Article can be accessed online at http://www.publishingindia.com

Comparative Study of Adaptive Learning Rate with Momentum and Resilient Back Propagation Algorithms for Neural Net Classifier Optimization

Saduf Afzal*, Mohd. Arif Wani**

Abstract

Learning algorithms are generally used to optimize the convergence of neural networks. We need to optimize the convergence of neural networks in order to increase the speed and accuracy of decision making process. One such algorithm that is used to facilitate the optimization process is back propagation learning algorithm. The objective of this study is to compare the performance of two variations of back propagation learning algorithm (Adaptive learning rate with momentum and Resilient). Both the algorithms are experimented on a variety of classification problems in order to assess the efficiency of these two learning approaches. Experimental results reveal that during testing and training Resilient propagation algorithm outperforms back propagation with Adaptive learning rate and momentum.

Keywords: ANN, Back-propagation, RPROP, Learning Rate, Momentum

1. INTRODUCTION

Classification is one of the most difficult decision making tasks. In classification, a set of instances are given called a training set, where each instance consists of several features or attributes. Attributes are either continuous or categorical. One of the attribute which is called the classifying attribute indicates the class to which each instance belongs. The objective of the classification problem is to build a model of classifying attribute based upon the other attributes. Most of the problems in Marketing, biological science, industry and medicine can be treated as classification problems. There are a number of classification methods, however the latest research in neural classification (Zhang, 2000) have established that neural networks are better alternatives to various traditional classification methods. Artificial Neural Network (ANN) is a mathematical model of the networks of neuron in the brain, sharing similar functionalities such as receiving input, processing it and then generating the output. The popular structure of neural networks is the feed-forward multi-layer perceptron (MLP) in which neurons are placed in some layers and signal is transmitted in one direction. A multilayer feed-forward neural network is made up of multiple layers i.e. input layer, one or more intermediary layers (hidden layers) and an output layer of neurons. The hidden layer is responsible for performing the intermediate computation on the inputs and then directs them to the output layer. Every node in a layer is connected to every other node in the neighbouring layer. The output of each layer serves as input to the next layer. The main purpose of training an ANN is to produce desired output when a set of input values are applied; the most commonly used training algorithm for feed forward neural network is back-propagation (BP). Back-Propagation algorithm is the most popular supervised learning algorithm proposed by (Rumelhart, 1986).

2. BACK PROPAGATION LEARNING ALGORITHM (BP)

BP is an optimization procedure based on gradient descent that adjusts weights to reduce the system error. During the learning process a BP network is partitioned

* Department of Computer Sciences,University of Kashmir, Srinagar, India. E-mail: Sadaf.afzal.123@gmail.com
** Department of Computer Sciences, University of Kashmir, Srinagar, India. E-mail: awani@uok.edu.in

into two stages: Forward pass and backward pass. In Forward pass the feed forward back-propagation network is presented with a training set vector which consists of both 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 neurons in hidden layer. These hidden layer outputs become inputs to the output neurons. The final output of the network is obtained from the activations of the output layer. The pattern that is obtained from the network is then compared with the desired pattern and based on the difference an error is calculated. This error is then used to adjust the weights backwards through the neural network. After calculating the error the algorithms steps one layer back and recalculates the weights of the output layer (the weights between the last hidden layer and the neurons of the output layer). The algorithm then computes the output at the last hidden layer and computes new values for its weights (the weights between the last and second last hidden layers). The algorithm continues calculating the error and computes new weight values, moving from one layer to another in backward direction towards the input layer (saduf, 2013). This procedure is repeated for each training pattern. The new weight vector W $(_{t+1})$ is adjusted as:

$$W_{(t+1)} = W_t\,\eta\,\frac{\partial E}{\partial W} + \mu\,W_{(t-1)} \tag{1}$$

Where η is the learning rate, $\frac{\partial E}{\partial W}$ is the error gradient, μ is the momentum coefficient.

Each iteration through the entire training set is called a cycle or an epoch. The process is then repeated until the stopping criteria of the training algorithm are met. There are numerous variations of BP; two of these are BP with adaptive learning rate and momentum (BP-ALM) and resilient propagation algorithm (RPROP).

A, Back propagation with Adaptive learning rate and Momentum (BP-ALM)

Learning rate is one of the most important factors that influence the convergence rate of a learning algorithm. In standard BP learning rate is kept fixed. A higher learning rate reacts quickly to input changes and can make network unstable. The changes can be too extreme and it can cripple the networks prediction ability. In the contrary if the learning rate is too low, it can increase the training time of the network to a large extent. However, high learning rate increases the risk that the weight search may jump over a minimum error condition. The consensus among researchers is that adaptive learning rates stabilize the risk of failure and accelerate the training process. An adaptive learning rate increases the learning rate only to the extent that the network can learn without a significant increase in error value. It also decreases the learning rate as the error decreases or until stable learning is resumed. In this particular study BP-ALM is implemented by traingdx (Howard, 2002) in MATLAB. In this algorithm weight is adjusted in the same way as standard BP but with varying learning rate. The new weight vector W $_{(t+1)}$ is updated as follows:

$$W_{(t+1)} = W_t\,\eta\,\frac{\partial E}{\partial W} + \mu\,W_{(t-1)} \tag{2}$$

Where

$$\eta(t+1) = \begin{cases} \eta(t) * \mathrm{lr\_d} & \text{if } E(t) > (t-1) \\ \eta(t) * \mathrm{lr\_i} & \text{if } E(t) < (t-1) \\ \eta(t) & \text{if } E(t) = (t-1) \end{cases}$$

lr_d and lr_i are the increase /decrease factor by which the learning rate is changed. Learning rate is increased if the new error is less than old error by a predefined ratio and vice versa. The values are set as: the predefined ratio is set to 1.04, lr_i is set to 1.05 and lr _d is set to 0.7.

B. Resilient Propagation algorithm (RPROP)

In RPROP (Riedmiller, 1993) (Riedmiller, 1994) only the sign of derivative is used to determine the weight update value. If the partial derivative of the corresponding weight has the same sign for the two consecutive iterations, the weight update is increased by a factor □+ otherwise the weight update value is decreased by a factor □ˉ .if the derivative is zero, then the weight update value remains same. However if the weight continues to change in the same direction i.e. if the derivative is positive, the weight is decreased by its update value otherwise the update value is added. The new weight vector in RPROP is given as

$$W_{(t+1)} = W_t\,\eta\,(t) * sign\left(\frac{\partial E}{\partial W}\right) \tag{3}$$

The pseudo code of algorithm is given as follows:

```
For all weights and biases {
If $((\frac{\partial E}{\partial W}(t-1) * \frac{\partial E}{\partial W}(t) > 0)$ then
{
    $\Delta(t) = \text{minimum}(\Delta(t-1) * \eta^{+}, \Delta\max)$
        $\Delta w(t) = -\text{sign}(\frac{\partial E}{\partial W}(t)) * \Delta(t)$
        $W(t+1) = w(t) + \Delta w(t)$
}
Else if ($\frac{\partial E}{\partial W}(t-1) * \frac{\partial E}{\partial W}(t) < 0$) then
{
    $\Delta(t) = \text{maximum}(\Delta(t-1) * \eta^{-}, \Delta\min)$
        $W(t+1) = w(t) - \Delta w(t-1)$
        $\frac{\partial E}{\partial W}(t) = 0$
}
Else if ($\frac{\partial E}{\partial W}(t-1) * \frac{\partial E}{\partial W}(t) = 0$) then
{
        $\Delta w(t) = -\text{sign}(\frac{\partial E}{\partial W}(t)) * \Delta(t)$
        $W(t+1) = w(t) + \Delta w(t)$
} }
```

Where maximum (minimum) returns the maximum (minimum) number. Sign operator returns +1 if the argument is positive else returns -1. Δ is the weight update and η is the increase or decrease factor.

3. EXPERIMENTS AND RESULTS

In order to compare the performance of RPROP and BP with adaptive learning rate and momentum, we have simulated them on three classification problems: Numeral recognition, Iris and Breast cancer classification problem. For each problem, 10 different trials were made to eliminate the possible problems associated with random weight setting. For each run, the numbers of epochs required for convergence are reported and from them mean of the number of epochs for 10 trials are collected see Table 1. A network is said to be converged when the selected error criterion is met. The criteria used to assess the performance of these algorithms focuses on the speed of convergence i.e. the number of epochs required to converge. For each problem, a neural network with one hidden layer with a sigmoid activation function is used. In order to assess the performance of algorithms a table is presented which summarizes the performance of the algorithms. The attributes of table include the minimum number of epochs, the mean value of epochs and the maximum number of epochs that NN takes to learn. The maximum epoch limit is set to 1000. If an algorithm fails to converge within the above limit, it is considered as a failure. All the simulations have been carried out using MATLAB R2009a.

Table 1: Network Description

Network description	*Numeric Recognition*	*Iris*	*Cancer*
no. of layers	3	3	3
no. of hidden layers	1	1	1
no. of neurons in input layer	45	4	9
no. of neurons in output layer	10	3	2
no. of neurons in hidden layer	12	5	5

A. Numeral Recognition:

One of the common applications of neural networks is in handwriting or numeral recognition (pattern recognition) (Muntaser, 2008). This enables a computer to translate character images into a text file, it allows us to take printed document and put it into the computer in editable form. In this paper we have evaluated the performance of RPROP and BP with adaptive learning rate and momentum on numeral recognition problem. For the sake of brevity, we limit our task to the recognition of digits from 0 to 9. Each digit is represented by a 5 X 9 bitmap as shown in figure 2. The numbers of neurons in the input layer are decided by the number of pixels in the bitmap. The bitmap in our case consists of 45 pixels and thus 45 input vectors are needed .the output layer has 10 neurons, one neuron for each digit to be recognised. Thus each target vector is a 10 element vector with 1 at the position of the number it represents and the other positions are 0. In practice, network does not always receive a perfect vector as input so it must be able to deal with noise. Noisy vectors are the input patterns that are distorted by adding random values chosen from a normal distribution to the binary vectors that represent the bitmaps of ten digits. In order to create a network that can handle noisy vectors, we train the network on both noisy as well as perfect vectors. Both the algorithms were evaluated on noisy vector as well as perfect vector. The selected architecture of the network for numeral recognition is 45-12-10., with the target error set to 0.001. Learning rate is set to 0.1 and the momentum coefficient is set to 0.9. The binary input vectors representing the bitmaps of the respective digits are directly fed into the network. Figure 3 and Figure 4 demonstrates the results of training BP-ALM and RPROP for one trial. The performance evaluation of the network trained with BP-ALM and RPROP are demonstrated in figure 5 and figure 6;

Figure 2. Bitmap of 9

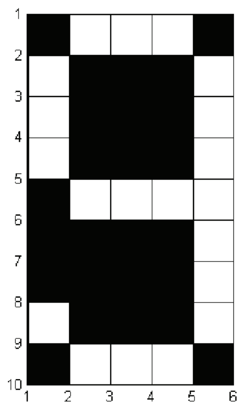

Figure 3. Error Performance of BP-ALM

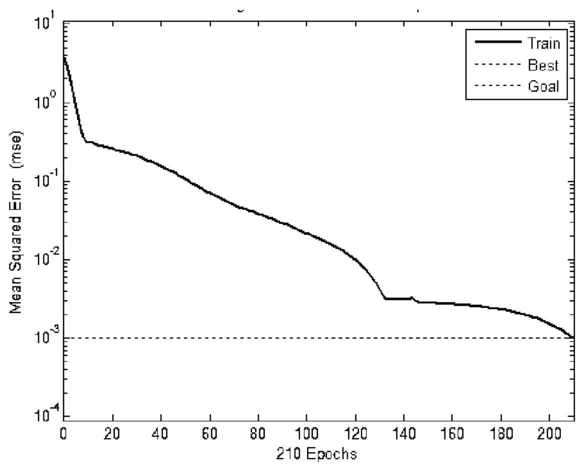

Figure 4. Error Performance of RPROP

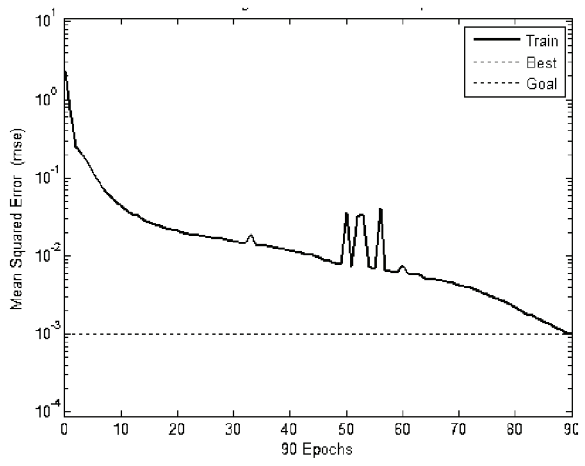

Figure 5. Percentage Error Graph of BP-ALM

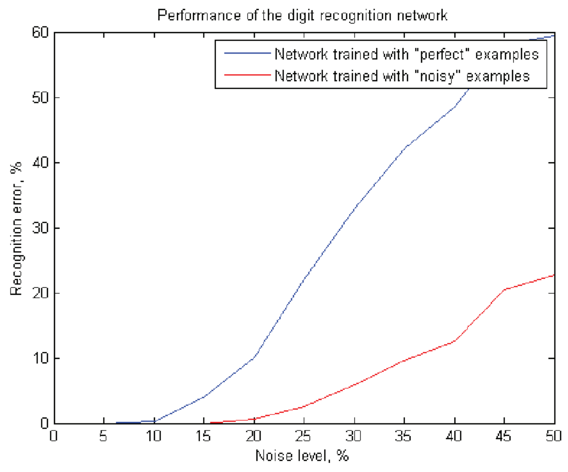

Figure 6. Percentage Error Graph of RPROP

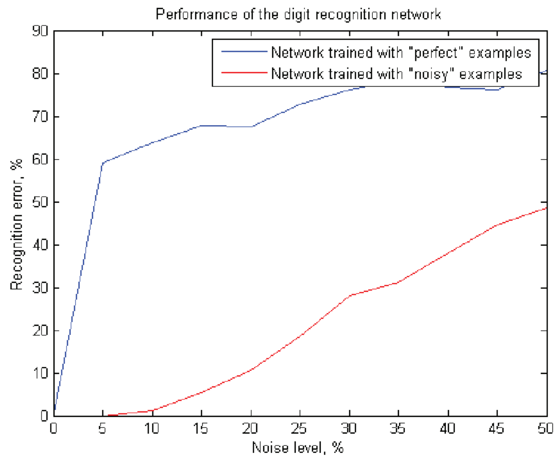

Table 2: Algorithm Performance on Perfect Vectors

Algorithm	*Mean*	*Max*	*Min*
BP-ALM	286	449	210
RPROP	102	146	64

Table 3: Algorithm Performance on Noisy Vector

Algorithm	*Mean*	*Max*	*Min*
BP-ALM	111	200	44
RPROP	84	128	14

From the TABLE 2 and TABLE 3 it can be clearly seen that RPROP shows better performance in reaching the desired target error value than the BP-ALM. For the perfect as well as noisy data set RPROP performs significantly better than BP-ALM with RPROP's mean being 102 and 84 epochs for perfect and noisy vectors. This indicates that for numeral recognition problem RPROP is a better choice than BP-ALM for training neural networks.

B. IRIS classification problem IRIS flower data set classification problem was made famous by (Fisher, 1936) who used it to illustrate principles of discriminant analysis. This data set consists of 150 samples that contain three classes of Iris plants: Iris Setosa, Iris Versicolor, and Iris Verginica. Each plant in the dataset is represented by four variables: sepal length, sepal width, and petal length and petal width. The sepal length ranges between 4.3 and 7.9cm, sepal width between 2.0 and 4.4cm, petal length between 1.0 and 6.9cm and petal width between 0.1 and 2.5cm. The selected architecture of network is 4-5-3 i.e. network has 4 input neurons, 5 hidden neurons and 3 output neurons. Sigmoid activation function is used for hidden and output neurons. Learning rate is set to 0.1 and the momentum coefficient is set to 0.9 in BP-ALM., the target error is set to 0.001.

Table 4: Algorithm Performance on IRIS Problem

Algorithm	*Mean*	*Max*	*Min*
BP-ALM	190	346	104
RPROP	34	44	20

From Table 4, it can be seen that RPROP algorithm performs quicker than BP-ALM. It takes less epochs to reach the desired target error with a mean of 34 epochs, which is significantly better than BP-ALM.

C. Breast Cancer Classification Problem

This dataset was generated at university of Wisconsin Hospital by (Wolberg, 1990) and is available at UCI Machine Learning Repository. The cancer dataset problem is a realistic classification problem. The objective is to classify whether a tumour is benign or Malignant based on the input pattern that is presented. The input pattern consists of following nine attributes: clump thickness, uniformity of cell size, uniformity of cell shape, marginal adhesion, single epithelial cell size, bare nuclei, blond chromatin, normal nucleoli and mitosis. The selected architecture of the neural network is 9-5-2 i.e. the network has 9 input neurons, 5 hidden neurons and 2 output neurons. The target error is set to 0.001. Learning rate and momentum coefficient are set to 0.1 and 0.9.

Table 5: Results of Simulation on Breast Cancer Classification Problem

Algorithm	*Mean*	*Max*	*Min*
BP-ALM	164	219	130
RPROP	27	41	17

For the Breast cancer classification problem there is significant difference between the training times of RPROP and BP-ALM. The difference is well indicated in the TABLE 5, with RPROP mean being 27 epochs and BP-ALM mean being 164 epochs.

4. CONCLUSION

This paper studies the performance of RPROP and BP with adaptive learning rate and momentum. The study is performed on Numeral recognition, Iris and Breast cancer classification problem. Study reveals that RPROP has a better convergence speed than BP with adaptive learning rate and momentum. This concludes that for training neural networks RPROP is a better choice. Further research can be done in this field by evaluating the performance of RPROP on other classification problems.

REFERENCES

[1] Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annual Eugenics*, 7(2), 179-188.

[2] Howard, D. & Mark, B. (2002). *Neural Network Toolbox for use with Matlab*. User's Guide Version 4.

[3] Wahed, M. A. (2008). Adaptive learning rate versus Resilient back propagation for numeral recognition. *Journal of Al-Anbar University for Pure Science*, 2(1), 94-105.

[4] Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). *Learning Internal Representations by Error Propagation*. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Cambridge: MIT Press.

[5] Braun, R. H. (1993). *A Direct Adaptive Method for Faster Back-Propagation Learning: The RPROP Algorithm*. Paper presented at the Proceedings of the International Conference on Neural Networks (pp. 586-591).

[6] Riedmiller, M. (1994). RPROP Description and Implementation Details Technical Report. University of Karlsruhe W-76128 Karlsruhe.

[7] Saduf, M. A. W. (2013). Comparative study of back-propagation algorithms for neural networks. *International Journal of Advanced Research in Computer Science & Software Engineering,* 3(12), 1151-1156.

[8] Wolberg, W. H. & Mangasarian, O. L. (1990). *Multi-surface Method of Pattern Separation for Medical Diagnosis Applied to Breast Cytology.* Paper presented at National Academy of Sciences, 87, 919-996.

[9] Zhang, G. P. (2000). *Neural Networks for Classification: A Survey*. IEEE Transactions on Systems Man and Cybernetics. Part C: Applications and Reviews, 30(4), 451-462.