K} w_{nk} g_k(\mathbf{x}) - \theta_n), \tag{6}$$

where v_n is the weight connecting the output unit and the n th unit of the second hidden layer.

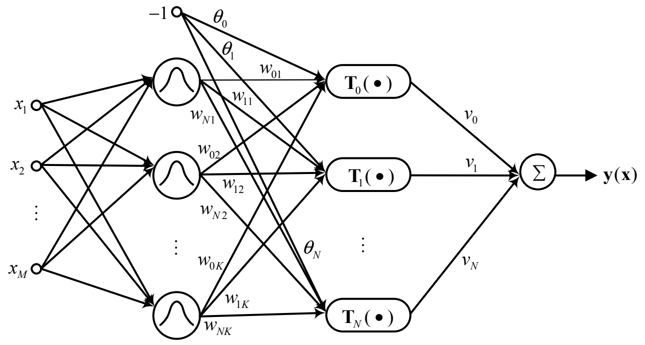

Figure 1. Gauss-Chebyshev Neural Network

3.2. Learning Algorithm

The modified back propagation algorithm [7] is used for training Gauss-Chebyshev neural network. The error function of the learning algorithm is as follows:

$$E = \frac{1}{2}\sum_{p}[y(\mathbf{x}_p) - f(\mathbf{x}_p)]^2, \tag{7}$$

where $p = 1, 2, \ldots, P$ is the number of the training data, while $y(\mathbf{x}_p)$ and $f(\mathbf{x}_p)$ are the network output and the target output for the p th input sample $\mathbf{x}_p$, respectively.

So the changes of the weights and parameters at each iteration are as follows: $\Delta v_n = -\eta \frac{\partial E}{\partial v_n}$, $\Delta w_{nk} = -\eta \frac{\partial E}{\partial w_{nk}}$, $\Delta \theta_n = -\eta \frac{\partial E}{\partial \theta_n}$, $\Delta c_{km} = -\eta \frac{\partial E}{\partial c_{km}}$, $\Delta \sigma_{km} = -\eta \frac{\partial E}{\partial \sigma_{km}}$, where η is the learning rate, while

$\frac{\partial E}{\partial v_n}$, $\frac{\partial E}{\partial w_{nk}}$, $\frac{\partial E}{\partial \theta_n}$, $\frac{\partial E}{\partial c_{km}}$, and $\frac{\partial E}{\partial \sigma_{km}}$ are the partial derivatives of the error function E with respect to v_n, w_{nk}, θ_n, c_{km}, and σ_{km}, respectively. According to the algorithm [7], finally, the changes of the weights and parameters are as follows:

$$\Delta v_n = -\eta \frac{\partial E}{\partial v_n} + \alpha \Delta v_n^{old}, \tag{8}$$

$$\Delta w_{nk} = -\eta \frac{\partial E}{\partial w_{nk}} + \alpha \Delta w_{nk}^{old}, \tag{9}$$

$$\Delta \theta_n = -\eta \frac{\partial E}{\partial \theta_n} + \alpha \Delta \theta_n^{old}, \tag{10}$$

$$\Delta c_{km} = -\eta \frac{\partial E}{\partial c_{km}} + \alpha \Delta c_{km}^{old}, \tag{11}$$

$$\Delta \sigma_{km} = -\eta \frac{\partial E}{\partial \sigma_{km}} + \alpha \Delta \sigma_{km}^{old}, \tag{12}$$

where α is the momentum constant. Finally, the learning algorithm of Gauss-Chebyshev neural network can be induced as follows:

- *Step 1*. Initialize the centers $c_k = (c_{k1}, c_{k2}, \ldots, c_{kM})^T$ and the width parameters $\sigma_k = (\sigma_{k1}, \sigma_{k2}, \ldots, \sigma_{kM})^T$, then initialize the weights w_{nk}, v_n and the threshold θ_n, where $k = 1, 2, \ldots, K$ and $n = 1, 2, \ldots N$. Finally, set the learning rate η and the momentum constant α be values between 0 and 1;
- *Step 2*. Compute the outputs $y(\mathbf{x}_p)$ of the network according to the equation (5) and (6) for the inputs $\mathbf{x}_p$, where $p = 1, 2, \ldots, P$. Then compute the error $E = \frac{1}{2}\sum_p [y(\mathbf{x}_p) - f(\mathbf{x}_p)]^2$ with the network outputs $y(\mathbf{x}_p)$ and the target outputs $f(\mathbf{x}_p)$. If the error is less than a given threshold, go to Step 3, else go to Step 4;
- *Step 3*. If the times of iteration exceed the maximum times, go to Step 4, else modify the weights and parameters as follows: $v_n^{new} = v_n^{old} + \Delta v_n$, $w_{nk}^{new} = w_{nk}^{old} + \Delta w_{nk}$, $\theta_n^{new} = \theta_n^{old} + \Delta \theta_n$, $c_{km}^{new} = c_{km}^{old} + \Delta c_{km}$, $\sigma_{km}^{new} = \sigma_{km}^{old} + \Delta \sigma_{km}$. And turn to Step 2;
- *Step 4*. End.

4. Numerical Experiments

To testify the higher performance of Gauss-Chebyshev neural network, several experiments are conducted as follows. Three of them are performed to compare Gauss neural network with Gauss-Chebyshev neural network and one for comparing Gauss-Sigmoid neural network with Gauss-Chebyshev neural network.

4.1. Experiments with Gauss and Gauss-Chebyshev Neural Networks

In the first experiment, the training data are generated by the following function [2], namely Func 1:

$$f(x) = \begin{cases} 1.8 & x \in [-6, -3] \\ -x - 1.2 & x \in [-3, 0] \\ 3e^{-0.1x}\sin(0.2x^2) - 1.2 & x \in [0, 2] \\ 0.6 & x \in [2, 6] \end{cases}.$$

121 training data are selected uniformly from the above function on the domain [-6, 6] by step length chosen to be 0.1. The number of units in each layer of the Gauss neural network is 1-27-1, while the Gauss-Chebyshev neural network 1-10-5-1. The learning rates η for the two networks are both 0.001 and the momentum constants α 0.001 too. The times of maximum iteration for the two networks are both 3000. Then the approximation results of the Gauss neural network and the Gauss-Chebyshev neural network are shown in Figure 2(a) and Figure 2(b), respectively. Note that the free parameters of Gauss neural network contain v_n, c_{km}, and σ_{km}, while Gauss-Chebyshev neural network v_n, w_{nk}, θ_n, c_{km}, and σ_{km}. The total numbers of free parameters for both networks are shown in the captions of their corresponding figures, which are denoted by signs N_P.

It is shown in Figure 2(a) and Figure 2(b) that in the intervals [-6, 0] and [2, 6] the approximation results of the Gauss-Chebyshev neural network is much better than that of the Gauss neural network. The reason is that a hidden layer composed by Chebyshev polynomials is added into the Gauss neural network, which can improve the accuracy of Gauss neural network for approximating a function with linear segments or even constant values in some intervals.

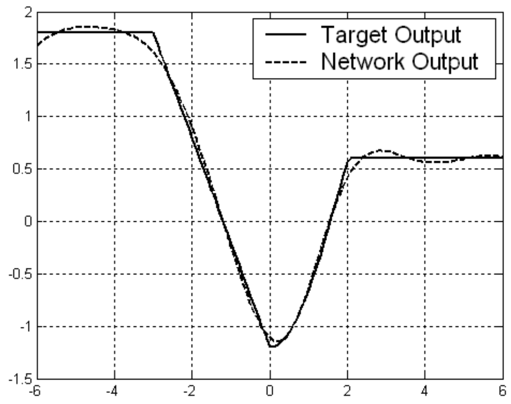

Figure 2(a). Gauss Neural Network (N_P =81)

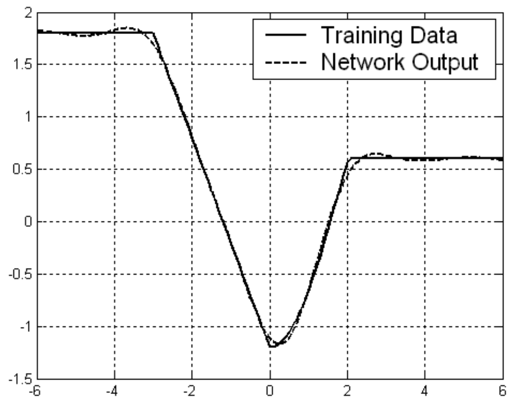

Figure 2(b). Gauss-Chebyshev Neural Network (N_P =80)

Another experiment is conducted to demonstrate the influences of the noises to the two networks. The function to be approximated is: $f(x)=\sin(2\pi x)$, $x\in[0,1]$, namely Func 2. 21 training data are selected from $f(x)+\varepsilon$ on the domain [0, 1] by step length 0.05, where $\varepsilon \sim N(0,0.1^2)$ is a Gaussian white noise. The number of units in each layer of the Gauss neural network is 1-15-1, while the Gauss-Chebyshev neural network is 1-6-4-1. The parametric settings for the two networks are same with the above experiment except that the times of maximum iteration. Then the approximating errors E between the network outputs and the target outputs for different iteration epochs are shown in Table 1. The sign N_I in the table denotes the number of iteration for the training of the two networks.

Table 1. The approximation errors of different epochs for the Gauss and Gauss-Chebyshev neural networks

N_I	Gauss (N_P =45)	Gauss-Chebyshev (N_P =44)
3000	0.0378	0.0299
6000	0.0393	0.0245
12000	0.0502	0.0245
24000	0.0735	0.0271

It is shown in Table 1 that after 3000 epochs, the approximation error of the Gauss-chebyshev neural network goes on decreasing while the Gauss neural network begins to increasing, which means that the convergence speed of the Gauss neural network is faster than the Gauss-Chebyshev one in this experiment. However, the noises begin to producing impacts on the two networks after 3000 and 12000 epochs respectively. It can be deduced that the performance of anti-noise ability of the Gauss-Chebyshev neural network is better than the Gauss one because its increasing rate of approximation error after 6000 epochs is slower than that of the Gauss neural network after 3000 epochs.

Finally, to compare the generalization ability of Gauss and Gauss-Chebyshev neural networks, two functions are utilized for generating training and testing data. They are shown in Table 2. The training and testing data are selected uniformly from the two functions on their corresponding domains without noises. The numbers of training and testing data are shown in Table 2.

Table 2. Functions are used for generating training and testing data

Name	Function	Domain	Train data	Test data
Func 3	$f(x)=\sin(\pi x)$	[-1, 1]	21	201
Func 4	$f(x)=e^{-(x-1)^2}+e^{-(x+1)^2}$	[-2.5, 2.5]	51	101

The number of units in each layer of the Gauss neural network is 1-15-1, while the Gauss-Chebyshev neural network 1-6-4-1. The learning rates η and the momentum constants α for the two networks are both 0.001. For approximating the underlying function 3, the times of maximum iteration for the two networks are both 3000. For approximating the underlying function 4, the times of maximum iteration for the Gauss-Chebyshev neural network is 1000 while 3000 for the Gauss neural network. Similar to the second experiment, the numbers of the total free parameters N_P for the two networks are 45 and 44, respectively.

After repeated 100 times, the box plots of the training and testing errors for the two networks are shown in Figure 3. The upper part of Figure 3 contains the training and testing errors of the two networks for the underlying function 3, while the lower part for the underlying function 4. It is shown in Figure 3 that both the mean values of the training error and the mean values of the testing errors of the Gauss-Chebyshev neural network are less than that of the Gauss neural network for the two functions. Furthermore, the margins between the testing errors of the two networks are much larger than that between the training errors. Then it can be declared that the generalization ability of Gauss-Chebyshev neural network is superior to Gauss neural network.

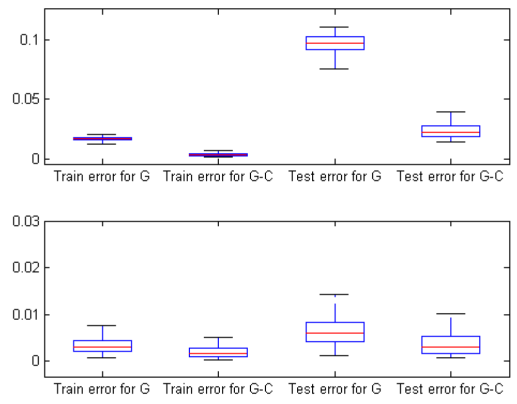

Figure 3. The training and testing errors for Gauss neural network (N_P =45) and Gauss-Chebyshev Neural Network (N_P =44) for the underlying function 3 (Upper half part) and function 4 (Lower half part)

4.2. Experiments with Gauss-Sigmoid and Gauss-Chebyshev Neural Networks

In this experiment, five functions are used for generating training data. They are shown in Table 3. Note that the training data are all selected uniformly from the five functions on their corresponding domains. The numbers of units in each layer of the Gauss-Sigmoid and the Gauss-Chebyshev neural networks are both set to be 1-10-5-1. The learning rates η and the momentum constants α for the Gauss-Chebyshev neural networks are both 0.001, while 0.01 for the Gauss-Sigmoid neural networks. For the function 7 and 8 in Table 3, the times of maximum iteration for both networks are 1000, while 3000 for the other two functions. Note that the numbers of the total free parameters N_P for both networks are both 80.

Table 3. Functions are used for generating training data

Name	Function	Domain	Train data
Func 5	$f(x)=\sin(x)e^{-0.4x}$	[0, 10]	201
Func 6	$f(x)=\dfrac{1}{1+(x-2)^2}$	[0, 4]	401
Func 7	$f(x)=e^{x\sin(\pi x)}$	[-1, 1]	201
Func 8	$f(x)=\sin(x)/x$	[-10, 10]	100

As the training stopped, the errors between the target outputs and the corresponding network outputs for the Gauss-Sigmoid and Gauss-Chebyshev neural networks are shown in Table 4.

Table 4. The errors for the Gauss-Sigmoid and Gauss-Chebyshev neural networks after training

Model	Func 5	Func 6	Func 7	Func 8
Gauss- Sigmoid (N_P =80)	0.9115	0.1294	3.9704	0.4320
Gauss- Chebyshev (N_P =80)	0.0759	0.0015	0.0129	0.0093

It is shown in the Table 4 that after the same epochs, the approximation errors of the Gauss-Chebyshev neural networks are much less than the Gauss-Sigmoid neural networks for the five underlying functions.

To compare the convergence speeds of the Gauss-Sigmoid and Gauss-Chebyshev neural networks, the function 2 is used again. 101 training data are chosen uniformly from the function 2 on its domain by step length 0.01. The parameters of the two networks are same with the above experiment except that the times of the maximum iteration for the Gauss-Sigmoid neural network is 6000. Then the approximation results of the Gauss-Sigmoid neural network and the Gauss-Chebyshev neural network are shown in Figure 4(a) and Figure 4(b), respectively.

It is shown in the upper parts of the Figure 4(a) and Figure 4(b) that though after 6000 epochs, the approximation error of the Gauss-Sigmoid neural network is still bigger than the error of the Gauss-Chebyshev neural network which after only 1000 epochs. Furthermore, from the lower parts of the Figure 4(a) and Figure 4(b) we are further confirmed that the convergence speed of the Gauss-Chebyshev neural network is more quickly than the Gauss-Sigmoid neural network.

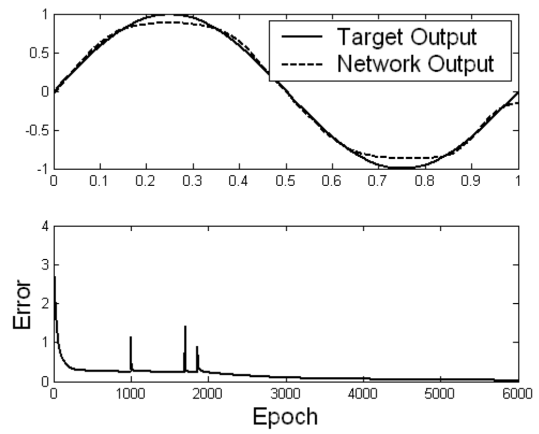

Figure4 (a). The approximation result and error curve of the Gauss-Sigmoid neural network (N_p =80)

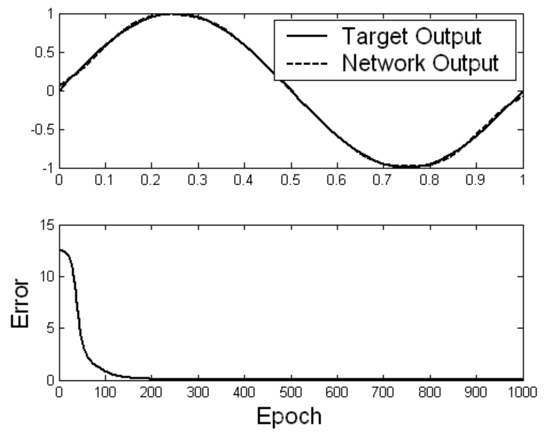

Figure4 (b). The approximation result and error curve of the Gauss-Chebyshev neural network (N_p =80)

5. Final Remarks

The numerical studies in Section 4 show that Gauss-Chebyshev neural network can improve the generalization, approximation, and anti-noise abilities of Gauss neural network. With the same topological structure and parametric setting, comparing with Gauss-Sigmoid neural network, Gauss-Chebyshev neural network can approximate a function more quickly and exactly. However, there exist issues for the further study on Gauss-Chebyshev neural network as follows:

a. To investigate the generalization ability of Gauss-Chebyshev neural network from nonlinear analysis;

b. To develop a methodology of forming the structure of the network;

c. To enhance the training speed of Gauss-Chebyshev neural network by a more efficient learning algorithm.

Acknowledgements

This work is supported by Natural Science Foundation of China (#60275025, #60121302).

The authors would like to thank Yu-Jiu Yang and Yong Wang for their useful discussions.

References

[1] Z. –O. Wang, and T. Zhu, "An efficient learning algorithm for improving generalization performance of radial basis function neural network", Neural Networks, Vol. 13, pp. 545-553, 2000.

[2] J. –R. Tsai, P. –C. Chung, and Ch. -I Chang, "A sigmoidal radial basis function neural network for function approximation", Proceedings of IEEE International Conference on Neural Networks, Vol. 1, pp. 496-501, 1996.

[3] K. Shibata, and K. Ito, "Gauss-Sigmoid neural network", Proceedings of International Joint Conference on Neural Networks, Vol. 2, 1203-1208, 1999.

[4] A. Namatame, and N. Ueda, "Pattern classification with Chebyshev neural networks", Int. J. Neural Networks, Vol. 3, pp. 23-31, Mar. 1992.

[5] P. Akritas, I. Antoniou, and V. V. Ivanov, "Identification and prediction of discrete chaotic maps applying a Chebyshev neural network", Chaos, Solitons and Fractals, Vol. 11, pp. 337-344, 2000.

[6] B. –G. Hu, H. –J. Xing, and Y. –J. Yang, "Geometric Interpretation of nonlinear approximation ability for feedforward neural networks", F. Yin, J. Wang, and C. Guo (Eds.): Advances in Neural Networks-ISNN2004, Springer-Verlag Berlin Heidelberg, Vol. 1, pp. 7-13, 2004.

[7] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal by error progation", D. E. Rumelhart, J. L. McClelland (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, MIT Press, Cambridge, Vol. 1, 1986.