2012 International Conference on Computer Technology and Science (ICCTS 2012)
IPCSIT vol. 47 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V47.6

Training of Sigmoidal FFANN using Zero Weight Initialization and Gaussian Learning Rates

Veenu [1+] and Pravin Chandra [2]

[1] Division of Computer Engineering, Netaji Subhas Institute of Technology (NSIT), Dwarka, New Delhi [India]. (Pursuing MTech(IT) 6th sem from USICT, GGSIPU, New Delhi).

[2] University School of Information and Communication Technology (USICT), Guru Gobind Singh Indraprastha University (GGSIPU), Dwarka, New Delhi [India].

Abstract. Neural networks has varying applications in the field of control problems to pattern recognition problems. They have been used to solve non linear transformation problems. The common paradigm is for randomizing the various parameters like random initialization of weights to zero. The first order or second order derivatives are used to minimize the error. In this paper the neural network parameter weights are initialized to zero. This is a bias able method for randomization of the problem by initialization of weights to zero.

Keywords: ANN – Artificial Neural Network, FFANN – Feed Forward Artificial Neural Network, Eta – Learning Rate parameter, BP – Back Propagation algorithm.

1. Introduction

The artificial neural networks model is described as a relation given below. [16]

{ANN Model} = {Architecture} + {Training/Learning Paradigm}

Many models of ANN and training / learning paradigms exists. One of the most widely used models is the sigmoidal feedforward artificial neural networks. These types of networks have non-recurrent architecture and also have supervised training / learning paradigm. Sigmoidal FFANN's are popular because of the universal approximation results concerned with it, intuitive appeal, simple coding and implementation of back propagation algorithm.

The sigmoidal feedforward artificial neural networks (FFANN's) are popular due to the universal approximation results for these types of networks [17]. The universal approximation results primarily exist in nature. They assure that if a given function is to be approximated, there exists an appropriate FFANN that approximates it arbitrarily well. The work is to find such type of network. The problem of finding appropriate network is solved by the FFANN training procedures. The training algorithm involve arbitrary choices for the parameters of the network.

Back propagation algorithm is the most widely used algorithm for supervised learning with multi-layered feedforward networks. The basic principle of back propagation learning algorithm is repeated application of chain rule to compute the influence of each weight in the network with respect to the error function. The learning rate has important effect in the training of the network as described in the paper by Riedmiller and Braun[18]. The algorithm used deals with the weight updates by doing parameter adaptation during learning. The adaptation can be done using global adaptation or local adaptation strategy. Global technique uses the knowledge of the entire network where as local technique use only weight specific parameters. Numbers of

[+] corresponding author
E-mail address: veenu.d@rediffmail.com

algorithms for performing modification of learning rate according to observed error function behaviour are Delta-bar-Delta Technique or SuperSAB algorithm. Adaptive learning is used for weight calculation.

Typically the training of the neural network is performed by initializing the weights to random small values typically in the range (-1,1) or (-½, ½) . After initialization a non linear optimization method is utilized (usually) to minimize the mismatch / error between the desired output and the obtained output from the network. In this paper we demonstrate that learning / training of the neural network can be done by initializing the weights to zero and introducing randomness in the learning rates. The brief description of the architecture of FFANN is given in section II. Section III demonstrates the experiment performed for the sigmoidal FFANN using back propagation algorithm. Section IV shows the output error table when sigma and eta are varied. Lastly section V concludes the work.

2. Architecture of FFANN

The architecture of the neural network used is as shown in Fig.1. A feedforward artificial neural network (FFANN) is chosen for the experimental purpose. A network has input layer nodes with one hidden layer and a single output layer node.

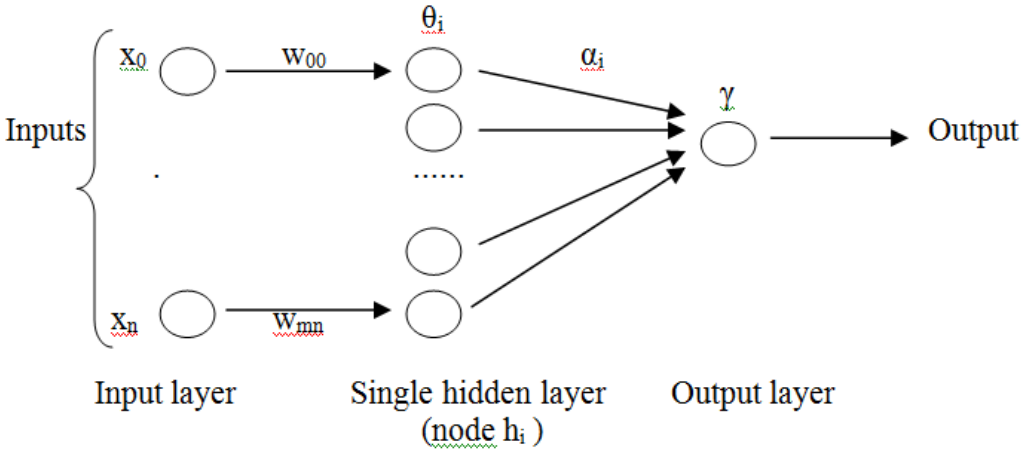

Fig. 1: Architecture of FFANN.

The input dimension is defined as n and the number of the hidden nodes be m. The matrix W denotes the input to hidden nodes connection weight, w_{ij} is the ij^{th} element of the matrix W representing the connection weight between the j^{th} input and the i^{th} hidden layer node and w_i represents the weight vector associated with the i^{th} hidden node (and the inputs). So, net input to the i^{th} hidden layer node is given by

$$net_i = \sum_{j=1}^{n} w_{ij} x_j + \theta_i \quad (1)$$

Where θ_i is the threshold/bias of the i^{th} hidden layer node. The output from the i^{th} hidden layer node is given by

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

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

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

Where α_i represents the connection strength between i^{th} hidden layer node and the output node, while γ is the threshold/bias of the output node. By considering threshold/bias equal to zero for the auxiliary input node, equ (1) can be defined as follows:

$$net_i = \sum_{j=0}^{n} w_{ij} x_j = W_i . x \quad (4)$$

and similarly introducing an auxiliary hidden node (i=0) such that $h_0 = 1$ for any input redefine equ (3) as

$$net = \sum_{i=0}^{m} \alpha_i h_i = a.h \quad (5)$$

Where, $\alpha \equiv \gamma$. The output of FFANN is given as

$$y = f(net) \quad (6)$$

The basic principle behind the Back propagation (BP) learning algorithm is repeated application of chain rule to compute the influence of each weight in the network with respect to the error function.

The various other parameters along with weight used for back propagation algorithm [14] are:

1. The number of input(s) and output(s) nodes defined in the input and the output layers are fixed. In this algorithm the number of nodes in the input layer is two and the number of nodes in the output layer is one. There is one hidden layer and the number of nodes in the hidden layer is fixed to 25.
2. The weights of the network are initialized to (small) uniform random values in the range [-r, r] where r>0 and r Є R (R is the set of all real numbers).
3. Randomized optimization algorithm is used.
4. The specific activation function(s) is used at the active nodes of the FFANN which is fixed before the FFANN is trained because of the limited number of sigmoidal functions. Sigmoids that are symmetric about the origin are preferred and the inputs are normalized because they are more likely to produce the outputs. In this linear function for network output and sigmoidal function for hidden node activation function is used. The sigmoidal function used is as

$$f(x)=1/(1+e^{-x}) \quad (7)$$

A number of modifications in the back propagation algorithm have been proposed. In this paper we modify the back propagation algorithm in which the weights are initialized to zero. Learning rate parameter (eta) is changed by alpha value which is generated by using Normal/ Gaussian distribution keeping mean fixed at zero.

$$eta_{new} = eta_{old} + alpha \quad (8)$$

where eta_{new} is the modified/changed value,

eta_{old} is the previous changed value ,

alpha = N (0, $sigma^2$) (Normal/ Gaussian distribution)

and sigma is varied between 0 and 1 with an interval of 0.025 ie. 0.025, 0.050, 0.075, 0.100, 0.125. Also the learning rate parameter, generated after every iteration is truncated in the range of 0.1 and 0.6.

3. Experiment Performed

A small experiment is conducted to demonstrate the effect on error (MSE) by using the following function approximation task [24].

Func1: y = sin(x1 * x2) ; x1,x2 uniform in [-2,2]

The data set for ANN are generated by uniform sampling. 200 data samples are used to train 30 networks for one problem. Each trained network is validated with new 200 data samples. Testing of the trained network is performed and test error is generated for the data sample. The network consists of two input, one hidden layer and one output node (Fig. 1). The architecture used is summarized in the table1 given below. The architecture is consisting of hidden layers from 1 to 30 and the architecture that gives the minimum error on training is used. All the hidden nodes use sigmoid activation function while the output nodes are linear.

Table1 : Network used

Sr. No	Function	No. of inputs	No.of hidden nodes	No.of Ouput nodes
1.	Func1	2	25	1

The back propagation algorithm is implemented in MATLAB 7.10 (R2010a). 200 random samples are generated from the input domain for the training purposes. For each problem 10,000 epochs of training are performed. The training is done for 30 networks for one function. The training data set is of 200 while the validation / test set size is 10,000. The values are randomly generated and scaled to (-1, 1).

4. Output

When the back propagation algorithm is executed the parameters used are:

1. Number of hidden nodes = 25
2. Eta value = 0.150
3. Sigma values = 0.025, 0.050, 0.075, 0.100, 0.125
4. Function 1 : y = sin(x1 * x2) ; x1,x2 uniform in [-2,2]

5. Mean = 0
6. Range of truncating eta = 0.1 to 0.6

Simple BP means when the Back propagation algorithm is executed without any changes. Modified BP means when the Back propagation algorithm is executed by initializing the weights to zero and varying sigma.

Table 2: Error obtained by using function 1

	Simple BP				Modified BP			
Sno	Sigma value	Training mean error	Validation mean error	Test mean error	Sigma value	Training mean error	Validation mean error	Test mean error
1	0	0.0130195	0.0134	0.0130	0.025	0.0296214	0.0250	0.0296
2	0	0.0130195	0.0134	0.0130	0.050	0.0214878	0.0198	0.0215
3	0	0.0130195	0.0134	0.0130	0.075	0.0281653	0.0239	0.0282
4	0	0.0130195	0.0134	0.0130	0.100	0.028285	0.0251	0.0283
5	0	0.0130195	0.0134	0.0130	0.125	0.0246377	0.0229	0.0246

5. Conclusion

In this paper we have taken a single application to demonstrate the possibility of zero weight initialization. The result indicates that it is feasible to perform training by initializing the weights to zero, though the result is higher approximately by a factor of two. In future we will validate the result on multiple problems and explore the feasibility of combining the random learning rate parameters with random weight initialization to get a better result. A theoretical analysis of the problem is being done and will be reported in the future.

6. References

[1] G. Cybenko, "Approximation by superposition of a sigmoidal function,"*Math. Contr., Signal, and Syst.*, vol. 5, pp. 233–243, 1989.

[2] K. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, pp. 183–192, 1989.

[3] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp.359–366, 1989.

[4] K. Hornik, M. Stinchcombe, and H. White , "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neural Networks*, vol. 3, pp. 551–560, 1990.

[5] M. Stinchcombe and H. White, "Approximating and learning unknown mappings using multilayer feedforward networks with bounded weights," in *Proc. IJCNN*, vol. III, 1990, pp. 7–16.

[6] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, 1991.

[7] A. R. Barron, "Universal approximation bounds for superposition of a sigmoidal function," *IEEE Trans. Inform. Theory*, vol. 39, pp. 930–945, 1993.

[8] T. Chen, H. Chen, and R.W. Liu, "Approximation capability in C(_R) by multilayer feedforward networks and related problems," *IEEE Trans.Neural Networks*, vol. 6, pp. 25–30, 1995.

[9] M. B. Stinchcombe, "Neural network approximation of continuous functionals and continuous functions on compactifications," *Neural Networks*, vol. 12, pp. 467–477, 1999.

[10] B. DasGupta and G. Schnitger, "The power of approximating:Acomparision of activation functions," in *Advances in Neural Information Processing Systems*, C. L. Giles, S. J. Hanson, and J. D. Cowan, Eds. San Mateo, CA: Morgan Kaufmann, 1993, vol. 5, pp. 615–622.

[11] K. Hornik, M. Stinchcombe, H. White, and P. Auer, "Degree of approximation results for feedforward networks

approximating unknown mappings and their derivatives,", Tech. Rep. NC-TR-95-004, 1995.

[12] L. Holmstrom and P. Koistinen, "Using additive noise in backpropagation training," *IEEE Trans. Neural Networks*, vol. 3, pp. 24–28, 1992.

[13] A. F. Murray and P. J. Edwards, "Synaptic weight noise during MLP training: Enhanced MLP performance and fault tolerance resulting from synaptic noise during training," *IEEE Trans. Neural Networks*, vol. 5,pp. 792–802, 1994.

[14] P. Chandra and Y. Singh, "Fault tolerance of feedforward artificial neural networks—A framework of study," in *IJCNN'03*, 2003, pp. 489–494.

[15] P. Chandra and Y. Singh , "Regularization and feedforward artificial neural network training with noise," in *IJCNN'03*, 2003, pp. 2366–2371.

[16] P. Chandra and Y. Singh, "Feedforward Sigmoidal Networks - Equicontinuity and Fault-Tolerance Properties," *IEEE Transaction on Neural Networks*, vol. 15, no. 6, November 2004.

[17] Pravin Chandra, " Sigmoidal function classes for feedforward Artificial Neural networks" , *Neural Processing Letters 18*, 185 – 195, 2003, printed in Netherland.

[18] 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

[19] Simom Haykin, "Neural Networks - A Comprehesive Foundation", Prentice Hall International. Inc**.,** New Jersey, 2nd edition, 1999.

[20] Simon Haykin,"Neural Networks and learning Machines", PHI learning Private limited, 2011, 3rd edition.

[21] A. P. Singh, P. Chandra, and C. S. Rai, "Empirical study of FFANNs tolerance to weight stuck at zero fault," *International Journal of Engineering & Technology*, vol. 2(2), 2010.

[22] A. P. Singh, C. S. Rai and P. Chandra, and "Empirical study of FFANNs tolerance to weight stuck at MAX/MIN fault," *International Journal of Artificial Intelligence and applications(IJAIA)*, Vol. 1, No. 2, April 2010.

[23] Y. Lecun, L. Bottou, G. Orr, and K. Muller, "Efficient backprop," *Neural Networks : Tricks of the trade*, Springer, 1998.

[24] V. Cherkassky, "Comparison of adaptive methods for function estimation from samples," *IEEE Transaction on Neural Networks*, vol.7(4), pp. 969-984, 1996.