ISSN XXXX XXXX © 2017 IJESC

Research Article Volume 7 Issue No.9

Determining Optimum Structure for Artificial Neural Network and Comparison between Back-Propagation and Levenberg-Marquardt Training Algorithms

Mohammed H. IBRAHIM[1], Kemal H. JIHAD[2], Lubna L. Kamal[3]
Ph.D. Teaching Assistant[1], MSc.Teaching Assistant[2], MSc. Student[3]
Department of Computer Engineering[1, 3], Department of Computer Science[2]
Necmettin Erbakan University, Faculty of Engineering and Architecture, Konya, Turkey[1]
Kirkuk University, Faculty of Science, Kirkuk, Iraq[2]
Selcuk University, Faculty of Engineering, Konya, Turkey[3]

Abstract:
Artificial neural networks (ANNs) became a modeling tool to solve the complicated classification and prediction problems in different disciplines and applications. In this study, the thermal model of ten datasets from UCI data warehouse is modeled with feed-forward artificial neural networks and the most successful network topology is determined as well. This model was trained with Back-propagation (ANN-BP) and Levenberg-Marquardt (ANN-LM) training algorithms and the test results are compared. According to the obtained results showed better performance of ANNs trained with BP training algorithm.

Keywords: Artificial Neural Networks, Learning Algorithm, Back-propagation, Levenberg-Marquardt.

I. INRODUCTION

ANNs is avital part of artificial intelligence, machine learning and cognitive sciences depend on ANNs to solve various nonlinear mappings relationships [1]. ANNs is normally used BP training algorithm to fix different problems on account of their approximation features. With respecting to the weights BP training algorithm computes explicit gradients of the error such as the mean of square error.

However, ANNs trained with gradient descent based learning algorithm generally fall of slow convergence and local minima. A lot of ANNs applications are developed for making the daily life easier even a model what exactly reflects the work process of the brain could not be developed. The studies in basic medical sciences included just efforts of mathematical modeling of neurons in the human brain become highly disciplined during the recent years [2].

The ANNs came in possession of a research subject in several different sciences such as physics, mathematics, electric and computer engineering, besides the employees of various occupational groups introduced the ANNs to their area of specialization [3].

Thus, it is seen that some concepts what appear as belong to human intelligence could be stated as numerical and the machines can do the learning and remember operations by the ways are impressively similar to the human brain.

The training of ANNs is usually done with BP and LM training algorithms, and these training algorithms directly affect the success of the ANNs classification algorithm [4]. So choosing the right training algorithm for ANNs training is very important. The early diagnosis of the diseases has a great importance concerning extension the life and also enhancing the life quality. The learning techniques as an early diagnosis are frequently used in the literature [4, 5]. For example, the incident ratio of the thyroid disease in our country is fairly high. The early diagnosis of the thyroid disease is so important to enhance the life quality.

Some measurements for the person who has disease doubt will give significant information about the diagnosis of the thyroid. A dangerous aspect of thyroid disease is that the thyroid patients cannot understand whether they have this disease [6, 7].

In this study, 10 datasets which are different characteristic from UCI datasets are used, and also there is data set of thyroid diseases in these data sets. The performance of the BP and LM training algorithms is evaluated on these datasets.

The study is organized as follows, The ANNs models are explained in part 2, the training algorithms of ANNs model are explained in part 3, the transfer functions in ANNs is explained in part 4, the normalization process is explained in part 5 and the experimental results are mentioned in part 5.

II. THE ARTIFICIAL NEURAL NETWORKS MODELS

They are two types of feed forward and recurrent networks based on the flow direction of the sign in *ANNs*. The *Feed-Forward networks* are the simplest and most primitive structure of the *ANNs*. The knowledge in this network secretly moves just forward and to the exit layer. The system is memory less.

This *ANNs* topology is also called static network. The multilayer *feed-forward network* topology is seen in Figure 1. As is seen in Figure, the signals are always carried forward. The weights of *feed-forward* connections could be changed during the training, but the connection weights of recycles could not be modified.

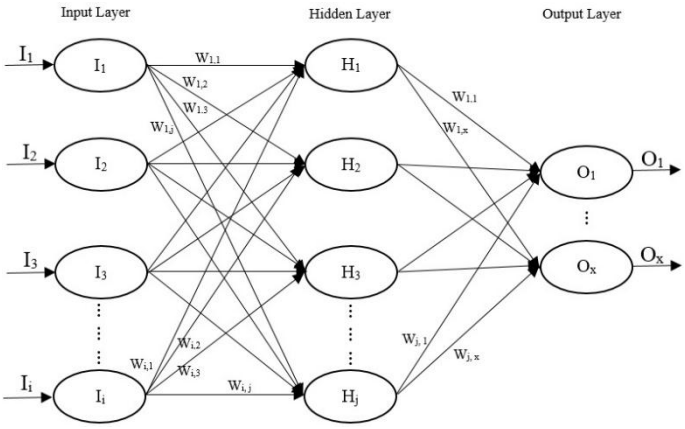

Figure.1. Network topology of feed-forward ANNs

III. THE TRAINING ALGORITHMS OF ARTIFICIAL NEURAL NETWORKS MODEL

The training in the *ANNs* is mostly seen as an optimization problem. Especially, the purpose of the training in *ANNs* is to minimize the error value by arranging the weights in a network. This error is calculated with various formulas, in this study *Mean Square Error* (*MSE*) formula is used. The *MSE* is given by formula 1 below.

$$E(w) = \frac{1}{n}\sum_{i=1}^{n}(d_i - y_i) \qquad (1)$$

Where d_i the targeted output value and y_i the value that is obtained from the output of the network. And each weight in the *ANNs* topology is updated according to this formula. The formula for updating weights is shown in formula 3 below.

$$w_{new} = w_{old} + \Delta w \qquad (2)$$

Δw is arranged to reduce the $E(w)$ as soon as possible. The calculation of Δw varies as the training algorithm. The most common used one in applications is the *BP* training algorithm. The errors in *BP* network expand backward via the connections in *feed-forward* mechanism by the derivative of *feed-forward* transfer transaction. There are disadvantages in standard *BP* training algorithm such as long computation time and the convergence risk at the local minimum. In this research, two of the high-performance *BP* training algorithm which enhances the performance by using different optimization techniques is analyzed. These algorithms are *LM* and *BP* training algorithms. These are shortly summarized in parts below.

LEVENBERG–MARQUARDT ALGORITHM

This algorithm is built on the idea of maximum method and generally not affected by the problems of slow convergence. This is the fastest learning method in *feed-forward* networks [1].

$$E(w) = \sum_{i=1}^{n}(d_i - y_i)^2 \qquad (3)$$

where d_i the targeted output value and y_i the value that is obtained from the output of the network. The target in *LM* training algorithm is to find the parameter vector (w) while the $E(w)$ is minimum. The new vector is calculated by using *LM* training algorithm as follows.

$$w_{k+1} = w_k + \Delta w_k \qquad (4)$$

The Δw_k is given in the equation below.

$$\Delta w_k = -\left(J_K^T g(w_k)\right)\left(J_K^T J_k + \lambda I\right)^{-1} \qquad (5)$$

where J_k is the Jacobian of $g(w_k)$ calculated by taking the derivative of $g(w_k)$ with respect to w_k, I is the Marquardt parameter. The *LM* training algorithm may be summarized as follows [2]:

Step 1: Calculate the $E(w)$

Step 2: Start with a small λ value (for example λ=0.01)

Step 3: Solve the Δw_k in equation and calculate the value of $E(w_k + \Delta w_k)$

Step 4: If the $E(w_k + \Delta w_k) \geq E(\Delta w_k)$, increase the λ by 10 times and go to step 3

Step 5: If the $E(w_k + \Delta w_k) < E(w_k)$ reduce λ 10 times, Update the w_k: $w_k \leftarrow w_k + \Delta w_k$ and go to step 3

Training multilayer neurons by using *LM* training algorithm to calculate the target output starts by assigning an initial value to the weight series. Then it continues by calculating the total of the square of the errors. Each of $(y_i - d_i)^2$ term means the square of the difference between the real output (y) and target output (d). The series of weight is adapted by being applied the *LM* training algorithm steps from (1) to (5) after obtaining all the $(y_i - d_i)^2$ error terms for a whole dataset.

BACK-PROPAGATION ALGORITHM

The *BP* training algorithm is the commonly used systematic method for multilayer neural networks nowadays. It has a high-power mathematical principle [3]. The *BP* training algorithm has caused to enlarge the problem range where the *ANNs* could be applied, and it enables for producing a large number of successful and powerful applications. Training the output layer in *BP* network is relatively straightforward. The problem is seen while training the interlayer weights out of any target vector. The solution for this problem could be found as providing the propagation of error signal more inner layers. If it needs to be talked about the mathematical base of *BP* training algorithm; it is an original weight reducing method in multilayer networks. The formal logic of the technique is to calculate the settable weight factor (w) for the input given. During the algorithm, each input data is passed from the hidden layer to produce a result. The expected result and the obtained one are compared to find the error at output level. Then, the derivative of the output error is passed from the output layer backward. The neurons set their weights to reduce their errors after being found the error value. The *BP* training algorithm uses the mean error square as an error measurement; this is not a necessity. Any differentiable error function could be utilized. But the selected error function must provide the measure if the difference between selected and real value. The *BP* training algorithm is given in Figure 2.

Begin

For each output layer node calculate

$$\delta_x = O_x(1 - O_x)(O_x - T_x)$$

For each hidden layer node calculate

$$\delta_j = OH_j\left(1 - OH_j\right)\sum_{x=1}^{n}\delta_x W_{j,x}$$

Update the weights by

$$\Delta W = -\eta\delta^t O^{t-1} + \alpha$$

Calculate new weight by

$$W_{new} = \Delta W + W_{old}$$

End

Figure.2. the Back-Propagation algorithm

This algorithm calculates the ΔW between two nodes. This statement is shown as below.

$$\delta_j = OH_j\left(1 - OH_j\right)\sum_{x=1}^{n}\delta_x W_{j,x} \qquad (6)$$

$$\delta_x = O_x(1 - O_x)(O_x - T_x) \quad (7)$$

here, δ_j is the error of nodes in hidden layer, $OH_j = \sum_{i=j=1}^{m} w_{ji}\, x_i$, δ_x is the error of nodes in output layer and T_x is the target output, O_x is the system output.

$$\Delta W = -\eta \delta^t O^{t-1} + \alpha \quad (8)$$

In the equation, let's ΔW represents weight between layers i and j. δ^t represents the node error in layer i and O^{t-1} represents the previous output in layer j. η represents the learning coefficient and α accounts for the momentum coefficient. The learning coefficient η is a constant determines the step measure of the training. Choosing the η as significant causes the oscillation. The first characteristics of the *BP* training algorithm are able to change the weight of values in its response to the errors. The derivative of the network output errors is transferred backward from the output layer to interlayer's. The characteristics of propagating backward are the eponym for this algorithm.

IV.TRANSFER FUNCTION IN ANNS

The transfer function that is also called threshold or signal function evaluates the result of the bindery function. Namely, it is a function specifies the output of the transaction element by considering the net input. It is mostly preferred to be a differentiable function. Different transfer functions are shown in Table 1. The bindery and transfer functions are selected based on the structure of the problem [10].

Table.1. Transfer functions

Name of transfer function	Formula of transfer function
Liner function	$f(x) = x$
Sinusoidal function	$f(x) = \sin(x)$
Sigmoid function	$f(x) = \frac{1}{1 + e^{-x}}$
Hiperbolik Tanjant function	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

V. NORMALIZATION PROCESS

In ANNs, the training of the data which are presented to ANNs could be made more efficient by applying some specific preprocess steps for entry and output of the network. The network input transaction functions make the network usage better. The normalization process is applied to raw data and effects for preparing the suitable data set for the training. The training of the ANNs can be so slow if the normalization process is not applied to the raw data set. The different techniques could be used in normalization processes. There are several data normalization types in literature. These are Min Rule, Max Rule, Median, Min-Max and Z-Score [11]. The normalization of the input and outputs of the Multilayer Network affects the performance of the network. Because the normalization makes the distribution of the values in data set regular. The inputs of ANNs may include extreme big and small values. They can be entered in an input set by mistake. These values can direct the network incorrect by causing to arise the extremely big and small values [12]. The scaling of all inputs in a specific range helps for both reducing the information from different environments on the same scale and eliminating the effect of extremely big and small values. Some investigators develop the scaling methods particular to their problems. A different scaling method could be used for each problem. Multilayer Network designers specify an approach so as to normalize the data on hands. Bringing a standard is not an efficient way. The normalization technique used is explained below. A part of the data set is analyzed by changing the numbers of hidden layer and neurons in ANNs. The obtained data is compared with the real value. Besides these operations, the normalization process is applied, and the classification success values are also found.

THE MIN-MAX NORMALIZATION

The method of the *Min-Max* normalizes the data linearly. The minimum is the lowest value that data can receive while the maximum means the highest value of data. The 9 numbered equation is used to reduce data to the range of 0 and 1 [4].

$$x' = \frac{x_i - x_{min}}{x_{max} - x_{min}} \quad (9)$$

In this equation, $\boldsymbol{x'}$ is normalized data, $\boldsymbol{x_i}$ is value of the input data, $\boldsymbol{x_{min}}$ the smallest number within the input set and $\boldsymbol{x_{max}}$ the biggest number within the input set.

VI. EXPERIMENTAL RESULTS

In the following experiment, we used 10 datasets from UCI datasets to test the performance of the BP and LM training algorithms [14], the characteristics of datasets and the structure of ANNs showed in table 1. The number of nodes in the hidden layer for the ANNs structure is determined by the highest classification accuracy achieved between one and the number of inputs multiplied by two. And for evolving the weights of ANNs we used BP and LM training algorithms.

Table.2. The characteristics of datasets and the structure of ANNs

Datasets	Number of			*ANNs* Structure
	Attributes	Classes	Samples	
Lymphography	18	4	148	18-5-4
Iris	4	3	150	4-6-3
Wine	13	3	178	13-4-3
Glass	9	6	214	9-12-6
New Thyroid	5	3	215	5-3-3
Shuttle-landing	6	2	253	6-3-2
Ionosphere	33	2	351	33-5-2
Balance-Scale	4	3	625	4-4-3
Breast cancer	9	2	699	9-8-2
Diabetes	8	2	768	8-8-2

The codes of the *BP* and *LM* training algorithms are coded in *visual studio C# 2015*. The screenshot of the program is shown in Figure 3 below.

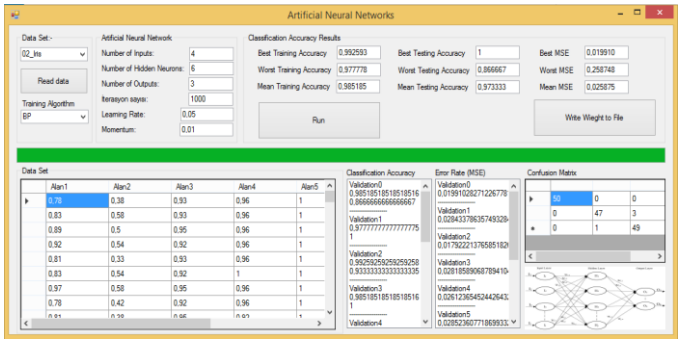

Figure. 3. Screenshot of the *ANNs* program

The parameters of *BP* and *LM* training algorithms used in the training of *ANNs* classification algorithm are given in Table 3 [2].

Table.3. Parameters of *BP* and *LM* training algorithms

Algorithm	Parameter	Value
LM	λ	0.01
BP	η	0.05
	α	0.01

The *MSE*, *training AC* and *testing AC* results in Table 4 are obtained using the parameters in table 2 and used *10-Fold cross-validation*.

Table .4. The results of *MSE*, *training AC* and *testing AC*.

Dataset	Algorithm	MSE	Training CA (%)	Testing CA (%)
Lymphography	ANN-BP	**0.027**	**98.80**	**76.94**
	ANN-LM	0.032	97.30	74.62
Iris	ANN-BP	**0.025**	**98.51**	**97.33**
	ANN-LM	0.051	96.83	95.74
Wine	ANN-BP	**0.000**	**99.71**	**97.83**
	ANN-LM	0.000	99.28	96.91
Glass	ANN-BP	**0.308**	**80.38**	**68.30**
	ANN-LM	0.601	78.29	64.82
shuttle-landing	ANN-BP	**0.011**	**99.60**	**96.84**
	ANN-LM	0.039	98.71	94.92
Ionosphere	ANN-BP	**0.023**	**98.92**	**88.62**
	ANN-LM	0.026	98.15	87.28
Balance-Scale	ANN-BP	**0.024**	**98.95**	**97.92**
	ANN-LM	0.038	96.37	93.91
Breast cancer	ANN-BP	**0.032**	**98.33**	**95.39**
	ANN-LM	0.048	97.86	94.17
Diabetes	ANN-BP	**0.290**	**78.65**	**77.53**
	ANN-LM	0.374	74.95	71.39
Thyroid	ANN-BP	**0.019**	**98.60**	**95.71**
	ANN-LM	0.035	97.14	92.38

When we analyze the results of training algorithms used for *ANNs* training, the *MSE* and *training AC* values are equal in Wine and Ionosphere datasets. In the other datasets the *BP* training algorithm gave better *MSE*, *training AC* and *testing AC* results than the *LM* training algorithm. In general the experimental results show that the classification accuracy of the *BP* training algorithm for all classification datasets is higher than the *LM* training algorithm. According to this result, it is more effective to use the *BP* training algorithm in the training of the *ANNs* classification algorithm.

VII. CONCLUSION

The training algorithm is so important for classification accuracy of *ANNs*. *BP* or *LM* training algorithms have been used to classify many problems in the *ANNs*. In this study, 10 classification benchmark data sets from *UCI* machine learning repository classified by using as *BP* and *LM* training algorithms. The obtained results from the both algorithms are compared in terms of *MSE* and classification accuracy. According to experimental results, it is observed that the *classification accuracy* increases and *MSE* decrease when *BP* is used in training of *ANNs*. As a result, the *BP* training algorithm performed a better performance in *ANNs* training.

VIII. REFERENCES

[1]. V. Kecman, Learning and soft computing: support vector machines, neural networks, and fuzzy logic models. MIT press, 2001.

[2].J. Han, J. Pei, and M. Kamber, Data mining: concepts and techniques. Elsevier, 2011.

[3].S. A. Kalogirou and M. Bojic, "Artificial neural networks for the prediction of the energy consumption of a passive solar building," Energy, vol. 25, no. 5, pp. 479-491, 2000.

[4].I. N. Daliakopoulos, P. Coulibaly, and I. K. Tsanis, "Groundwater level forecasting using artificial neural networks," Journal of Hydrology, vol. 309, no. 1, pp. 229-240, 2005.

[5].F. Amato, A. López, E. M. Peña-Méndez, P. Vaňhara, A. Hampl, and J. Havel, "Artificial neural networks in medical diagnosis," ed: Elsevier, 2013.

[6].Q. K. Al-Shayea, "Artificial neural networks in medical diagnosis," International Journal of Computer Science Issues, vol. 8, no. 2, pp. 150-154, 2011.

[7].F. Temurtas, "A comparative study on thyroid disease diagnosis using neural networks," Expert Systems with Applications, vol. 36, no. 1, pp. 944-949, 2009.

[8].G. Lera and M. Pinzolas, "Neighborhood based Levenberg-Marquardt algorithm for neural network training," IEEE Transactions on Neural Networks, vol. 13, no. 5, pp. 1200-1203, 2002.

[9].Ş. Ekinci, K. Çarman, and H. Kahramanlı, "Investigation and modeling of the tractive performance of radial tires using off-road vehicles," Energy, vol. 93, pp. 1953-1963, 2015.

[10].K. Debes, A. Koenig, and H.-M. Gross, "Transfer Functions in Artificial Neural Networks A Simulation-Based Tutorial," Brains, Minds and Media, vol. 2005, no. 1, 2005.

[11].T. Jayalakshmi and A. Santhakumaran, "Statistical Normalization and Back Propagation for Classification," International Journal of Computer Theory and Engineering, vol. 3, no. 1, p. 89, 2011.

[12].L. Al Shalabi, Z. Shaaban, and B. Kasasbeh, "Data mining: A preprocessing engine," Journal of Computer Science, vol. 2, no. 9, pp. 735-739, 2006.

[13].C. Saranya and G. Manikandan, "A study on normalization techniques for privacy preserving data mining," International Journal of Engineering and Technology (IJET), vol. 5, no. 3, pp. 2701-2704, 2013.

[14].A. Asuncion and D. Newman, "UCI machine learning repository," ed, 2007.