IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research

Website: www.ijdacr.com (Volume 4, Issue 8, March 2016) **ISSN 2319-4863**

Feed-Forward Neural Networks Training: A Comparison Between Two Approaches of Back-Propagation Learning Algorithm

Belqasem Aljafari

Email: belqasem.aljafari@gmail.com

***Abstract*— Back-propagation is one of the most famous training algorithms for multilayer perceptrons. Unfortunately, it can be very slow for practical applications. Over the last years many improvement strategies have been developed to speed up back propagation. It's very difficult to compare these different techniques, because most of them have been tested on various specific data sets. Most of the reported results are based on some kind of tiny and artificial training sets. It's very doubtful if these results hold for more complicate practical application. In this report an overview of many different speedup techniques is given. All of them were assessed by a very hard practical classification task, which consists of energy efficiency data set.**

***Keywords*— Back-propagation, Artificial Neural Networks (ANNs), Multi-Layer Perceptions (MLPs), and Momentum coefficient (α).**

I. Introduction

This report is intended to summarize our experience using many different speedup techniques for the back propagation algorithm. Most of these algorithms are using many parameters, which have to be tuned by hand. So hundreds of the tests runs have to be performed. It's beyond the scope of this paper to discuss every approach in detail. We rather group the different approaches in some classes of algorithms and discuss these classes. The proposed approach combines Artificial Neural Networks (ANNs) and Back-propagation analysis to assess and predict the heating and cooling energy efficiency of residential buildings.

II. Artificial Neural Networks (ANNs)

ANNs have been proven to be an efficient approach in many areas, such as aerospace, automotive, mathematics, engineering, medicine, economics, meteorology, psychology, neurology, and many others [1]. There are many learning algorithms used for ANNs. The most popular algorithm is Back-propagation (BP) neural network as shown in Figure 1. Fausett [2] stated that a BP neural network is also called Multi-Layer Perceptions (MLPs) and is based on gradient descent optimization approach in which the total squared error of the output signals is minimized. Indeed, I will use in my project two approaches of Back-propagation, which are Standard Back-propagation Algorithm and Batch Updating Algorithm.

III. Standard Back-Propagation Algorithm

BPLAs were proposed by Rumelhart [3]. They have since become famous learning algorithms among ANNs. In the learning process, to reduce the inaccuracy of ANNs, BPLAs use the gradient-decent search method to adjust the connection weights. The structure of a back-propagation ANN is shown in Figure 1. The output of each neuron is the aggregation of the numbers of neurons of the previous level multiplied by its corresponding weights. The input values are converted into output signals with the calculations of activation functions. Back-propagation ANNs have been widely and successfully applied in diverse applications, such as pattern recognition, location selection and performance evaluations.

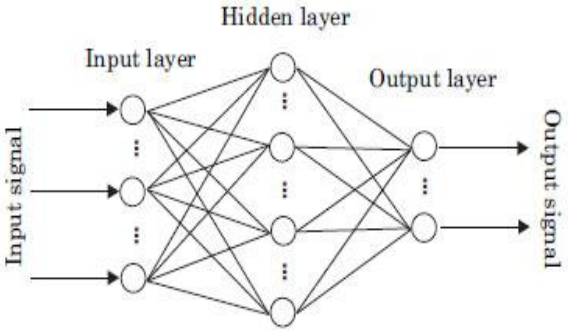

Figure 1. Structured chart of neural network.

Fig. 1 Structured Chart of Neural Network

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research

Website: www.ijdacr.com (Volume 4, Issue 8, March 2016) ISSN 2319-4863

The procedure of a BPLA is described step by step as follows:

Step 1: Normalize the data: The obtained data is mapped to the bound [0; 1], so as adjust the defined range of attributes and avoid the saturation of neurons.

Step 2: Set the network parameters: Number of hidden layers: In general, one or two hidden layers produce better convergence. Too few or too many hidden layers yield poor results [4]. Number of units in hidden layers: The larger the number of units in the hidden layers, the slower the speed of convergence becomes. Too small of a number of units in the hidden layer is not sufficient to respond to the interactions between input variables. On the other hand, a large number of units generates smaller error but the complexity of the network leads to slow convergence.

Learning rate (μ): In general, too fast or too slow of a learning rate is detrimental to network convergence. I selected values between 0.1 and 1.0 to test the networks.

Momentum coefficient (α): Momentum coefficients also affect network convergence. To avoid error fluctuations in the convergence process, I selected the values between 0.01 and 1.0 to test the networks.

Transfer function: Sigmoid function $F(x) = (\frac{1}{1+e^{-x}})$ is the transfer function that I will use in my project and its value is in the interval of [0, 1].

Step 3: Calculate the neuron's output signal: Each neuron's output signal is calculated by $etj = \sum_{i=1\sim m} Wji\ Xi + bj$, and sigmoidal function is used to convert netj for each neuron in the hidden layers. netj is the output of neuron j. wji is the weight on the connection from neuron i to j; xi is the input signal of the neuron i; and bi is the bias of neuron j.

Step 4: Calculate the error values: Steps 3-5 are repeated until the network is converged. The error is calculated by

$$Eq \triangleq \frac{1}{2}(dq - Xout^3)^T(dq - Xout^3)$$

$$\triangleq \frac{1}{2}\sum_{m=1}^{n=3}(dq - Xout^3)^2$$

Step 5: Updating the weight by this equation:

$$Wji^s(K+1) = Wji^s(K)\mu^s\delta j^s Xout, i^{s-1}$$

IV. Batch Updating Algorithm

Most of the improvement strategies are based on batched back-propagation. Using this update scheme the gradient of the complete error surface is known and can be used to calculate new connection strength more accurately than just using the gradient of the partial error surface given by a single training pattern. Batch updating takes the average of the input and use them as input in the second neuron. It will use 70% for (training), and 30% for independent validation. Using the training data to assess the final performance quality of the network can lead to overfitting this can be avoided by using independent validation [5](validation set, test set).

V. Methodology

A. Data Description

The dataset used in this project was obtained from the University of California-Irvine repository. The data was obtained by simulating 12 different building. The dataset comprises 768 samples and 8 features that will be used to predict two real values responses. The eight attributes or features include: Relative Compactness, Surface Area, Wall Area, Roof Area, Overall Height, Orientation, Glazing Area, and Glazing Area Distribution. The two outputs or response variables are heating load and cooling load.

TABLE I

Variable Type	**Variable Name**
Input Variables	Relative Compactness (X1) Surface Area (X2) Wall Area (X3) Roof Area (X4) Overall Height (X5) Orientation (X6) Glazing Area (X7) Glazing Area Distribution (X8)
Output Variables	Heating Load (Y1) Cooling Load (Y2)

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research

Website: www.ijdacr.com (Volume 4, Issue 8, March 2016) ISSN 2319-4863

VI. RESULTS AND DISCUSSION

Table 2 shows experiments of standard back propagation with ($\Delta\mu$).

$$\mu(k) = \mu o \frac{1}{1 + \frac{K}{Ko}}$$

TABLE II

Learning Rate	0.1	0.3	0.4	0.5	0.7	0.9
MSE	0.0735	0.0718	0.0722	0.0724	0.0758	0.0795
Epoch	590	584	580	490	530	546

As we can obtained from Table 1 that the smallest value was at $\mu = 0.3$ with

$MSE = 0.0718$ and 584 Epochs.

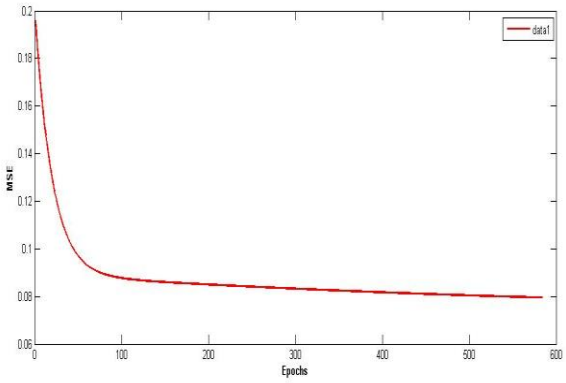

Fig. 2 Standard Back-propagation with ($\Delta\mu$)

This graph showes that the data stopped at MSE= 0.0718 at $\mu = 0.3$ and 584 Epochs.

Table 2 shows experiments of standard back propagation with variable momentum (Constant μ+$\Delta\alpha$).

$$\Delta Wji^s(K+1) = \mu^s \delta j^s(k) Xout, i^s(k) + \alpha \Delta Wji^s(k-1)$$

TABLE III

Learning Rate	0.3		
Momentum	0.01	0.5	0.9
MSE	0.0262	0.0174	0.0115
Epoch	284	291	280

As we can observe from Table 2 that momentum improves the convergence speed of the standard back propagation by introducing stabilization weight changes. Using variable momentum coefficient (MC) back propagation algorithm, the small average mean squared error obtained is 0.0115. ($\mu = 0.3$, and $\alpha = 0.9$) with 280 Epochs.

Table 3 shows experiments of batch updating with ($\Delta\mu$).

TABLE IV

Learning Rate	0.1	0.3	0.4	0.5	0.7	0.9
MSE	0.0917	0.083	0.1099	0.1088	0.1112	0.115
Epoch	261	144	285	270	242	221

It is obvious that batch updating can significantly speed up the convergence. It is very fast compare to standard back propagation with ($\Delta\mu$). Using variable learning rate (VLR) back propagation algorithm, the small average mean squared error obtained is 0.0832. The speed is found to be slower taking less than 200 epochs.

IJDACR
International Journal Of Digital Application & Contemporary Research
International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 4, Issue 8, March 2016) **ISSN 2319-4863**

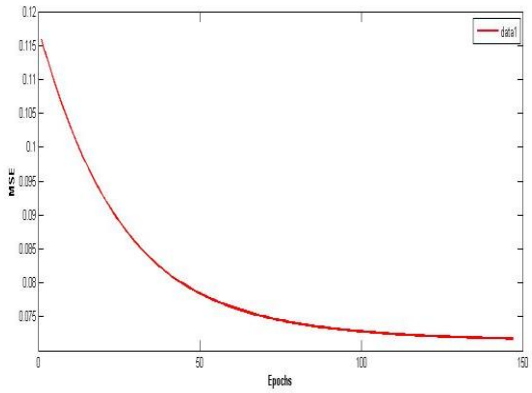

Fig. 3 batch updating with ($\Delta\mu$)

Table 4 shows experiments of batch updating with variable momentum (Constant μ+$\Delta\alpha$). $\Delta Wji^s(K+1) = \mu^s \delta j^s(k) Xout, i^s(k) + \alpha \Delta Wji^s(k-1)$

TABLE V

Learning Rate	0.3		
Momentum	0.01	0.5	0.9
MSE	0.0158	0.0154	0.0153
Epoch	93	96	87

The average mean squared error obtained using batch updating with momentum back propagation algorithm is 0.0153 indicating performance of batch updating with momentum is higher than VLR algorithm. Also, batch updating with momentum is much faster than VLR. Comparing the two parameters – speed and performance, batch updating with momentum is found to be better than Variable learning rate in all two parameters.

VII. CONCLUSION

In this paper we report the results of an experimental analysis of the use of different neural models for predicting two real values of energy efficiency, which are heating and cooling load, and meanwhile, there is no physical meaning of dataset. In particular we consider two approaches of Back-propagation: the Standard Back Propagation and batch Updating Algorithm. This project has compared the speed and performance of Standard Back-propagation algorithm and Batch Updating algorithm in terms of variable learning rate and momentum coefficient. Results show Batch Updating algorithm with variable learning rate to be better than Standard Back-propagation Algorithm. Furthermore, results show Batch Updating algorithm with momentum to be better in all parameters.

REFERENCES

[1] Kreider, J., 1991, "Artificial Neural Networks Demonstration for Automated Generation of Energy Use Predictors for Commercial Buildings," Ashrae Transactions, 97(1), 775–779.

[2] Fausett, L., 1994, "Fundamentals of Neural Networks: Architectures, Algorithms, and Applications," Prentice-Hall.

[3] D. E. Rumelhart, G. E. Hinton and R. J. Williams, Learning representations by back-propagating errors, Nature , vol.323, no.6088, pp.533-536, 1986.

[4] M. Li, X. Guo, X. D. Tan and J. H. Yuan, Predictive method for power growth based on improved BP neural network and rebuilding of new industry structure, Journal of Central South University (Science and Technology) , vol.38, no.1, pp.143-147, 2007.