Neural Network Performance Analysis Using Hanning Window Function as Dynamic Learning Rate

Md. Mehedi Hasan, Arifur Rahaman, Munmun Talukder, Mobarakol Islam, Mirza Md. Shahriar Maswood, Md. Mostafizur Rahman
Dept. of Electronics & Communication Engineering,
Khulna University of Engineering & Technology,
Khulna-9203, Bangladesh
{mehedi_ece, arifur_ece} @yahoo.com, summon.kuet@gmail.com, mobarak.islam@yahoo.com, shaonkuetece05@gmail.com, mostafiz963@yahoo.com

***Abstract*—In human brain the neurons are excited in a dynamic way. The response of different neurons varies widely because of the variation of electrical signal in every neuron. Backpropagation(BP) is a training algorithm where the learning of the Neural Network (NN) is done by a constant learning rate (LR). But to replicate the human brain function, the learning rate should be changed as the excitation of different neurons. In this paper a new learning algorithm is proposed called Hanning Window Neural Network (HWNN) to train the network. Here the window function is used to make the learning rate dynamic called Hanning learning rate (HLR) and for this dynamic learning rate the neural network outperforms than the existing BP algorithm. HWNN is extensively tested on five real world benchmark classification problems such as ionosphere, australian credit card, time series, wine and soybean identification. The proposed HWNN outperforms the existing BP in terms of generalization ability and also convergence rate.**

Keywords-- backpropagation; neural network; Hanning window function; learning rate; generalization ability; convergence rate.

I. INTRODUCTION

Machine learning is a scientific discipline that automatically learns to recognize complex patterns and make intelligent decisions based on data [1]. Artificial neural networks are the networks in where the artificial neurons are connected to imitate the original biological neural structure. So a neural network is a computational model that tries to simulate the structure and functional aspects of biological neural networks [2]. Although biological network is so much complex, very few of this complexity is considered to design the artificial neural network.

Back propagation algorithm learns the weights for a multilayer network, given a network with a fixed set of units and interconnections. So BP is a supervised learning technique used for training Multi-Layer Perceptrons (MLPs) [3]. It employs gradient descendent rule to attempt to minimize the squared error between the network output values and the target values for these outputs. In conventional BP, weights in the network are adjusted by the algorithm to make the error decrease along a descent direction [4]. Though BP is a well practiced algorithm, BP suffers from a number of problems [5]. The limitations are - it is extremely slow, it may be trapped into local minima before converging to a solution and having trapped into local minima BP may lead to failure in finding a global optimal solution [6], oscillations may occur during training (this usually happens when the learning rate is high). And the back propagation (BP) is trained by a constant learning rate so it has also poor generalization ability. There are so many attempts to enhance BP performance such as making LR and activation factor adaptive, the performance of a NN is increased [7], convergence speed can be increased for BP algorithm by using adaptive accuracy of weights, instead of using fixed weight accuracy [8], optimizing dynamic LR by using an efficient method of deriving the first and second derivatives of the objective functions with respect to the LR [9]. Also for better performance of BP, noise injection into the network [10] is done. It has been proved that the injection of noise into the hidden neurons has a similar effect as the injection of noise with the training patterns [11]. For faster convergence maximized gerdient function has been used [12]. So for having these limitations of BP algorithm a novel algorithm is proposed here which uses the Hanning window function as learning rate. The window function has a dynamic range of values. This window function is a sinusoidal function and it helps to make the Artificial Neural Network (ANN) more similar with the biological neural network.

In this paper, a new NN learning algorithm called Hanning Window Neural Network (HWNN) is proposed. Here the learning rate is made dynamic by using Hanning window function as learning rate called HLR. And so the artificial neurons also excited by almost a similar way to the biological neurons. In every single iteration the learning rate is different and so the training is faster than BP. HWNN is applied on five benchmark problems such as australian credit card, ionosphere, time series, wine and soybean. And for the

978-1-4799-0400-6/13/$31.00 ©2013 IEEE

dynamic value of learning rate, HWNN outperforms BP in terms of generalization ability and convergence rate.

The organization of this paper is as follows. In Section II, we describe about the proposed HWNN. Five experiments are studied in Section III to support our theory. The discussion on HWNN is presented in section IV. Finally Section V concludes the paper.

II. HWNN

In this work the learning rate is made dynamic by using Hanning window which is a sinusoidal function [13]. The window function varies with time so it produces the dynamic value of the learning rate and this is defined by HLR.

HLR is produced by the following equations:

$$d\,(t) = .5 - [.5 * \cos\{(2 * 180 * t)/3\}];$$

$$HLR(t) = d(t)\ /\beta;$$

β is user specified constant parameter. For this experiment β is taken greater than 1. Fig. 1 shows the Hanning window function and Fig. 2 shows the amplitude spectrum of Hanning window.

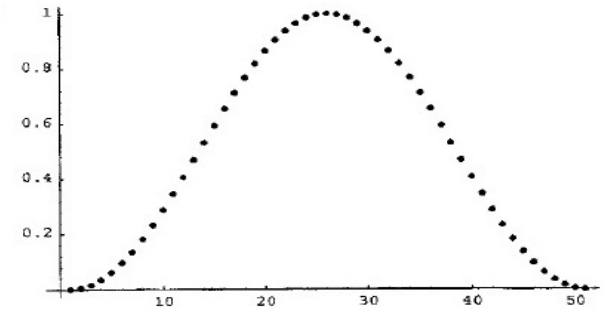

Figure 1. Hanning window

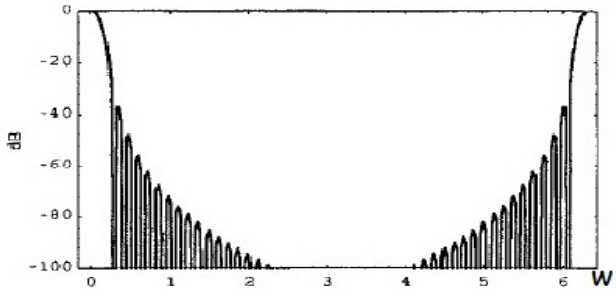

Figure 2. Amplitude spectrum of Hanning window

A feed-forward neural network with one input layer, one hidden layer, and one output layer is shown in Fig. 3. Let, the numbers of input, hidden and output units are I, J and K respectively. W_{ij} is the network weight that connects input unit i and hidden unit j, and W_{jk} is the network weight that connects hidden unit j and output unit k. The number of training examples is n and any arbitrary n-th training example is $\{x_{n1}, x_{n2}, \ldots\ldots, x_{pI}; y_{n1}, y_{n2}, \ldots\ldots, y_{nK}\}$, where x_n is the input vector and y_n is the target vector. h_{nj} and O_{nk} are the outputs of hidden unit j and output unit k for the n-th training example. Δ represents the difference between the current and new value of the network weights. The consecutive steps of HWNN are given below.

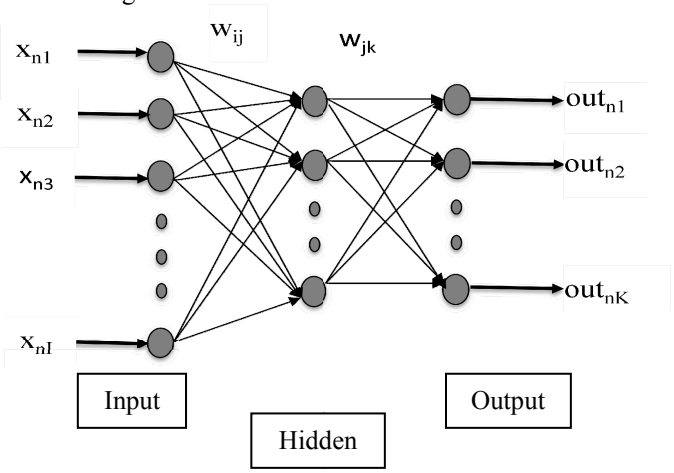

Figure 3. A feed-forward network with single hidden layer

Step 1: Initialize the network weights in an arbitrary interval (-w, +w). Set the iteration number, t = 0.

Step 2: Projected Hanning Learning Rate (HLR) with traditional learning rate η by using following equations:

$$d\,(t) = .5 - [.5 * cos\{(2 * 180 * t)/3\}];$$

$$HLR(t) = d(t)\ /\beta;$$

Here is used to rescale the HLR.

Step 3: Calculate and for the n-th training example by following equations:

$$h_{nj} = f(\sum_{i=1}^{i} W_{ij}\, X_{ni})$$

$$Out_{nk} = f\,(\sum_{i=1}^{i} W_{jk}\, h_{nj})$$

An adaptive activation function is used as following

$$f(x) = 1/(1 + e^{-x})$$

Step 4: Updating of the output weight w_{jk} & hidden weight w_{ij} by the following equations:

$$w_{jk} = w_{jk} + \Delta w_{jk}$$

$$w_{ij} = w_{ij} + \Delta w_{ij}$$

ΔW_{jk} and ΔW_{ij} are the changes of weights for the output and hidden layer,

$$\Delta W_{jk} = -HLR(t)\,\frac{\partial E_n}{\partial W_{jk}}$$

$$= -HLR(t)\,\partial_{nk} h_{nj}$$

$$\Delta W_{ij} = -HLR * \partial_{nj} * X_{ni}$$

$$\text{where, } E_n = \frac{1}{2}\sum_{k=1}^{k} [Out_{nk} - T_{nk}]^2$$

$$\partial_{nk} = (Out_{nk} - T_{nk})\, Out_{nk}(1 - Out_{nk})$$

$$\partial_{nj} = h_{nj}(1 - h_{nj}) \sum_{k=1}^{k} \partial_{nk}\, W_{jk}$$

When this process is completed for all the training examples, iteration is completed. Then number of iteration t is increased one and move to next Step 5.

Step 5: Calculate the mean square error for the iteration of Step 4 as follows:

$$E(t) = \frac{1}{2nk}\sum_{n=1}^{n}\sum_{k=1}^{k}[Out_{nk} - T_{nk}]^2$$

Check the termination condition, If E(t) satisfies the termination criteria then stop the training and test the network, otherwise go to Step 2.

III. EXPERIMENTAL STUDIES

A. Hanning Window LearningRrate

Hanning window is a sinusoidal window function. The learning rate is made dynamic by using the Hanning window function which called Hanning window learning rate (HLR). For this dynamic value of the HLR, the network achieves a faster convergence.

B. Characteristics of Benchmark datasets

HWNN is applied on five benchmark classification problems – ionosphere, australian credit card, time series, wine and soybean identification. The datasets are collected from the University of California at Irvine (UCI) Repository of the machine learning database [14] and PROBEN1 [15]. Some characteristics of the datasets are listed in TABLE I. For example, diabetes problem is a classification problem having a total examples or patterns of 768 with 8 attributes and 2 classes. Other problems are arranged in a similar fashion. The total examples of a dataset are divided into training examples, validation examples, and testing examples. The training examples are used to train the NN, validation examples are used to terminate the training process and the trained NN is tested with the testing examples.

TABLE I. CHARACTERISTIC OF BENCHMARK DATASETS.

Datasets	Number of		
	Total Examples	Input Attributes	Output Classes
Ionosphere	351	33	2
Credit Card	690	51	2
Time Series	740	3	2
Wine	222	13	3
Soybean	683	82	19

C. Experimental Process and Comparison

In order to investigate the convergence speed and generalization ability of NN learning with different algorithms mentioned above, simulations are carried out with wide range of benchmark problems. Since no analytical techniques are available to study the learning speed of BP algorithm, simulation with different problems and comparison are the usually adopted means to evaluate the effectiveness of a modification. Here investigation has been done with five different problems. A feed-forward NN with single hidden layer is taken for each problem. The numbers of input and output units of the NN are equal to the number of attributes and number of classes of the dataset respectively. The number of hidden units is taken arbitrarily for several datasets. The weights of the NN are initially randomized in the interval (-1, +1). The training is stopped when the mean square error on the validation set increases in consecutive four iterations and for few example training is terminated based on predefined training error. In order to check the generalization performance of trained NN, the 'testing error rate' (TER) is computed. TER is the ration of the number of classified testing examples to the number of testing examples. The experimental results are furnished in terms of TER and number of required iterations. Mean and SD indicate the average and standard deviation values of 20 independent trials. HWNN is compared with standard BP. To make a fair comparison, the LR(η) of BP is selected in such a way that, this fixed value of learning rate is always an intermediate point in the range [min(HLR) & max(HLR)] .

1) Ionosphere: A 33-4-2 NN is trained with first 175 examples, and 88 examples for validation, and last 88 examples for testing whereas the total number of examples for this problem is 351. The obtained results are reported in TABLE II and the error curves are shown in Fig. 4. When LR is 0.08, BP requires 92.00 iterations to obtain TER of 0.1040 where as HWNN requires only 51.20 iterations to obtain TER of 0.0892. Hence HWNN obtained good generalization ability than BP. Fig. 4 shows that HWNN converges the training faster than standard BP.

TABLE II. NUMBER OF REQUIRED ITERATIONS AND TERS WITH BP AND HWNN FOR IONOSPHERE PROBLEM OVER 20 INDEPENDENT RUNS.

Serial No.	Algorithm	η /HLR	Iteration		TER	
			Mean	SD	Mean	SD
01	BP	0.08	92.00	69.9321	0.1040	0.0216
	HWNN	0.08	51.20	18.1041	0.0892	0.0191
02	BP	0.12	52.30	18.9476	0.0966	0.0286
	HWNN	0.12	44.60	25.5351	0.0852	0.0202

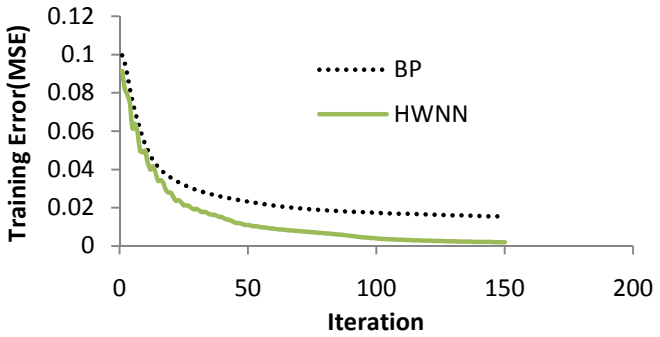

Figure 4. Training error vs. no. of iteration of BP and HWNN for ionosphere problem

2) Australian Credit Card: A 51-4-2 NN is trained with first 345 examples. The training is stopped with 173 validation examples and the NN is tested with last 172 testing examples. The numerical results are reported in Table III and the convergence curves of BP and HWNN are shown in Fig. 5. When LR is 0.08 then iteration and TER of BP is 35.50 & 0.1526 and for HWNN iteration and TER are 31.65 & 0.1422. These results certify that the generalization ability of HWNN is better than that of BP.

TABLE III. NUMBER OF REQUIRED ITERATIONS AND TERS WITH BP AND HWNN FOR CARD PROBLEM OVER 20 INDEPENDENT RUNS.

Serial No.	Algorithm	η/HLR	Iteration		TER	
			Mean	SD	Mean	SD
01	BP	0.08	35.50	7.7233	0.1526	0.0088
	HWNN	0.08	31.65	6.3267	0.1442	0.0100
02	BP	0.12	25.70	4.1845	0.1459	0.0078
	HWNN	0.12	20	2.9496	0.1107	0.0103

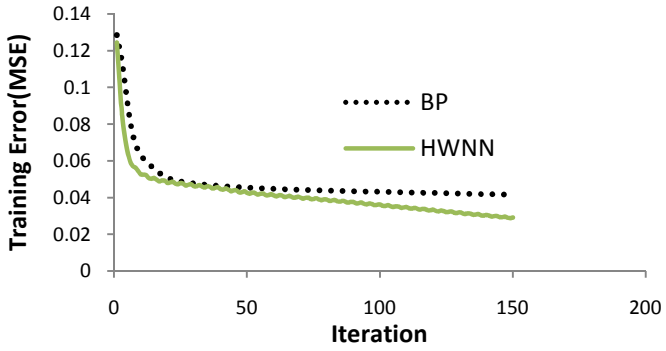

Figure 5. Training error vs. no. of iteration of BP and HWNN for card problem.

3) Time series: Time series is generated by Henon Map [16]. Henon Map is developed by the equations of $x_{n+1} = 1 + ax_n^2 + by_n$ and $y_{n+1} = x_n$, where initial value of a=-1.4, b=0.3, x(1)=0.63135448, y(1)=0.18940634 respectively. This problem holds 740 examples. The size of NN is considered as 3-2-2. Here the first 500 patterns belong to train and the last 240 examples are used for testing. The experimental results are reported in TABLE IV and the error curves are shown in Fig. 6. When LR is 0.08, iteration and TER are 39.75 & 0.2087 for BP and 34.65 & 0.2023 for HWNN respectively. This ensures that HWNN carries a faster convergence rate over BP.

TABLE IV. NUMBER OF REQUIRED ITERATIONS AND TERS WITH BP AND HWNN FOR TIME PROBLEM OVER 20 INDEPENDENT RUNS.

Serial No.	Algorithm	η /HLR	Iteration		TER	
			Mean	SD	Mean	SD
01	BP	0.08	39.75	18.4658	0.2087	0.0096
	HWNN	0.08	34.65	13.3089	0.2023	0.0083
02	BP	0.12	30.55	14.4273	0.2092	0.0075
	HWNN	0.12	23.85	8.7251	0.2019	0.0087

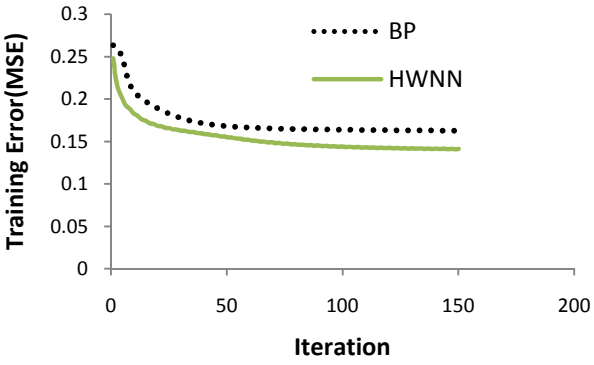

Figure 6. Training error vs. no. of iteration of BP and HWNN for time problem.

4) Wine: A NN size of 13-4-3 is chosen for training. The results are shown in TABLE V and the error curves are shown in Fig. 7. The first 178 patterns are used for training and last 44 patterns are used for testing. When the LR is 0.08 then iteration and TER of BP is 24.65 & 0.2193. But for HWNN these values are 23.55 & 0.1773 respectively. Which shows that the generalization ability and the convergence rate of HWNN is better than BP.

TABLE V. NUMBER OF REQUIRED ITERATIONS AND TERS WITH BP AND HWNN FOR WINE PROBLEM OVER 20 INDEPENDENT RUNS

Serial No.	Algorithm	η/HLR	Iteration		TER	
			Mean	SD	Mean	SD
01	BP	0.08	24.65	4.1264	0.2193	0.1182
	HWNN	0.08	23.55	6.3204	0.1773	0.1140
02	BP	0.12	20.40	4.0423	0.2568	0.1078
	HWNN	0.12	18.55	3.2323	0.2295	0.1046

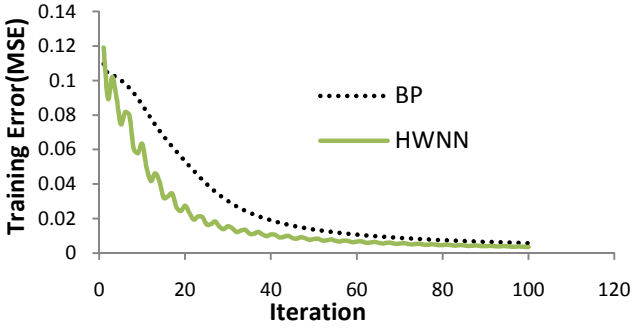

Figure 7. Training error vs. no. of iteration of BP and HWNN for wine problem.

5) Soybean: This problem has total 683 examples. The first 342 examples are used for training, 171 examples for validation and last 170 examples for testing.We take a NN of

82-4-19. The experimental results are listed in TABLE VI for different LR. When LR is 0.08, number of iterations and TER of standard BP are 839.0 & 0.2515 respectively while these are 303.45 & 0.2409 for HWNN. It is clear that HWNN outperforms than BP in terms of convergence rate and generalization ability. Fig. 8 shows the training error of BP and HWNN with respect to number of iteration. From the data and the curves it is certified that HWNN converges the training faster and also has better generalization ability than standard BP.

TABLE VI. NUMBER OF REQUIRED ITERATIONS AND TERS WITH BP AND HWNN FOR SOYBEAN PROBLEM OVER 20 INDEPENDENT RUNS.

Serial No.	Algorithm	η/HLR	Iteration		TER	
			Mean	SD	Mean	SD
01	BP	0.08	839.0	1173.9	0.2515	0.1090
	HWNN	0.08	303.45	182.1476	0.2409	0.0899
02	BP	0.12	702.1	667.9747	0.2209	0.0812
	HWNN	0.12	212.90	132.2452	0.2118	0.0792

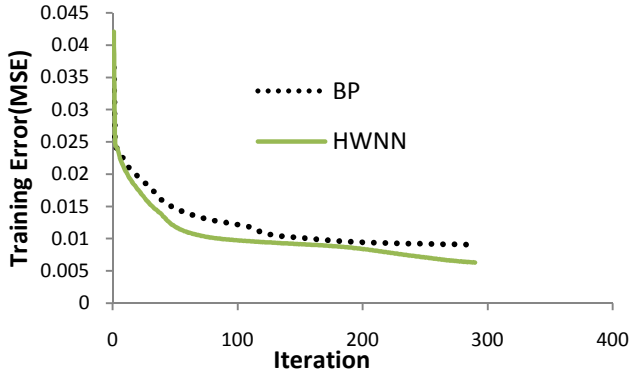

Figure 8. Training error vs. no. of iteration of BP and HWNN for soybean problem.

IV. DISCUSSION

Today it is a challenging job to make a neural network that has the characteristics almost similar to the biological network. Though it is not possible to imitate completely the biological network but it can be done to the neural network to response dynamically. Because human brain thinks or get excited in a dynamic way so by making the learning rate dynamic it is possible to make the artificial neural network responses in a dynamic way. In this work we made the learning rate dynamic. The difference between BP and HWNN is that BP trains NN with a fixed value of LR, while HWNN makes the LR dynamic during the training. In HWNN the LR is made dynamic by using the Hanning window function as LR (HLR). So HWNN outperforms in terms of convergence rate and generalization ability.

In the NN training examples which have given above are terminated based on validation examples. When the mean square error on the validation set starts to increase or satisfies the stopping criteria, HWNN stops the training.

V. CONCLUSION

In this paper a novel supervised algorithm called HWNN has been proposed. This proposed method consists of a dynamic learning rate called HLR which involves Hanning window function. Standard BP trains NNs with a constant value of LR. But in human brain the neurons get excited dynamically. For this purpose, HWNN trains NNs with dynamic LR. And for this dynamicity of LR HWNN gives better result than standard BP. Promising results are obtained with different kinds of real world benchmark problems such as ionosphere, australian credit card, time series, wine and soybean. The results show that the convergence is faster in HWNN than BP.

REFERENCES

[1] M. Xue and C. Zhu, "A Study and Application on Machine Learning of Artificial Intelligence", In the proceeding of International joint Conference on Artificial Intelligence, pp. 272-274, 2009.

[2] S.W. Li, "Analysis of Contrasting Neural Network with Small-World Network" , In the Proc. of International Seminar on Future Information Technology and Management Engineering, pp. 57 - 60, 2008.

[3] S. Hooshdar and H. Adeli, "Toward intelligent variable message signs in freeway work zones: A neural network approach", Journal of Transportation Engineering, pp. 83–93, 2004.

[4] . X. H. Yu and G. A. Chen, "Efficient backpropagation learning using optimal learning rate and momentum", Neural Networks, pp. 517–527, 1997.

[5] S. Haykin, "Neural Networks, A Comprehensive Foundation", New York 10022: IEEE Society Press, Macmillan College Publishing, 1994.

[6] M. Gori and A. Tesi, "On the problem of local minima in backpropagation," IEEE Trans. Pattern Anal. Machine Intell., vol. 14, pp. 76–86, 1992.

[7] C. C. Yu, and B. D. Liu, "A Backpropagation Algorithm With Adaptive LR And Momentum Coefficient", In the Proc. of International joint Conference on neural network, pp. 1218-1223, 2002.

[8] C. Gonzalo, Q. Tian, Y. Fainman, and S. H. Lee, "Fast Convergence of the Backpropagation Learning Algorithm by Using Adaptive Accuracy of Weights", Proceedings of the 35th Midwest Symposium on Circuits and systems, vol. 2, pp. 1221-1224, 1992.

[9] X. H. Yu, "Dynamic LR Optimization of the Backpropagation Algorithm", IEEE Transactions on Neural Networks, vol. 6, no. 3, May 1995.

[10] Kevin Ho, Chi-sing Leung and John Sum, "On Weight-Noise-Injection Training", Advances in Neural Information Processing, LNCS-5507, 2009.

[11] Nait Charif Hammadi and Hideo Ito, "Improving the Performance of Feedforward Neural Networks by Noise Injection into Hidden Neurons", Journal of Intelligent and Robotic Systems, 21, pp. 103–115, 1998.

[12] S.U. Ahmed, M. Shahjahan, K. Murase, "Maximization of the gradient function for efficient neural network training ", In the proc.of 13th International Conference on Computer and Information Technology (ICCIT), pp. 424 – 429, 2010.

[13] Alexander D. Poularikas "The Handbook of Formulas and Tables for Signal Processing" , Boca Raton: CRC Press LLC, 1999.

[14] A. Asuncion and D. Newman, ''UCI Machine Learning Repository", Schl. Inf. Comput. Sci., Univ. California, Irvine, CA, 2007.

[15] L. L. Prechelt, "PROBEN1-A set of neural network benchmark problems and benchmarking rules", Technical Report 21/94, Faculty of Informatics, University of Karlsruhe, 1994.

[16] Pan Ying, Li Bo, Liu Yuncheng, "T-S fuzzy identical synchronization of a class of generalized Henon hyperchaotic maps", In the proc.of IEEE International Conference on Information and Automation (ICIA), pp. 623 - 626, 2010.