Volume 3, Issue 12, December 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Comparative Study of Back Propagation Learning Algorithms for Neural Networks

Saduf, Mohd Arif Wani
Dept.of Computer Sciences
University of Kashmir
Srinagar, j&k, India

Abstract—back propagation method is a technique used in training multilayer neural networks in a supervised manner. The back propagation method, also known as the error back propagation algorithm, is based on the error-correction learning rule; it is most often used learning algorithm in neural networks as it deals with continuous data and differentiable functions for both single and multilayer models. Slower convergence and longer training times are the disadvantages often mentioned when the conventional back propagation algorithm is compared with other competing techniques. Over years many improvements and modifications of the learning algorithm have been reported to overcome these shortcomings. In this paper the back propagation algorithm and several variations to improve the performance of the algorithm have been thoroughly reviewed.

Keywords— back propagation; momentum; gain; local minima; Activation function; network model

I. INTRODUCTION

In the last few decades, extensive research has been carried out in developing the theory and the application of the Artificial Neural Networks(ANNs).ANN has emerged to be a powerful mathematical tool for solving various practical problems like pattern classification and recognition, medical imaging, speech recognition and control. Of the many Neural Network (NN) architectures proposed, the Multi Layer Perceptron(MLP)with back propagation(BP)learning algorithm is found to be effective for solving a number of real world problems. Artificial Neural Networks (ANN) is a mathematical or computational model based on biological networks. It consists of interconnected group of artificial neurons, and processes information using a connectionist approach to computation. In most cases, an ANN is an adaptive system that changes its structure based on external or internal information which flows through the network during learning phase. In more practical terms, neural networks are non linear statistical data modelling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. As human being, we learn how to write, read, understand speech, recognize and distinguish between pattern-all by learning from examples. In the same way ANNs are trained rather than programmed. A multilayer feed-forward neural network consists of an input layer, hidden layer and an output layer of neurons. Every node in a layer is connected to every other node in the neighbouring layer. A feed forward network has no memory and the output is solely determined by the current input and weight values. A feed forward neural network consists of one or more layers of usually non-linear processing units (can use linear activation functions as well).the output of each layer serves as input to the next layer. The objective of training a NN is to produce desired output when a set of input is applied to the network; the training of FNN is mainly undertaken using the back propagation (BP) based learning. Back Propagation Neural Network (BPNN) algorithm is the most popular and the oldest supervised learning multilayer feed forward neural network algorithm proposed by [1].

II. CRITICAL REVIEW OF BP ALGORITHM

A. *Overview of the Algorithm*

Back propagation is a method of training multilayer artificial neural networks which use the procedure of supervised learning. Supervised algorithms are error-based learning algorithms which utilise an external reference signal (teacher) and generate an error signal by comparing the reference with the obtained output. Based on error signal, neural network modifies its synaptic connection weights to improve the system performance. In this scheme, it is always assumed that the desired answer is known a priori.The traditional Back-propagation Neural Network (BPNN) Algorithm is widely used in solving many practical problems. The BPNN learns by calculating the errors of the output layer to find the errors in the hidden layers. Due to this ability of Back-Propagating, it is highly suitable for problems in which no relationship is found between the output and inputs. Due to its flexibility and learning capabilities, it has been successfully implemented in wide range of applications. A Back-Propagation network consists of at least three layers of units: an input layer, at least one intermediate hidden layer, and an output layer. Typically, units are connected in a feed-forward fashion with input units fully connected to units in the hidden layer and hidden units fully connected to units in the output layer. The input pattern is presented to the input layer of the network. These inputs are propagated through the network until they reach the output units. This forward pass produces the actual or predicted output pattern. Because back propagation is a supervised learning algorithm, the desired outputs are given as part of the training vector. The actual network outputs are subtracted from the desired outputs and an error signal is produced. This error signal is then the basis for the back propagation step, whereby the errors are passed back through the neural network by computing the contribution of each hidden processing unit and deriving the corresponding adjustment needed to produce the correct output. The connection weights are then adjusted and the neural network has just "learned" from an experience. Once the network is trained, it will provide the desired output for any of the input patterns. The change in the weights may be done at the time each input vector is presented. This is known as sequential mode of training. The change can also be averaged and weights can be changed after all the input vectors are applied. This type of training is called batch mode of training.

B. *Training in Back-Propagation Algorithm*

The feed forward back-propagation network undergoes supervised training, with a finite number of pattern pairs consisting of an input pattern and a desired or target output pattern. An input pattern is presented at the input layer. The neurons here pass the pattern activations to the next layer neurons, which are in a hidden layer. The outputs of the hidden layer neurons are obtained by using a bias, and also a threshold function with the activations determined by the weights and the inputs. These hidden layer outputs become inputs to the output neurons, which process the inputs using an optional bias and a threshold function. The final output of the network is determined by the activations from the output layer. The computed pattern and the input pattern are compared, a function of this error for each component of the pattern is determined, and adjustment to weights of connections between the hidden layer and the output layer is computed. A similar computation, still based on the error in the output, is made for the connection weights between the input and hidden layers. The procedure is repeated with each pattern pair assigned for training the network. Each pass through all the training patterns is called a cycle or an epoch. The process is then repeated as many cycles as needed until the error is within a prescribed tolerance. The adjustment for the threshold value of a neuron in the output layer is obtained by multiplying the calculated error in the output at the output neuron and the earning rate parameter used in the adjustment calculation for weights at this layer. After a BP network has learned the correct classification for a set of inputs from a training set, it can be tested on a second set of inputs to see how well it classifies untrained patterns. Thus; an important consideration in applying BP learning is how well the network generalizes.

1) *Problems with BP Learning:* Back propagation has some problems associated with it which include network paralysis, local minima and slow convergence.
   - Perhaps the best known is called "Local Minima". This occurs because the algorithm always changes the weights in such a way as to cause the error to fall. But the error might briefly have to rise as part of a more general fall, If this is the case, the algorithm will "get stuck" (because it can't go uphill) and the error will not decrease further.
   - Network paralysis occurs when the weights are adjusted to very large values during training, large weights can force most of the units to operate at extreme values, in a region where the derivative of the activation function is very small.
   - A multilayer neural network requires many repeated presentations of the input patterns, for which the weights need to be adjusted before the network is able to settle down into an optimal solution.

II. APPROACHES FOR ADDRESSSING PROBLEMS WITH BP LEARNING

Different authors have addressed the problem of BP learning in different ways (using better energy function [2], choosing dynamic learning rate and momentum term [3] or modifying the optimization strategy and/or employing adaptation rules other than the gradient descent [4]. It is rather very difficult to compare the performance of these algorithms, because most of them have been tested only on very special and relatively simple benchmark problems. Most of the variations that are proposed to improve the BP algorithm involve the use of learning rate, momentum and gain tuning[5] of the activation function to speed-up the network convergence and avoid getting stuck at local minima or by substituting BP with more efficient algorithms examples of which are Levenberg-Marquardt algorithm, Resilient back propagation algorithm and many others**.** A summary of proposed solutions along with possible weak features is discussed under different sections below:

A. *Learning rate and momentum coefficient.*

Learning rate can limit or expand the extent of weight adjustment in a training cycle. A higher learning rate can make the network unstable and can cripple the networks prediction ability. In the contrary if the learning rate is low, the training time of the network is increased. A high learning rate is used to accelerate learning until the weight adjustments begin to reach plateau. A constant learning rate is in efficient as BP needs a gradually declining rate of convergence. Nevertheless, the best choice of learning rate is problem dependent and may need some trial-and-error before a good choice is found. In fact, for best results, different values of learning rate should be used for each weight during the training process since no single value is optimal for all dimensions. The variable or adaptive learning rate increases the learning rate only to the extent that the network can learn without large error increase. It also decreases the learning rate as the error decreases or until stable learning is resumed. Reference [6] has put forward for consideration the use of adaptive coefficients in which the value of learning rate is a function of the error derivatives on consecutive updates. Another possible way to improve the rate of convergence is by adding some momentum to the adjustment expression. This can be accomplished by adding a fraction of the previous weight change to the current weight change. Low momentum causes weight oscillations and instability and thus preventing the network from learning. High momentum can cripple the network adaptability. For stable BP the momentum factor should be kept less than one. Momentum factors close to unity is needed to smooth error oscillation. During the middle of training, when steep slope occurs, a small momentum factor is recommended. However, during the end of training, large momentum values are desirable. Some researchers kept the momentum-coefficient fixed but later studies on static momentum-coefficient revealed that Back-propagation with Fixed Momentum (BPFM) shows acceleration results when the current downhill of the error function and the last change in weights are in similar direction, when the current gradient is in an opposing direction to the previous update, BPFM will cause the weight direction to be updated in the upward direction instead of down the slope as desired, so in that case it is necessary that the momentum-coefficient should be adjusted adaptively instead of keeping it fixed [7], [8]. Over the past few years several adaptive-momentum modifications are proposed by researchers. One such modification is BP with adaptive learning rate and momentum term [9].This algorithm converges more rapidly than those of using a fixed parameter, but at the penalty of extra computation. In this algorithm each weight has its own learning rate and momentum factor. In addition, both learning rates and momentum factors are adaptive at each iteration. Meanwhile Simple Adaptive Momentum (SAM) [10] was proposed to further improve the convergence capability of BPNN. SAM works by scaling the momentum-coefficient according to the similarities between the changes in the weights at the current and previous iterations. If the change in the weights is in the same 'direction' then the momentum-coefficient is increased to accelerate convergence to the global minima otherwise momentum-coefficient is decreased. SAM is found to lower computational overheads then the Conjugate Gradient Descent and Conventional BPNN. In 2009, R. J. Mitchell adjusted momentum-coefficient in a different way than [10], the momentum-coefficient was adjusted by considering all the weights in the Multi-layer Perceptrons (MLP). This technique was found much better than the previously proposed SAM [10] and helped improve the convergence to the global minima possible [11].

B. *Gain parameter.*

In feed forward algorithm the slope of the activation function is directly influenced by a parameter referred to as gain. For larger gain values the activation function approaches a step function. Whereas for smaller gain values the output values change from zero to unity over a large range of the weighted sum of the input values and the sigmoid function approximates a linear function. Some researchers kept the gain value fixed while some changed it adaptively.. In 2007, Nawi et al. [12] found that by varying the gain value adaptively for each node can significantly improve the training time of the network. Based on this concept [13] proposed an improvement over conventional back propagation which involves changing the momentum value adaptively by keeping the gain value fixed. The performance of this algorithm was compared with back propagation with adaptive gain and back propagation with simple momentum by simulating them on five classification problems and it was found that back propagation with adaptive momentum outperforms them. They further proposed an improvement by adjusting activation function of neurons in the hidden layer in each training patterns. The activation functions are adjusted by the adaptation of gain parameters together with adaptive momentum and adaptive learning rate value during the learning process. Results in [14] demonstrate that the learning rate, momentum coefficient and the gain of the activation function have a significant impact on training speed

C. *Activation Function.*

The choice of activation functions may strongly influence complexity and performance of neural networks and have been said to play an important role in the convergence of the learning algorithms [15]-[18]. Some types of activation functions have been proposed. Reference [19] used a combination of various functions, such as polynomial, periodic, sigmoidal and Gaussian functions. Reference [20] proposed Gaussians bars activation functions. Reference [21]-[23] used nonpolynomial activation functions. Reference [21] used rational transfer functions with very good results. Reference [24] used Lorentzian transfer functions. Reference [25] proposed a new class of sigmoidal functions and proved that they satisfy the requirements of the universal approximation theorem. Reference [26] proposed a new sigmoidal activation function with good results for modelling of dynamic, discrete time systems. Reference [27] used Hermite Polynomial with very satisfactory results. Reference [28] proposed a Max-Piecewise-Linear (MPWL) Neural Network for function approximation. Reference [29] introduced two new activation functions labelled sincos and sinc.Reference[30] used complementary log-log and probit functions to show that when the data follow a binomial distribution with characteristics of complementary log-log and probit functions using the logistic sigmoid function in the neural networks models is inadequate.

D. *Neural Network Model.*

Defining the topology of the neural network is the most important criteria in building a successful model able to converge and to generalize. Deciding the parameter of the neural network is not a straight forward task, several decisions should be taken e.g. the activation function to use, type of the multilayer network (e.g. fully connected or not), number of hidden layers and the number of neurons in each hidden layer. The number of hidden layers and the number of neurons in each one of these layers is a very critical decision to make. The number of nodes in the hidden layers has a big effect on the outcome. Network with too few hidden nodes may suffer under fitting and will not be able even to learn the training dataset. While a network with a big number of hidden nodes would suffer from Over fitting. Over fitting occurs when the limited amount of information contained in the training set is not enough to train all of the neurons in the hidden layers. It is also worth to mention that a large number of hidden neurons would dramatically increase the time needed for training. Several methods have been used to determine the best topology for the neural network. For instance,

[31], [32] have proposed Genetic Algorithm (GA) based methods to determine the best topology of the neural network models.

E. Levenberg-Marquardt Algorithm.

The Levenberg–Marquardt (L-M) algorithm [33]-[35] outperforms BP algorithm and many other conjugate gradient methods in a wide variety of problems. L-M is a blend of local search properties of Guass-Newton with consistent error decrease provided by gradient descent algorithm. The training of feed forward networks based on L-M is considered as an unconstrained optimization problem. The main disadvantage of L-M algorithm is its increased memory requirements to calculate jacobian matrix of the error function, determining the inverse of matrix with dimensions equal to the number of weights of neural network is cumbersome. Another disadvantage of L-M is that it does not always guarantee global optimum for a unconstrained optimization problem. The whole training process should be restarted, when the solution is acceptable-M method does not guarantee global optimum and it is just a heuristic that works extremely well in practical problems. The main drawback of this algorithm is its computational complexity to calculate matrix inversion. Generally inverse of Hessian is implemented by pseudo inverse method or singular value decomposition approach.

F. Resilient back propagation algorithm (RPROP).

RPROP [36] has been used successfully in training multi-layer perceptron neural networks, where it was shown that it can outperform the typical gradient descent learning procedure on a number of benchmark problems. RPROP takes advantage of the signs of the gradient in order to increase the learning rate for the descent in cases where consecutive gradients are of the same sign, and in order to decrease the learning rate for the descent when consecutive signs of the gradient are alternating. Furthermore, RPROP backtracks when two consecutive signs of the gradient are alternating. Only the sign of the derivative is used to determine the direction of the weight update; the magnitude of the derivative has no effect on the weight update. The size of the weight change is determined by a separate update value. The update value for each weight and bias is increased by a factor delt_inc whenever the derivative of the performance function with respect to that weight has the same sign for two successive iterations. The update value is decreased by a factor delt_dec whenever the derivative with respect to that weight changes sign from the previous iteration. If the derivative is zero, then the update value remains the same. Whenever the weights are oscillating the weight change will be reduced. If the weight continues to change in the same direction for several iterations, then the magnitude of the weight change will be increased**.**

III. COMPARISON OF RESULTS OF VARIOUS BP LEARNING ALGORITHMS

The performance of these improved versions of BP algorithm was verified by simulating them on certain problems. Following algorithms were analysed and simulated.

- The conventional Back Propagation (BP).
- The Back Propagation with Adaptive Gain (BP-AG).
- Back Propagation with Adaptive Gain, Adaptive Momentum and Adaptive Learning Rate (BP-AGAMAL).
- Back propagation with Adaptive momentum (BP-AM).
- Back propagation with momentum and adaptive learning rate (BP-AL).
- Levenberg-Marquardt (L-M***).***

TABLE I
PERFORMANCE COMPARISON FOR IRIS CLASSIFICATION PROBLEM

	BP	**BP-AG**	**BP-AGAMAL**
Accuracy	91.9	90.3	93.1
Failures	2	0	0

TABLE II
PERFORMANCE COMPARISON FOR CREDIT CARD CLASSIFICATION PROBLEM

	BP-AL	**L-M**
Mean Square Error	0.3186	0.2117
Network Training Time(sec)per epoch	0.00203	0.1022

TABLE III
MEAN SQUARE ERROR AND NETWORK TRAINING TIME FOR BP-AL and L-M

	BP	**BP-AG**	**BP-AM**
Accuracy	94.28	91.05	96.60
Failures	1	1	0

IV. DIRECTIONS FOR FUTURE WORK

The problem of local minima can be solved by implementing the mechanism of diverse curiosity. This detects the phenomenon of being trapped in the local minima and the occurrence of premature convergence. Upon detection of such phenomenon, the network would be able to escape from local minima with the use of stochastic disturbance, and become active again to explore better solution in its search space, thus the problem of local minima will be solved and the convergence rate of BP will be increased to some extent.

V. CONCLUSION

The Back-propagation Neural Network (BPNN) is a supervised learning neural network model highly applied in different applications around the globe. Although it is widely implemented in the most practical ANN applications and performs relatively well, it is suffering from a problem of slow convergence and convergence to local minima. This makes Artificial Neural Network's application very challenging when dealing with large problems. In the present paper, a deep and thorough study of improved versions of back propagation algorithm for feed forward NN is carried out. At the end of the paper, a comparison between the traditional BP network learning algorithm and the improved BP network learning algorithm is given. Although the improved versions of BP algorithm perform relatively better than standard BP algorithm but no one guarantees the global optimum solution and this question still needs to be answered.

REFERENCES

[1] Rumelhart, D. E., Hinton, G.E., Williams, R. J.," Learning internal representations by error propagation, *"Parallel Distributed Processing: Explorations in the Microstructure of Cognition. (1986).*

[2] R. J. Schalkoff, *Artificial Neural Networks*, McGraw-Hill, 1997.

[3] N. K. Bose, and P. Liang, *Neural Network Fundamentals with Graph Algorithms and Applications*, McGraw-Hill, 1996.

[4] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[5] Zweiri, Y.H., Seneviratne, L D., Althoefer, K," Stability analysis of three term back propagation algorithm,"*J.Neural Networks* 18, 1341-1347(2005).

[6] R. A. Jacobs, "Increased rates of convergence through learning adaptation, *"Neural Networks*,Vol.1 pp.295-307,1988.

[7] Shao, H., and Zheng,H.," A new bp algorithm with adaptive momentum for FNNs training," in*: GCIS 2009, Xiamen, China*, pp. 16--20 (2009)

[8] Rehman, M. Z., Nawi, N.M., Ghazali, M. I, "Noise-induced hearing loss (NIHL) prediction in humans using a modified back propagation neural network," in: *2nd International Conference on Science Engineering and Technology*, pp. 185--189 (2011)

[9] Chien cheng Yu,and Bin Da Liu,"A backpropogation algorithm with adaptive learning rate and momentum coefficient," *IEEE,2002*.

[10] Swanston, D. J., Bishop, J.M., and Mitchell, R. J," Simple adaptive momentum: new algorithm for training multilayer perceptrons," *J. Electronic Letters*. 30, 1498-1500 (1994)

[11] Mitchell, R. J.,"On simple adaptive momentum," in: *CIS 2008*, London, United Kingdom, pp.01--06 (2008)

[12] Nawi, N. M., Ransing, M. R. and Ransing, R. S,"An improved conjugate gradient based learning algorithm for back propagation neural networks," *J. Computational Intelligence*. 4, 46--55 (2007)

[13] M. Z. Rehman , N. M. Nawi ," Improving the Accuracy of Gradient Descent Back Propagation Algorithm (GDAM) on Classification Problems," *(IJNCAA)* 1(4): 838-847ns, (2011) (ISSN: 2220-9085)

[14] Maier, H. R. and Dandy, G. C., "The effect of internal parameters and geometry on the performance of back-propagation neural networks," an empirical study. *Environmental Modelling and Software*. 13(2): p. 193-209. (1998).

[15] Chandra P, Singh Y.," An activation function adapting training algorithm for sigmoid feed forward networks," *Neurocomputing 61*:429–437(2004).

[16] Duch W, Jankowski N," Survey of neural transfer functions," *Neural Comput Appl 2*:163–212, 1999.

[17] Duch W, Jankowski N.," Transfer functions: hidden possibilities for better neural networks, "in *9th European symposium on artificial neural networks*, pp 81–94, 2001.

[18] Singh Y and Chandra P, "A class? 1 sigmoidal activation functions for FFANNs*," J Econ Dynamic Control* 28(1):183–187,2003.

[19] Pao YH, "Adaptive *pattern recognition and neural networks*,"2nd edn. Addison-Wesley, New York, 1989.

[20] Hartman E, Keeler JD, Kowalski, "Layered neural networks with Gaussian hidden units as universal approximations," *Neural Comput Appl* 2(2):210–215, 1990.

[21] Hornik K, "Approximation capabilities of multilayer feed forward networks. Neural Net 4(2):251–257, 1991.

[22] Hornik K , " Some new results on neural network approximation," Neural Net 6(9):1069–1072.

[23] Leshno M, Lin VY, Pinkus A, Schocken S., " Multilayer feed forward networks with a nonpolynomial activation function can approximate any function," *Neural Net* 6(6):861–867,1993.

[24] Leung H, Haykin S, " Rational function neural network," *Neural Comput Appl* 5(6):928–938,1993.

[25] Giraud B, Lapedes A, Lon C, Lemm J , " Lorentzian neural Nets," Neural Net 8(5):757–767,1995.

[26] Skoundrianos EN, Tzafestas SG, "Modelling and FDI of dynamic discrete time systems using a MLP with a new sigmoidal activation function," J Intell Robotics Syst 41(1):19–36,2004

[27] Ma L, Khorasani K (2005), " Constructive feed forward neural networks using hermite polynomial activation functions," *IEEE Trans Neural Net* 16(4):821–833

[28] Wen C, Ma X , " A max-piecewise-linear neural network for function approximation,"*Neurocomputing* 71:843–852,2005.

[29] Efe MO , " Novel neuronal activation functions for feed forward neural networks," *Neural Process Lett* 28:63–79,2008.

[30] Gomes GSS, Ludermir TB, "Complementary log-log and probit: activation functions implemented in artificial neural networks,"in: 8th International conference on hybrid intelligent Systems," *IEEE Computer Society*, pp 939–942,2008.

[31] Gwang-Hee, K., Jie.-Eon., Y., Sung-Hoon., A., Hun-Hee, C. & Kyung-In, K., "Neural network model incorporating a genetic algorithm in estimating construction costs," *Building and Environment*, Vol. **39**(11), pp. 1333 – 1340, 2004.

[32] White, D. & Ligomenides, " GANNet: A genetic algorithm for optimizing topology and weights in neural network design *,"New Trends in Neural Computation. Springer Berlin / Heidelberg*, Vol. **686**, pp. 322-327, 1993.

[33] Bazaraa, M. S., Sherali, H. D., & Shetty, C. M, *Nonlinear programming. Theory and algorithms*, 2nd ed., India, Wiley2004.

[34] Madsen, K., Nielsen, H. B., & Tingleff, O,*Methods for non-linear least squares problems* ,(2nd ed.). Technical University of Denmark, 2004.

[35] Marquardt, D. W, "An algorithm for least-squares estimation of nonlinear Parameters," *Journal Society Industrial Applied Mathematics*, 11(2), 431–441, 1963.

[36] M. Riedmiller, H. Braun, "A direct adaptive method for faster back-propagation learning: the RPROP algorithm, "in: *Proc. of the International Conference on Neural Networks*, 1993, pp. 586–591.