International Journal of Computational Intelligence Systems, Vol. 8, No. 6 (2015) 1004-1016

Polynomial Based Functional Link Artificial Recurrent Neural Network adaptive System for predicting Indian Stocks

D. K. Bebarta
G.M.R. Institute of Technology,
Rajam, A.P., India

Birendra Biswal
G.M.R. Institute of Technology,
Rajam, A.P., India

P. K. Dash
Siksha 'O' Anusandhan University,
Bhubaneswar, Odisha, India

www.gmrit.org

Received 7 January 2015

Accepted 24 August 2015

Abstract

A low complexity Polynomial Functional link Artificial Recurrent Neural Network (PFLARNN) has been proposed for the prediction of financial time series data. Although different types of polynomial functions have been used for low complexity neural network architectures earlier for stock market prediction, a comparative study is needed to choose the optimal combinations of the nonlinear functions for a reasonably accurate forecast. Further a recurrent version of the Functional link neural network is used to model more accurately a chaotic time series like stock market indices with a lesser number of nonlinear basis functions. The proposed PFLARNN model when trained with the well known gradient descent algorithm produces reasonable accuracy with a choice of range of weight parameters of the network. However, to improve the accuracy of the forecast further, the weight parameters of the recurrent functional neural network are optimized using an evolutionary learning algorithm like the differential evolution (DE). A comparison with other well known neural architectures shows that the proposed low complexity neural model can provide significant prediction accuracy for one day advance and speed of convergence using the International Business Machines Corp. (IBM) stock market indices.

Keywords: PFLARNN, Polynomial functions, backpropagation learning algorithm, differential evolution, IBM stock indices, MAPE, AMAPE.

1. Introduction

Financial time series data are more complicated than other statistical data due to the long term trends, cyclical variations, seasonal variations and irregular movements. Predicting such highly fluctuating and irregular data is usually subject to large errors. So developing more realistic models for predicting financial time series data to extract meaningful statistics from it more effectively and accurately is of great interest in financial data mining research. A significant amount of studies has been done in this field that includes hybrid combinations of soft computing technology and data mining analysis applied to stock data prediction over a time frame varying from one day ahead to several days ahead. Traditionally, the linear statistical models like autoregressive moving average (ARMA) or ARIM [1-3] used for time series forecasting are simple but suffer

Copyright: the authors

from several shortcomings due to the nonlinearity of data. Hence researchers have developed more efficient and accurate soft computing methods like ANN [4, 5], Fuzzy set theory, Support Vector Machine, Rough Set theory, etc. for financial forecasting.

Various ANN based methods like Multi Layer Perception Network (MLP) [6], Radial Basis Function Neural Network (RBF) [7], Wavelet Neural Network (WNN)[8] , Recurrent Neural Network (RNN) [9] and Functional Link Artificial Neural Network (FLANN) [10-13] are extensively used for stock market prediction due to their inherent capabilities to identify complex nonlinear relationship present in the time series data based on historical data and to approximate any nonlinear function to a high degree of accuracy. A major benefit of neural networks is that it incorporates prior knowledge in ANN to improve the performance of stock market prediction. It also allows the adaptive adjustment to the model parameters depending on the nonlinear description of the problem. Chang & Fan [14] proposed a hybrid model by integrating a wavelet and TSK Fuzzy rules and an adaptive based ANFIS model [15, 16] for stock price forecasting. These ANN or hybrid system networks with several layers of neurons involve large computational complexity during training and testing of the model for forecasting non-linear time series data. In order to have better command on prediction, the intelligent functional link artificial neural network (FLANN) initially proposed by Pao, Y. H., & Takefji, Y. (1992) [17] has a nonlinear functional block to provide an expanded input space that introduces both nonlinearity and high dimensionality. This network has less computational complexity and provides better prediction performance in comparison to the widely used MLP network.

A wide variety of FLANNs [18-20] with functional expansion using orthogonal trigonometric, Chebyshev polynomial, Laguerre polynomial, Legendre orthogonal polynomial, and power series polynomial functions have been discussed in the literature. In this paper, the detailed architecture and mathematical modeling of various polynomial and trigonometric FLANNs along with the recurrent version are used for stock market prediction. It is already well known that the recurrent neural networks (RNNs) usually provide a smaller architecture than most of the nonrecursive neural networks like MLP [21], RBFNN, etc. Also their feedback properties make them dynamic and more efficient to model nonlinear systems accurately which are imperative for nonlinear prediction and time series forecasting. Many of the ARMA processes have been accurately modeled by RNNs for nonlinear dynamic system identification. The accuracy and robustness of forecasting the stock prices and volatilities can, however, be improved further by optimizing the weights of the PFALRNN by using some evolutionary techniques like GA, PSO, or the DE [22-24] and their adaptive versions. From the above evolutionary techniques, DE is chosen because it is simple but powerful global optimization method that converges faster than PSO. Further DE operates through the same computational steps as employed by any standard evolutionary algorithm (EA), but unlike traditional evolutionary techniques, DE employs the difference of the parameter vectors to explore the objective function landscape. Additionally, DE takes very few control parameters like the population size, the scale factor, and the crossover rate, which makes it easy to use. Apart from stock market prediction the proposed hybrid PFLARNN and DE model is quite efficient in predicting other time series databases like the highly fluctuating electricity price or currency exchange rates.

This paper is organized in six sections. In Section 2, the architecture of the Polynomial recurrent FLANN model is proposed and various polynomial basis functions are outlined. Section 3 is about briefing of differential evolution algorithm which is used to optimize the free parameters of our proposed architecture. In section 4, presents detailed description of sentiment of stock time series indices, technical indicator generation, complete features generation for our prediction model, and performance criteria's to assess the performance of the suggested architecture with comprehensive features. Section 5 summarizes the experimental results of the forecasting system obtained using different comprehensive features and finally in discussion the results are compared using suggested performance criteria. Finally, conclusion is given in Section 6 and followed by references are in section 7.

2. Development of a stock forecasting system using PFLARNN Model

Copyright: the authors

As shown in Fig.1 the PFLARNN comprises a nonlinear functional expansion block ($NFEB_1$, $NFEB_2$, , $NFEB_n$) for past stock indices, a few technical indicators, and an one step delayed output fed to the input space. The function of the NFEBs is to provide a nonlinear expansion of the inputs to increase the dimensionality of the input pattern, which ultimately results in the reduction of a number of layers in the network and improvement in the computational efficiency and speed of convergence. This architecture is chosen because it stores time-delayed information of the output that is suitable for the modeling of sequential data. Further the importance of the proposed network is that it enjoys adaptive on-line training to track the sudden changes in the financial time series data. It is already well known that the recurrent neural networks (RNNs) usually provide a smaller architecture than most of the non-recursive neural networks like MLP, RBFNN, etc. Also their feedback properties make them dynamic and more efficient to model nonlinear systems accurately which are imperative for nonlinear prediction and time series forecasting.

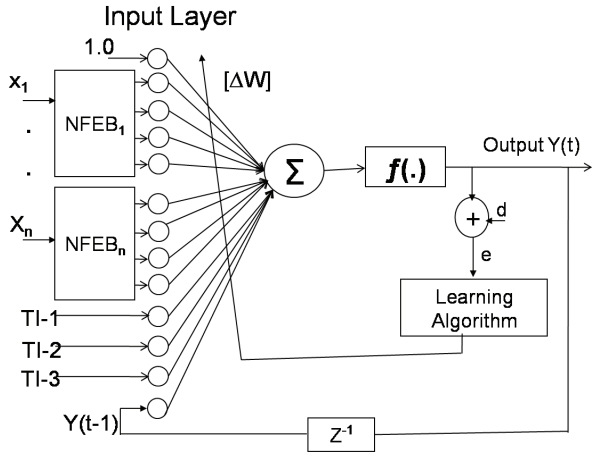


Fig.1: The architecture of PFLARNN model

The number of past stock indices is arranged in a vector form as

$$X(k) = [x_1(k), x_2(k), ..., x_i(k), ..., x_n(k)] \quad (1)$$

Each element of the above input vector is functionally expanded to increase the input dimensionality and is represented as

$$C = [c_1, c_2, ..., c_p] \ \ where \ 1 \le p \le M \ and \ M = 4n \quad (2)$$

In the input layer, the input variables comprise one input for the bias, the output vector C of the non-linear functional expansion block, three popular technical indicators, and one input for the recurrent link obtained by feeding back one step delayed output. Thus the final input vector is obtained as

$$I(k) = [1.0, \ [C], TI\text{-}1, TI\text{-}2, TI\text{-}3, y(k-1)] \quad (3)$$

The vector C is obtained by using the following polynomial functions in the functional block.

2.1. Polynomial Basis functions

2.1.1 Chebyshev Polynomial Function

The lower and higher order Chebyshev Polynomials are given in equations (4) and (5) as

$$\begin{aligned} c_1(x_i) &= x, & c_2(x_i) &= 2x^2 - 1 \\ c_3(x_i) &= 4x^3 - 3x, & c_4(x_i) &= 8x^4 - 8x^2 + 1 \end{aligned} \quad (4)$$

The higher order Chebyshev Polynomials can be generated using the generalized recursive formula given in equation (5).

$$c_{r+1} = 2 * x * c_r(x) - c_{r-1}(x_i) \quad (5)$$

The architecture of the PFLARNN using Chebyshev Polynomials is termed as C-PFLARNN.

2.1.2 Laguerre Polynomial Function

The lower and the higher order Lagurre Polynomials are represented as

$$\begin{aligned} c_1(x_i) &= -x + 1 \\ c_2(x_i) &= x^2/2 - x + 1 \end{aligned} \quad (6)$$

The higher order Laguerre Polynomials can be generated by the recursive formula

$$c_{r+1}(x) = 1/(r+1) * ((2r+1)c_r(x) - rc_{r-1}(x)) \quad (7)$$

2.1.3 Legendre Polynomial Function

This model uses the basis function Legendre Polynomials. Legendre Polynomials are a set of

Copyright: the authors

orthogonal polynomials obtained from the solution of the Legendre differential equation.
The lower order and the higher order Legendre Polynomials are given in equations (15) & (16).

$$c_1(x) = x, \qquad c_2(x) = \frac{(3x^2 - 1)}{2}$$
$$c_3(x) = \frac{(5x^3 - 3x)}{2}, \quad c_4(x) = \frac{(35x^4 - 30x^2 + 3)}{8} \tag{8}$$

The higher order Legendre Polynomials can be generated by the recursive formula.

$$c_{r+1}(x) = \frac{1}{(r+1)} * ((2r+1) * x * c_r(x) - r c_{r-1}(x)) \tag{9}$$

2.1.4 Power Series Polynomial Function

This model uses the basis function as Power series. Power series is implemented to enhance the input pattern given to PFLARNN model is given in equation (10).

$$c_p(x_i) = (x_i)^p \qquad where 0 \le p \le 6 \tag{10}$$

2.2. Conventional learning process of the model

The weights of the PFLARNN model are updated at the end of each experiment by computing the error between the desired output and the estimated output. The weight vector is represented as

$$W = [w_0, w_1, w_2, w_3, ...w_{5M+4}] \tag{11}$$

The error at the k[th] time step of the L[th] experiment is expressed as

$e(k) = d(k) - y(k)$, where $d(k)$ is the desired output and the cost function is

$$E(k) = \frac{1}{2} e^2(k) \tag{12}$$

Also the output of the model is obtained as

$$y(k) = f(S(k)) = f\{[I(k)]^T * W\} \tag{13}$$

Where $S(k) = [I(t)]^T * W$

And $$f(S) = \frac{1}{(1 + e^{(-0.5 * S(k))})} \tag{14}$$

Using gradient descent algorithm, the weight vector is updated recursively as

$$W(k+1) = W(k-1) + \alpha e(k)(1 - y(k))^2 I(k) \tag{15}$$

where α is the learning rate.

3. Differential Evolution Algorithm (DEA)

Differential Evolution (DE) [19] is a population-based stochastic function optimizer, which uses a rather greedy and less stochastic approach for problem solving in comparison to classical evolutionary algorithms, such as genetic algorithms, evolutionary programming, and PSO. DE combines simple arithmetical operators with the classical operators of recombination, mutation, and selection to evolve from a randomly generated starting population to a final solution. Differential Evolution algorithm uses four parameters like population (*Pop_Size*), crossover probability (*CR*), and scaling factors (*F1* and *F2*). The objective function for optimization is chosen in equation (16) as

$$Obj = \frac{1}{NN} \sum_{k=1}^{N} (d(k) - y(k))^2 \tag{16}$$

Where *NN* is the total number of data samples used for training.

4. Simulation study

4.1. The selection of stock time series data

The stock data for the experiment has been collected from International Business Machines Corp. (IBM). This data set covers more than 4000 trading days ranging from 09/06/1995 to 31/12/2012. The data set includes the opening price, highest price, lowest price, closing price, and total volume which are considered as basic information's to forecast the stock time series data. We have considered the attributes like the closing price and volume of the data set for the model to forecast. The data set shown in Fig.2 is partitioned into a data set for training the model and a test set for validating whether the predictions are valid outside the training samples. The 2000 trading days of data and the following 300

Copyright: the authors

trading days of data are used for training and testing, respectively.

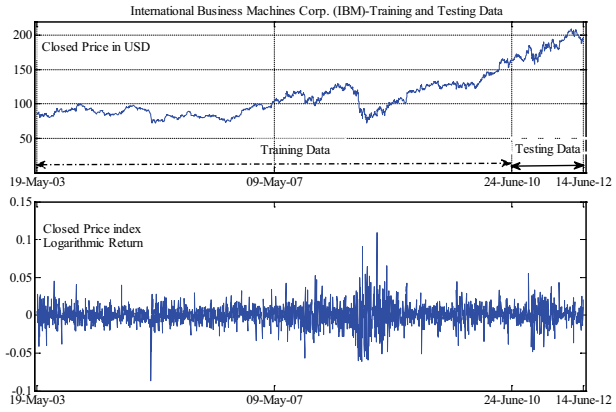


Fig.2: The IBM Data Set

4.2. The calculation of three technical indicators

To reduce the over learning problem of the proposed model and to reduce the complexity of the architecture, three important technical indicators namely the Relative Strength Indicator RSI (5), Price Volume Change Indicator PVC (5), and the Moving Average Volume Indicator MAVI (5, 20) are tried as inputs to the proposed PFLARNN model. The reason of choosing these three variables is that they are focused on price, volume, and combination of both price and volume. The following three equations (17-19) are used to compute as RSI, PVC, and MAVI to reduce the over learning of the proposed model.

TI-1: Relative Strength Indicator

$$RSI(5) = \sum_{i=1}^{n} Max(\Delta x_i, 0) \Big/ \sum_{i=1}^{n} |\Delta x_i| \tag{17}$$

TI-2: Price Volume Change Indicator

$$PVC(5) = \sum_{i=1}^{n} (\Delta x_i \times \Delta v_i) \Big/ \sum_{i=1}^{n} (|\Delta x_i \times \Delta v_i|) \tag{18}$$

TI-3: Moving Average Volume Indicator

$$MAVI(5,20) = MV_5 \Big/ MV_{20} \tag{19}$$

where x_i is the closing price on the ith day, $\Delta x_i = x_i - x_{i-1}$, $\Delta v_i = v_i - v_{i-1}$, MV5 is 5 day moving average of trading volume, and MV20 is 20 day moving average of trading volume. n is the number of samples used for calculating the indicators.

4.3. Training and testing of the proposed model

The data set is preprocessed before training and testing the model. The weights are initialized using evolutionary algorithm and further optimized by using back propagation learning algorithm.

The entire input data patterns including technical indicators are normalized to values between 0.0 and 1.0. The normalization formula in equation (20) is used to express the data in terms of the minimum and maximum value of the dataset.

$$u = (x_i - x_{\min}) / (x_{\max} - x_{\min}) \tag{20}$$

where, 'u' and 'x' represent normalized and actual value, respectively.

The stock time series data is found to be noisy, random, volatile, and is prone to high fluctuations for several economic and non-economic factors. At the model design stage, different variants are tried to generate high forecasting accuracy from the proposed model. To explore the impact of the model on the performance of the PFLARNN architecture, the following cases are considered:

1. Proposed model with weights varying in the range of -1.0 to +1.0 or -2.0 to +2.0 or -3.0 to +3.0 are considered randomly for training and testing.

2. For Differential Evolution Algorithm the parameters are initialized to the following values:

   No. of generations=50; Maximum number of iterations=100; cross over parameter *CR* = 0.8, scaling factors *F1* = 0.5 and *F2*=0.3; Weight ranges as output: -1.0 to 1.0, -2.0 to 2.0 and -3.0 to 3.0.

   The estimated weight ranges as output: -1.0 to 1.0, -2.0 to 2.0 and -3.0 to 3.0 using DE algorithm used in the proposed model for training and testing.

4.4. Performance Metrics

To validate whether the architecture has the generalization capability, we have considered the IBM stock data in two parts (shown in Fig. 2)

a. Training Period: 2000 daily observations from 20/05/2003 to 08/04/2011 are used as training data.
b. Testing Period: 300 daily observations from 08/04/2011 to 18/06/2012 are used for testing.

Copyright: the authors

The PFLARNN architecture is validated with some testing results based on the various measurements to study the accuracy of forecasting results. The Mean Absolute Percentage Error (MAPE) in equation (21) represents the absolute average prediction error between actual and forecast value. This gives the repulsive effect of very small prices; the Average Mean Absolute Percentage Error (AMAPE) in equation (22) is, therefore, adopted and compared. The impact of the model uncertainty needs to be measured on forecast results, the variance of forecast errors (σ_{Err}^2) in equation (23) is used for the purpose. The smaller the variance, the more accurate is the forecast result and more confidence is obtained on the model.

$$. \ MAPE = \left(\frac{1}{N} \sum_{j=1}^{N} \frac{abs(e(j))}{y(j)} \right) \times 100\% \tag{21}$$

$$AMAPE = \left(\frac{1}{N} \sum_{j=1}^{N} \left[\frac{abs(e_j)}{\bar{y}} \right] \right) \times 100\%$$

$$where \quad \bar{y} = \frac{1}{N} \sum_{j=1}^{N} [y(j)] \tag{22}$$

$$\sigma_{Err}^2 = \left(\frac{1}{N} \sum_{j=1}^{N} \left[\frac{abs(ee(j))}{\bar{y}} \right] - AMAPE \right)^2 \tag{23}$$

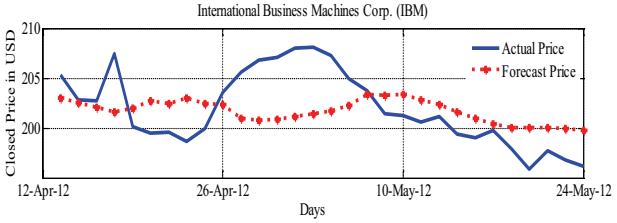
where $ee(k) = y(k) - \bar{y}$, and $y(k)$ represents the actual value; $\bar{y}$ is the average value of actual price; σ_{Err}^2 variance of forecast error, and N is the number of forecasted data samples.

5. Application and Experimental Results

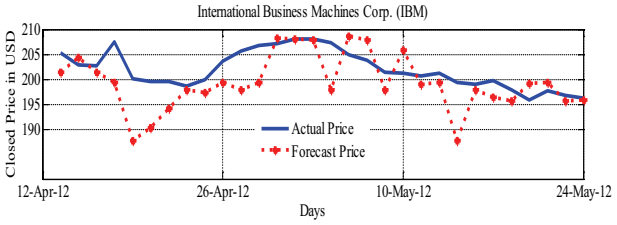
In this section, we have tested our model as per the case studies planned in section 4.3. The reason of selecting these case studies is to show that the proposed model can provide vigorous performance under different types of stock affinity.

5.1. Test Results of PFLARNN without DE optimization

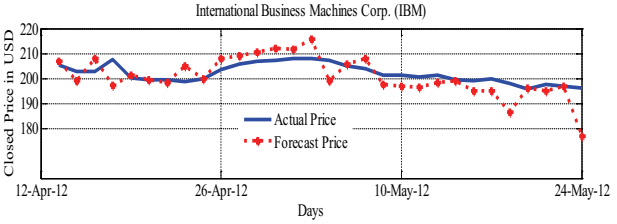
Figs.3 to 6 shows the forecasting results of the PFLARNN architecture using Lagurre, Chebyshev, Power series, and Legendre Polynomial Functions, respectively, with various network weights ranging from -1.0 to +1.0, -2.0 to +2.0, and -3.0 to +3.0. These figures also illustrate the testing results of 30 days taken between 12/04/2012 to 24/05/2012 on real data. These 30 days are considered ahead of 250 days from the training period. From the figures it is observed that the performance for day ahead forecasting of the IBM stock is the best with weight range from -3 to +3 in comparison to other weight ranges. This is further corroborated from Table-1 which shows the least MAPE for this weight range.



(a): Forecasting performance during the testing period with network weights -1.0 to +1.0.



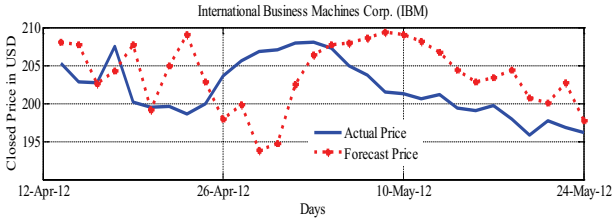
(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.



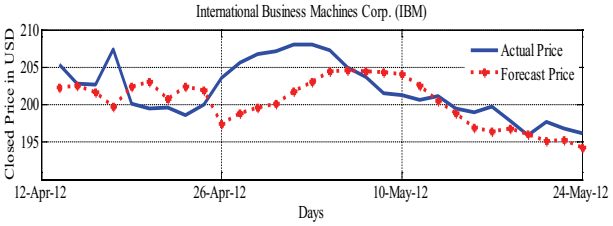
(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

Fig.3: Effect of varying network weights on the forecasting performance using IBM dataset using Lagurre Polynomial Function.

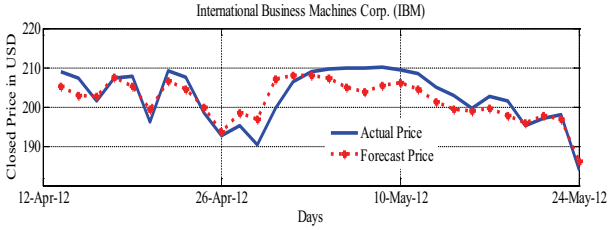
Copyright: the authors



(a): Forecasting performance during the testing period with network weights -1.0 to +1.0.

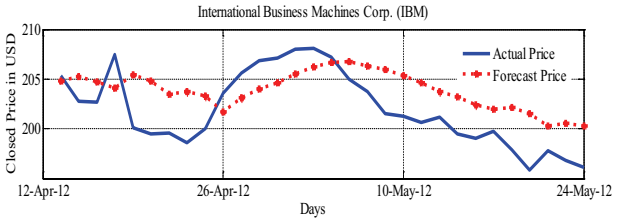


(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.

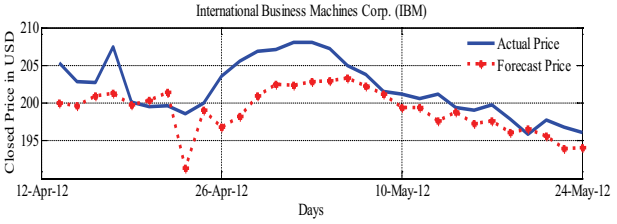


(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

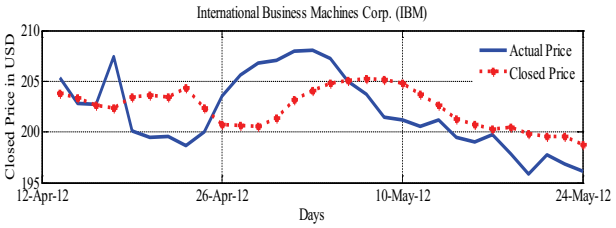
Fig.4: Effect of varying network weights on the forecasting performance using IBM dataset using Chebyshev Polynomial Function.



(a): Forecasting performance during the testing period with network weights -1.0 to +1.0.

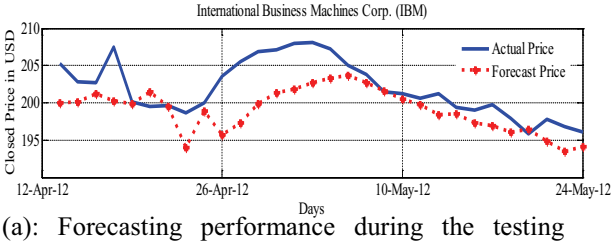


(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.

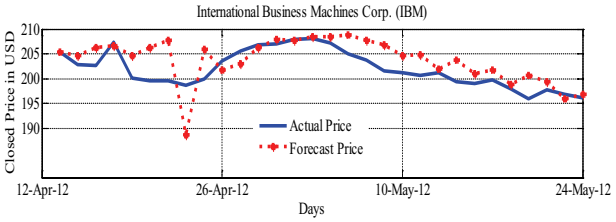


(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

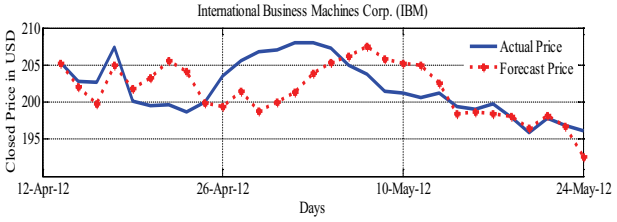
Fig.5: Effect of varying network weights on the forecasting performance using IBM dataset using Power series Polynomial Function.



(a): Forecasting performance during the testing period with network weights -1.0 to +1.0



(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.



(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

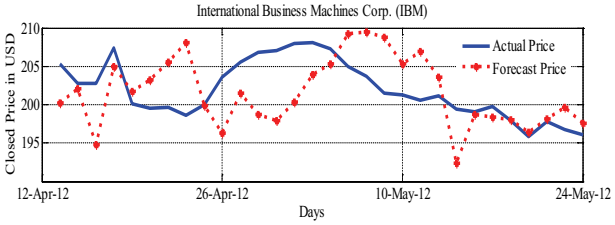
Fig.6: Effect of varying network weights on the forecasting performance using Legendre Polynomial function

The next section describes the use of differential evolution (DE) in the adaptation of the weights of the PFLARNN and the prediction performance for different sets of weight ranges.

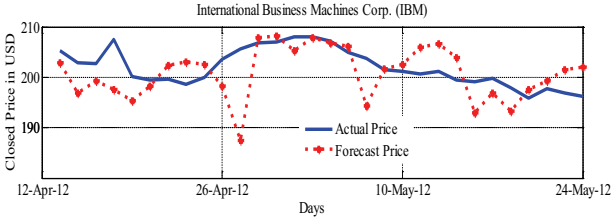
Copyright: the authors

5.2. Test Results for Hybrid PFLARNN and DE optimization

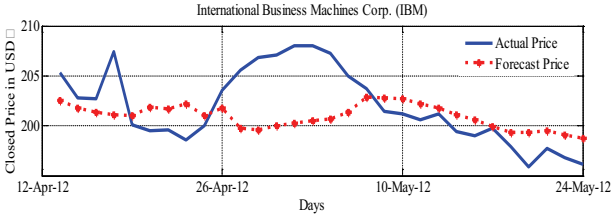
The following results are obtained with various network weights ranging from -1.0 to +1.0, -2.0 to +2.0, and -3.0 to +3.0 for the hybrid PFLARNN using DE optimization technique and the convergence is achieved in just 50 iterations. Figs.7 to 10 exhibit the 30-days testing results for different weight ranges. The prediction performance of the Lagurre Polynomial Function based PFLARNN is shown in Fig.7. Figs.8 and 9 illustrate the testing results of the same 30 days for different weight range using Chebyshev Polynomial Function based PFLARNN, and the Power series Polynomial function based PFLARNN trained by DE algorithm, respectively. From the figures 7 to 10 it is clearly seen that the weight range -3 to +3 produces the best performance prediction and this is also verified from Table-1, where all the performance metrics like MAPE, AMAPE, and variances have the least magnitudes.



(a): Forecasting performance during the testing period with network weights -1.0 to +1.0

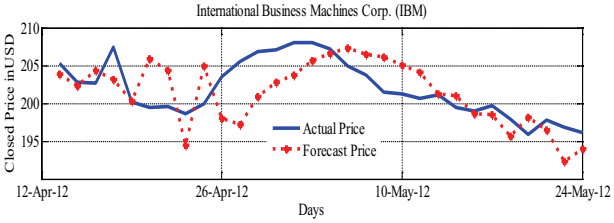


(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.

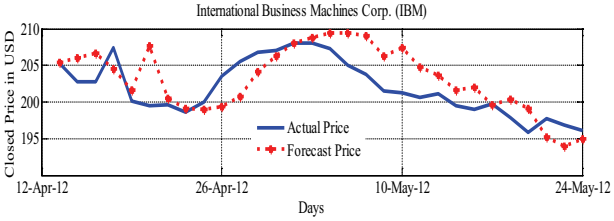


(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

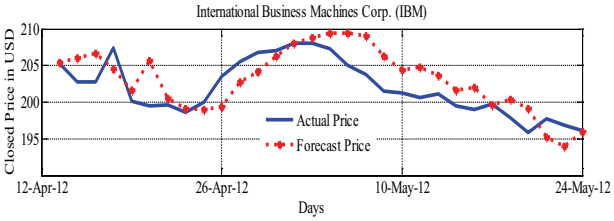
Fig.7: Effect of varying network weights on the forecasting performance using Laguerre Polynomial Function.



(a): Forecasting performance during the testing period with network weights -1.0 to +1.0

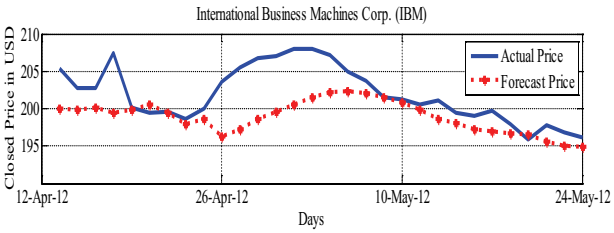


(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.

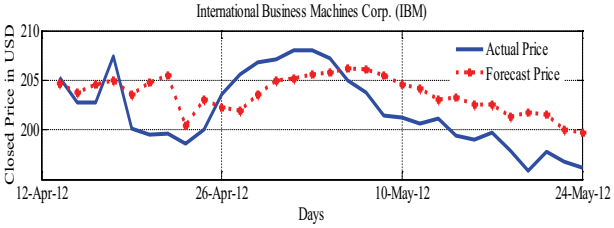


(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

Fig.8: Effect of varying network weights on the forecasting performance using Chebyshev Polynomial Function.

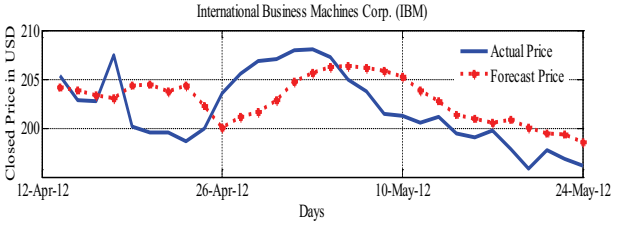


(a): Forecasting performance during the testing period with network weights -1.0 to +1.0



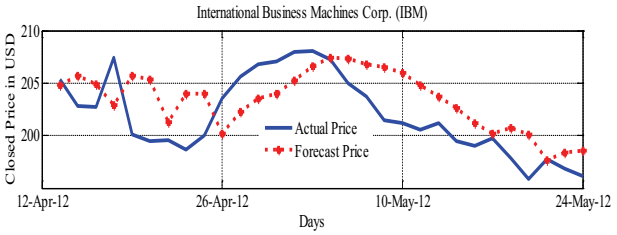
(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.

Copyright: the authors

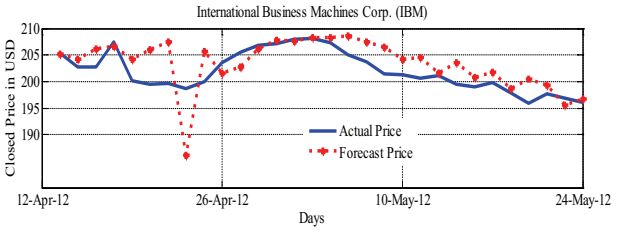


(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

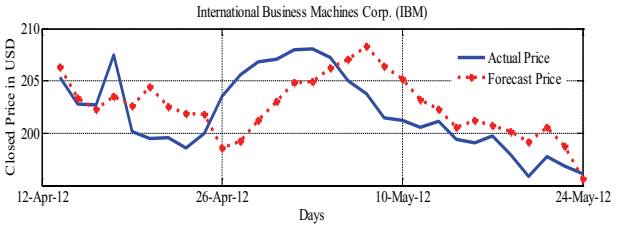
Fig.9: Effect of varying network weights on the forecasting performance using Power series Polynomial function.



(a): Forecasting performance during the testing period with network weights -1.0 to +1.0



(b): Forecasting performance during the testing period with network weights -2.0 to +2.0.



(c): Forecasting performance during the testing period with network weights -3.0 to +3.0.

Fig. 10: Effect of varying network weights on the forecasting performance using Legendre Polynomial Function.

Finally we have tried to show with a chart in Fig. 11 to present the benefits of the increased accuracy in the prediction in an economical term to give a clear increased profitability.

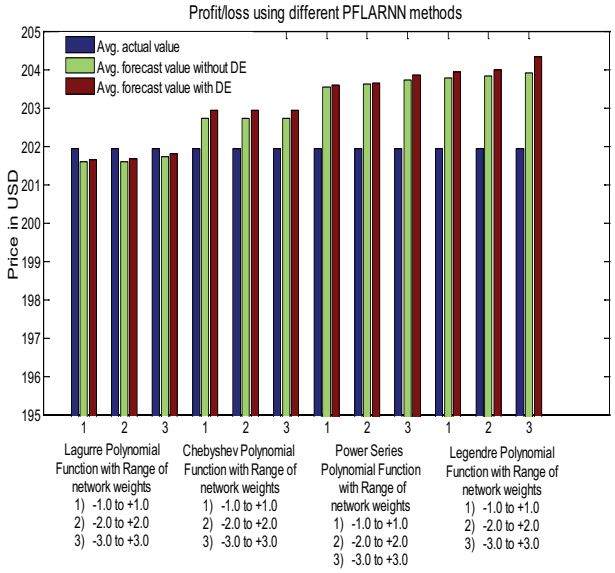


Fig. 11: Chart that shows the enhanced accuracy of profitability.

5.3. Discussion

The various performance indices like the MAPE, AMAPE, and the variance of forecast errors are given in Table-1. From this Table it is observed that the performance of the recurrent Polynomial FLANN is significantly better in the case of DE based learning than the conventional gradient descent algorithm. The best performance, however, is obtained using the Legendre Polynomial Function with optimized parameters of weights ranging from -3.0 to +3.0. Further in Table-2 it provides the benefits of the increased accuracy in the prediction value under various methods proposed in our research. Finally in Table-3, it provides a comparison between the widely used neural networks like the MLP, ANFIS, WNN, and RBFNN, etc. and the proposed PFLARNN in forecasting the IBM stock price over a time frame varying from 1 day ahead to 15 days ahead. Also for performance comparison the well known statistical model ARMA is used for forecasting IBM stock indices for the same period as considered for several neural architectures. From the Table it is observed that the performance of the RBFNN is slightly better than PFLARNN for forecasting when the time frame is increased to more than one day. As expected the ARMA produced the worst performance metric MAPE in comparison to the neural network models. This clearly

Copyright: the authors

demonstrates the universal approximation properties of the neural network over a widely used statistical model like ARMA. However, in the RBFNN the choice of hidden layer neurons pose a problem, which is to be settled by trial and error. WNN is described in the Appendix. After predicting the stock indices, it is easy to find out the stock trend movement which will be useful for stock trading and ultimate profit booking.

6. Conclusion

Functional Link Artificial Neural Network is a single layer ANN structure, in which the hidden layers are eliminated by transforming the input pattern to a high dimensional space by using a set of polynomial basis functions giving rise to a low complexity neural model. Different variants of FLANN can be modeled depending on the type of basis functions used in functional expansion. Also a recurrent version of the polynomial FLANN is presented in this paper to improve the speed of convergence and forecasting accuracy. A comparison with other neural architectures has been provided to validate the accuracy of the proposed neural network inspite of its simplicity and low complexity architecture. Further differential evolution (DE) is used to optimize the weight parameters of the PFLARNN to provide an improved accuracy for different architectures or functional expansions.

7. References

1. Contreras, J., Rosario, E., Nogales, F. J., Conejo, A. J. (2003) 'ARIMA Models to Predict Next-Day Electricity Prices', *IEEE Transactions on Power Systems*, Vol. 18, pp. 1014-1020.
2. Rojas, O. Valenzuelab, F. Rojas, A. Guillen, L.J. Herrera, H. Pomares, L. Marquez, M. Pasadas, (2008) 'Soft-computing techniques and ARMA model for time series prediction', *Neural Computing, Elsevier*, Vol-71, pp. 519-537.
3. Juan J. Flores, Mario Graff, Hector Rodriguez, (2012), 'Evaluative design of ARMA and ANN models for time series forecasting', *Renewable Energy, Elsevier*, Vol-44, pp. 225-230.
4. Chen, A. S., Leung, M. T., Daouk, H. (2003) 'Application of Neural Networks to an emerging financial market: Forecasting and trading the Taiwan Stock Index', *Computers & Operations Research on Elsevier*, Vol-30, pp. 901-923.
5. Fagner Andrade de Oliveira, Cristiane Neri Nobre, (2011), "The Use of Artificial Neural Networks in the Analysis and Prediction of Stock Prices", IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 2151-2155.
6. M.P.Singh, Sanjeev Kumar, N.K.Sharma, 'Mathematical formulation of second derivative of backpropagation error with nonlinear output function in feedforward neural networks', International Journal of Information and Decision sciences, 2010, Vol.2, no.4, pp.352-374.
7. Sun, Y.F., Liang, Y.C., Zhang, W.L., Lee, H.P. Lin, W.Z. and Cao, L.J.,'Optimal partition algorithm for the RBF neural network for financial time series forecasting', Neural Computing and Applications, 2005,Vol. 14, No. 1, pp.36-44.
8. Hsieh, T. J., Hsiao, H. F., Yeh, W.C., 'Forecasting stock markets using wavelet transforms and recurrent neural networks: an integrated system based on artificial bee colony algorithm', Applied Soft Computing, 2011, Vol. 11, No. 2, pp.2510-2525.
9. Yumlu, S., Gurgen, F.S. and Okay, N., 'A Comparison of global, recurrent and smoothed-piecewise neural models for Istanbul stock exchange prediction, Pattern Recognition Letters, 2005,Vol. 26, No. 13, pp.2093–2103.
10. Majhi Banshidhar, Shalabi Hasan, Fathi Mowafak, 2005, 'FLANN Based Forecasting of S&P 500 Index', Information Technology Journal, vol. 4, no. 3, pp. 289-292.
11. Patra, J. C., Bornand, C., Meher, P. K. (2009) 'Laguerre Neural Network-based Smart Sensors for Wireless Sensor Networks', *Instrumentation and Measurement Technology Conference, 2009. I2MTC '09. IEEE*, pp. 832-837.
12. Nanda, S. K., Tripathy, D. P., Mahapatra, S. S. (2011) 'Application of Legendre Neural Network for Air Quality Prediction', *the 5th PSU-UNS International Conference on Engineering and Technology (ICET-2011)*.
13. Patra, J. C. (2011) 'Chebyshev Neural Network-Based Model for Dual-Junction Sollar Cells', *IEEE Transactions on Energy Conversion*, Vol. 26, pp. 132-139.

Copyright: the authors

14. Chang, P.C. and Fan, C.Y., 'A Hybrid System Integrating a Wavelet and TSK Fuzzy Rules for stock price forecasting, IEEE Transactions on Man, Machine, and Cybernetics: Part-C,2008, Vol. 38, No. 6, pp.802-815.
15. Chang, P. C., Fan, C. Y. (2008) 'A Hybrid System Integrating a Wavelet and TSK Fuzzy Rules for Stock Price Forecasting', *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. 38, pp. 802-815.
16. Boyacioglu, M. A., Avci, D. (2010) 'An Adaptive Network Based Fuzzy Inference System (ANFIS) for the Prediction of Stock Market Return: The Case of the Istanbul Stock Exchange', *Experts Systems with Applications, ELSIVER*, Vol. 37, pp. 7908–7912.
17. Pao, Y. H., Takefji, Y. (1992) 'Functional-Link Net Computing', *IEEE Computer Journal,* pp.76-79.
18. Majhi, R., Panda, G., Sahoo, G. (2008) 'Development and performance evaluation of FLANN based model for forecasting of stock markets', Vol. 36, pp. 6800-6808.
19. Bebarta, D.K., Rout, A.K., Biswal, B., Dash P.K. (2012) 'Forecasting and classification of Indian stocks using different polynomial functional link artificial neural networks', *India Conference (INDICON), 2012 Annual IEEE*, pp. 178-182.
20. Bebarta, D.K., Biswal, B., Dash P.K. (2012) 'Comparative study of stock market forecasting using different functional link artificial neural networks' *International Journal of Data Analysis Techniques and Strategies, Inderscience Publishers*, Vol. 4, Number 4/2012, pp. 398-427.
21. Khan Asif Ullah, Bandopadhyaya T. K., Sharma Sudhir, 2008, 'Comparisons of Stock Rates Prediction Accuracy using Different Technical Indicators with Back propagation Neural Network and Genetic Algorithm Based Back propagation Neural Network', First International Conference on Emerging Trends in Engineering and Technology, pp. 575-580.
22. Hatem Abdul Kadar, Mustafa Abdul Salam, 2012, ' Evaluation of Differential Evaluation and Particle swarm Optimization Algorithms at Training of Neural Network for stock Prediction', International Arab Journal of e-Technology, vol. 2, no. 3, pp. 145-151.
23. Faissal Mili, Manel Hamdi, 2012, 'A Hybrid Evolutionary Functional Link Artificial Neural Network for Data Mining and Classification', International Journal of Advanced Computer Science and Applications, vol. 3, no. 8, pp. 89-95.
24. Yadav J. S., Patidar N. P., Panda J. Singhai Sidhartha, 2009, 'Differential Evolution Algorithm for Model Reduction of SISO Discrete Systems', World Congress on Nature & Biologically Inspired.

Appendix

Wavelet Neural Network

The output of wavelet neural network is given by

$$y_k(x) = \sum_{m=1}^{M} \omega_m \psi_m(x) = \sum_{m=1}^{M} \omega_m |a_m|^{-\frac{1}{2}} \psi_m\left(\frac{x - b_m}{a_m}\right)$$

where ψ_m is the wavelet activation function of the m^{th} unit of the hidden layer and ω_m is the weight connecting the m^{th} unit of the hidden layer to the output layer unit. The $\psi_m(x)$ is called as mother wavelet and the parameters a_m and b_m are the scale and translation parameters and x represents inputs to the WNN model.

The mother wavelet is given by

$$\psi_m(x) = -x \exp\left(-\frac{x^2}{2}\right)$$

WNN is a type of basis function neural network in the sense that the wavelets consists of the basis function. For higher dimension problems WNN requires more hidden units because a large number of basis function units have to be employed to approximate a given system.

Copyright: the authors

Table 1: Performance indices of PFLARNN using different polynomial basis functions

Method	Duration (30 Days)	Range of network weights	Performance Measurements					
			MAPE		AMAPE		variance (σ_{Err}^2)	
			Without DE	With DE	Without DE	With DE	Without DE	With DE
Laguerre Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	2.125	2.040	2.125	2.033	39.06×10^{-5}	33.26×10^{-5}
		-2.0 to +2.0	1.983	1.861	1.980	1.811	28.00×10^{-5}	21.71×10^{-5}
		-3.0 to +3.0	1.814	1.731	1.812	1.722	26.29×10^{-5}	12.42×10^{-5}
Chebyshev Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	1.602	1.564	1.595	1.511	12.65×10^{-5}	11.31×10^{-5}
		-2.0 to +2.0	1.479	1.218	1.478	1.341	10.36×10^{-5}	9.09×10^{-5}
		-3.0 to +3.0	1.434	1.391	1.433	1.327	8.00×10^{-5}	7.98×10^{-5}
Power Series Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	1.576	1.555	1.575	1.553	19.27×10^{-5}	14.95×10^{-5}
		-2.0 to +2.0	1.508	1.483	1.506	8.61×10^{-5}	11.78×10^{-5}	8.61×10^{-5}
		-3.0 to +3.0	1.417	1.443	1.416	1.444	8.95×10^{-5}	7.94×10^{-5}
Legendre Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	1.542	1.453	1.539	1.455	15.85×10^{-5}	15.01×10^{-5}
		-2.0 to +2.0	1.441	1.430	1.435	1.417	17.24×10^{-5}	14.63×10^{-5}
		-3.0 to +3.0	1.415	1.382	1.411	1.386	8.36×10^{-5}	6.29×10^{-5}

Copyright: the authors

Table 2: Accuracy reflects in terms of profit/loss using different PFLARNN methods by using without and with optimization based evolutionary method DE for 30 days.

Method	Duration (30 Days)	Range of network weights	Avg. actual value per unit share in USD	Avg. forecast value per unit share in USD (Profitability)	
				Without DE	With DE
Laguerre Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	201.9463	201.5835	201.6551
		-2.0 to +2.0	201.9463	201.5866	201.6817
		-3.0 to +3.0	201.9463	201.7245	201.7951
Chebyshev Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	201.9463	202.7154	202.9437
		-2.0 to +2.0	201.9463	202.7154	202.9437
		-3.0 to +3.0	201.9463	202.7154	202.9437
Power Series Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	201.9463	203.5511	203.5923
		-2.0 to +2.0	201.9463	203.6281	203.6543
		-3.0 to +3.0	201.9463	203.7324	203.8514
Legendre Polynomial Function	12/04/2012 to 24/05/2012	-1.0 to +1.0	201.9463	203.7665	203.9245
		-2.0 to +2.0	201.9463	203.8221	203.9848
		-3.0 to +3.0	201.9463	203.8953	204.3417

Table 3: MAPE errors during testing IBM data set using different ANN models along with ARMA optimized by DE

Days ahead	MLP	ANFIS	PFLARNN	RBFNN	WNN	ARMA
1	1.766	1.7212	1.382	1.562	1.743	1.935
3	2.371	2.176	1.764	1.897	2.132	2.562
5	2.834	2.316	2.083	2.005	2.659	3.273
7	3.181	2.995	2.409	2.394	3.125	3.982
15	4.613	3.946	3.142	3.096	4.232	5.877

Copyright: the authors