EMPIRICAL STUDY OF FFANN TOLERANCE TO WEIGHT STUCK AT MAX/MIN FAULT

Amit Prakash Singh, Chandra Shekhar Rai[1] and Pravin Chandra[2]

[1]University School of Information Technology, GGS Indraprastha University
Kashmere Gate, Delhi – 110 403, India
aps.ipu@gmail.com, csrai_ipu@yahoo.com
[2]Iinstitute of Informatics & Communication, University of Delhi South Campus
Delhi, India
pc_ipu@yahoo.com

ABSTRACT

Fault tolerance property of artificial neural networks has been investigated with reference to the hardware model of artificial neural networks. Weight fault is an important link, which causes breakup between two nodes. In this paper three types of weight faults have been explained. Experiments have been performed to demonstrate fault tolerance behavior of feedforward artificial neural network for weight-stuck-MAX/MIN fault. Effect of weight-stuck-MAX/MIN fault on trained network has been analyzed in this paper. The obtained results suggest that networks are not fault tolerant to this type of fault.

KEYWORDS

Artificial Neural Network, Fault Tolerance, Weight Fault

1. INTRODUCTION

An Artificial Neural Network (ANN) is an information-processing paradigm that is inspired by the biological nervous system, i.e., the human brain [2]. The key element of this paradigm is the novel structure of the information processing system. Artificial neural network have the potential for parallel processing due to the implementation on Application Specific Integrated Circuit (ASIC) [4] or Field Programmable Gate Array (FPGA) [5][6][7]. The input-output function realized by neural network is determined by the value of its weights.

In the case of biological neural network, tolerance to loss of neurons has high priority, since a graceful degradation of performance is very important for survival of the organism. Fault tolerance measures the capacity of neural network to perform the desired task under given fault condition. It also maintains their computing ability when a part of the network is damaged or removed. In [8], the study of fault tolerant properties of the neurons has been reported for partial fault tolerance by replication and training and the assertion is that Triple Modular Replication (TMR) leads to a fault tolerant network. This is a one of the popular technique in digital system.

Fault tolerances of ANN have been studied in [1][15]. Fault tolerance of ANN may be characterized/categorized on the following aspects:

(i) Weight error: Weight stuck at zero/max/min
(ii) Neuron error: Node stuck at zero/max/min

(iii) Input pattern errors: injecting noise during the training phase.

The focus of this paper is to study effect of weight stuck at MAX/MIN fault in the FFANN.. Experiments have been performed to demonstrate behavior of network on weight stuck at MAX/MIN faults. Detailed experimentation has been explained in section 6 to demonstrate behavior of the network under weight stuck at MAX/MIN fault.

The analysis of fault tolerance of a network normally requires study of fault under weight, node and external faults [24]. *Stuck-at* model is a popular technique to study effect of fault on a given network [22], where a faulty gate delivers a constant logic one or logic zero at its output or acts as if one of its inputs is stuck at a fixed logic value. Neural networks process analog function values, and thus the range of possible faults may be even larger. *Weight stuck at MAX/MIN* fault has been chosen for the experimentation purpose, in order to make fault analysis manageable.

The emphasis of the fault tolerance investigation of ANNs has been focused on the demonstration of non-fault tolerant behavior of these networks and/or the design of paradigm for making a network fault tolerant to specific faults. This paper aims to present an effect of weight fault specifically stuck at MAX/MIN fault on a trained network. .

In this paper, Section II discusses related work; Section III explained architecture of FFANN, Section IV discusses fault model and weight fault. Section V discusses the experiments and the obtained results and fault measuring metrics for the weight-*stuck-MAX/MIN* fault while conclusion is presented in Section VI.

2. RELATED WORKS

With the widespread usage of the chip-based device of the ANN as controller [9], it has become imperative to study the behavior of these circuits under various faults, i.e., study of their fault tolerance behavior must be undertaken. The available literature on the fault-tolerance behavior of feedforward ANNs may be summarized as:

1. Demonstration of non-fault-tolerance to specific faults [8][11].
2. Regularization during training [12].
3. Enhancement of fault tolerance by design of algorithms for embedding fault-tolerance into the network, during training [13][16].
4. Redesigning the network architecture (after training) by replication of nodes and their associated weights and usage of majority voting [3][14].

Piuri [11] asserts that the network can not be considered to be intrinsically fault tolerant. Edwards and Murray [12], use the regularization effect of weight noise to design a fault tolerant network. Chin et. al. [16] demonstrate a training algorithm that uses weight value restriction (and addition of additional nodes), fault injection during training and network pruning to achieve a fault tolerant network, while [3] and [10] redesign the trained network to achieve a fault tolerant network. Chu and Wah [20] has introduced the fault tolerant neural network with hybrid redundancy that comprised spatial redundancy, temporal redundancy, and coding.

Phatak and Koren [14] devised measures to quantify the fault tolerance as a function of redundancy. Bolt *et. al*. [17] indicated that the network trained by backpropagation algorithm seldom distribute information to connection uniformly. Due to this information few connection are key components, whose failure will cause great loss to the networks. A method to improve the fault tolerance of backpropagation networks is presented in [21], which restrained the magnitudes of the connections during training process. Hammadi and Ito [13] demonstrate a training algorithm that reduces the relevance of weight. In [13], relevance of weight in each training epoch was estimated, and then decreases the magnitude of weight

3. ARCHITECTURE OF FFANN

The architecture of the proposed fault diagnosis neural network is illustrated in Figure 1. A feed forward artificial neural network (FFANN) has been chosen for the experiment purpose. Output of the network is defined as:

$$y = \sum_{i=1}^{N} \alpha_i h_i + \gamma \qquad (1)$$

Where, α_i is defined as connection strength between *i*th hidden layer node and the output node, while γ is the threshold/bias of the output node while h_i is the output of the ith hidden layer node, and it is defined as

$$h_i = \sigma(\sum_{i=1}^{m} w_{ij} x_j + \theta_i) \qquad (2)$$

In eq. (2), x_j is an input vector applied to network; w_{ij} is defined as connection strength between *j*th inputs to *i*th hidden layer node, while Θ_i is the threshold/bias of the *i*th hidden layer node and $\sigma(.)$ is the activation function used as a non linear transformation at the nodes of the hidden layer.

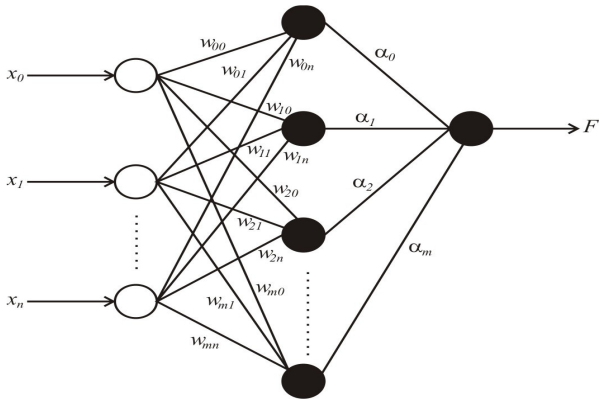

Figure 1: Schematic Architecture of FFANN with one hidden layer of nonlinear nodes

In this paper, the network output function may be called a linear function as no transformation of the net input to the output node is performed. A hyperbolic tangent sigmoid transfer function [23] has been chosen as an activation function for hidden layer node and it is defined as $\frac{e^x - e^{-x}}{e^x + e^{-x}}$ where x is the net input to the node.

4. FAULT MODELS AND METRICS

Three types of fault model exist in neural networks system [10]. Fault models are categorized weight fault, Input faults and Node faults. Missing link of interconnection between two nodes is called weight fault. The weight and node faults are often modelled as stuck-at-0 and stuck-at-

MAX/MIN and most often occur during a memory disappearance or a link disconnection in VLSI. Categorization of weight faults is explained in section 5.
Any incorrectness in the input to the adaptive machine is defined as an input fault/error. This faults/error occurs due to external disturbance or noise. Mainly these types of fault affect input vector of the machine.
Node fault is a similar type of fault as weight fault. Node faults are categorized in two types of faults, namely hidden node faults and output node faults. Three types of node faults happen in FFANN. Node fault categorized as follows:

1. Node stuck at zero
2. Node stuck at one
3. white noise in node

In this paper we consider only weight stuck at MAX/MIN fault. This fault corresponds to stuck-MAX/MIN fault in the hardware, refers to interconnection between two nodes become permanently on its maximum/minimum value.

4.1 Weight Fault

Weight fault/errors for the FFANN are defined:
(a) Weight stuck at zero (WSZ): This fault corresponds to an open fault or connection breakage between two nodes.
(b) Weight stuck at maximum/minimum (WSMa/ WSMi): Weight stuck at a value of $\pm|W|_{max}$, where $|W|_{max}$ is the maximum magnitude weight in the system. A -ve weight will pushed to $-|W|_{max}$, while a +ve weight will be pushed towards $+|W|_{max}$, which is demonstrated as WSMAXPOS in this paper. Weight stuck as a value of $\pm|W|_{min}$ is defined as, $|W_{|min}$ is the minimum magnitude weight in the system. A –ve weight will push to $-|W|_{min}$., while a +ve weight will be pushed towards $+|W|_{min}$, which is demonstrated as WSMINPOS in this paper. This allows us to model weight faults at substantially large values. Experiment demonstrates the effect of WSMax/Min fault on the trained network in section 5.
(c) White noise in weights (WNW): The presence of white noise (zero mean gaussian with finite variance) may be taken as a reflection of thermal noise or circuit degradation. This noise is different from node output noise as it is not correlated in weights leading from the same node.

5. EXPERIMENTS AND RESULTS

A small experiment was conducted to demonstrate the applicability of weight-stuck-MAX/MIN (WSMax/Min) fault on a trained neural network. The mean squared error (MSE) is used to measure the effect of WSMax/Min faults. The percentage of misclassification is suggested as a measure of fault/error, for classification problem. 30 nos. of networks were trained for the following function approximation tasks [18].

Fn1 : $y = \sin(x_1 * x_2)$; x_1,x_2 uniform in [-2,2]

Fn1: $y = \exp(x_1 * \sin(\pi * x_2))$; x_1,x_2 uniform in [-1,1]

The data set for ANN are generated by uniform sampling. The network consists of two inputs, one hidden layer and one output node (Figure 1). The detail of the architecture used is summarized in Table 1. The architecture was identified by exploratory experiments where the size of the hidden layer was varied from 5 to 30 (that is, the number of nodes in the hidden layer were varied from 5 to 30 in steps of 5) and the architecture that give the minimum error on training was used. All the hidden nodes use tangent hyperbolic activation function while the output nodes are linear.

Table 1: Architecture of network used

Sr. No.	Function	Inputs	Hidden nodes	Output nodes	No. of weight
1.	Fn1	2	25	1	101
2.	Fn2	2	15	1	61

The resilient propagation (RPROP) [19] algorithm as implemented in MATLAB 7.2 Neural Network toolbox is used with the default learning rate and momentum constant. For training the network 200 random samples were generated from the input domain of the functions for training purposes. 5000 epochs of training was conducted for each problem. 30 nos. of networks has been trained with the above procedure.

Table 2: Weight Stuck at MAX/MIN Summary Data

Fault Metric	**WSMAXPOS**		**WSMINPOS**	
MINMAX	**Fn1(7)**	**Fn2(5)**	**Fn1(2)**	**Fn2(18)**
MIN_MSE	9.48E-06	0.0007	0	0
MAX_MSE	63.5805	43.8689	0.700451	0.475097
MEAN_MSE	11.0509	8.7114	0.129236	0.103277
MEDIAN_MSE	0.81206	0.685754	0.0667618	0.0484024
STD_MSE	18.4019	14.6401	0.147783	0.115438
MINMEAN	**Fn1(7)**	**Fn2(14)**	**Fn1(22)**	**Fn2(11)**
MIN_MSE	9.48E-06	0.0014205	0	0
MAX_MSE	63.5805	44.449	1.03956	0.573863
MEAN_MSE	11.0509	7.04836	0.100539	0.0936884
MEDIAN_MSE	0.81206	0.34773	0.028788	0.0191068
STD_MSE	18.4019	12.0506	0.183387	0.151413

Table 2 provides the summary statistics for the network chosen. From the value obtained we may infer that these networks do not show very good fault tolerance behavior for the WSMax/Min fault. Though, some of the weights do not affect the network computation, as under these fault, zero value of MSE demonstrate error of network due to WSMax/Min fault is zero. From this it infer that few weights are not utilized in computation and can easily be pruned.

Figures 2 and 3 represents the behavior of all 30 networks for the average MSE and maximum MSE for (any) single WSMAXPOS fault for Fn1 and Fn2 respectively. Based on the analysis, we have chosen network no. 7 and 5 for function 1 and 2 respectively to study fault metric in MINMAX and network no. 7 and 14 for function 1 and 2 respectively, has been chosen to study fault metric in MINMEAN for WSMAXPOS fault.

In a similar way figures 4 and 5 represents the behavior of all 30 networks for the average MSE and maximum MSE for (any) single WSMINPOS fault. . From the figures, we may infer that, the network no. 2 and 18 for the function 1 and 2 respectively, the value of the error metric is lowest out of 30 networks. So we have chosen network no. 2 and 18 for further analysis of fault metric and summary data is presented in table 2. In a similar way, network no. 22 and 11 for function 1 and 2 respectively gives the lowest error amongst the 30 networks.

We choose the network (out to 30), with the minimum maximum MSE (MINMAX) and minimum mean MSE (MINMEAN), for further analysis for each of the two tasks; that is, to demonstrate network with the best fault tolerance behavior is chosen for further analysis.

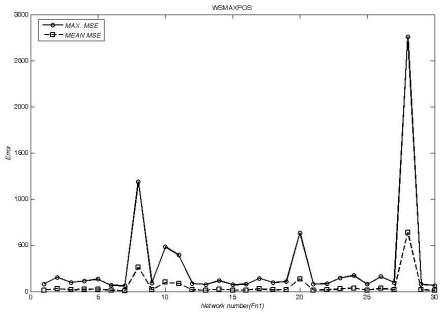

Fig. 2: Behavior of 30 network for Fn1 under WSMAXPOS

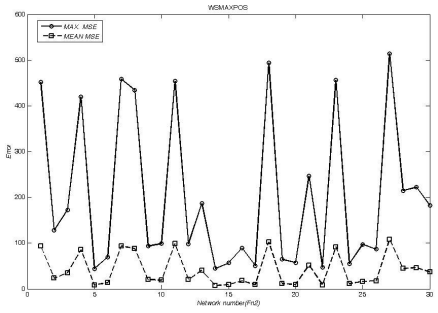

Fig. 3: Behavior of 30 network for Fn2 WSMAXPOS

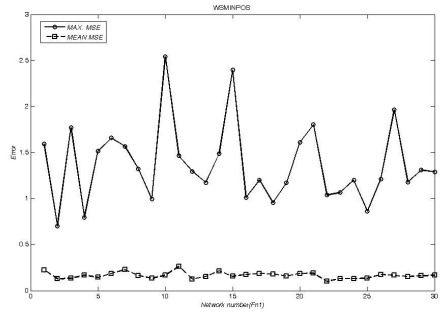

Fig. 4: Behavior of 30 network for Fn1 under WSMINPOS

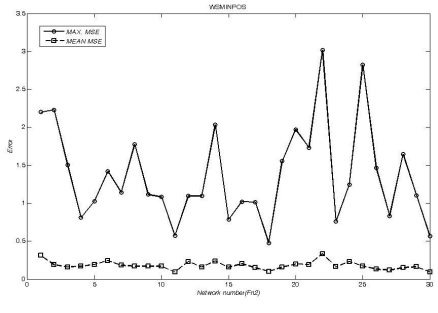

Fig. 5: Behavior of 30 network for Fn2 WSMINPOS

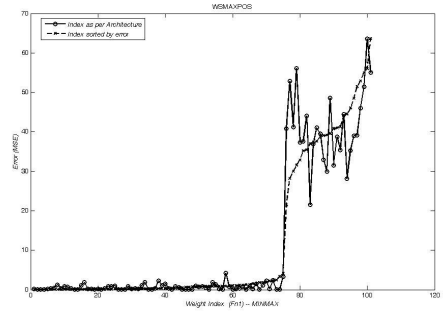

Fig. 6: Weight distribution (MINMAX) for network 7 under Fn1 for WSMAXPOS

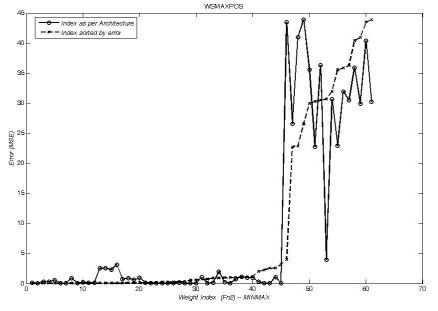

Fig. 7: Weight distribution (MINMAX) for network 5 under Fn2 for WSMAXPOS

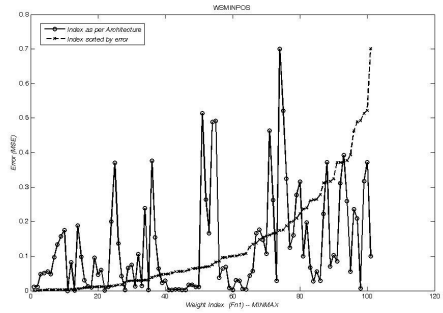

Fig. 8: Weight distribution (MINMAX) for network 2 under Fn1 for WSMINPOS

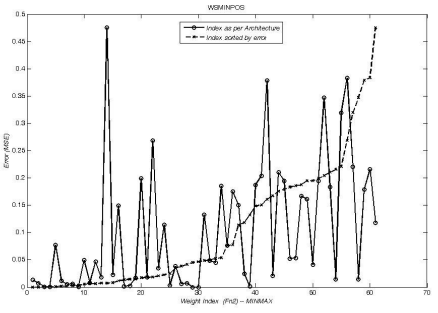

Fig. 9: Weight distribution (MINMAX) for network 18 under Fn2 for WSMINPOS

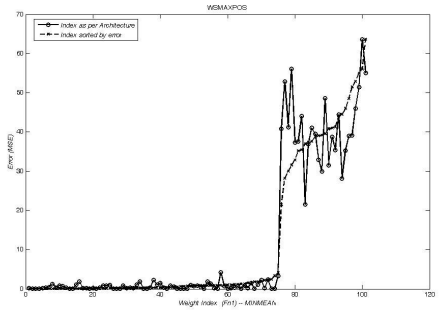

Fig. 10: Weight distribution (MINMEAN) for network 7 under Fn1 for WSMAXPOS

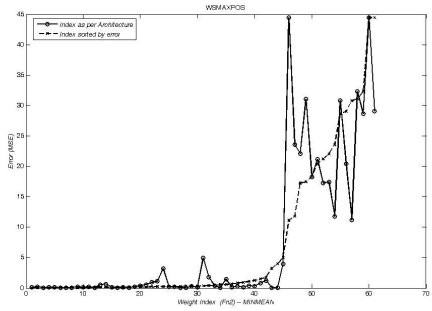

Fig. 11: Weight distribution (MINMEAN) for network 14 under Fn2 for WSMAXPOS

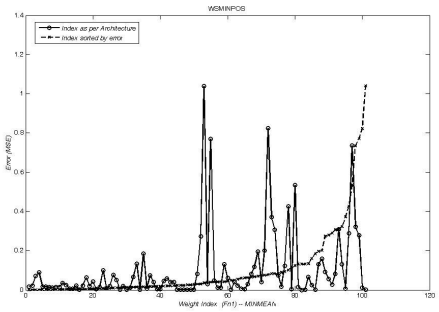

Fig. 12: Weight distribution (MINMEAN) for network 22 under Fn1 for WSMINPOS

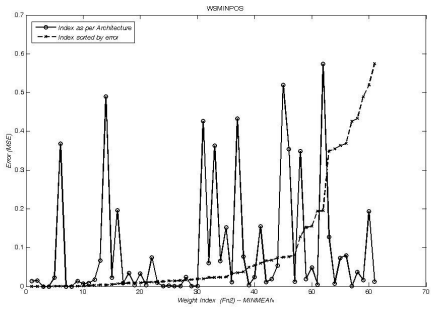

Fig. 13: Weight distribution (MINMEAN) for network 11 under Fn2 for WSMINPOS

From figure 6-9, it is demonstrated that faults in some weights is tolerated (that is, the induced error under fault is small), but the figures also shows that fault in some weights are critical (that is, the faults in these weights lead to large computational error in the network output), for the MINMAX metric, a similar behavior is seen in figure 10-13 for the MINMEAN metric. Figure 6-13 also provides the information of weight distribution across the error in a particular network under specific task.

From the results obtained in Table 2, it is apparent that these networks trained using the RPROP[19] algorithm can not be called fault tolerant to the WSMax/Min fault.

6. CONCLUSION

This paper has presented empirical results on weight stuck-at-Max/Min faults for sigmoidal FFANNs. From the obtained results we may conclude:

1. Some weight does not affect the output of the network, which can be pruned.
2. Some weights lead to small change in output, so it may be infered that partial fault tolerance can exist in the network.
3. Some weights are critical for max/min value computation and faults in these are not well tolerated.

In our opinion, the next step in the analysis of these networks is to device a mechanism that distributes the computational importance of the critical weights through realignment of weight or by addition of new nodes (hidden) and corresponding weights. Moreover, the effect of initial weights on the fault tolerance behavior of FFANN to weight stuck at Max/Min faults needs to be further investigated.

REFERENCES

[1]. F. M. Dias and A. Antunes, "Fault Tolerance of Artificial Neural Networks: an Open Discussion for a Global Model," *International Journal of Circuits, Systems and Signal Processing*, Naun, July, 2008.

[2]. R. P. Lippmann, "An Introduction to Computing with Neural Nets," *IEEE ASSP Magazine*, pp. 4-22, April 1987.

[3]. F. M. Dias and A. Antunes, "Fault Tolerance Improvement through architecture change in Artificial Neural Networks," *Engineering Applications of Artificial Intelligence*, 2007.

[4]. S. Satyanarayana, Y. P. Tsividis, and H. P. Graf, "A Reconfigurable VLSI Neural Network," *IEEE Journal of Solid State Circuits*, vol. 27, no. 1, January 1992.

[5]. M. A. Cavuslu, C. Karakuzu, and S. Sahin, "Neural Network Hardware Implementation using FPGA," *Neural Information Processing*, Lecture Notes in Computer Science, Berlin Germany: Springer, 2006, vol. 4234.

[6]. R. Raeisi and A. Kabir, "Implementation of Artificial Neural Network on FPGA," *American Society for Engineering Education*, Illinois-Indiana and North Central Joint Section Conference , April, 2006

[7]. S. Sahin, Y. Becerikli, and S. Yazici, "Neural Network Implementation in Hardware Using FPGAs," *Neural Information Processing*, Lecture notes in Computer Science, Berlin Germany: Springer, 2006, vol. 4234.

[8]. E.B. Tchernev, R. G. Mulvaney, and D.S. Phatak, "Investigating the Fault Tolerance of Neural Networks," *Neural Computation*, vol. 17, no. 7, pp. 1646-1664, July 2005.

[9]. F. M. Dias, A. Antunes, and A. Mota, "Artificial Neural Networks: a Review of Commercial Hardware," *Engineering Applications of Artificial Intelligence*, IFAC, vol. 17(8), pp. 945-952, 2004.

[10]. P. Chandra and Y. Singh, "Fault Tolerance of Feedforward Artificial Neural Networks - A Framework of Study," *Proceedings of the International Joint Conference on Neural Networks*, vol. 1, pp. 489-494, July 2003.

[11]. V. Piuri, "Analysis of Fault Tolerance in Artificial Neural Networks," Journal *of Parallel and Distributed Computing*, pp. 18-48, 2001.

[12]. P. J. Edwards and A. F. Murray, "Fault Tolerance via Weight Noise in Analog VLSI Implementations of MLP's – A Case study with EPSILON," *IEEE Transaction on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 45, no. 9, September 1998.

[13]. N. C. Hammadi and H. Ito, "A Learning Algorithm for Fault Tolerant Feedforward Neural Networks," *IEICE Trans. Information and Systems*, vol. E80-D, no.1, pp.21-27, 1997.

[14]. D.S. Phatak and I. Koren, "Complete and Partial Fault Tolerance of Feedforward Neural Nets," *IEEE Transaction on Neural Networks*, vol. 6, no. 2, pp. 446-456, March, 1995.

[15]. C. Alippi, V. Piuri, and M. Sami, "Sensitivity to Errors in Artificial Neural Networks: A Behavioral Approach," *IEEE Transaction on Circuits and Systems-I: Fundamental Theory and Applications*, vol. 42, no. 6, June 1995.

[16]. C. T. Chiu, K. Mehrotra, C.K. Mohan, and S. Ranka,, "Training Techniques to obtain fault-tolerant neural network," 24th International Symposium on Fault-Tolerant Computing, pp360-369, June 1994.

[17]. G. Bolt, "Investigating Fault Tolerance in Artificial Neural Networks," University of York, Department of Computer Science, Technical Report YCS 154, Heslington, York, England, 1991.

[18]. V. Cherkassky, "Comparison of Adaptive methods for function estimation from samples," *IEEE Transaction on Neural Networks,* vol. 7, no. 4, July 1996.

[19]. M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," *Proceedings of the IEEE International Conference on Neural Networks (ICNN)*, pp. 586-591, San Francisco, 1993

[20]. L.C. Chu and B.W. Wah, "Fault tolerant neural networks with hybrid redundancy", IEEE *Int. Joint Conference on Neural Networks*, San Diego, CA. vol. 2, pp. 639-649,1990

[21]. S. S. Yeung and X. Sun, "Using Function Approximation to analyze the sensitivity of MLP with antisymmetric squashing activation function", IEEE Transaction on neural networks, vol.13, no.1, January 2002.

[22]. A. D. Friedman and P. R. Memon, "Fault Detection in Digital Circuits", Prentice-Hall, Englewood Cliffs, NJ 1971.

[23]. T. P. Vogl, J.K. Mangis, A.K. Rigler, W.T. Zink, and D.L. Alkon, "Accelerating the convergence of the backpropagation method," *Biological Cybernetics*, Vol. 59, 1988, pp. 257-263

[24]. A. P. Singh, P. Chandra, and C. S. Rai, "Fault Models for Neural Hardware", IEEE First International Conference on Advances in System Testing and Validation Lifecycle (VALID 2009), held during September 20-25, 2009 in Porto, Portugal.

Authors

Amit Prakash Singh is an Assistant Professor in University School of Information Technology, Guru Gobind Singh Indraprastha University, Delhi. He is pursuing Ph.D. from GGS Indraprastha University, Delhi. His area of research is Neural Hardware and Embedded System.

Dr. Chandra Shekhar Rai is an Associate Professor in University School of Information Technology, GGS Indraprastha University, Delhi. He has 10 years of teaching and research experience. He has published many research papers in the journal of international repute. His area of research is artificial neural network and signal processing.

Dr. Pravin Chandra is an Associate Professor in Institute of Informatics & Communication, University of Delhi South Campus, Delhi. He has earlier worked in GGS Indraprastha University, Delhi. He has published many research papers in the area of Artificial Neural Network in journal of international repute.