IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 13, Issue 4 (Jul. - Aug. 2013), PP 34-38
www.iosrjournals.org

Using Multi-layered Feed-forward Neural Network (MLFNN) Architecture as Bidirectional Associative Memory (BAM) for Function Approximation

Manisha Singh[1], Somesh Kumar[2]
[1]*M.Tech. (CSE) Student*
[2]*Professor, Noida Institute of Engineering & Technology, Greater Noida, India*

Abstract : *Function approximation is to find the underlying relationship from à given finite input-output data. It has numerous applications such as prediction, pattern recognition, data mining and classification etc. Multi-layered feed-forward neural networks (MLFNNs) with the use of back propagation algorithm have been extensively used for the purpose of function approximation recently. Another class of neural networks BAM has also been experimented for the same problem with lot of variations. In the present paper we have proposed the application of back propagation algorithm to MLFNN in such a way that it works like BAM and the result thus presented show greater and speedy approximation for the example function.*
Keywords: *Neural networks, Multilayered feed-forward neural network (MLFNN), Bidirectional Associative Memory (BAM), Function approximation*

I. INTRODUCTION

Capturing the implicit relationship between the input and output patterns such that when test input pattern is given, the pattern corresponding to the output of the generalized system is retrieved, is called the problem of pattern mapping. Therefore the very objective of pattern mapping problem is to capture the implied function and mimic its behavior. Alternatively, function approximation is the task of learning or constructing a function that generates approximately the same outputs from input vectors as the process being modeled, based on available training data. The approximate mapping system should give an output which is fairly close to the values of the real function for inputs close to the current input used during learning. There exist multiple methods that have been established as function approximation tools, where an artificial neural network (ANNs) is the one which has been very popular recently. A trained neural network is expected to capture the system characteristics in their weights. Such network is supposed to have generalized from the training data, if for a new input the network produces the same output which the system would have produced.

Mainly the two categories of networks used for pattern mapping (thus function approximation) task are – multilayered feed-forward neural networks trained using back-propagation algorithm [1-2], and the bidirectional associative memory (BAM) [3] trained using Hebb rule. Due to design, MLFNN with hidden layers is used for the classification purposes and hence when used for pattern mapping task, suffers the problem of generalization. Generalization is possible only if the mapping function satisfies certain constraints.

Otherwise, the problem of capturing the mapping function from the training data set becomes an ill-posed problem [Tikhonov and Arsenin, 1977]. To overcome this, a radial basis function network (RBFN) and counter propagation neural network (CPN) has been used. The underlying algorithms which modify weights separate such networks from back propagation. In back-propagation, the error and corresponding weight modification are propagated backwards from the output layer to the input layer. Different training samples may exert opposite forces in increasing or decreasing the same weight, and the net effect is that weight changes are often of extremely small magnitude in networks with learning rates small enough to have some assurance of converging to a local minimum of the mean squared error.

II. RELATED WORK

Function approximation using feed-forward neural networks (FNNs).has been studied and discussed in detail by so many authors in literature (Cybenko, 1989; Hecht-Nielsen, 1989; Carroll and Dickinson, 1989; Hornik, 1990, 1993; Park and Sandberg, 1991, 1993; Barron, 1993). FNNs are established structures capable of approximating generic classes of functions, including continuous and bounded. The structure studied varied in the terms of number of hidden layers for a particular function and different classes of FNNs were defined for approximating different classes of functions as per a set of approximation criteria. It is well known that a two-layered FNN, i.e. one that does not have any hidden layers, is not capable of approximating generic nonlinear continuous functions (Widrow, 1990). On the other hand, four or more layer FNNs is rarely used in practice.

Hence, almost all the work deal with the most challenging issue of the approximation capability of three-layered FNNs.

The dynamics of multilayer feed-forward neural networks is simpler compared to recurrent neural networks **[4-6]** and BAMs. Since for storage and recall, Hebbian learning rule is used in BAM and therefore it has limited storage and mapping capability. There has been reported substantial work ([7]-[24]) in the literature to overcome this limitation. In the high-order BAM [7], high-order nonlinearity is applied to forward and backward information flows to increase the memory capacity and improve error correction capability. The exponential BAM [8] employs an exponential scheme of information flow to exponentially enhance the similarity between an input pattern and it's nearest stored pattern. It improves the storage capacity and error correcting capability of the BAM, and has a good convergence property. In [9], a high capacity fuzzy associative memory (FAM) for multiple rule storage is described. A weighted-pattern learning algorithm for BAM is described in [10] by means of global minimization. As inspired by the perceptron-learning algorithm, an optimal learning scheme for a class of BAM is advanced [11], which has superior convergence and stability properties. In [12], the synthesis problem of bidirectional associative memories is formulated as a set of linear inequalities that can be solved using the perceptron training algorithm. A multilayer recursive neural network with symmetrical interconnections is introduced in [14] for improved recognition performance under noisy conditions and for increased storage capacity. In [15], a new bidirectional hetero-associative memory is defined which encompasses correlational, competitive and topological properties and capable of increasing its clustering capability. Other related work can be found in [16], [17],[18] and [19] in which either by adding dummy neuron, increasing in number of layers or manipulating the interconnection among neurons in each layer, the issue of performance improvement of BAM is addressed. Even some new learning algorithms were introduced to improve the performance of original BAM and can be found in detail in [20]-[24].

III. PROPOSED MODEL

All methods discussed in section II use different algorithms which work only in one direction i.e. forward direction and the error thus calculated is propagated backward and weights are adjusted accordingly. We propose a method based on Back Propagation algorithm which works in two phases: ***Phase-I*** and ***Phase-II***. Since in both the phases the error is calculated and weights are adjusted accordingly, once in forward direction and then in backward direction, therefore there is a significant improvement in convergence. In this section we outline the theory behind the proposed approach.

3.1 Bidirectional Associative Memory (BAM)

The Bidirectional associative memory is hetero-associative, content-addressable memory. In this type of memory the neurons in one layer are fully connected to the neurons of second layer. In fact if we take a feed-forward network with one hidden layer and connections are done among various neurons of layers (**Fig. 1**). If we perform error minimization using back propagation algorithm, then this corresponds to the BAM. The weights are taken symmetrical.

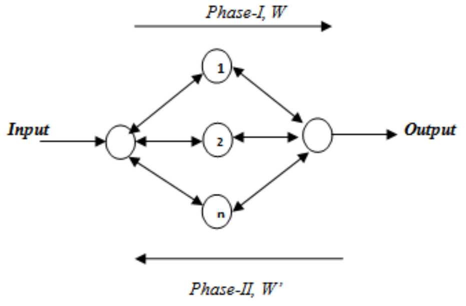

Figure 1: *A typical 1-n-1 feed-forward architecture as Bidirectional Associative Memory*

3.2 The standard BP Algorithm for *1-n-1* feed-forward architecture

- *Create a feed-forward network with one input,* ***n*** *hidden units and one output.*
- *Initialize all the weights to small random values.*
- *Repeat until* $(E_{avg} = \sum_{k=1}^{Q} Ek < \tau)$ *where* τ *is the tolerance level of error*

1. Input pattern $\mathbf{X_k}$ from training set and compute the corresponding output $\mathbf{S_y}$
2. For corresponding output, calculate the error as

$$\delta_o = S_y(D - S_y)(1 - S_y)$$

3. For each hidden unit **h**, calculate

$$\delta_h = S_h(1 - S_h)\, W_{hj}\, \delta_o$$

4. Update each network weight between hidden and output layer W_{hj} as follows:

$$W_{hj} = W_{hj} + \eta\Delta W_{hj} + \alpha\, \Delta W_{hj}old$$

where $\Delta W_{hj} = S_h * \delta_o$ and $\Delta W_{hj}old = \Delta W_{hj}$

5. Update each network weight between input and hidden layer W_{ih} as follows:

$$W_{ih} = W_{ih} + \eta\Delta W_{ih} + \alpha\, \Delta W_{ih}old$$

where $\Delta W_{ih} = \delta_h * X_k$ and $\Delta W_{ih}old = \Delta W_{ih}$

3.3 The proposed Two-Phase BP Algorithm

Phase-I

The standard BP algorithm

Phase-II

- *Repeat until* $(E_{avg} = \sum_{k=1}^{Q} Ek < \tau)$ *where* τ *is the tolerance level of error*

1. $W_{hi} = W'_{ih}$ *and* $W_{jh} = W'_{hj}$
2. *Input pattern* D_k *for corresponding* X_k *and compute the corresponding output* S_u
3. *For corresponding output* S_u *, calculate the error as*

$$\delta_i = S_u(X - S_u)(1 - S_u)$$

4. *For each hidden unit* ***h***, *calculate*

$$\delta_h = S_h(1 - S_h)\, W_{hi}\, \delta_i$$

5. *Update each network weight between hidden and input layer* W_{hi} *as follows:*

$$W_{hi} = W_{hi} + \eta\Delta W_{hi} + \alpha\, \Delta W_{hi}old$$

where $\Delta W_{hi} = S_h * \delta_i$ *and* $\Delta W_{hi}old = \Delta W_{hi}$

6. *Update each network weight between output and hidden layer* W_{jh} *as follows:*

$$W_{jh} = \Delta W_{jh} + \eta\Delta W_{jh} + \alpha\, \Delta W_{jh}old$$

where $\Delta W_{jh} = \delta_i * D_k$ *and* $\Delta W_{jh}old = \Delta W_{jh}$

IV. SIMULATION DESIGN AND EXPERIMENTS

A three-layered feed-forward neural network has been created for the experimentation. At the input and output layers only one neuron has been considered since we are trying to approximate a single variable function ***sin(x)***. The hidden layer consists of variable number of neurons ranging from 2 to 10. The summary of the various parameters used is shown in **Table 1.**

At different learning rates and fixed momentum value (0.6), the convergence of the function has been studied with tolerance value .05 by both algorithms discussed in **section 3.1 & 3.2** - standard back propagation and the proposed two-phase back propagation algorithm (which let the MLFNN work as BAM). The results have been tabulated in **Table 2.** One epoch in standard BP algorithm means when all the training patterns are presented once, while one epoch in proposed two-phase BP algorithm means when for all patterns both forward and backward phases run once.

Table 1: *The parameters used in simulation*

Parameter	***Number/Values***
Input Neurons	1
Hidden Neurons	2/3/4/5/6/7/8/9/10
Number of Patterns	5
Learning rate, η	0.7/0.8/0.9/1.0
Momentum, α	0.6
Tolerance, τ	0.005

Table 2: *Number of epochs required for convergence for approximating the function* ***y=sin(x).***

Architecture *(input-hidden-output)*	*Learning rates*							
	η=0.7		***η=0.8***		***η=0.9***		***η=1.0***	
	FFN	BAM	FFN	BAM	FFN	BAM	FFN	BAM
1-2-1	4480	1268	4126	1184	3759	1095	3232	1041
1-3-1	2562	532	2640	461	2398	424	2061	370
1-4-1	2201	487	2252	387	2071	346	1597	353
1-5-1	2194	423	1837	399	1445	381	1524	342
1-6-1	1354	419	1267	336	1229	348	1264	330
1-7-1	1459	406	1368	377	1363	322	1407	329
1-8-1	1742	395	1785	329	1380	349	1444	320
1-9-1	1309	347	1102	312	1025	300	839	276
1-10-1	1226	343	1022	308	950	494	801	**272**
1-11-1	1270	376	1133	327	998	572	915	312

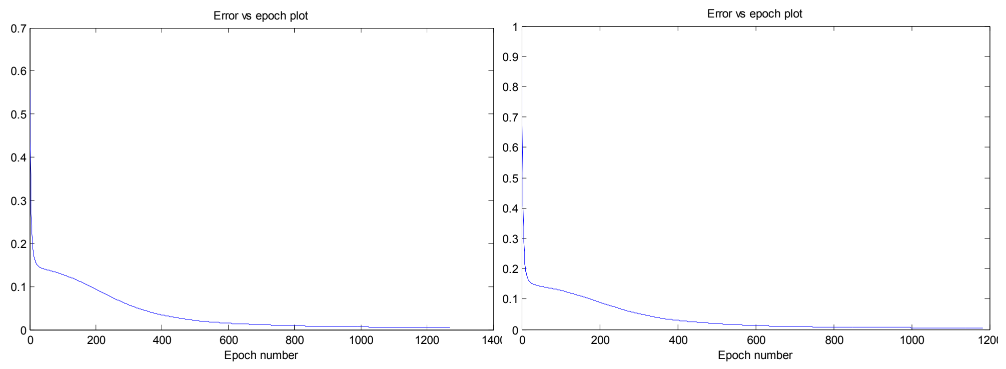

Figure 2: Two *typical plots of the instantaneous squared error on an epoch by epoch basis as training proceeds*

V. RESULTS AND DISCUSSION

The results shown in **Table 2** clearly indicate that the proposed two-phase back propagation algorithm for BAM architecture simply outperform the MLFNN architecture with standard back propagation algorithm. The number of epochs taken for the convergence of the taken function is substantially low in the case of proposed two-phase BP algorithm. Moreover, if we increase the number of neurons in the hidden layer, from 2 to 10, the convergence is achieved early that means in lesser number of epochs in both the cases. Another obvious observation comes out that on increasing the learning rate from 0.7 to 1.0 the convergence is being achieved in lesser number of epochs. The results show that the best suited BAM architecture for the taken function **sin(x)** is 1-10-1 with learning rate 1.0 and momentum 0.6.

VI. CONCLUSION AND FUTURE SCOPE

The new two-phase BP algorithm has been presented which when applied to multi-layered feed-forward neural network (MLFNN) let it behave like bidirectional associative memory (BAM). As discussed above, the proposed algorithm is better suited for function approximation and results obtained are quite encouraging. Still the work is in early stage and more experiments with the various parameters like number of hidden layers with variable number of neurons in each hidden layer, number of inputs, value of momentum etc. are to be undertaken and their effect on achieving the convergence is to be studied.

References

[1] Rumelhan D.E., Hinton G.E., and Williams R. J.**,** Learning Internal Representations by Error Propagation, Parallel Distributed Processing, Vol. I Foundations, MIT Press, Cambridge, MA, pages 318-364. 1986.

[2] Tang C. Z**.** and Kwan H**.** K., Parameter effects on convergence speed and generalization capability of back-propagation algorithm International Journal of Electronics**,** 74(1): 35-46, January 1993.

[3] Kosko B., Neural Networks and Fuzzy Systems **(Prentice**-Hall. Inc., New Jersey, 1992)

[4] Pearlmutter B.A., Gradient calculations for dynamic recurrent neural networks: a survey. IEEE Trans. On Neural Networks**,** 6 (5):1212-1228, Sept. 1995

[5] Prokhorov D.V., Feldkamp L.A., and Tyukin I.Y.**,** Adaptive behavior with fixed weights in RNN: an overview. Proceedings of International Joint Conference on Neural Networks, 2002, pages 201 8-2022

[6] Kwan H. K. and Yan J., Second-order recurrent neural network for word sequence learning, Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing**,** Hong Kong, May 2-4. 2001, pages 405-408.

[7] Tai H.-M.. Wu C.-H., and Jong, T.-L., High-order bidirectional associative memor': Elecrronics krlers, 25(21):1424-1425, 12th Oct. 1989.

[8] Jeng Y.-J., Yeh C.-C., and Chiueh T.D., 'Exponential bidirectional associative memories", Electronics Letters, 26(11):717-718,24th May1990.

[9] Chung F.L. and Lee Tong, **'**On fuzzy associative memory with multiple-rule storage capacity". IEEE Trans. on Fuzzy Systems**,** 4(3):375-384, August 1996.

[10] **[**Wang T.. Zhuang X., and Xing X., 'Weight learning of bidirectional associative memories by global minimization", IEEE Trans. on Neural Networks, 3(6):1010-1018, Nov. 1992.

[11] Shanmukh K**.** and Venkatesh Y.V., 'Generalized scheme for optimal learning in recurrent neural networks': IEE Proc: Vision, Image**,** and Signal Processing. 142(2):71-77, April 1995.

[12] Salih, I. Smith, S.H., and Liu**,** D., 'Design of bidirectional associative memories based on the perceptron training technique". Proceedings **of** IEEE International Symposium **on** Circuits and Systems, 1999, Vol. **5,** pages 355-358.

[13] Kwan H. K**.** and Tsang P. C., 'Multi -layer recursive neural network': Proceedings of Canadian Conference on Electrical and Computer Engineering, Montreal, Canada, September 17-20, 1989, Vol. **1.** pages 428-431.

[14] Widrow B**.** and Winter R., 'Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition'', IEEE Computer Magazine, pages 25-39. March 1988.

[15] Chartier S., Giguere G. and Langlois D., "A new bidirectional heteroassociative memory encompassing correlational, competitive and topological properties", Neural Networks 22 (2009), pp 568-578, 2009

[16] Wang Y.F., Cruz J.B. and Mulligan J.H., "Two coding strategies for bidirectional associative memory", IEEE Trans. Neural Networks, vol. 1, no. 1, pp 81-92, 1990

[17] Wang Y.F., Cruz J.B. and Mulligan J.H., "Guaranteed recall of all training pairs for bidirectional associative memory", IEEE Trans. Neural Networks, vol. 2, no. 6, pp 559-567, 1991

[18] Kang H., "Multilayer associative neural network (MANN's): Storage capacity versus perfect recall", IEEE Trans. Neural Networks, vol. 5, pp 812-822, 1994

[19] Wang Z., "A bidirectional associative memory based on optimal linear associative memory", IEEE Trans. Neural Networks, vol.45, pp 1171-1179, Oct.1996

[20] Hassoun M.H. and Youssef A.M., "A high performance recording algorithm for Hopfield model associative memories", Opt. Eng., vol. 27, no. , pp 46-54, 1989

[21] Simpson P.K., "Higher-ordered and interconnected bidirectional associative memories", IEEE Trans. Syst. Man. Cybern., vol. 20, no. 3, pp 637-652, 1990

[22] Zhuang X., Huang Y. and Chen S.S., "Better learning for bidirectional associative memory", Neural Networks, vol.6, no.8, pp1131-1146, 1993

[23] Oh H. and Kothari S.C., "Adaptation of the relaxation method for learning bidirectional associative memory", IEEE Trans. Neural Networks, vol.5, pp 573-583, July 1994

[24] Wang C.C. and Don H.S., "An analysis of high-capacity discrete exponential BAM", IEEE Trans. Neural Networks, vol. 6, no. 2, pp 492-496, 1995