Empirical Study of Least Sensitive FFANN for Weight-Stuck-at Zero Fault

Amit Prakash Singh
University School of IT
GGS Indraprastha University
Delhi – 110 403 [India]

Pravin Chandra
Institute of Informatics & Comm.
University of Delhi South Campus
New Delhi – 110 021 [India]

Chandra Shekhar Rai
University School of IT
GGS Indraprastha University
Delhi – 110 403 [India]

ABSTRACT
An important consideration for neural hardware is its sensitivity to input and weight errors. In this paper, an empirical study is performed to analyze the sensitivity of feedforward neural networks for Gaussian noise to input and weight. 30 numbers of FFANN is taken for four different classification tasks. Least sensitive network for input and weight error is chosen for further study of fault tolerant behavior of FFANN. Weight stuck-at zero fault is selected to study error metrics of fault tolerance. Empirical results for a WSZ fault is demonstrated in this paper.

General Terms

Artificial Neural Network

Keywords
Artificial Neural Network, Fault models, Sensitivity analysis.

1. INTRODUCTION

Sensitivity analysis provides ability to analyze faulty behavior of neural hardware [3]. Error/fault in feedforward artificial neural network FFANN is classified as [20]:

1) node error/fault
   (a) Output node error/fault
   (b) Hidden layer node error/fault
2) weight error/fault
3) Input error/fault

Study of fault tolerance and robustness in neural networks needs following three areas to be explored [18]:

1) The capability of neural networks to performs in case of failure of internal components (node/weight)
2) Tolerance capability of FFANN to input noise
3) Tolerance capability of FFANN to noise in weight

Design of fault tolerant neural network for failure of an internal component is studied in [19], where redundancy technique has been proposed. Another technique to design fault tolerant neural networks is given in [21], where magnitude of weight is initially kept low, so that any error due to weight will be negligible. While fault tolerance networks for the noise perturbed in input data and weight requires understanding the effects due to small error in input and weight.

Sensitivity analysis is required to study the effect of noise in input and weight [1][4][22]. In [10], Alippi *et. at*. have studied the effect of error in input and weight in the neural networks. In a similar way, further research has been made in [5] to compute effect of noise perturbation in weight and input by Choi and Choi. Zeng and Yeung have demonstrated in [2], where sensitivity of a neural network also depends on the structural configuration of the network.

Sensitivity analysis is widely used for the pruning of the network. Redundant inputs can easily be deleted by analyzing the effect of the input on the network output [23]. Zurada *et al*. [6] [8] and Engelbrecht *et al*. [7], used sensitive analysis results to delete redundant inputs and prune the architecture of the MLP. Deletion of unimportant weight from a network using sensitive analysis is given in [9].

In this paper, we analyze the sensitivity for an input error and weight error of a feedforward neural network and demonstrated a fault metric for weight *stuck-at* zero fault. In section II, architecture of feedforward artificial neural network is described. Sensitivity is explained in section III, fault model to be considered in this paper is explained in section IV. Section V explained experiments and results and paper is concluded in section VI.

2. ARCHITECTURE OF FFANN

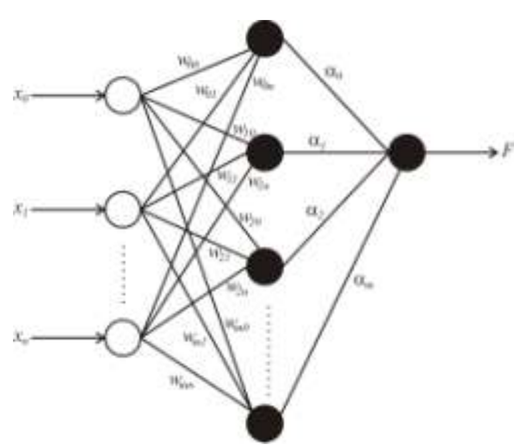

Figure 1. Architecture of FFANN

The networks considered in this paper are one hidden layer networks with a single output. The net input to the *i*th hidden layer node is given by

$$net_i = \sum_{j=1}^{n} w_{ij} x_j + \theta_i = w_i \cdot x + \theta_i \qquad (1)$$

Where,

w_{ij} is the *ij*th element of the matrix **W** representing the connection strength between *j*th input and the *i*th hidden layer node. Θi is the threshold/bias of the *i*th hidden layer node. The output from the ith hidden layer node is given by

$$h_i(x) = \sigma(net_i) \qquad (2)$$

We have used tan hyperbolic tangent sigmoidal function for the transformation from hidden layer node to output node.

The net input to the output node may be defined similarly to (1) as follows:

$$net = \sum_{i=1}^{m} \alpha_i h_i + \gamma = \mathbf{a} \cdot \mathbf{h} + \gamma \qquad (3)$$

Where, α_i represents the connection strength between the *i*th hidden layer node and the output node, while γ is the threshold/bias of the output node.

Inclusion of auxiliary input node $x_0 = 1$ allows the redefinition of (1) as below

$$net_i = \sum_{j=1}^{m} \mathrm{w}_{\mathrm{ij}} \mathrm{x}_{\mathrm{j}} = \mathbf{W_i} \cdot \mathbf{x} \qquad (4)$$

Where W_i is the weight vector w_i augmented by the zeroth term corresponding to the bias. And, similarly introducing an auxiliary hidden node (*i*=0) such that h0 = 1 for any input allows us to redefine (3) as follows:

$$net = \sum_{i=1}^{n} \alpha_i h_i = \mathbf{A} \cdot \mathbf{h} \qquad (5)$$

The notations are explained in figure 1. The figure 1 represents a m-input, n-hidden node and one-output FFANN.

3. DEFINITION OF SENSITIVITY

Sensitivity analysis of the network can be done with reference to various input parameters of neural networks which affect performance of output of the network. A definition of sensitivity in BP-networks has been suggested in [16] [17].

Sensitivity analysis is based on the measurement of the effect that is observed in the output y_k due to the change that is produced in the input x_i. Thus, the greater the effect observed in the output, the greater the sensitivity present with respect to input.

Obtaining the Jacobian matrix [11] by the calculation of the partial derivatives of the output y_k, with respect to the input xi ,

$$\text{i.e. } S_{ik} = \frac{d \ln y_k}{d \ln x_i} \qquad (6)$$

$$= \frac{x_i}{y_k} \frac{dy_k}{dx_i} \qquad (7)$$

eq.(6) constitute the sensitivity analysis of the network, where S_{ik} represents the sensitivity of the percentage change of the output y_k due to percentage changes in the input variable x_i.

The values for the Jacobian matrix [11] do not depend only on the input-output but it also depends on the value stored in a hidden node and layer connection. It also depends on the activation function of the neuron of the hidden layer. Since different input pattern can provide different values for the slope, the sensitivity is generally found by calculating mean squared error (MSE) and mean absolute percentage error (MAPE). The same procedure is required to be followed for the study of weight sensitivity due the fact that each input pattern provides different new weight updated value.

4. FAULT MODELS AND FAULT METRICS

Framework to study fault tolerance of a FFANN is given in [14]. Broad classifications of faults of a FFANN are input, weight and node faults. Sensitive analysis for a single network on four classification problem for various fault models is given in [15].

Missing link of interconnection between two nodes is called weight fault. The weight and node faults are often modeled as *stuck-at-0* fault. Any incorrectness in the input to the adaptive machine is defined as an input fault/error. These faults occur due to external disturbance or noise. Mainly these types of fault affect input vector of the machine. Effect of input error on a network is an important parameter to study sensitive analysis of a FFANN.

Node fault is a similar type of fault as weight fault. Node faults are categorized in two types of node faults, namely hidden node faults and output node faults. Three types of node faults happen in node faults. 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-0* fault to measure fault metric for the chosen least sensitive network.

To measure the effect of faults/errors on the network output enumerated above, the following types of error/fault/parameter sensitivity measures may be defined:

4. 1 MSE, MAPE and Other Global Measures:

The mean squared error (MSE) and the mean absolute percentage error (MAPE) should be used to measure the effect of all types of faults if the output of the FFANN is real. The percentage of misclassification is suggested as a measure of fault/error, for classification problem. Error of the network is reported as term of minimum (MIN), maximum (MAX), mean (MEAN), median (MEDIAN) and standard deviation (STD) error for MSE and MAPE both.

5. EXPERIMENTS AND RESULTS

In this section we apply sensitivity analysis to four approximation tasks. A small experiment was conducted to demonstrate the performance of the network for the weight suck-0 fault. Networks were trained for the following function approximation tasks [12].

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

Fn2: $y = \exp(x_1 * \sin(\pi * x_2))$; x_1, x_2 uniform in [-1,1]

Fn3:
$$a = 40*\exp(8*(x_1 - 0.5)^2 + (x_2 - 0.5)^2)$$
$$b = \exp(8*(x_1 - 0.2)^2 + (x_2 - 0.7)^2)$$
$$c = \exp(8*(x_1 - 0.7)^2 + (x_2 - 0.2)^2)$$
$$y = a/(b+c)$$
; x_1, x_2 uniform in [0,1]

Fn4:
$$y = 1.3356[1.5(1 - x_1) + \exp(2x_1 - 1)\sin(3\pi(x_1 - 0.6)^2) + \exp(3(x_2 - 0.5))\sin(4\pi(x_2 - 0.9)^2)]$$
; x_1, x_2 uniform in [0,1]

Performance of the trained network for perturbed input and weight is demonstrated in this work. The data set for ANN are generated by uniform sampling of the domain of definition of the functions. 30 nos. of networks is designed for each of the above classification tasks. Best network is selected to demonstrate the effect of WSZ fault in this paper.

The network consists of two input, 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
3.	Fn3	2	20	1	81
4.	Fn4	2	10	1	41

The resilient propagation (RPROP) [13] algorithm as implemented in MATLAB 7.2 Neural Network toolbox is used with the default learning rate and momentum constant. 200 random samples were generated from the input domain of the functions for training purposes. 1000 epochs of training was conducted for each problem

For each function we have trained 30 networks. An initial data set of 200 input–target pairs were generated to create trained set of network. A random generated new 200 data is given in the network for the validation purpose and another 1600 input is used for the testing of the network. Sensitivity error is evaluated on each type of network i.e. Trained, Validated and testing network.

Table 2: Performance of the network in a respective function for the perturbation in input

Function Number	Sensitivity	Network Number
Fn1	3412.263352	28
Fn2	2.456094639	22
Fn3	8.403130056	29
Fn4	7.235289441	24

Sensitivity performance is evaluated for 30 networks for all functions. The detailed summary of least sensitive network under input perturbation for each function is given in Table 2, while Table 3 illustrates the least sensitive network for the perturbation in weight for each function.

Table 3: Performance of the network in a respective function for the perturbation in weight

Function Number	Sensitivity	Network Number
Fn1	5228.975007	22
Fn2	3.471157934	2
Fn3	9.753132308	29
Fn4	8.460312299	24

From Table 2 and 3, it infers that network number 2, 22, 24, 28 and 29 are least sensitive network for the respective function. Figures 2 and 3 depict the sensitivity profile for the small perturbation in input and weight respectively for all four functions. It is clearly shown that for few networks in various functions has very high sensitivity. It shows that, the high sensitive network will give bigger error under small error/fault.

Error of the least sensitive network for respective function is reported in Table 4. Error of the network has been evaluated for different set of data which has been taken for validation, Testing and Training purpose of these networks.

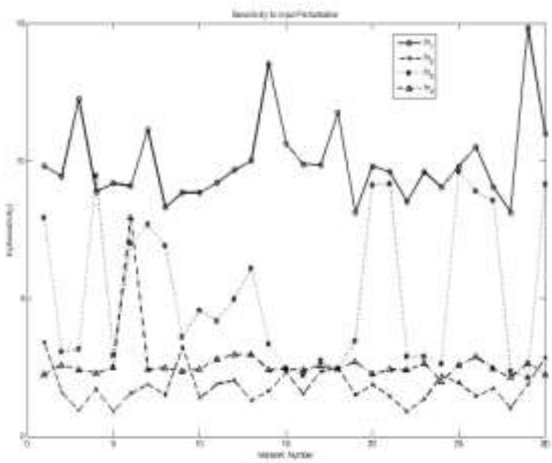

Figure 2: Sensitivity Performance for perturbation in input

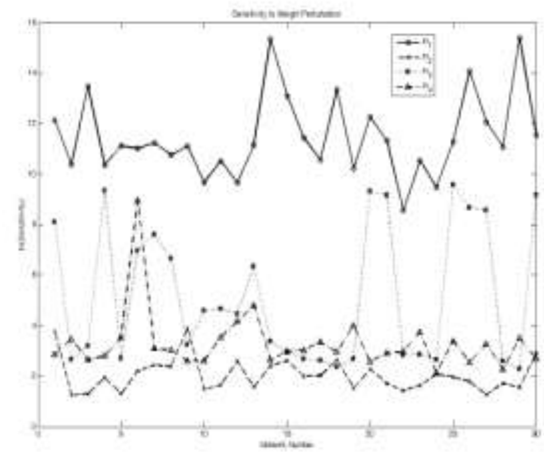

Figure 3: Sensitivity performance for perturbation in weight

From Table 4 it infers that the network which gives least validated error will be the best network for testing and training error. Network no. 22 is the best network under input noise and network no. 2 is best network under weight error. Fault metric is evaluated for each of the above selected network (Table 4) and summary data for various parameters, defined as minimum (MIN), maximum (MAX), average (MEAN), MEDIAN, standard deviation (STD) are given in Table 5. Table 5 infers that zero error can be achieved for the WSZ fault in network no. 29 for the function 3 and 4.

Table 4: Error of the different network during Validation, Testing and Training data.

Network No.	**Validated Error**	**Testing Error**	**Training Error**	**Network No.**	**Validated Error**	**Testing Error**	**Training Error**
28	0.01489	0.01205	0.00211	22	0.01096	0.01157	0.00343
22	0.00083	0.00071	0.00052	2	0.00129	0.00169	0.00092
29	0.01708	0.02670	0.00668	29	0.01708	0.02670	0.00668
24	0.04676	0.05336	0.03083	24	0.04676	0.05336	0.03083

Table 5: Error summary data for WSZ fault in best least sensitive network for various functions.

Functions	**Fn1**		**Fn2**		**Fn3**	**Fn4**
Network No.	**28**	**22**	**22**	**2**	**29**	**24**
MSE						
MAX	2.3709355	0.847630137	1.22903257	1.40029995	4.30610395	12.4319445
MIN	2.57E-05	0.000187753	6.31E-05	1.13E-05	0	0.000277212
MEAN	0.154443728	0.143512299	0.08686692	0.154132433	0.723229633	1.02186233
MEDIAN	0.066829645	0.060372458	0.045210111	0.042621345	0.346940272	0.112749482
STD	0.309626563	0.191064937	0.165677106	0.301854507	0.923509994	2.23362283
MAPE						
MAX	1702.96366	788.848961	119.630249	116.780046	80.7444284	0

MIN	1.14586119	3.17763977	0.184222094	0.236371538	0	215.436978
MEAN	295.852227	203.323983	18.7602784	24.3895412	23.3270594	0.454999627
MEDIAN	145.850795	115.660476	11.5486072	13.6305486	17.657623	35.5177626
STD	344.509372	209.228319	19.2948895	27.3671241	19.6062565	45.4333725

6. CONCLUSION

In this paper, we have demonstrated empirical results of sensitive analysis for FFANNs. Least sensitive network is identified from 30 trained networks for each classification tasks. Sensitivity to input and weight error for all the 30 networks and for each classification talks is demonstrated. Validated error, testing error and training error for selected network is demonstrated in Table 4. The same network is also placed under WSZ fault and empirical result for various errors for the selected network is given in Table 5.

REFERENCES

[1]. Stevenson, M., Winter, R., and Widrow, B. 1990. Sensitivity of feedforward neural networks to weight errors. IEEE Transaction on Neural Networks, vol. 1, no. 1, pp. 71-80.

[2]. Zeng, X., and Yeung, D.S. 2001. Sensitivity analysis of multilayer perceptron to input and weight perturbations. IEEE Transaction on Neural Networks, vol. 12, no. 6, pp. 1358-1366.

[3]. Tian, Z., Lin, T.T.Y., Yang, S. and Tong, S. 1997. The faulty behavior of feedforward neural networks with hard-limiting activation function. Neural Computation, vol. 9, no. 5, pp. 1109-1126.

[4]. Oh, S. H. and Lee, Y. 1995. Sensitivity analysis of single hidden-layer neural networks with threshold functions. IEEE Transaction on Neural Networks, vol. 6, no. 4, pp. 1005-1007.

[5]. Choi, J. Y. and Choi, C. –H. 1992. Sensitivity analysis of multilayer perceptrons with differentiable activation functions. IEEE Transaction on Neural Netowkorks, vol. 3, pp. 101-107.

[6]. Zurada, J.M., Malinowski, A., and Usui, S. 1997. Perturbation method for deleting redundant inputs of perceptron networks. Neurocomputing, vol. 14, pp. 177-193.

[7]. Engelbrecht, A. P., Fletcher, L., and Cloete, I. 1999. Variance analysis of sensitivity information for pruning multilayer feedforward neural networks. In Proceedings of IEEE IJCNN'99, Washington, DC. Vol. 3, pp. 1829-1833.

[8]. Zurada, J. M., Malinowski, A., and Cloete, I. 1994. Sensitivity analysis for minimization of input data dimension for feedforward neural network. IEEE International Symposium on Circuits and Systems, New York. pp. 447-450.

[9]. Karnin, E. D. 1990. A simple procedure for pruning backpropagation trained neural networks. IEEE Transaction on Neural networks. vol. 1, no. 2. pp. 239-242.

[10]. Alippi, C., Piuri, V., and Sami, M. 1995. Sensitivity to errors in artificial neural networks: A behavioral approach. IEEE Transaction of Circuits and Systems-I Fundamental Theory and Applications, vol. 42, no. 6, pp. 358-361.

[11]. Arrowsmith, D. K. and Place, C. M. 1992. Dynamical System. Chapman & Hall, London.

[12]. Cherkassky, V. 1996. Comparison of Adaptive methods for function estimation from samples. IEEE Transaction on Neural Networks. vol. 7, no. 4, pp. 969-984.

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

[14]. Chandra, P. and Singh, Y. 2003. Fault tolerance of feedforward artificial neural networks- A framework of study. In proceeding of IEEE International Joint Conference on Neural Networks, vol. 1, pp. 489-494.

[15]. Singh, A. P., Chandra, P., and Rai, C. S. 2009. Fault models for neural hardware. IEEE International Conference on Advances in System Testing and Validation Lifecycle (VALID 2009), Porto. pp. 7-12.

[16]. Zurada, J. M., Malinowski, A., and Cloete, I. 1994. Sensitivity analysis for minimization of input data dimension for feedforward neural network. ISCAS'94, vol. 6, pp. 447-450.

[17]. Aggarwal, K. K., Singh, Y., Chandra, P., and Puri, M. 2005. Sensitivity analysis of fuzzy and neural network models. ACM SIGSOFT Software Engineering Notes, vol. 30, no. 4. pp. 1-4.

[18]. Sequin, C. H. and Clay, R. D. 1990. Fault tolerance in feedforward neural network. Technical Report – 90-031.International Computer Science Institute, UE Berkely, CA.

[19]. Phatak, D.S. 1995. Fault tolerant artificial neural network. In proceeding of the 5th dual Use Technologies and Applications Conference, Utica/Rome, NY. Pp. 193-198.

[20]. Bolt, G. 1991. Investigating fault tolerance in artificial neural networks. Technical Report: YCS 154. University of York.

[21]. Chiu, C. T., Mehrotra, K., Mohan, C. K., and Ranka, S. 1994. Training Techniques to obtain fault tolerant neural networks. Proceeding of International Symposium of Fault Tolerant Computing, Austin TX, pp. 360-369.

[22]. Piche, S. W. 1995. The selection of weight accuracies for Madalines. IEEE Transaction on Neural Networks, vol. 6, no. 2, pp. 432-445.

[23]. Takenaga, H., Abe, S., Takatoo, M., Kayama, M., Kitamura, T., and Okuyama, Y. 1990. Optimal input selection of neural networks by sensitivity analysis and its application to image recognition. IAPR workshop on Machine Vision Applications, pp. 117-120.