A COMPARATIVE STUDY OF BACKPROPAGATION ALGORITHMS IN FINANCIAL PREDICTION

Salim Lahmiri[1]

[1]Department of Computer Engineering, University of Quebec at Montreal, Montreal, Canada
lahmiri.salim@courrier.uqam.ca

ABSTRACT

Stock market price index prediction is a challenging task for investors and scholars. Artificial neural networks have been widely employed to predict financial stock market levels thanks to their ability to model nonlinear functions. The accuracy of backpropagation neural networks trained with different heuristic and numerical algorithms is measured for comparison purpose. It is found that numerical algorithm outperform heuristic techniques.

KEYWORDS

Neural Networks, Backpropagation, Stock Market, Forecasting

1. INTRODUCTION

Forecasting the stock market price movements has been a major challenge for both investors and scholars. Indeed, financial time series are highly volatile across time, and many factors affect their dynamics; mainly investor's expectations based on actual and future economic and political conditions. In the past, statistical methods such as the autoregressive integrated moving average (ARIMA) model [1] were widely employed to predict stock market time series. However, ARIMA processes are not adequate to predict such a noisy and nonlinear data. Therefore, soft computing techniques such as artificial neural networks (ANN) were largely adopted to predict the stock market movements [2]. The artificial neural networks are adaptive nonlinear systems capable to approximate any function. Theoretically, a neural network can approximate a continuous function to an arbitrary accuracy on any compact set [3]-[5]. The backpropagation (BP) algorithm that was introduced by Rumelhart [6] is the well-known method for training a multilayer feed-forward artificial neural networks. It adopts the gradient descent algorithm. In the basic BP algorithm the weights are adjusted in the steepest descent direction (negative of the gradient). However, the backpropagation neural network (BPNN) has a slow learning convergent velocity and may be trapped in local minima. In addition, the performance of the BPNN depends on the learning rate parameter and the complexity of the problem to be modelled. Indeed, the selection of the learning parameter affects the convergence of the BPNN and is usually determined by experience. Many faster algorithms were proposed to speed up the convergence of the BPNN. They fall into two main categories. The first category uses heuristic techniques developed from an analysis of the performance of the standard steepest descent algorithm. The second category uses standard numerical optimization techniques. The first category includes the gradient descent with adaptive learning rate, gradient descent with momentum, gradient descent with momentum and adaptive learning rate, and the resilient algorithm. In the standard steepest descent, the learning rate is fixed and its optimal value is always hard to find. The heuristic techniques allow the optimal learning rate to adaptively change during the training process as the algorithm moves across the performance surface. Therefore, the performance could be improved. The second category includes conjugate gradient, quasi-Newton, and Levenberg-Marquardt (L-M) algorithm. In the conjugate gradient algorithms, a search is performed along conjugate directions; therefore the convergence is faster than steepest descent directions. Quasi-Netwon method often converges faster than conjugate gradient methods since it does not require calculation of second derivatives. For instance, it updates an approximate Hessian matrix at each iteration. Finally, The L-M method combines the best features of the Gauss-Newton technique and the steepest-descent method. It also converges faster than conjugate gradient methods since the Hessian Matrix is not computed but only approximated. For instance, it uses the Jacobian that requires less computation than the Hessian matrix.

In science and engineering problems, there are many papers in the literature that examined the effectiveness of each category of algorithms on the performance of the BPNN. For instance, authors in [7] compared the performance of Levenberg-Marquardt, BP with momentum and BP with momentum and adaptive learning rate to classify the transformer oil dielectric and cooling state. They found that the BP with momentum and adaptive learning rate improves the accuracy of the BP with momentum and also gives a fast convergence to the network. The authors in [8] compared Levenberg-Marquardt, conjugate gradient and resilient algorithm for stream-flow forecasting and determination of lateral stress in cohesionless soils. They found that Levenberg-Marquardt algorithm was faster and achieved better performance than the other algorithms in training. The authors in [9] considered the problem of breast cancer diagnosis and compared the classification accuracy of the standard steepest descent against the classification accuracy of the gradient descent with momentum and adaptive learning, resilient BP, Quasi-Newton and Levenberg-Marquardt algorithm. The simulations show that the neural network using the Levenberg-Marquardt algorithm achieved the best classification performance. In their research, the authors in [10] employed three neural networks with different algorithms to the problem of intrusion detection in computer and network systems. The learning algorithms considered by the authors were the standard, the batch, and the resilient BP algorithm. They conclude that the resilient algorithm had a better performance to the application. Finally, authors in [11] compared the performance of the standard BP with and Levenberg-Marquardt algorithm to the prediction of a radio network planning tool. They found that the standard BP algorithm achieved the minimum error and then outperforms the Levenberg-Marquardt algorithm.

In the context of stock market forecasting, the BP is; indeed; the most employed algorithm to train artificial neural networks [2][12][13]. However, the problem of comparison of the accuracy of BP training algorithms in financial prediction was not considered. Therefore, the purpose of this paper is to examine the accuracy of BPNN trained with different heuristic and numerical techniques.

The rest of this paper is organized as follows. Section 2 presents basic concepts about BPNN and methodology. Experimental results are presented in Section 3. Finally, a conclusion is given in Section 4.

2. BACKPROPAGATION AND ALGORITHMS

The Multi-layer perceptron (MLP) networks trained using BP algorithm [6] are the most popular choice in neural network applications in finance [2]. The MLP consists of three types of layers. The first layer is the input layer and corresponds to the problem input variables with one node for each input variable. The second layer is the hidden layer used to capture non-linear relationships among variables. The third layer is the output layer used to provide predicted values. In this paper, the output layer has only one neuron corresponding to the prediction result. The relationship between the output y_t and the input x_t is given by:

$$y_t = w_0 + \sum_{j=1}^{q} w_j \cdot f\left(w_{0,j} + \sum_{i=1}^{p} w_{i,j} \cdot x_t \right)$$

where $w_{i,j}$ (i=0,1,2,…,p;j=1,2,…,q) and w_j (j=0,1,2,…,q) are the connection weights, p is the number of input nodes, q is the number of hidden nodes, and f is a nonlinear activation function that enables the system to learn nonlinear features. The most widely used activation function for the output layer are the sigmoid and hyperbolic functions. In this paper, the hyperbolic transfer function is employed and is given by:

$$f(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$$

The MLP is trained using the BP algorithm and the weights are optimized. The objective function to minimize is the sum of the squares of the difference between the desirable output ($y_{t,p}$) and the predicted output ($y_{t,d}$) given by:

$$E = 0.5\sum_{t=1} \left(y_{t,p} - y_{t,d}\right)^2 = 0.5\sum_{t=1} e^2$$

The training of the network is performed by BP [6] algorithm trained with the steepest descent algorithm given as follows:

$$\Delta w_k = -\alpha_k . g_k$$

where, Δw_k is a vector of weights changes, g_k is the current gradient, α_k is the learning rate that determines the length of the weight update. Thus, in the gradient descent learning rule, the update is done in the negative gradient direction. In order to avoid oscillations and to reduce the sensitivity of the network to fast changes of the error surface [14], the change in weight is made dependent of the past weight change by adding a momentum term:

$$\Delta w_k = -\alpha_k . g_k + p.\Delta w_{k-1}$$

where, p is the momentum parameter. Furthermore, the momentum allows escaping from small local minima on the error surface. Unfortunately, the gradient descent and gradient descent with momentum do not produce the fastest convergence, and even are often too slow to converge. Heuristic and numerical algorithms [15][16][17] are employed to train the standard BP with gradient descent and faster its convergence. Heuristic techniques employed in this study include the gradient descent with adaptive learning rate, gradient descent with momentum, gradient descent with momentum and adaptive learning rate, and the resilient algorithm. On the other hand, numerical techniques employed in this paper include quasi-Newton (Broyden-Fletcher-Goldfarb-Shanno, BFGS), conjugate gradient (Fletcher-Reeves update, Polak-Ribiére update, Powell-Beale restart), and Levenberg-Marquardt algorithm. The description of all these techniques is given in Table 1, and the reader may consult [15][16][17] for details. Figure 1 exhibits the prediction system. For instance, past index prices are fed to the BPNN to predict future price index. In other words, we make the hypothesis that lagged prices values of the stock market help forecasting its future level (price).

Table 1. List of algorithms

Algorithms	**Adaptation**	**Description**
Gradient descent (standard)	$\Delta w_k = -\alpha_k \cdot g_k$	The weights and biases are updated in the direction of the negative gradient of the performance function.
Gradient descent with momentum	$\Delta w_k = -\alpha_k \cdot g_k + p.\Delta w_{k-1}$	Momentum is added by a fraction change of the new weight.
Gradient descent with adaptive learning rate	$\Delta w_k = \alpha.\frac{\Delta E_k}{\Delta w_k}$	The initial network output and error are calculated. Using the current learning rate, new weights and biases are calculated at each epoch.
Gradient descent with momentum and adaptive learning rate	$\Delta w_k = p.\Delta w_{k-1} + \alpha.p.\frac{\Delta E_k}{\Delta w_k}$	Adaptive learning rate and momentum training are combined.
Resilient BP	$\Delta w_k = -sign\left(\frac{\Delta E_k}{\Delta w_k}\right).\Delta_k$	Only the sign of the partial derivative is considered to determine the direction of the weight update multiplied by the step size.
Fletcher-Reeves (conjugate)	$p_0 = -g_0$ $\Delta w_k = \alpha_k p_k$ $p_k = -g_k + \beta_k p_{k-1}$ $\beta_k = \frac{g'_k g_k}{g'_{k-1} g_{k-1}}$	Iteration starts by searching in the steepest descent direction. A search line method[1] is employed to find the optimal current search direction α. Next (update) search direction β is found such that it is conjugate to previous search directions.
Polak-Ribiere (conjugate)	$p_0 = -g_0$ $\Delta w_k = \alpha_k p_k$ $p_k = -g_k + \beta_k p_{k-1}$ $\beta_k = \frac{\Delta g'_{k-1} g_k}{g'_{k-1} g_{k-1}}$	Update is made by computing the product of the previous change in the gradient with the current gradient divided by the square of the previous gradient.
Powell-Beale restarts (conjugate)	$\lvert g'_{k-1} g_k \rvert \geq 0.2 \lVert g_k \rVert^2$	Update of Search direction is reset to the negative of the gradient only when this condition is satisfied.
BFGS quasi-Newton	$\Delta w_k = -H'_k g_k$	*H* is the Hessian (second derivatives) matrix.
Levenberg-Marquardt (LM)	$\Delta w_k = -H'_k g_k$ $H' = J'J$ $g = J'e$	*J* is the Jacobian matrix (first derivatives) and e is a vector of network errors.

[1] The search line method proposed in [20] is adopted in our study.

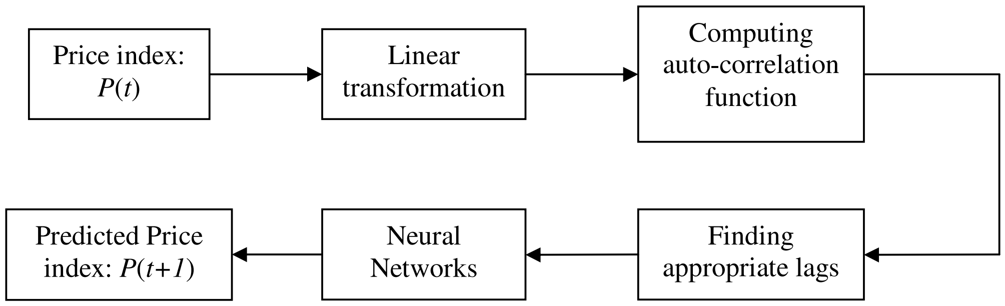

Figure 1. Prediction system

The auto-correlation function is employed to find the appropriate lags to be fed to BPNN following the methodology in [18][19] for prediction task. Finally, the performance of the prediction system is evaluated using the following common statistics: root mean of squared errors (RMSE), mean absolute error (MAE), and mean absolute deviation (MAD). They are given as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(P_t - F_t)^2}$$

$$MAE = \frac{1}{N}\sum_{t=1}^{N}\left|P_t - F_t\right|$$

$$MAD = \frac{1}{N}\sum_{t=1}^{N}\left|F_t - \overline{F}\right|$$

3. Experimental Results

The data consists on the S&P500 daily prices from October 2003 to January 2008. All neural networks are trained with 80% of the entire sample and tested with the remaining 20%. Based on the methodology presented in [18][19], we found that the S&P500 daily prices are auto-correlated up to two lags. For instance, actual price index $P(t)$ is auto-correlated with $P(t\text{-}1)$ and $P(t\text{-}2)$. Therefore, the following model is approximated using BP neural networks:

$$P(t) = f(P(t-1), P(t-2))$$

where *f*(.) is a nonlinear function to be approximated with BPNN trained with heuristic and numerical techniques. Figure 2 shows the root mean of squared errors and the mean absolute error statistic; and Figure 3 (left) shows the mean absolute deviation statistic. The simulation results show strong evidence of the superiority of BFGS quasi-Newton and L-M algorithm according to RMSE and MAE statistics. On the other hand, gradient descent with adaptive learning rate (GDALR), gradient descent with momentum and adaptive learning rate (GDMALR), Fletcher-Reeves, and Polak-Ribiére perform the worst. A surprising finding is that the standard gradient descent (GD) performs much better than all heuristic techniques used in the study and much better than Fletcher-Reeves, and Polak-Ribiére which are conjugate gradient approaches. According to the mean absolute deviation statistic, the L-M algorithm performs the best followed by the BFGS and resilient and gradient descent. In addition, Fletcher-Reeves, and Polak-Ribiére perform the worst. Finally, the Polak-Ribiére is the fastest algorithm followed by gradient descent with adaptive learning rate (GDALR), gradient descent with momentum and adaptive learning rate (GDMALR), and gradient descent (Figure 3, left).

Surprisingly, the resilient algorithm was the lowest to converge. In sum, the findings show that backpropagation neural networks (BPNN) trained with conjugate (BFGS) and the Levenberg-Marquardt (LM) provide the best accuracy according to RMSE, MAE, and MAD. Therefore, numerical techniques are suitable to train BPNN than heuristic methods when the problem of S&P500 stock market forecasting is considered.

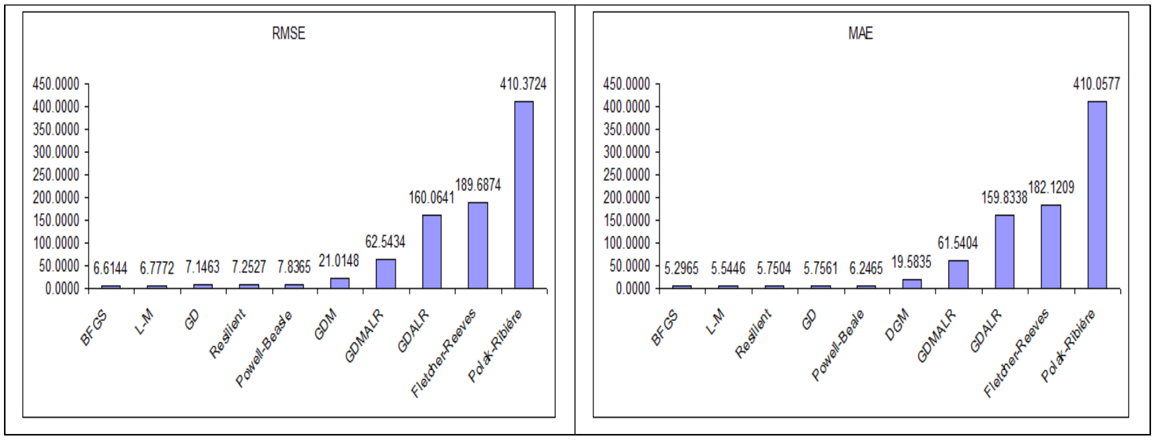

Figure 2. RMSE and MAE statistics

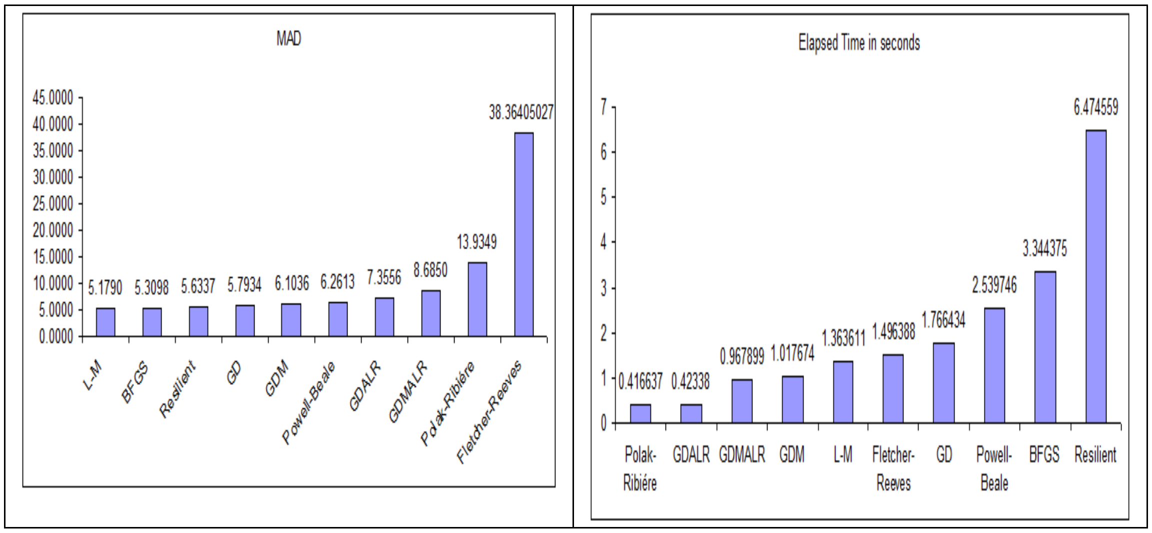

Figure 3. MAD statistic and elapsed time.

3. Conclusions

The problem of S&P500 price index prediction is considered. Backpropagation neural networks (BPNN) are trained with standard steepest descent, heuristic and numerical techniques; and accuracy measures are computed for comparison purpose. It is found that numerical techniques are suitable to train BPNN than heuristic methods when the problem of S&P500 stock market forecasting is considered. In particular, BFGS conjugate algorithm and Levenberg-Marquardt are the best in terms of accuracy. In addition, the standard steepest gradient descent was found to perform much better than some heuristic and numerical algorithms.

REFERENCES

[1] Box, G. E. P., Jenkins, G. M. (1994) Time series analysis: Forecasting and control, (3rded.). Englewood Cliffs: Prentice Hall.

[2] Atsalakis G.S., Valavanis K.P. (2008) ''Surveying Stock Market Forecasting Techniques –Part II: Soft Computing Methods'' Expert Systems with Applications, 36, 5932-5941.

[3] Funahashi, K.-I., (1989) "On the Approximate Realization of Continuous Mappings by Neural Networks" Neural Networks, Vol. 2, pp. 183-192.

[4] Hornik, K., (1991) "Approximation Capabilities of Multilayer Feedforward Networks" Neural Networks, Vol, 4, pp. 251-257.

[5] Cybenko, G., (1989) "Approximation by Superpositions of Sigmoidal Function," Math. Contr. Signals Syst., Vol. 2, pp. 303-314.

[6] Rumelhart D.E., Hinton G.E., Williams R.J. (1986) ''Learning Representations by Back–Propagating Errors'' Nature, 323, 533-536.

[7] Mokbnache L., Boubakeur A. (2002) ''Comparison of Different Back-Propagation Algorithms used in The Diagnosis of Transformer Oil'' IEEE Annual Report Conference on Electrical Insulation and Dielectric Phenomena, 244-247.

[8] Kişi Ö., Uncuoglu E. (2005) ''Comparaison of Three Back-Propagation Training Algorithms for Two Cases Studies'' Indian Journal of Engineering & Materials Sciences, 12, 434-442.

[9] Esugasini E., Mashor M.Y., Isa N., Othman N. (2005) ''Performance Comparison for MLP Networks Using Various Back Propagation Algorithms for Breast Cancer Diagnosis'' Lecture Notes in Computer Science, 2005, 3682/2005, 123-130, DOI: 10.1007/11552451_17

[10] Iftikhar A., Ansari M.A., Mohsin S. (2008) ''Performance Comparison between Backpropagation Algorithms Applied to Intrusion Detection in Computer Network Systems'' 9th WSEAS International Conference on Neural Networks (NN'08), 231-236.

[11] Nouir Z., Sayrac Z.B., Fourestié B., Tabbara W., Brouaye F. (2008) ''Comparison of Neural Network Learning Algorithms for Prediction Enhancement of a Planning Tool''. http://confs.comelec.telecom-paristech.fr/EW2007/papers/1569014736.pdf

[12] Kwon, Y.-K., & Moon, B.-R. (2007) ''A hybrid neurogenetic approach for stock forecasting'' IEEE Transactions on Neural Networks, 18(3), 851-864.

[13] Qi, M., & Zhang, G. P. (2008) ''Trend timevseries modeling and forecasting with neural networks'' IEEE Transactions on Neural Networks, 19(5), 808–816.

[14] Jang J.-S.R., Sun C-T., Mizutani E. (1997) Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, NJ: Prentice-Hall.

[15] Scales L.E. (1985) Introduction to Non-Linear Optimization, New York, Springer-Verlag.

[16] Jorge N., Wright S.J. (2006) Numerical Optimization, 2nd Edition. Springer.

[17] Simon Haykin , Bart Kosko (2001) Intelligent Signal Processing, Wiley-IEEE Press, first edition.

[18] Greene W.H. (2002) Econometric Analysis, Prentice Hall; 5th edition.

[19] Brockwell P.J., Davis R.A. (2002) Introduction to Time Series and Forecasting, Springer; 8th Printing.

[20] Charalambous C. (1992) Conjugate Gradient Algorithm for Efficient Training of Artificial Neural Networks, IEEE Proceedings, 139 (3), 301-310.

Author

Salim Lahmiri (M. Eng, Ph.D, Canada) is interested in signal and image processing.