International Journal of Engineering & Technology, 7 (3.28) (2018) 124-130

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Improved the Speed Up Time and Accuracy Training in the Batch Back Propagation Algorithm via Significant Parameter

Mohammed Sarhan Al_Duais*, Fatma Susilawati Mohamad, Mumtazimah Mohamad

Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Terengganu, Malaysia
**Corresponding Author E-mail: sarhan2w@gmail.com*

Abstract

Although the batch backpropagation (BBP) algorithm is a new style for weight updating, it is slow training and there are several parameters that needed to be adjusted manually. The most significant parameter for improving the efficiency of the batch back propagation algorithm is learning rate. The drawbacks of the BBP algorithm are its slow learning rate and easy convergence to the local minimum. To overcome this problem, we have created a new dynamic learning rate (LR) to escape the local minimum, which enables a faster training time for the batch back propagation algorithm. The dynamic batch backpropagation (DBBPL) algorithm, which uses this dynamic learning rate, is presented in this paper. This technique was implemented using a sigmoid function, and the two-dimensional exclusive OR problem, the balance dataset, and the Iris dataset were used as benchmarks with different structures to test the efficiency of the dynamic learning rate. The real datasets were divided into a training set and a testing set. 75 experiments were carried out using MATLAB software (2012a). From the experimental results, the DBBPL algorithm provides superior performance in terms of training and quickly training with the high level of accuracy compared to the BBP algorithm, whereas the accuracy rates of the structures were 98.7% and 99.1%, and processing times of the improved algorithm were 3936 and 4755 times faster, respectively than the BBP algorithm, and with existing works.

Keywords: *artificial neural network; batch back-propagation algorithm; local minimum; processing time improved; dynamic learning rate.*

1. Introduction

Artificial neural networks (ANNs) have been proven to be suitable for many tasks in pattern recognition and machine learning [1]. The BP algorithm is very popular in supervisor training and learning [2-3] and the batch BP algorithm is widely used in many applications, it covers robotics, automation, and weight changes in ANNs [4-6]. The BP algorithm has led to tremendous breakthroughs in applications involving multilayer perceptions [7]. Gradient descent is commonly used to adjust the weight by using a change in E; however, this approach is not guaranteed to converge the global minimum error [8-9].

The BP algorithm is accurate in terms of training [10,11]; however, the primary problem with this algorithm is its slow learning rate and easy convergence to a local minimum and its tendency towards training saturation [12-13]. In addition to these disadvantages of the BP algorithm, several parameters need to be adjusted manually, such as learning rate and momentum term [14]. To solve this problem, several techniques have been developed to improve learning, to speed up the BP algorithm or to escape a local minimum, such as the heuristic approach. Heuristic methods are currently the most widely used methods for improving the learning rate of the BP algorithm. Learning rate is the most significant parameter affecting the weight update of a neural network. Many studies of this have been carried out such as that in [15], proposing a new algorithm involving adjusting the weight training through a penalty for avoiding the local minima. The relationship between the penalty and the learning rate has been made. The penalty affects the updating of weight. The values 0.013 and 0.001 were selected manually for learning rate and penalty respectively. The results were compared with the backpropagation (SBP) algorithm. The study in [16] improves the batch BPAP algorithm using a proposed dynamic learning rate with a penalty. The penalty coefficient is set at $\lambda > 0$ and the learning rate set at $\lambda = 0.15$. The algorithm has a 2:2:1 structure and uses a sigmoid activation function. The weight update in the batch BPAP algorithm is bounded during training. The experimental results show that the BPAP has faster training than the BP, with a fixed learning rate. The study in [17] investigated the effects of the input parameters using three different structures: the BP algorithm, the BP algorithm with a momentum factor and the BP algorithm using a conjugate gradient descent. A sigmoid function was used as the activation function. The goals of this research were to study the ways in which differences in learning rates affected the recognition rate; the experiments, therefore, began with a sample structure and varied the value of the learning rate. The second method used various values of learning rates and hidden nodes, while the third method used a BP algorithm with a conjugate gradient descent. The experimental results indicated that the BP algorithm with the momentum factor provided the highest recognition rate, whereas the recognition rate for the other methods was 0.99. The work in [18] presented a dynamic BP algorithm for training with a boundary. In this case, the weight was updated under the constraints of this boundary; a sigmoid function was used as the activation function. The boundary helps to increase the rate of training of the BP algorithm and enhances the classification rate; in this study, the value of the classification correction was 91.1%. The researchers in [19] presented a specific penalty for obtaining the proportion of the norm of the weight or to prove the boundedness of the weights in the network training process. The initial weight is chosen in the range [-0.5, 0.5] and the learning rate was fixed using an equation

as an either a small constant (0.05) or an adaptive series. The penalty factor was set as 0.001. The results showed that the BBPAP obtained better convergence than prior work. In [20] improved the BP algorithm using a two-layer structure with a dynamic learning rate, which depends on both the current and previous values of the error training. The created learning rate depends on the value of the learning rate that is fixed at values η = 1.0, 0.4 and 1.4. The results show that the improved BP algorithm gave superior performance in terms of training than the standard BP algorithm.
The remainder of this paper is organized as follows: Section 2 describes the materials and method used; Section 3 presents the implementation; Section 4 presents the experimental results; Section 5 is a discussion; Section 6 explains about evaluation of the performance of improving algorithm; Section 7 give the conclusions of the study and Section 8 discusses future work.

2. Materials and Method

This kind of this research belongs the heuristic method. The heuristic method included two parameters such learning rate and momentum term. This study will be creating a dynamic function for each learning rate and momentum term to increase speed up back propagation algorithm. There are many steps which appear as follows.

2.1. Neural Network Model

We propose an ANN model, which consist of three-layer neural network that has an input, hidden and output layer. In this study, there are two different structure. The input layer is considered to be { X_1, X_2,..., X_i }, which represents the nodes; the nodes depend on the types or attributes of the data. The hidden layer is made of two layers with four nodes. Whereas, the L_h and LL_k are the first and second layer respectively. The output layer Y_r is made of one layer with one neuron. Three basis, two of them are used in the hidden and one in the output layer, which is denoted by u_{0j}, v_{0k} and w_{0r}. v_{hj} is the weight between neuron h from hidden layer L and neuron j from the hidden layer LL. u_{ih} is the weight between neuron i in the input layer and neuron h in the hidden layer. Finally, the sigmoid function is employed as an activation function.

2.2. Create Dynamic Learning Rate

There are two primary techniques for selecting the value for the learning rate. The first is to set it to a small constant value in the interval [0,1]; the second is to use a series value in the range [0,1] [21]. The learning rate should be sufficiently large to allow escape from the local minimum and to facilitate fast convergence to minimize error training [22]; however, the value of the learning rate which is too large leads to fast training with an oscillation in error training. To ensure a stable learning BP algorithm, the learning rate must be small [23]. The weight update between neuron *k* from the output layer and neuron *j* from the hidden layer is as follows:

$$\Delta w_{jk}(t+1) = w_{jk}(t) - \gamma \frac{\partial E}{\partial W_{jk}(t)} \quad (1)$$

where $\Delta w_{jk}(t)$ is a weight change. The weight is updated for each epoch in (1), and the speed of the training depends on a parameter that affects the updating of the weight. To enhance the BP algorithm as given in (1), to avoid local minima and training saturation, a dynamic function can be used to obtain an adaptive learning rate. Several studies have used an adaptive learning rate using a monotonicity function; the work in [24-25], for instance, used an exponential function to increase the speed of the BP algorithm. The exponential function used in these studies was a monotonic function, whereas the current work focuses on the boundary function and the creation of a dynamic learning rate using a penalty. We propose a dynamic learning rate as follows:

$$\gamma_{dmic} = k + \frac{1}{\sin(1-o_r)} \quad (2)$$

where K is the average of the activation function. The activation function used in this study is a sigmoid function. This formula uses (1-o_r) as an implicit function in γ to ensure that the expression $\sin(1-o_r)$ (the boundary function) is a positive value for every value of o_r. The dynamic learning rate has both an upper and a lower bound; updating of the weight in the BP algorithm is bounded. We substitute γ_{dmic} from (2) into (1) to obtain:

$$\Delta w_{jk}(t+1) = w_{jk}(t) - [k + \frac{1}{\sin(1-o_r)}] \frac{\partial E}{\partial W_{jk}(t)} \quad (3)$$

The weight update is automatic for every layer under effect the dynamic learning rate (γ_{dmic}).

2.3. Dynamic Batch Back Propagation (DBBPL) Algorithm

In this part, we will focus on implements of DBBPL algorithm and validate of the formwork of mathematics of the DBBPL algorithm. There are three stages of training in the DBBPL algorithm as follows:

2.3.1. Forward Phase

In the forward phase just calculate the weight in the each hidden layer $L_1,...,L_i$ as follows:

$$L_{-inh} = u_{0h} + \sum_{i=1}^{n} x_i u_{ih} \quad (4)$$

Compute the output of first hidden layer of L_h

$$L_h = f(L_{-inh}) \quad (5)$$

Send the result to LL_j (LL_j $j = 1,2,...,p$) and calculate the input signal

$$LL_{-inh} = v_{0h} + \sum_{i=1}^{n} L_h v_{hj} \quad (6)$$

Calculate the output layer of second hidden LL

$$LL_j = f(LL_{-inj}) \tag{7}$$

Send the output LL to input Y_{-inr} and then get the input layer of Y_{-inr}

$$Y_{-inr} = w_{0r} + \sum_{i=1}^{n} LL_i w_{jr} \tag{8}$$

Compute the output layer Y_r

$$Y_r = f(Y_{-inr}) \tag{9}$$

2.3.2. Backward-Pass Phase

This stage starts, when the weight reaches to the output of the last hidden as follows:
Compute the error training

$$e_r = \sum_{r=1}^{n} (t_{jk} - Y_r) \tag{10}$$

Calculates the local gradient of output layer Y_r at neuron r

$$\delta_r = e_r f(Y_{-inr}) \quad f(Y_{-inr}) = Y_{-inr}(1 - Y_{-inr}) \tag{11}$$

Compute the weight correction term (update w_{jr} latter)

$$\Delta w_{jr} = -[k + \frac{1}{\sin(1-Y)_r} \delta_r YY_r \tag{12}$$

Compute, bias correction term (update w_{0r} later)

$$\Delta w_{0r} = -[k + \frac{1}{\sin(1-Y)_r} \delta_r \tag{13}$$

Send δ_r to hidden layer each hidden unite ($j = 1, ..., p$). Compute the single error or error back propagation for layer above

$$\delta_{-inj} = \sum_{r=1}^{m} \delta_r \mathrm{w}_{\mathrm{jr}} \tag{14}$$

Calculates the local gradient of hidden layer LL_j at neuron

$$\delta_j = \delta_{-inj} f(LL_{-inj}) \tag{15}$$

Compute weight correction term (update v_{hj} later)

$$\Delta v_{hj} = -[k + \frac{1}{\sin(1-Y_r)}] \delta_j L_h \tag{16}$$

Calculate, bias collection term (update v_{0j} latter)

$$\Delta v_{0j} = -[k + \frac{1}{\sin(1-Y_r)}] \delta_j \tag{17}$$

Send δ_j to first hidden L_h $h = 1, ..., q$. Compute the single error as below

$$\delta_{-\mathrm{inh}} = \sum_{j=1}^{b} \delta_j v_{hl} \tag{18}$$

Compute the local gradient of hidden layer L_h compute as follows

$$\delta_h = \delta_{-inh} f(L_{-inh}), \quad f(L_{-inh}) = L_{-inh}(1 - L_{-inh}) \tag{19}$$

Compute weight correction (update u_{ih} later)

$$\Delta u_{ih} = -[k + \frac{1}{\sin(1-Y_r)}] \delta_h x_i \tag{20}$$

Calculates bias weight corrective (update u_{0h} later)

$$\Delta u_{0h} = -[k + \frac{1}{\sin(1-Y_r)}] \delta_h \tag{21}$$

2.3.3. Update the Weight Phase

At the weight update stage, all of the layers are adjusted simultaneously. The weight update is calculated as follows:

For each output layer $(\mathrm{j} = 0,1,2,\ldots\ \mathrm{p};\ \mathrm{r} = 1\ldots,\mathrm{m})$

$$w_{jr}(t+1) = w_{jr}(t) + [k + \frac{1}{\sin(1-Y_r)}] \delta_r LL_J \tag{22}$$

For bias

$$w_{0r}(t+1) = w_{0r}(t) + [k + \frac{1}{\sin(1-Y_r)}] \delta_r \tag{23}$$

For each hidden layer (LL_j h = 0,…,q; j = 1,…,p)

$$\mathrm{v_{hj}}(\mathrm{t+1}) = v_{hj}(t) + (\ k + \frac{1}{\sin(1-o_r)}) \delta_j z_h \tag{24}$$

For bias

$$\mathrm{v_{0j}}(\mathrm{t+1}) = v_{0j}(t) + (\ k + \frac{1}{\sin(1-o_r)}) \delta_j \tag{25}$$

For each hidden layer, $L_h (h = 1, \ldots, n; h = 1, ..., q)$

$$u_{ih}(t+1) = u_{ih}(t) + (k + \frac{1}{\sin(1-o_r)})\delta_h x_i \quad (26)$$

For the biases,

$$u_{0h}(t+1) = u_{0h}(t) + (k + \frac{1}{\sin(1-o_r)})\delta_h \quad (27)$$

3. Implementation of the Dynamic Algorithms

The dataset is very significant for the confirmation to improve the BP algorithm. All datasets in this study are taken from the UCI Machine Learning Repository. The data sets are available online at https://archive.ics.uci.edu/ml/index.html. These real data sets are divided into two parts, a training set and a testing set. The BP algorithm was implemented using a fixed value for the learning rate from the range [0,1], while the DBBPL algorithm is trained using a dynamic function for the learning rate.

4. Results and Discussion

This section describes the experiments carried out to validate our proposed algorithm. The range of the error limit affects training time, as shown in [26]. We calculate the accuracy of training as follows [27]:

$$\text{Accuracy (\%)} = \frac{1-\text{absolut}(T_1-O_1)}{UP-LW} * 100$$

where *UP* and *LW* are the upper bound and lower bound of the activation function. A sigmoid function was used and thus *UP* = 1 and *LW* = 0.

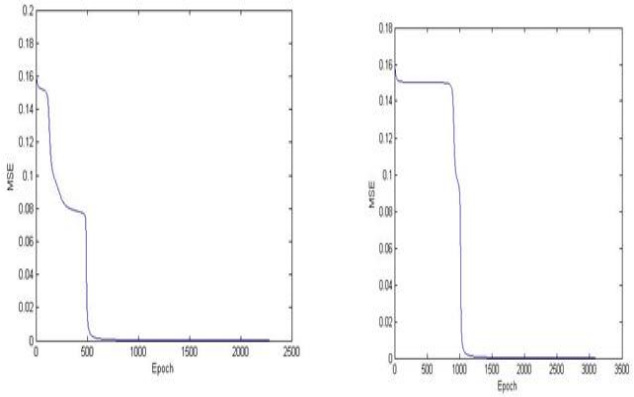
4.1. Experimental Results for the DBBPL Algorithm with the XOR Problem

Ten experiments (EX) were carried out using MATLAB, and the average (AV) was taken of several criteria used in this study for measurement of the training performance. The experimental results are tabulated in Table 1.

Table 1: Average the performance of DBBPL algorithm with XOR with different structure

	First Structure			Second Structure		
Item	Time (Sec)	Epoch	Accuracy	Time (Sec)	Epoch	Accuracy Training
A.V	4.7147	2279	0.9821	7.714	3086	0.9807
S.D	0.2948	0	0	1.174	2.4	0

From Table 1, it can be seen that the use of the formula in (2) enhances the performance of the backpropagation algorithm in terms of the training for both structures. It also reduces the time required for training; for the first structure, the average time for training is t = 4.7147 s and the epoch is 2279, while for the second structure the average training time is t = 7.7141 s and the epoch is 3086. There is no significant different between the two structures in terms of accuracy training. The small value of standard deviation (SD) indicates that the scatter in time and error training is very close for each experiment; however, that of the first structure which uses a single hidden layer, is better than that of the second structure for each time training and error time, whereas SD = 0 for both time and error training. The training curve is shown in Figure 1.



(a) (b)

Fig1: Training Curve of Dynamic algorithm with XOR

From Fig. 1(a), for the first structure, it can be seen that the curve training is a daisy with index epoch to meet the global minimum. The dynamic learning rate helps the BP algorithm to avoid training saturation. For the second structure as shown in Fig. 1(b), the weight training does not change before 500 epochs. This means that the DBBPL algorithm undergoes training saturation, although following this, the training curve converges quickly to give the minimum error.

4.2. Experimental Results for the BBP Algorithm with XOR Problem

We implemented the BBP algorithm, manual values of learning rate. The experiment results are tabulated in Table 2.

Table 2: Average the performance of the training of BBP algorithm with XOR Problem

	First Structure		Second Structure	
Values of γ	Time (Sec)	Epoch	Time (Sec)	Epoch
AV	225.7362	1707567	288.8325	2091897
S.D	127.12952	2541604	260.26521	2258757

From Table 2, for first structure, the average of time is $225.7362 \approx 226$ s with average epoch is 1707567, while the second structure the average of time training is $288.8325 \approx 2889$ s with 2091897 epoch.

4.3. Experimental Results for the DBBPL Algorithm with the Balance Training Dataset

We test the performance of our proposed method, given in (2), using the Balance data. This is one of the most widely known databases in pattern recognition and has 625 patterns. The experimental results are presented in Table 3.

Table 3: Average the performance of DBBPL algorithm with balance - Training Set

	First Structure			Second Structure		
Item	Time (Sec)	Epoch	Accuracy	Time (Sec)	Epoch	Accuracy Training
AV	8.2777	50	0.99996	0.9416	1	0.99995
S.D	0.2948	0	4.89E-5	0.0841	0	5E-05

From Table 3, it can be seen that for the first structure which uses a single hidden layer, the average training time is 8.2777 s with 50 epochs. The average of the training accuracy is 0.99996 and the accuracy is therefore very close to one. For the second structure which has two hidden layers, the average training time is 0.9416 with 1 epoch. The average of the training accuracy is 0.99995, meaning that the training accuracy is also very close to one for this structure. The high values for accuracy indicate that the dynamic

learning rate helps the back-propagation algorithm to avoid training saturation, to achieve a higher learning rate and to reach the global minimum.

4.4. Experimental Results for the BP Algorithm with the Balance Training Dataset

The performance was tested using 250 patterns as a form of training. The results of the simulation are given in Table 4.

Table 4: Average the performance of BP algorithm with Balance-Training set

	First Structure		Second Structure	
Values of γ	Time (Sec)	Epoch	Time (Sec)	Epoch
AV	477.2925	2804	240.7302	767
S.D	693.2013113	4029.163	265.0520825	889.7967

From Table 4, it can be seen that the average training time is for first structure is 477.2925 second $\approx$ 477 second with 2804 epoch. For second structure, the average time is 240.7302 $\approx$ 241 seconds with 769 epoch.

4.5. Experimental Results for the DBBPL Algorithm with the Balance Testing Dataset

The DBBPL algorithm was implemented using the Balance dataset for testing. The input layer represents the attributes of the data. The experimental results are recorded in Table 5.

Table 5: Average the performance of DBBPL algorithm with Balance-Testing set

	First Structure			Second Structure		
Item	Time (Sec)	Epoch	Accuracy	Time (Sec)	Epoch	Accuracy Training
AV	6.6576	66	0.99929	0.2723	1	0.99624
S.D	0.9397845	1.9E-6	0.000114	0.061616	0	0.000249

From Table 5, it can be seen that the dynamic approach for learning rate reduces the time required for training and enhances the convergence of the time training. For the first structure, the average training time was 6.6576 s with an average of 66 epochs. For the second structure, the average training time was 0.2723 s with an average of 1 epoch. Both structures gave high training accuracy and the average SD of time for both structures was less than one.

4.6. Experimental Results for the BBP Algorithm with the Balance Testing Dataset

In this section, we implement the backpropagation algorithm using 250 patterns, representing the testing dataset. The experimental results are given in Table 6.

Table 6: The performance of the training of BBP algorithm with Balance-Testing set

	First Structure		Second Structure	
Values of γ	Time (Sec)	Epoch	Time (Sec)	Epoch
AV	1229.051333	6991	1110.9285	14636
S.D	1676.5149	11334	1439.474401	23117

From Table 6, for first structure the average training time is 1229.051333 seconds $\approx$ 1229 with16991 epoch. For the second structure, the average training time 1110.9285 second $\approx$ 1111with 14636 epoch. The S.D for both structure is greater than one.

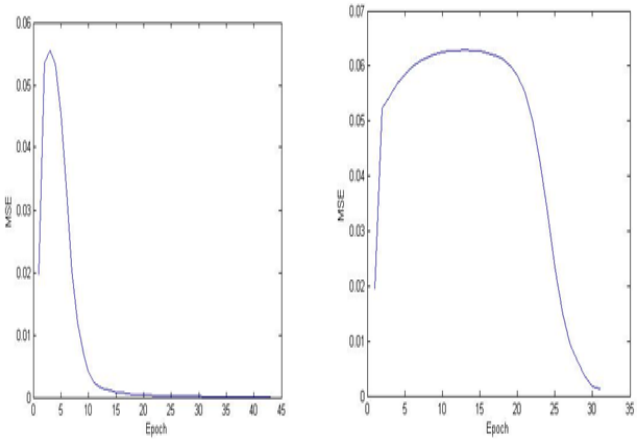
4.7. Experimental Results for the DBBPL Algorithm with the Iris Training Dataset

Ten experiments were carried out in MATLAB, and the average of several criteria was used for the measurement of the performance of training. The experimental results are given in Table 7.

Table 7: Average Training BP algorithm with Iris – Training

	First Structure			Second Structure		
Item	Time (Sec)	Epoch	Accuracy	Time (Sec)	Epoch	Accuracy Training
AV	2.165	43	0.9955	2.358	42	0.95542
S.D	0.1092	0	0	0.74775	12	0.01867

From Table 7, it can easily be seen that the dynamic learning rate has the effect of reducing the training time for the BP algorithm. For both structures, the average training time is very short: 2.165 s for the first structure, and 2.358 s for the second structure. In addition, the number of epochs for both structures is very low. This indicates that the new dynamic learning rate has the effect of removing training saturation for the BP algorithm and allowing the global minimum training to be reached. In addition, the average accuracy of the dynamic algorithm for both structures is 0.9955 and 0.95542 respectively. The training curve is shown in Figure 2.



(a) (b)

Fig. 2: Training Curve of Dynamic algorithm with Iris –Training Dataset

From Fig. 2, it can be seen that the dynamic learning rate allows the BP algorithm to escape the flat-spot problem appearing in both structures. For the first structure in Fig. 2(a), the dynamic learning rate allows the BP algorithm to avoid the flat spot after 10 epochs, while in Fig. 2(b), training does not change before 30 epochs for the second structure. For both structures, the training curve converges quickly to give the minimum error.

4.8. Experimental Results for the BBP Algorithm with the Iris Training Dataset

Table 8: Average Training BP algorithm with Iris – Training

	First Structure		Second Structure	
Values of γ	Time (Sec)	Epoch	Time (Sec)	Epoch
A.V	5108.759	254670	735.555	16062
S.D	10185.	10185.22	742.6004044	17109.799

From Table 8, it can be seen that for the first structure which uses one hidden layer, the average of the training time is 5108.759 seconds $\approx$ 5109 with 254670 epoch. For the second structure, the average training time 735.555 second $\approx$ 16062 second with 14636 epoch. The S.D for both structure is greater than one.

4.9. Experimental Results for the DBBPL Algorithm with the Iris Testing Dataset

The performance of training are given in Table 9.

Table 9: Average Training Improve algorithm with Iris – Testing

	First Structure			Second Structure		
Item	Time (Sec)	Epoch	Accuracy	Time (Sec)	Epoch	Accuracy Training
AV	2.1983	63	0.9960	3.9108	103	0.98726
S.D	0.2074536	1	0.00021931	0.625271	16.271	0.00074188

From Table 9, easily can see that the dynamic learning rate helps BP algorithm for reducing the time training. Both the structures the average of the training time is very short . The average time is 2.1983 seconds for the first structure while the average time for the second structure is 3.9108 for second structure. In addition, the average accuracy of a dynamic algorithm for both structures is 0.9960 and 0.98726 respectively. That indicates that the new dynamic learning helps the BP algorithm to remove the saturation training and reach the global minimum training.

4.10. Experiments the BBP Algorithm with Iris - Testing

Table 10: Average Training BBP algorithm with Iris – Testing

	First Structure		Second Structure	
Values of γ	Time (Sec)	Epoch	Time (Sec)	Epoch
AV	2087.4398	63339	1379.29975	10751
S.D	2038.59631	65565.41043	1639.369388	5747.54

From Table 10, for first structure the average training time is 2087.4398 seconds $\approx$ 2087 with 163339 epoch. For the second structure, the average training time 1379.29975 second $\approx$ 1379 with 14636 epoch. The S.D for both structure is greater than one.

5. Discussion

To validate the performance of the DBBPL algorithm through compression with BBP algorithm for speeding up training time and to determine which is superior. Several criteria are used to compare the DBBPL algorithm with the BBP algorithm. We calculate the Processing Time Improved of training using the following formula [28-29].

$$\text{Processing Time Improved (PTI)} = \frac{\text{Execution time of } BP \text{ algorithm}}{\text{Execution time of } BPDR \text{ algorithm}}$$

5.1. Performance Training of the DBBPR Algorithm versus the BBP Algorithm for the First Structure

To validate the improved BP algorithm, we compare the performance between the DBBPR algorithm and the BP algorithm. The speed-up obtained in training is shown in Table 11.

Table 11: Processing Time Improved the DBBPL algorithm versus BP with first structure

	Dynamic DBBPL Algorithm		BBP Algorithm		
	AV Time (Sec)	AV Epoch	AV Time (Sec)	AV Epoch	Processing Time Improved = (BBP/DBBPL)
XOR	4.7147	2279	225.7362	1707567	47.87
Balance Training Balance Testing	8.2777	50	477.2925	2804	57.66004
	6.6576	66	1229.5131	6991.239	184.6781
Iris Training Iris Testing	2.165	43	5109.759	254670	2360.166
	2.1983	63	1379.2997	2087	627.4393

From Table 11, it is evident that the dynamic algorithm outperforms the BBP algorithm. The period of the training time of the DBBPL algorithm is $2.165 \leq t \leq 8.2777$ s; which is considered a narrow interval that means that the DBBPL algorithm takes a short time with few epochs, to convergence into the global minimum. The period of the training time of BBP algorithm is in the range $225.7362 \leq t \leq 5109.759$ s; that can be considered a wide indicating that the BBP algorithm has a high level of training saturation and requires a long training time. At its maximum time training, the DBBPL algorithm is 2360.166 2360 times faster than the BBP algorithm.

5.2. Performance Training of the DBBPL algorithm versus the BBP Algorithm for the Second Structure

Table 12: Processing Time Improved the DBBPL algorithm versus BP with first structure

	Dynamic DBBPL Algorithm		BBP Algorithm		
	AV Time (Sec)	AV Epoch	AV Time (Sec)	AV Epoch	Processing Time Improved (BBP/DBBPL)
XOR	7.7141	3086	288.8325	2091897	37.44
Balance Training Balance Testing	0.9416	1	240.7302	769	255.6608
	0.2723	1	1110.9285	14636	4079.796
Iris Training Iris Testing	2.358	42	526.3825	16062.4	223.2326
	3.3580	103	526.3825	24713	156.7548

From Table 12, it is clear that the DBBPL algorithm outperforms the BBP algorithm. The period of the training time of the DBBPL algorithm is $0.2723 \leq t \leq 7.7141$s; which is considered a narrow interval that means that the DBBPL algorithm takes a short time with few epochs, to convergence into the global minimum. The period of the training time of BBP algorithm is in the range $240.8325 \leq t \leq 1110.9285$ s; that can be considered a wide interval, indicating that the BBP algorithm has a high level of training saturation and requires a long training time. At its maximum time training, the DBBPL algorithm is 4079.796 4080 times faster than the BBP algorithm.

6. Evaluation of the Performance of Improved Batch BP Algorithm

To evaluate the performance of the improved DBBPL algorithm in terms of the increased rate of training of the dynamic DBBPL algorithm presented in this study. The performance of the improved DBBPL algorithm is also compared to previous research works. The limited error or stop training is set at 0.0001 in this study; elsewhere in the literature such as in [16], this value is set at 0.0001 respectively. For the purpose of comparison between the results of this study and previous work, the fit was rerun with stop training values corresponding to those in previous work. The results of the current study show that the dynamic (DBBPL) algorithm outperforms the previous studies in terms of the training time, number of epochs and accuracy.

7. Conclusion

The batch backpropagation (BBP) algorithm is commonly used in many applications, including robotics, automation and global posi-

tioning systems. However, it requires a long training time to get global minimum training. To overcome this problem, a dynamic learning rate is proposed here. The learning rate is a widely used technique for improving the batch BP algorithm and an important parameter for controlling the weight training. This study introduces the DBBPL algorithm which training by learning rate. The dynamic learning rate affects the weight for each hidden layer and output layer and eliminates training saturation. One of the main advantages of the training used in the DBBPL algorithm is that it reduces training time, error training and the number of epochs. The experimental results show that the DBBPL algorithm gives superior performance compared with the BBP algorithm.

8. Future Work

This study was carried out using a dynamic learning rate algorithm for improving BBP algorithm. However, it is possible to create a dynamic training rate and momentum factor that have to integrate with each other to keep the weight adjusted as moderate.

References

[1] M. Sheikhan, M. Abbasnezhad Arabi and D. Gharavian "Structure and weights optimization of a modified Elman network emotion classifier using hybrid computational intelligence algorithms: A comparative study" Connection Science, 27(4), (2015), 340-357.

[2] H. Azami and J. Escudero, "A comparative study of breast cancer diagnosis based on neural network ensemble via improved training algorithms" Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, (2015), pp. 2836-2839.

[3] C. C. Cheung, S. C. Ng, A. Lui and S. S. D. Xu, "Further enhancements in WOM algorithm to solve the local minimum and flat-spot problem in feed-forward neural networks", Proceedings of the International Joint Conference on Neural Networks, (2014), pp. 1225-1230.

[4] M. S Al_Duais and F. S. Mohamad. A review on enhancements to speed up training of the batch back propagation algorithm. Indian Journal of Science and Technology, 9(46), (2016).

[5] [5] Y. Bassil, "Neural network model for path-planning of robotic rover systems", International Journal of Science and Technology, 2, (2012).

[6] H. H.Örkcü and H. Bal, "Comparing Performances of backpropagation and genetic algorithms in the data classification. Expert Systems With Applications" 38(4), (2011), 3703-3709.

[7] P. Moallem, "Improving Back-Propagation via an efficient Combination of A Saturation Suppression Method", Neural Network World, 20(2), (2010), 207.

[8] D. Xu, H. Shao and H. Zhang, "A new adaptive momentum algorithm for split-complex recurrent neural networks', Neurocomputing, 93, (2012), 133-136.

[9] B. Gong, "A novel learning algorithm of the back-propagation neural network," Proceedings of the International Conference in Control, Automation and Systems Engineering, 2009, pp. 411-414.

[10] S. Nandy, P. P. Sarkar and A. Das, "An Improved Gauss-Newtons Method based Back-propagation algorithm for fast convergence", International Journal of Computer Applications, 39(8), (2012), 1206-4329.

[11] J. M. Rizwan, P. N. Krishnan, R. Karthikeyan and S. R. Kumar," Multi layer perception type artificial neural network based traffic control", Indian Journal of Science and Technology, 9(5), (2016).

[12] Y. Shao, C. Zhao, Y. Bao and Y. He," Quantification of nitrogen status in rice by least squares support vector machines and reflectance spectroscopy", Food and Bioprocess Technology, 5(1), (2012), 100-107.

[13] L. Wang, Y. Zeng and T. Chen, "Back propagation neural network with adaptive differential evolution algorithm for time series forecasting", Expert Systems with Applications, 42(2), (2015) , 855-63.

[14] Q. Dai and N. Liu, "A two-phased and Ensemble scheme integrated Backpropagation algorithm", Neurocomputing, 9(4), (2014), 1124-1135.

[15] E. Noersasongko, F. T. Julfia, A. Syukur, R. A. Pramunendar and C. Supriyanto, "A tourism arrival forecasting using genetic algorithm based neural network", Indian Journal of Science and Technology, 9(4), (2016).

[16] Y. Liu., Z. Li, D. Yang, K. S. Mohamed, L. Wang and W. Wu, "Convergence of batch gradient learning algorithm with smoothing L 1/2 regularization for Sigma–Pi–Sigma neural networks", Neurocomputing, 151, (2015), 333-341.

[17] H. Shao, J. Wang, L .Liu, D. Xu and W. Bao," Relaxed conditions for convergence of batch BPAP for feed forward neural networks", Neurocomputing, 153, (2015), 174-79,153.

[18] C. Kaensar, "Analysis on the parameter of back propagation algorithm with three weight adjustment structure for hand written digit recognition", Proceedings of the 10th International Conference on Service Systems and Service Management, (2013), pp. 18-22.

[19] H. Zhang, W. Wu and M. Yao," Boundedness and Convergence of Bach BackPropagation Algorithm with Penalty for Feedforward Neural Networks", Neurocomputing, 89, (2012), 141-146 .

[20] S. J. Abdulkadir, S. M. Shamsuddin and R. Sallehuddin, "Three term back propagation network for moisture Prediction" Proceedings of the International Conference on Clean and Green Energy , (2012), pp. 103-707.

[21] Y. Huang," Advances in artificial neural networks–methodological development and application", Algorithms, 2(3), (2009), 973-1007.

[22] Y. Huang, "Advances in artificial neural networks–methodological development and application", Algorithms, 2(3), (2009), 973-1007.

[23] J. Li, L. Lian and Y. Jiang, "An Accelerating Method of Training Neural Networks Based on Vector Epsilon Algorithm", Proceedings of the 3rd International Conference on Information and Computing, 4, (2010), 292-295.

[24] M. Negnevitsky, "Multi-Layer Neural Networks with Improved Learning Algorithms", Proceedings of the Digital Imaging Computing: Techniques and Applications, (2005), pp. 34-34.

[25] H. Shao and G. Zheng, "A new BP algorithm with adaptive momentum for FNNs training", Global Congress on Intelligent Systems, 4, (2009), 16-20.

[26] C. Yang and R. Xu ," Adaptation Learning Rate Algorithm of Feed-Forward Neural Networks", Proceedings of the International Conference in Information Engineering and Computer Science, (2009), pp. 1-3.

[27] M. S. Al_Duais and F. S. Mohamad, (2017). Dynamically-adaptive Weight in Batch Back Propagation Algorithm via Dynamic Training Rate for Speedup and Accuracy Training. Journal of Telecommunications and Information Technology.

[28] N. M. Nawi, N. A. Hamid, R. S Ransing, R. Ghazali and M. N. M. Salleh, "Enhancing Back Propagation Neural Network Algorithm with Adaptive Gain on Classification Problems", Networks, 4(2), (2011).

[29] F. Saki, A. Tahmasbi, H. Soltanian-Zadeh and S. B. Shokouhi, "Fast opposite weight learning rules with application in breast cancer diagnosis", Computers in Biology and Medicine, 43(1), (2013), 32-34.