2nd International Conference on Electrical, Computer & Telecommunication Engineering (ICECTE)
8-10 December 2016, Rajshahi-6204, Bangladesh

Stock Market Prediction Using an Improved Training Algorithm of Neural Network

Mustain Billah
Department of Information
And Communication Technology
Mawlana Bhashani Science
And Technology University
Tangail, Bangladesh

Sajjad Waheed
Department of Information
And Communication Technology
Mawlana Bhashani Science
And Technology University
Tangail, Bangladesh

Abu Hanifa
Department of Information
And Communication Technology
Mawlana Bhashani Science
And Technology University
Tangail, Bangladesh

***Abstract*—Predicting closing stock price accurately is an challenging task. Computer aided systems have been proved to be helpful tool for stock prediction such as Artificial Neural Network(ANN), Adaptive Neuro Fuzzy Inference System (ANFIS) etc. Latest research works prove that Adaptive Neuro Fuzzy Inference System shows better results than Neural Network for stock prediction. In this paper, an improved Levenberg Marquardt(LM) training algorithm of artificial neural network has been proposed. Improved Levenberg Marquardt algorithm of neural network can predict the possible day-end closing stock price with less memory and time needed, provided previous historical stock market data of Dhaka Stock Exchange such as opening price, highest price, lowest price, total share traded. Morever, improved LM algorithm can predict day-end stock price with 53% less error than ANFIS and traditional LM algorithm. It also requires 30% less time, 54% less memory than traditional LM and 47% less time, 59% less memory than ANFIS.**

***Index Terms*—Stock prediction, Neural Netwok, Training algorithm, Levenberg Marquardt, Dhaka Stock Exchange.**

I. Introduction

Stock markets are one of the important part of the economy of a country. Actually it's the most important way for the companies to raise capital. Not only the investors but also the common peoples are also finding it as an investment tool. As stock market influences individual and national economy heavily, predicting the future values of stock market is essential task while taking the correct decision whether to buy or sell the share [3]. But it was very difficult to predict the stock price trends [14] efficiently because many factors such as economics, politics, environment etc were deciding parameters.

For many years, traditional statistical prediction methods [11] such as linear regression, time series analysis, chaos theory were popular. But for the uncertainity in stock market [15], these methods were failure or partially successfull. Soft computing techniques such as neural network [8], [5], [4], fuzzy systems [9] have been applied to solve this problem as they capture the non-linearity of stock market.

Adaptive Network-Based Fuzzy Inference System (ANFIS) has been used for stock prediction of Istambul Stock Exchange [2]. [1] also uses an ANFIS based model for stock price prediction. A three stage stock market prediction system is introduced in paper [13]. [5] presents an integrated system where wavelet transforms and recurrent neural network (RNN) based on artificial bee colony (abc) algorithm (called ABC-RNN) are combined for stock price forecasting. A review of used data mining techniques used in this purpose is analized in [10].

However, though there are many research works on stock prediction on different Stock Exchange trend, there are a very litte works on Bangladesh perspective. [12] worked on stock prediction of Dhaka Stock Exchange (DSE) using neural network and ANFIS where ANFIS showed better accuracy. By modifying training algorithm of neural network this performance can be improved. So the goal of this paper is to acquire better performance then existing works. In this paper, an improved of Levenberg Marquardt training algorithm of neural network has been proposed. This improved LM algorithm shows better result than ANFIS. It also requires less computing time and allocate less memory than traditional LM algorithm of neural network.

The following section are as follows: Section II gives a brief description of neural network. In section III improved LM algorithm is presented. Section IV contains the experimental results. A conclusion is also included in section V.

II. Neural Network

Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. We can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements.Figure shows a typical neuron structure: This neuron contains an input layer, a hidden layer and an output layer. Input x_j is multiplied by its weight w_{ij} giving a product $x_j w_{ij}$. The transfer function f is applied on this

978-1-5090-5785-6/16/$31.00 2016 IEEE

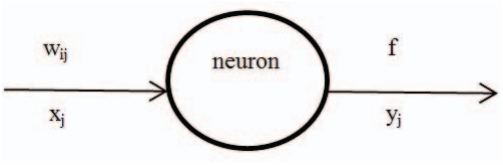

Fig. 1. Typical Neuron in a neural network system

product giving an output y_j which can be represented by :

$$y_j = f(x_j w_{ij}) \tag{1}$$

Here, i = index of neurons in the hidden layer.

j = index of an input to the neural network.

Typically, neural networks are adjusted, or trained based on a comparison of the output and the target, until the network output matches the target. After the artificial neural network has been trained wirh known values, then it can perform decision making.

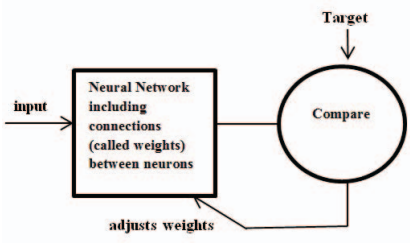

Fig. 2. ANN working principle

Many methods have already been developed for neural-networks training. The steepest descent algorithm, also known as the error backpropagation (EBP) algorithm is one of the most significant breakthroughs for training neural networks. But it is known as an inefficient algorithm because of its slow convergence.

The slow convergence of the steepest descent method can be greatly improved by the GaussNewton algorithm. The GaussNewton algorithm can find proper step sizes for each direction and converge very fast; especially, if the error function has a quadratic surface, it can converge directly in the first iteration. But this improvement only happens when the quadratic approximation of error function is reasonable. Otherwise, the GaussNewton algorithm would be mostly divergent.

The LevenbergMarquardt algorithm combines the steepest descent method and the GaussNewton algorithm. It inherits the speed advantage of the GaussNewton algorithm and the stability of the steepest descent method. Its more robust than the GaussNewton algorithm, because in many cases it can converge well even if the error surface is much more complex than the quadratic situation.

A. Levenberg Marquardt algorithm

Derived from steepest descent method and Newton algorithm, the update rule of LM algorithm is:

$$\Delta w = (J^T J + \mu I)^{-1} J^T e \tag{2}$$

where w is the weight vector, I is the identity matrix, is the combination coefficient, J is the Jacobian matrix and e is the error vector.

$$J = \begin{vmatrix} \frac{\partial e_{11}}{\partial w_1} & \frac{\partial e_{11}}{\partial w_2} & \cdots & \frac{\partial e_{11}}{\partial w_N} \\ \frac{\partial e_{12}}{\partial w_1} & \frac{\partial e_{12}}{\partial w_2} & \cdots & \frac{\partial e_{12}}{\partial w_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_{1M}}{\partial w_1} & \frac{\partial e_{1M}}{\partial w_2} & \cdots & \frac{\partial e_{1M}}{\partial w_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_{P1}}{\partial w_1} & \frac{\partial e_{P1}}{\partial w_2} & \cdots & \frac{\partial e_{P1}}{\partial w_N} \\ \frac{\partial e_{P2}}{\partial w_1} & \frac{\partial e_{P2}}{\partial w_2} & \cdots & \frac{\partial e_{P2}}{\partial w_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_{PM}}{\partial w_1} & \frac{\partial e_{PM}}{\partial w_2} & \cdots & \frac{\partial e_{PM}}{\partial w_N} \end{vmatrix} \tag{3}$$

where P is the number of training patterns, M is the number of outputs, and N is the number of weights. Elements in error vector are calculated by:

$$e_{pm} = d_{pm} - O_{pm} \tag{4}$$

where d_{pm} and O_{pm} are the desired output and actual output, respectively, at network output when training pattern.

The algorithm adjusts according to whether E is increasing or decreasing as follows:

1) Do an update as directed by the rule above.
2) Evaluate the error at the new weight vector.
3) If the error has increased as a result of the update, reset the weights to their previous values and increase μ. Then go to (1) and try an update again.
4) If the error has decresed as a result of the update, then accept the step (i.e. keep the weights at their new values) and decrease μ.

III. Improved Levenberg Marquardt

Though Levenberg-Marquardt method is considered efficient, computing large Jacobians needs a large memory. The large matrixes needed to be inverted for computation, results in bigger computation time. To reduce computation cost, changes are introduced in Levenberg-Marquardt method in many paper [6], [4], [7].

Performance index to be optimized in Levenberg-Marquardt algorithm is given as

$$F(w) = \sum_{p=1}^{p} \left[\sum_{m=1}^{m} e_{pm}^2 \right] \tag{5}$$

Instead of the performance index given by (5) [4] introduces a new performance index in Levenberg-Marquardt method as follows: :

$$F(w) = \sum_{p=1}^{p} \left[\sum_{m=1}^{m} e_{pm} \right]^2 \tag{6}$$

This leads to major reduction in matrix size, thereby reducing computation cost.

Again, a new scheme proposed by [7] proves that, for larger problem, decreasing μ by a factor of 5 and raising by a factor of 1.5 results into better result.

[6] proposes a new scheme for improving Levenberg-Marquardt algorithm. Update rule in this scheme is as follows:

$$\Delta w = (Q + \mu I)^{-1} g \tag{7}$$

where Q is the Quasi-Hessian matrix and g is the gradient vector. Q can be calculated as the sum of submatrices q_{pm}.

$$Q = \sum_{p=1}^{p} \sum_{m=1}^{m} q_{pm} \tag{8}$$

$$q_{pm} = j_{pm}^{T} j_{pm} \tag{9}$$

$$j_{pm} = \left[\begin{array}{cccc} \frac{\partial e_{pm}}{\partial w_1} & \frac{\partial e_{pm}}{\partial w_2} & \cdots & \frac{\partial e_{pm}}{\partial w_N} \end{array} \right] \tag{10}$$

gradient vector g can be calculated as the sum of gradient subvector η_{pm}.

$$g = \sum_{p=1}^{p} \sum_{m=1}^{m} \eta_{pm} \tag{11}$$

$$\eta_{pm} = j_{pm} e_{pm} \tag{12}$$

With the improved computation, both quasi-Hessian matrix and gradient vector can be computed directly, without Jacobian matrix storage and multiplication. During the process, only a temporary vector j_{pm} with N elements needs to be stored; in other words, the memory cost for Jacobian matrix storage is reduced by (P$\times$ N) times.

However, analyzing the above improvements proposed, a combined model of improved levenberg marquardt algorithm has been proposed:

Algorithm 1 Improved Levenberg Marquardt Algorithm

Step 1: Set Q = 0, g = 0 and Performance Index,

$$F(w) = \sum_{p=1}^{p} \left[\sum_{m=1}^{m} e_{pm}^{2} \right]^{2};$$

Step 2: According to performance Index F(w), compute j_{pm};
Step 3: Compute q_{pm};
Step 4: Compute η_{pm};
Step 5: Compute Δ w according to

$$\Delta w = (Q + \mu I)^{-1} g;$$

Step 6: Evaluate the error at the new weigh vector;
Step 7: If error <previous error

$$w = w + \Delta w$$

$$\mu = \mu \times 1.5$$

$$gotostep2;$$

else

$$\mu = \frac{\mu}{5}$$

$$gotostep4;$$

IV. Experiment and Result

A. Data Collection and preprocessing

Grameenphone data from Dhaka Stock Exchange (DSE) covering the period from January 2013 to april 2015. The data set contains daily opening price (BDT), closing price (BDT), highest price (BDT), lowest price (BDT) and total number of share traded (no.). After collecting, data were preprocessed to reduce garbage value and incorrect data were cleaned.

B. Experimental Structure of Neural Network

Neural Network tool of MATLAB 2010a was used for developing the ANN model. ANN have been trained with different number of hidden nodes number and for each node number different weights and bias were used at least ten times. However, the best model in training phase is not necessarily the best one in the testing phase. So averaging the performance of ten-time running is done to find out the error. Ten (10) neurons in the hidden layer performs better.

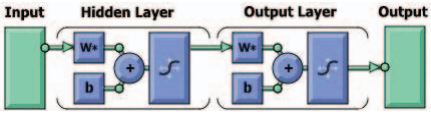

Fig. 3. Final neural network architechture

C. Experimental Structure of ANFIS

Different membership functions results in different outcomes of ANFIS. Variations in membership functions have been applied to justify the results. Finally, ANFIS showed better performance using 4 2 2 2 configuration of triangular MF for four input and constant MF for output variables.

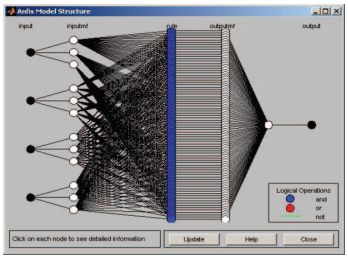

Fig. 4. Final ANFIS architechture

D. Result Analysis

Numerous statistical methods are used for measuring the accuracy and peformance of prediction models. In this study, root mesn square eror ($RMSE$) and the coefficient of multiple determinations (R^2) are used to compare predicted

and actual values.

The RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum_{n=1}^{n}(y_{\text{predicted}} - y_{\text{actual}})^2}{n}} \tag{13}$$

The R^2 is defined as:

$$R^2 = 1 - \frac{\sum_{n=1}^{n}(y_{\text{predicted}} - y_{\text{actual}})^2}{\sum_{n=1}^{n}(y_{\text{actual}} - y_{\text{mean}})} \tag{14}$$

Here, n is the total data sample, $y_{predicted}$ is the predicted value and y_{mean} is the average value of actual output, y_{actual} is the actual output. $RMSE$ values nearer to 0 predicts less error and R^2 values nearer to 1 indicates higher correlation.

Neural network (with traditional LM), Neural Network (with improved LM) and Adaptive Neuro Fuzzy Inference System (ANFIS) have been trained with the training data. Then their performance are measured against the test data. A comparative result of three are shown in the tables below.

TABLE I
ANN (IMPROVED LM ALGORITHM), ANFIS AND ANN (TRADITIONAL LM) TEST RESULT

	ANN (improved LM)	ANFIS Result	ANN (Traditional)
RMSE	0.87	1.86	1.881
R^2	0.99797	0.98116	0.98

Neural network trained with improved LM algorithm has the lowest $RMSE$ value which is 53% less than others methods.Again highest R^2 value indicates that, improved LM algorithm outperforms other methods.

Again, Table II shows that, computing time and memory needed for improved LM are also 30% and 54% less accordingly than traditional LM algorithm and 47% and 59% less accordingly than ANFIS.

TABLE II
MEMORY AND TIME COMPARISON OF PROPOSED LM AND TRADITIONAL LM

	Traditional LM	ANFIS	Improved LM
Time needed(s)	4.23	5.65	2.948
Memory allocated(kb)	5,674.00	6,453.00	2,592.00

So, improved LM algorithm is much more efficient than traditional LM algorithm and ANFIS.

V. CONCLUSION

In this paper, an improved Levenberg Marquardt(LM) algorithm of Artificial Neural Network(ANN) has been proposed. Then, this improved algorithm has been applied for stock market closing price prediction. Generally Adaptive Neuro Fuzzy Inference System(ANFIS) based model performs better prediction than ANN. But if ANN uses this improved algorithm for training, it shows 53% more accuracy in stock prediction than ANFIS. It also requires less memory allocation and computing time. This improved LM training algorithm proves neural network to be better computing tool for predicting closing stock price in Bangladesh Stock Exchange perspective. Moreover, it can be used for any prediction purposes. In future, we will use this concept in predicting network traffic prediction purpose.

REFERENCES

[1] I. Svalina, V. Galzina, R. Luji, and G. Imunovi, "An adaptive network- based fuzzy inference system (ANFIS) for the forecasting: The case of close price indices," *Expert systems with applications*, vol. 40, no. 15, pp. 60556063, 2013.

[2] M. A. Boyacioglu and D. Avci, "An adaptive network-based fuzzy inference system (ANFIS) for the prediction of stock market return: the case of the Istanbul stock exchange," *Expert Systems with Applications*, vol. 37, no. 12, pp. 79087912, 2010.

[3] E. F. Fama and K. R. French, "Common risk factors in the returns on stocks and bonds," *Journal of financial economics*, vol. 33, no. 1, pp. 356, 1993.

[4] B. M. Wilamowski, Y. Chen, and A. Malinowski,"Efficient algorithm for training neural networks with one hidden layer,*"in IJCNN. Proc. 1999 International Joint Conference*, 1999, vol. 3, pp. 17251728.

[5] T.-J. Hsieh, H.-F. Hsiao, and W.-C. Yeh, "Forecasting stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm," *Applied soft comput- ing*, vol. 11, no. 2, pp. 25102525, 2011.

[6] B. M. Wilamowski and H. Yu, "Improved computation for Leven- bergMarquardt training," *IEEE Transactions on Neural Network*, vol. 21, no. 6, pp. 930937, 2010.

[7] M. K. Transtrum and J. P. Sethna, "Improvements to the Levenberg- Marquardt algorithm for nonlinear least-squares minimization," arXiv preprint arXiv:1201.5885, 2012.

[8] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. De Jess,"Neural network design," vol. 20. PWS publishing company Boston, 1996.

[9] B. Kosko,"Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence/book and disk," Prentice Hall, Upper Saddle River, 1992.

[10] B. S. Arasu, M. Jeevananthan, N. Thamaraiselvan, and B. Janarthanan, "Performances of data mining techniques in forecasting stock indexevi- dence from India and US," *Journal of the National Science Foundation of Sri Lanka*, vol. 42, no. 2, 2014.

[11] M. H. Pesaran and A. Timmermann, "Predictability of stock returns: Robustness and economic significance," *The Journal of Finance*, vol. 50, no. 4, pp. 12011228, 1995.

[12] M. Billah, S. Waheed, and A. Hanifa,"Predicting Closing Stock Price using Artifical Neural Network and Adaptive Neuro Fuzzy Inference System (ANFIS: The Case of the Dhaka Stock Exchange,"*International Journal of Computer Applications*, vol. 129, no. 11, pp. 15, 2015.

[13] D. Enke, M. Grauer, and N. Mehdiyev, "Stock market prediction with multiple regression, fuzzy type-2 clustering and neural networks," *Procedia Computer Science*, vol. 6, pp. 201206, 2011.

[14] D. G. McMillan, "Stock return, dividend growth and consumption growth predictability across markets and time: Implications for stock price movement," *International Review of Financial Analysis*, vol. 35, pp. 90101, 2014.

[15] D. Avramov, "Stock return predictability and model uncertainty, "*Journal of Financial Economics*, vol. 64, no. 3, pp. 423458, 2002.