Neural Comput & Applic (2017) 28:101–110
DOI 10.1007/s00521-015-2039-0

ORIGINAL ARTICLE

An adaptive local linear optimized radial basis functional neural network model for financial time series prediction

A. Patra[1] · S. Das[2] · S. N. Mishra[3] · M. R. Senapati[4]

Received: 10 April 2015 / Accepted: 12 August 2015 / Published online: 23 August 2015

Abstract For financial time series, the generation of error bars on the point of prediction is important in order to estimate the corresponding risk. In recent years, optimization techniques-driven artificial intelligence has been used to make time series approaches more systematic and improve forecasting performance. This paper presents a local linear radial basis functional neural network (LLRBFNN) model for classifying finance data from Yahoo Inc. The LLRBFNN model is learned by using the hybrid technique of backpropagation and recursive least square algorithm. The LLRBFNN model uses a local linear model in between the hidden layer and the output layer in contrast to the weights connected from hidden layer to output layer in typical neural network models. The obtained prediction result is compared with multilayer perceptron and radial basis functional neural network with the parameters being trained by gradient descent learning method. The proposed technique provides a lower mean squared error and thus can be considered as superior to other models. The technique is also tested on linear data, i.e., diabetic data, to confirm the validity of the result obtained from the experiment.

Keywords Local linear radial basis functional neural network (LLRBFNN) · Radial basis functional neural network (RBFNN) · Multilayer perceptron (MLP) · Recursive least square (RLS) · Mean squared error (MSE)

1 Introduction

The financial market is the most unpredictable, volatile and random in nature. In the past years, there are huge ups and downs in the stock market. If accurate forecasting of the stock indices is done, it would be very helpful to the people who are regular investors in the stock market and the investor can get a clean and confident idea about the appropriate time to buy, hold or sell. It is known that the stock market data are one of the noisiest and inconsistent data so investors are always in a mess when they invest in the stock market considering the risk involved and they do not feel confident enough to get a profit out of it. This has lead different researchers to motivate their minds toward the study of stock market data prediction. The motivation in this field is to efficiently predict the stock market or efficient market hypothesis (EMH) [1]. The stock market is affected not only by economic factors but also by political situation of a country, terrorists attack and even on individual investor's psychology [2].

✉ S. Das
aug10.soumya@gmail.com

A. Patra
a_patra_2005@yahoo.com

S. N. Mishra
sarose.mishra@gmail.com

M. R. Senapati
manassena@gmail.com

[1] Department of Computer Science Engineering, Gandhi Institute for Education and Technology, Bhubaneswar, India

[2] Department of Computer Science and Engineering, Government College of Engineering, Kalahandi 766002, India

[3] Department of Computer Science Engineering, Indira Gandhi Institute of Technology, Saranga, Dhenkanal, India

[4] Department of Computer Science and Engineering, Centurion University of Technology and Management, Bhubaneswar 752050, India

Neurophysiologist Warren Mcculloh and logician Walter pits were the pioneers in the field of neural networks and established it in 1943. Neural networks are less sensitive to error term assumptions, and they can tolerate noise and chaotic components. Bank and other financial institutions are investing heavily in the development of neural network model and have started to implement it in the financial trading arena.

In the past decades, many approaches have been dedicated toward the development of the models of forecasting the market trends. One of the most popular and traditional method to predict the market data is time series methods. These traditional methods analyze the past data to find out the pattern of changes in the variable. The time series model using linear approach are exponential smoothing, moving average, neural network and fuzzy models [3]. One of the most adopted statistical techniques is autoregressive integrated moving average (ARIMA)-based models [4]. Stock markets are dynamic in nature and are not perfectly linear, so these models are not the successful in predicting the stock market. There are some nonlinear models such as autoregressive conditional heteroskedasticity (ARCH) and general autoregressive conditional heteroskedasticity (GARCH) [5]. Exponential GARCH was used in forecasting the market, but they do not have any specified steps. After the introduction of neural network, many complex financial models are easily implemented and shortcoming of the traditional models is observed. By using neural network model, an ideal model can be implemented to predict and forecast the stock market.

Local linear radial basis functional neural network (LLRBFNN) is a modification of radial basis function neural network (RBFNN) and is a recent technique. Not much work has been reported on this technique. The authors have used this technique [6] for time series prediction, and the reported result in terms of mean squared error (MSE) is 0.0047. In the year 2014, authors [7] have used this for breast cancer classification getting an accuracy of 98.74. In this paper, the technique is used to forecast the closing price of YahooInc. A detail discussion of the technique is provided in Sect. 3.

An adaptive neural network is a system where the system parameters change according to the input. Accurate forecasting of time series data requires the system to be more adaptive, i.e., the system parameters can change at a faster rate according to the input. In our study, the parameters such as centers, weights and variance change based on the error calculated in the output layer. Unlike RBFNN, in case of LLRBFNN, the number of neurons in the hidden layer is known and is equal to the number of inputs.

The rest of the sections are organized as follows: The Literature review is presented in Sect. 2, and the concept LLRBFNN is described in Sect. 3.1. Brief discussion about RBFNN and multilayer perceptron (MLP) is provided in Sects. 3.2 and 3.3, respectively. Sections 4.2 and 4.5 show the simulation results and the comparative study and discussion, and finally, the conclusion is described in Sect. 5.

2 Literature review

In the ANN approach, the different models used are MLP [8], recurrent neural network (RNN) and wavelet neural network [9]. These models provide robust performance when the dataset is huge. In addition, the learning algorithms such as backpropagation, least mean square method and recursive least square (RLS) method help to train the model and evaluate the value of unknown parameters [9]. In the year 2009, forecasting of stock price using backpropagation (BP) neural network was analyzed by the researcher Li [10], by utilizing three-layered feedforward neural networks as well as topology of network, principles of determining the number of hidden layers, selection and pretreatment of sample data and determination of preliminary parameters. In order to promote convergence speed, Hung [11] proposed the model for forecasting of volatility of stock markets using adaptive fuzzy–GARCH and particle swarm optimization (PSO). Shen [12] selected a RBFNN to train data and forecast the stock indices of the shanghai stock exchange, and they introduced the artificial fish swarm algorithm (AFSA) to optimize RBF in the year 2011. In the year 2012, Adebiyi et al. [13] presented the model based on technical and fundamental indicators, and expert's opinions using neural network architecture. The aim was to yield more accurate results in stock price prediction [13]. Author Chen [14] proposed a hybrid ANFIS model to forecast stock price of 160 electronics companies listed by the Taiwan stock exchange corporation in the year 2013. There are various methods available in the literature of neural network, such as MLP model, RBFNN [15], functional link artificial neural network (FLANN) [16], adaptive neurofuzzy information system (ANFIS) and LLRBFNN [6]. Using these methods, the finance market can be predicted before 1 day, 1 week or 1 month. Many researchers have conducted several researches to check whether artificial neural network (ANN) models are actually capable of handling the nonlinear pattern of the financial data, which includes some hybrid models that combine statistics with neural network, combination of wavelet transform and fuzzy logic with neural networks. Models using this neural network are widely used to design systems to forecast both nonlinear and linear data [17]. RLS [18, 19], compared to gradient decent method and least mean square (LMS) algorithm, has a faster rate of convergence algorithm and hence can be used to train the

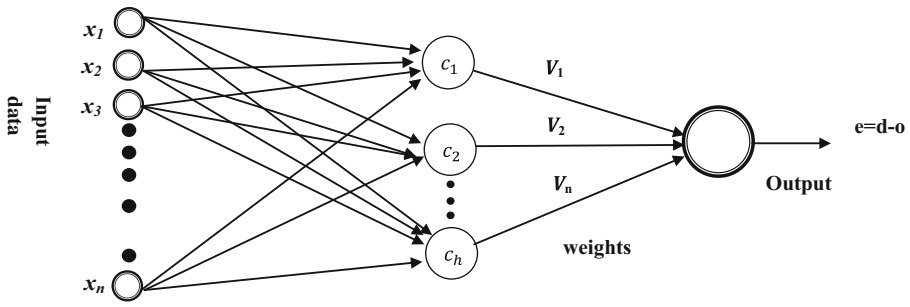

Fig. 1 Architecture of LLRBFNN

model. Therefore, we applied a hybrid technique using RLS and backpropagation to train LLRBFNN.

In this paper, a LLRBFNN [6] is presented where weights are trained using RLS algorithm and centers using error backpropagation. The basic difference between RBFNN and LLRBFNN model is that the connected weights between the hidden layer and the output layer are replaced by a linear model [6]. In comparison with LLRBFNN and RBFNN, the LLRBFNN is more efficient and has more ability to approximation than with the RBFNN as it uses the local linear model instead of the hidden layer units [6].

Large amounts of time series data are produced routinely in a number of application areas, such as finance, economy, process industry, experimental sciences, medicine and monitoring of electrical distribution network disturbances. Analysis of such data is required to discover important knowledge in data, and hence, mining of non-stationary data assumes significance and usefulness.

3 Time series prediction

3.1 Prediction using local linear RBFNN

RBFNN is the one of the powerful feedforward network models used for approximation, regulation, classification and pattern recognition. The RBFNN with local linear model in the hidden layer is called LLRBFNN [6]. There are no connected weights between input layer and hidden layer. The output layer of the LLRBFNN is given by:

$$y = \sum_{j=1}^{n} v_j z_j(x), \tag{1}$$

$$v_j = w_{j1}x_1 + w_{j2}x_2 + \cdots + w_{jn}x_n \tag{2}$$

$$z_j(x) = \exp\left(\frac{-\|x - c_j\|^2}{2\sigma_j^2}\right) \tag{3}$$

where $x = [x_1, x_2, \ldots, x_n]$ are the inputs, c_j is the center of jth activation function, σ_j is the spread which is responsible to control the smoothness of the activation function and $\|x - c_j\|$ is the Euclidean distance between the inputs and the center of the function [15]. In the proposed model, the centers are trained by using gradient descent learning and the equation to update the centers is given by:

$$C_{ij}(t+1) = C_{ij}(t) + \eta_1 + \frac{\partial E}{C_{ij}}, \quad \text{for } i = 1 \text{ to } p \text{ and } j = 1 \text{ to } h \tag{4}$$

In this model, the weights are present in between the hidden layer and the output layer and they are updated using the RLS algorithm, discussed in Sect. 3.2 (Fig. 1).

The cost function, i.e., the error function, is given by:

$$E = \frac{1}{2}\sum (d - o)^2 \tag{5}$$

where E is the sum of the error square, d is the desired output and o is the output obtained from the network.

3.1.1 Recursive least square (RLS)

The RLS is a parameter identification technique. This technique is implemented here as it has a faster rate of convergence in comparison with other available techniques [6, 9]. In this algorithm, there are two variables involved in the recursion process (those with the time index $n - 1$): $\hat{w}(i-1)$, P_{i-1}. We have to provide initial values for these variables in order to start the recursion. The weight updating rule is given below:

$$k(i) = \frac{P(i-1)\varphi T(i)}{\lambda + p(i-1)\varphi T(i)} \tag{6}$$

$$w(i) = w_j(i-1) + k(i)\left[d_j(i) - w_j(i-1)\varphi T(i)\right] \tag{7}$$

$$P(i) = \frac{1}{\lambda}[P(i-1) - k(i)\varphi(i)P(i-1)] \tag{8}$$

where λ is the forgetting factor, P is the filter order, φ is the value to initialize $P(0)$ and W is the weights between hidden layer and output layer.

Here λ is real number between 0 and 1 and $P(0) = a^{-1}I$, in which a is a small positive number and $w_j\ (0) = 0$.

The computational steps involved in implementing the LLRBFNN for forecasting are as follows:

1. *For each input data x,*
2. *Update the LLRBFNN center by gradient descent learning.*
3. *Update the weights of the model using RLS.*
4. *Calculate the mean error by using the cost function*
5. *if the error is within the specified limit*
   go to step 6
   else go to step 2
6. *stop*

The process of updating continues till the centers and weights are converged and the updated parameters act as a new training input to the LLRBFNN model in the beginning of the iteration.

3.2 Prediction using radial basis function neural network

The RBFNN model is applied in classification, approximation, pattern recognition, etc. Each RBFNN center approximates a cluster of training of data vectors that are closed to each other in Euclidian space [12, 18]. When a signal is inputted into the model, the nearer center of that vector gets activated, resulting activating certain output neurons.

This neural network model consists of mainly three layers: An input layer, which is made up of source nodes; hidden layer, which is made up of an appropriate dimension; and an output layer, which provides the output of the network to the activation pattern implemented in the input layer. The nodes are strongly connected to the previous layer nodes along with weights in between the hidden layer and the output layer.

3.3 Prediction using multilayer perceptron (MLP)

The MLP is one of the most popular neural network architecture. The MLP is a universal approximate and is taken as the benchmark for comparing the performance of other neural network architectures. A MLP uses connected weights in both layers, from input to hidden layer and hidden layer to output layer. In most of the models, activation function used is sigmoid function, but here the activation function used is the tangent function. Training of backpropagation requires optimal selection of the parameter vectors of the weights (between input layer to hidden layer and hidden layer to output layer). Weights are multiplied to the inputs and passed through an activation function [Activation function = *f*(Sum(inputs * weights))].

The backpropagation algorithm is as follows:

initialize network weights (often small random values)
do
for each training example x_i
prediction = neural-net-output(network, x_i*) // forward pass*
actual = obtained output(x_i*)*
compute error (prediction - actual) at the output units
compute Δw_t *for all weights from hidden layer to output layer // backward pass*
compute Δw_i *for all weights from input layer to hidden layer // backward pass continued*
update network weights using back propagation
until all examples classified correctly or another stopping criterion satisfied
return the network.

The weights update in MLP is done by using the following rules:

The rule to update the weights from input to hidden layer is given by:

$$w_{(t+1)} = w_t + 2\mu e\left(1 - \text{out}^2\right)h_i \tag{9}$$

where e is the error, w_t is the old weight, $w_{(t+1)}$ is the updated weight and μ is the learning rate.The rule to update the weights from hidden layer to output layer is given by:

$$\frac{\mathrm{d}E}{\mathrm{d}w_{ij}} = -2e\left(1 - \text{out}^2\right)\left(1 - h_i^2\right)w_i \tag{10}$$

After derivation, the generalized equation to update the weights of hidden layer to output layer is given by:

$$w_{ij(t+1)} = w_{ij} + 2\mu\left(1 - \text{out}^2\right)(1 - h_i)w_i \tag{11}$$

4 Implementation

The Yahoo Inc dataset is used, in order to analyze the performance of the above-mentioned algorithm [20]. The normalized data are used for training all the three models, i.e., MLP, RBFNN and LLRBFNN. The testing is carried out after the mean error and weights have converged. The centers and weights of RBFNN were updated using gradient descent learning technique. The LLRBFNN model has four neurons in the input layer, four neurons in the hidden layer and one neuron in the output layer. The centers of the LLRBFNN model are updated using the gradient descent learning method, and the weights are updated using the RLS algorithm. The initial weights are initialized to ones, and the weights are updated once after each input pattern is applied to the network. The training of the networks continues till there is no change in the weights. The error function is the cost function, and the performance of the network is determined by MSE. The MSE can be calculated and given by:

$$\text{MSE} = \frac{\frac{1}{2}\sum(d - o)^2}{N} \tag{12}$$

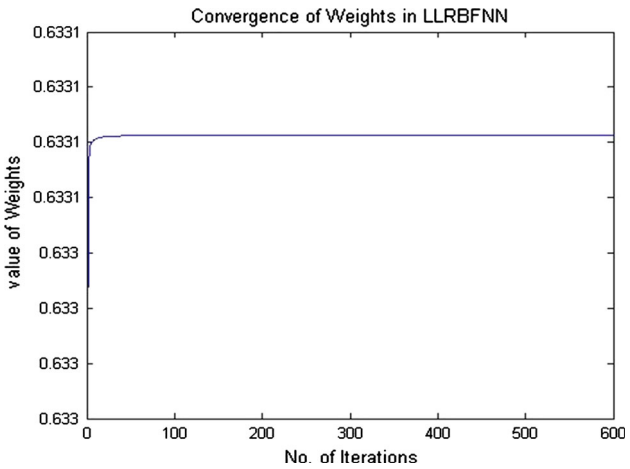

Fig. 2 Weight convergence of LLRBFNN

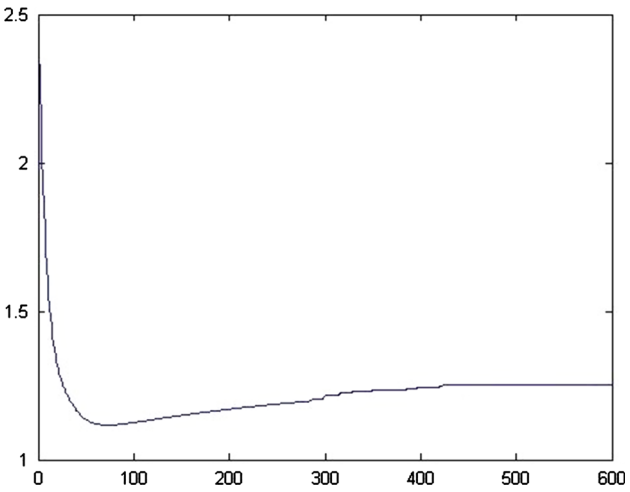

Fig. 3 Weight convergence of RBFNN

where d is the desired output, o is the actual output and N is the number of sample data.

4.1 Details of dataset

The dataset used in the network model for training and testing is the historical data of stock for Yahoo Inc [20] from January 1, 2007, to January 1, 2013. The dataset contains 1500 samples divided into seven columns named as date, open price, close price, high, low, volume and adj. close. Out of these columns, two columns, such as date and adj.close are discarded as they are not required and close is discarded as it is used in the output node for comparison and to calculate the error for pattern by pattern. The dataset containing 1500 samples is divided into two parts: One is for training and other is for testing of the network. Approximately 64 % of the whole data are used as training data and rest 36 % data are used as testing data. The actual data are in the form of large whole numbers, so they have been normalized and converted to the value range between [0, 1] with the help of min–max normalization. The formula is given by:

$$X_{\text{norm}} = \frac{X_{\text{original}} - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{13}$$

4.2 Result and comparative study

Figures 2, 3 and 4a, b show the convergence of weights that were obtained during the training of the models of LLRBFNN, RBFNN and MLP. Figure 5a, b shows the comparison between forecasted and the actual index value of the stock market during training and testing phase of the model using LLRBFNN. Figure 6a, b shows the compar-

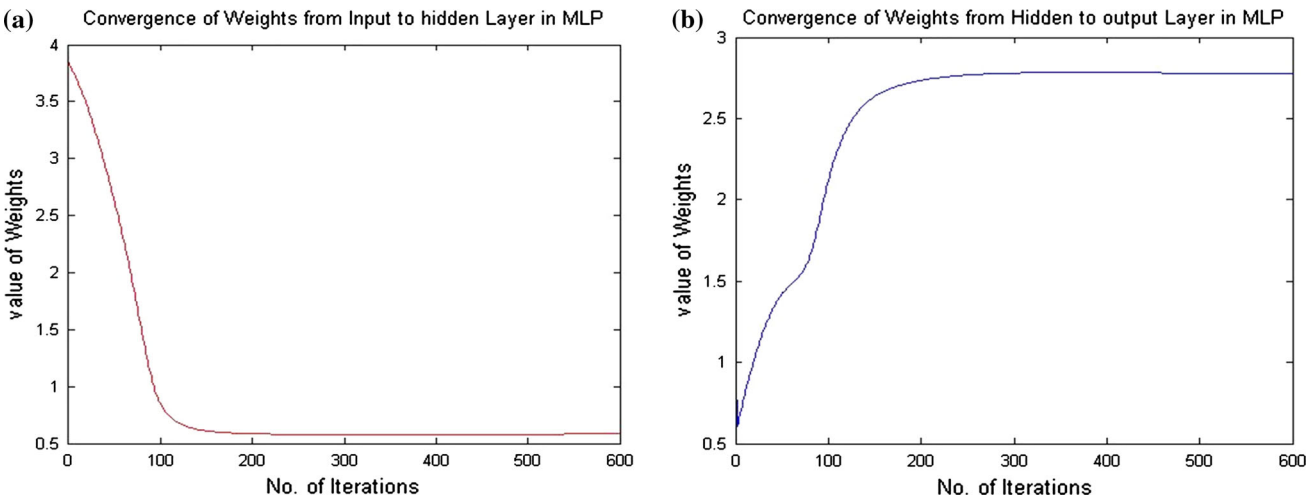

Fig. 4 **a** Weight convergence between input and hidden layer of MLP and **b** weight convergence between hidden and output layer of MLP

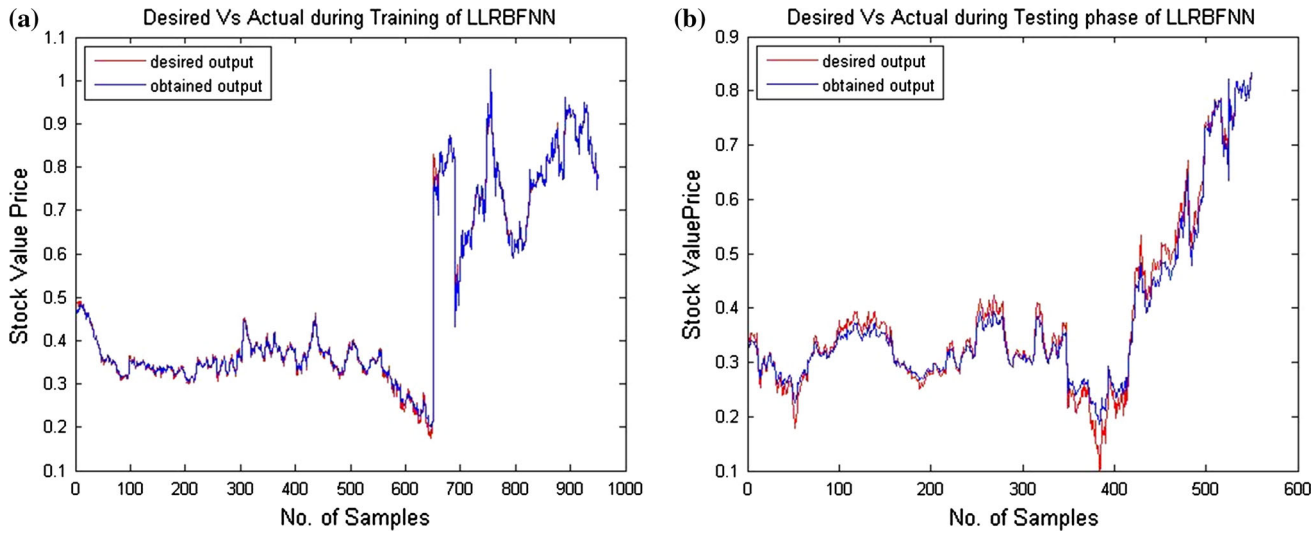

Fig. 5 **a** Desired verses actual LLRBFNN (training) and **b** desired verses actual LLRBFNN (testing)

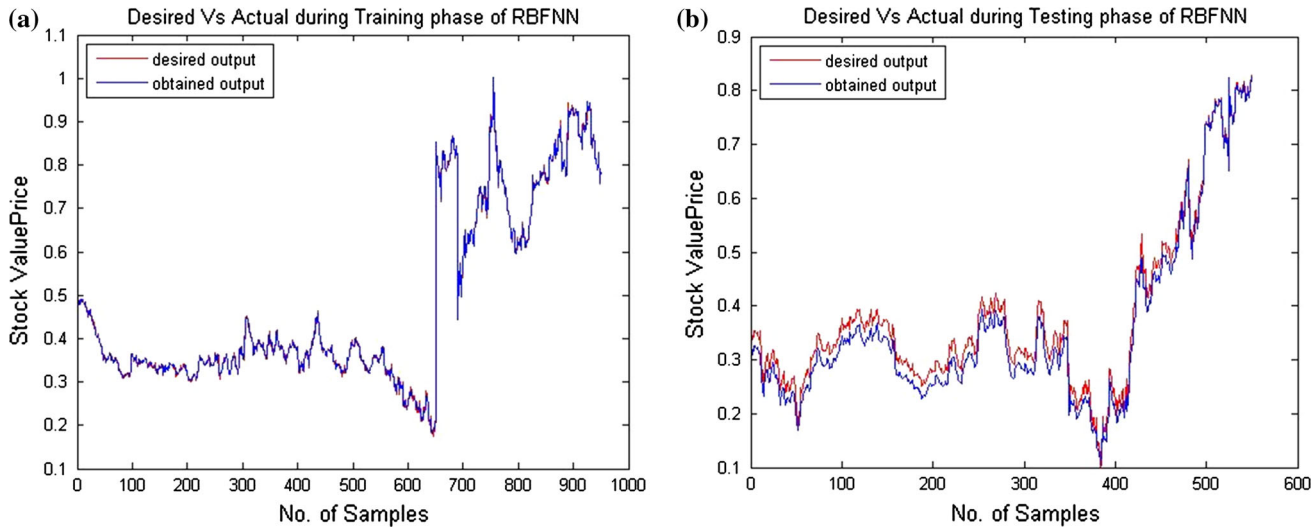

Fig. 6 **a** Desired verses actual RBFNN (training) and **b** desired verses actual RBFNN (testing)

ison between forecasted and actual index value of the stock market obtained during the training and testing phase of the model using RBFNN. In the same manner, Fig. 7a, b gives us idea about the forecasted and actual index value of the stock market of the model using MLP. Optimized values of weights of the MLP, RBFNN and LLRBFNN models, and the centers of RBFNN and LLRBFNN models are listed in Table 1. The average distance between actual and forecasted price in a group of 50 is given in Table 2, and the MSE values during training as well as in testing of different techniques are given in Table 3.

4.3 Comparison with linear (diabetic) dataset

To compare the proposed technique with other linear datasets, we used one popular dataset (diabetes) from AIM hospital downloaded from UCI repository [21] for classification.

The diabetes dataset consists of a total of 978 exemplars [21, 22]. For our classification purpose, 768 exemplars were used for training and the rest 210 exemplars for testing.

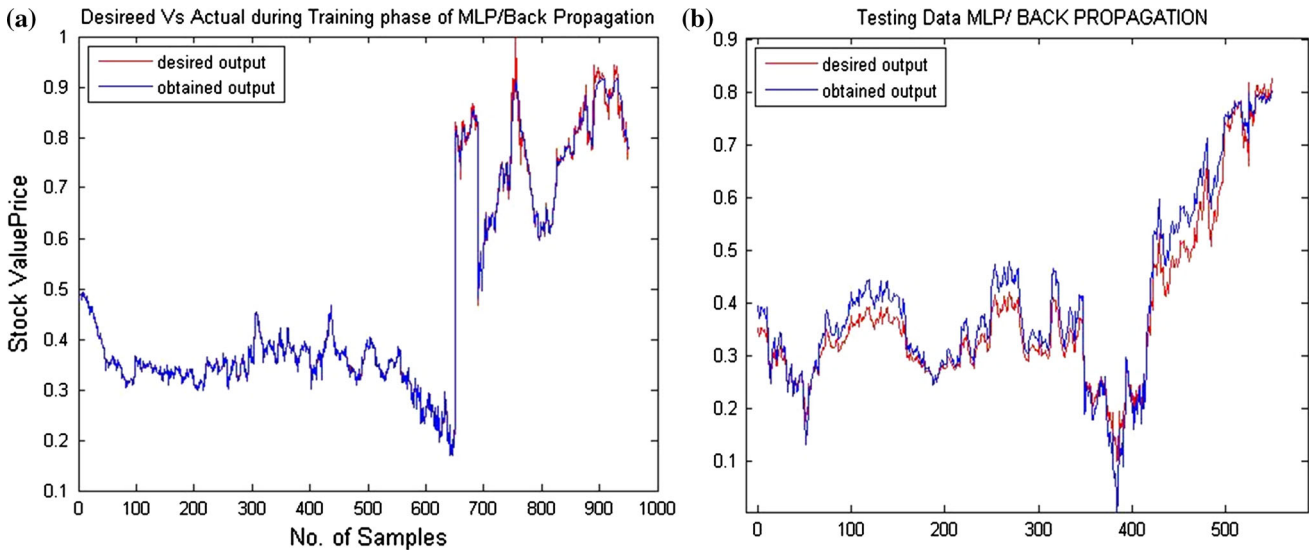

Fig. 7 **a** Desired verses actual MLP (training) and **b** desired verses actual MLP (testing)

Table 1 Value of weights and centers after training phase

Technique	Execution time in seconds	Weights between input and hidden layer	Weights between hidden and output layer	Centers
LLRBFNN-RLS	154.6733	NA	0.4212	0.6796
			0.5318	0.6759
			0.1300	0.8078
			−0.4498	2.6074
RBFNN	107.9096	NA	0.4041	0.8459
			0.4051	0.9316
			0.4054	0.7821
			0.4058	1.1184
MLP	103.8777	0.046	0.9277	NA
		0.0286	0.9297	
		−0.0453	0.9365	
		0.5681	0.9287	

Table 2 Performance comparison between different models (average distance between forecasted and actual price during testing)

Number of iterations	LLRBFNN	RBFNN	MLP
1–50	0.0028025744	0.0279210060	0.0124950488
51–100	0.0002256634	0.0269105210	0.0179989039
101–150	0.0178242565	0.0300106648	0.0446204503
151–200	0.0022996622	0.0277497219	0.0132845595
201–250	0.0025146153	0.0284467002	0.0210631671
251–300	0.0152513494	0.0295768559	0.0406010535
301–350	0.0086893765	0.0287510662	0.0311769843
351–400	0.0311034265	0.0199460819	0.0274371475
401–450	0.0119041259	0.0250712527	0.0377427926
450–500	0.0281871337	0.0165772913	0.0608545078
501–550	0.0044050568	0.0028010096	0.0038842155

Table 3 Simulation results obtained from the study

Technique	Learning algorithm used	Mean squared error (training)	Mean squared error (testing)
LLRBFNN	Backpropagation and RLS	0.00011	0.004767267
RBFNN	Backpropagation	0.00013	0.024022056
MLP	Backpropagation	0.00014	0.025341032

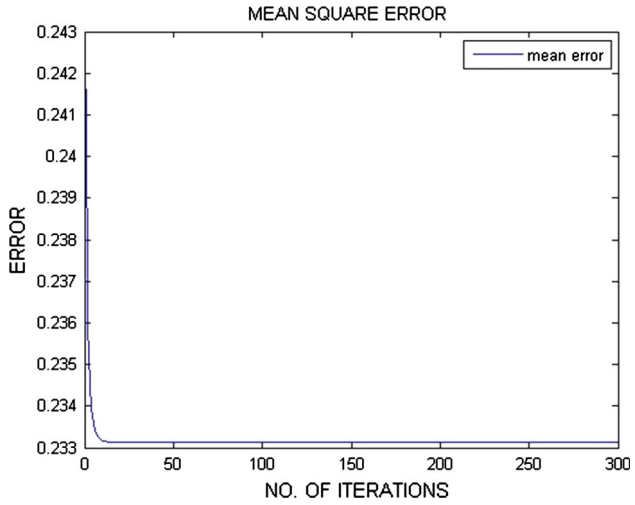

Fig. 8 MSE convergence of LLRBFNN

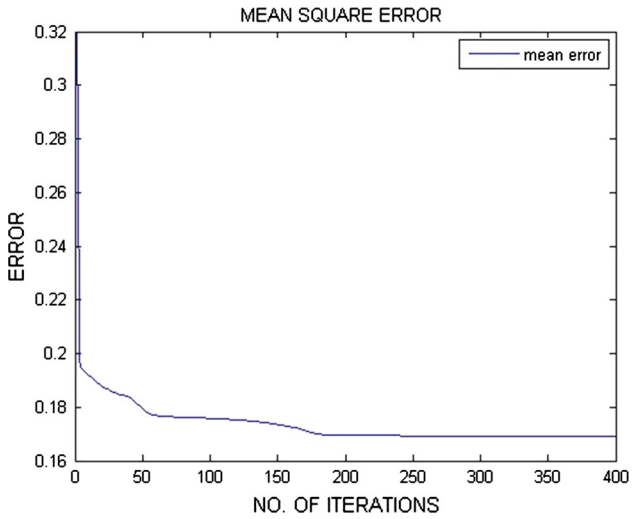

Fig. 10 MSE convergence of MLP

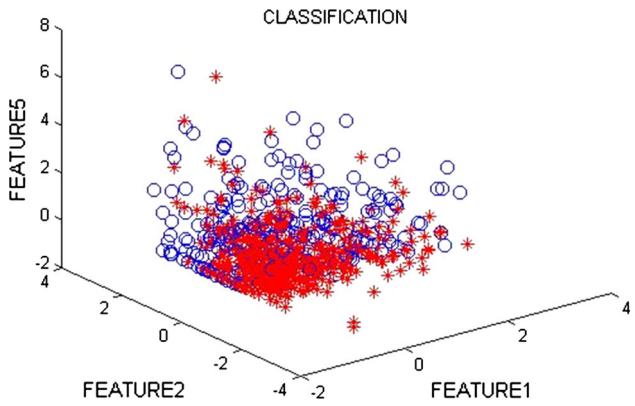

Fig. 9 Classification using LLRBFNN

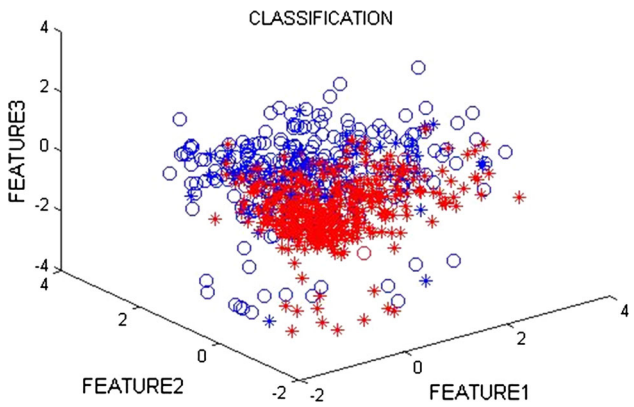

Fig. 11 Classification using MLP

4.4 Results and comparison using diabetic dataset

Figure 8 represents the MSE convergence curve, and Fig. 9 represents the classification using LLRBFNN. Similarly, Figs. 10 and 11 represent the MSE curve and classification using MLP. Figures 12 and 13 represent the MSE curve and classification using RBFNN, respectively. Table 4 describes confusion matrix for all the three models.

4.5 Discussion

From Tables 1, 2 and 3, we observe the followings

1. The average distance during testing, in case of LLRBFNN, is the minimum in comparison with RBFNN and MLP.

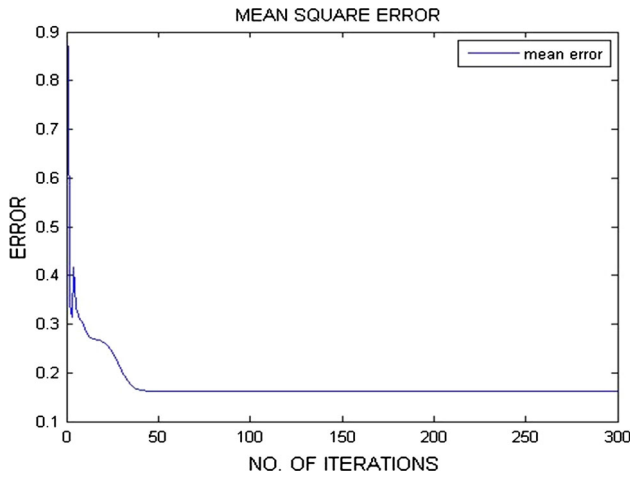

Fig. 12 MSE convergence of RBFNN

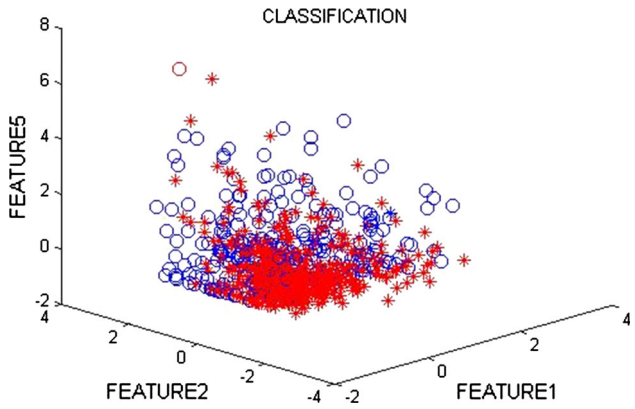

Fig. 13 Classification using RBFNN

Table 4 Confusion matrix obtained for the three models

	Class 1	Class 2
Confusion matrix LLRBFNN		
Class 1	500	9
Class 2	0	259
% of classification	100.00	96.64
Overall percentage is 98.83 %		
Confusion matrix MLP		
Class 1	428	57
Class 2	71	210
% of classification	85.60	78.36
Overall percentage is 83.07 %		
Confusion matrix RBFNN		
Class 1	484	18
Class 2	16	249
% of classification	96.80	92.91
Overall percentage is 95.44 %		

2. The LLRBFNN is giving the lowest MSE in case of training and testing in comparison with RBFNN and MLP.
3. This justifies that the accuracy of LLRBFNN is more than that of RBFNN and MLP.
4. Also it is observed that the time taken by LLRBFNN is more than that of the RBFNN or MLP. This is because LLRBFNN has more number of neurons in the hidden layer.

We draw our conclusion in the following section by utilizing the useful information shown in Tables 1, 2 and 3.

5 Conclusion

In this paper, LLRBFNN is used to forecast the nonlinear dynamics of stock price time series. The LLRBFNN provides a parsimonious interpolation in high-dimension space, and it is more efficient than RBFNN. Also the local linear model gives more ability than a constant weight, and thus in the proposed LLRBFNN, only a few of radial basis functions are necessary for time series predicting. In other words, the LLRBFNN provides better results than RBFNN and MLP. This claim has been demonstrated in the simulation of three networks (LLRBFNN, RBFNN and MLP) with the same inputs. The objective of the study was to test the effectiveness of LLRBFNN.

Simulation results for financial time series prediction problem show the effectiveness of the presented approach for prediction. We have tested all the three techniques on diabetics data downloaded from UCI repository to show the effectiveness of LLRBFNN and to confirm to the result of the study carried on time series data. It is observed that this technique suffers from a major drawback. Its training time is more as compared to the training time of RBFNN or MLP. In future, research need to be carried out to reduce the training time of LLRBFNN. It is a known fact that data mining techniques are more suitable to larger databases; we intend to use this technique on a large database to validate the findings of this study.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Chakravarty S, Dash PK (2009) Forecasting stock market indices using hybrid network. Nat Biol Inspir Comput. doi:10.1109/NABIC.2009.5393749 **(ISBN 978-1-4244-5053-4/09/IEEE)**
2. Majumdar M, Hussain MDA (2007) Forecasting of Indian stock market index using artificial neural network. Inf Sci 98–105

3. Zhang GP (2003) Time series forecasting using a hybrid ARIMA and neural network model. Neurocomputing 50:159–175
4. Wang J-H, Leu J-Y (1996) Stock market trend prediction using ARIMA-based neural networks. In: IEEE international conference on neural networks, 1996, vol 4. IEEE, pp 2160–2166
5. Liu H-C, Hung H-C (2010) Forecasting S&P-100 stock index volatility: the role of volatility asymmetry and distributional assumption in GARH models. Expert Syst Appl 37(7):4928–4934
6. Nekoukar V, Beheshti MTH (2010) A local linear radial basis function neural network for financial time series forecasting. Appl Intell 33(3):352–356. doi:10.1007/s10489-009-0171-1
7. Senapati MR, Das SP, Champati PK, Routray PK (2014) Local linear radial basis function neural networks for classification of breast cancer data. In: PISER 16, vol 02, pp 033–042
8. Carpinteiro OAS, Leite JPRR, Pinheiro CAM, Lima I (2012) Forecasting models for prediction in time series. Artif Intell Rev 38(2):163–171. doi:10.1007/s10462-011-9275-1
9. Senapati MR, Dash PK (2013) Intelligent system based on local linear wavelet neural network and recursive least square approach for breast cancer classification. Artif Intell Rev 39:151–163. doi:10.1007/s10462-011-9263-5
10. Li F, Li C (2009) Application study of BP neural network on stock market prediction. In: Ninth international conference on hybrid intelligent systems. IEEE, pp 174–177. doi:10.1109/HIS.2009.248
11. Hung JC (2011) Adaptive fuzzy-GARCH model applied to forecasting the volatility of stock markets using particle swarm optimization. Inf Sci. doi:10.1016/j.ins.2011.02.027
12. Shen W (2011) Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm. Knowl Based Syst 24(3):378–385. doi:10.1016/j.knosys.2010.11.001
13. Adebiyi AA, Ayo CK, Adebiyi MO, Otokiti SO (2012) An improved stock price prediction using hybrid market indicators. Afr J Comput ICT 5(5):124–135 **(ISSN-2006-1781.IEEE)**
14. Chen M-Y (2013) A hybrid ANFIS model for business failure prediction using PSO and subtractive clustering. Inf Sci 220:180–195. doi:10.1016/j.ins.2011.09.013
15. Samantray SR, Dash PK, Panda G (2006) Fault classification using HS-transformation and radial basis function neural network. Electr Power Syst Res 76:897–905
16. Bahrammizaee A (2010) A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert systems and hybrid intelligent system. Neural Comput Appl 19(8):1165–1195
17. Kawaji S, Chen YH (2001) Evolving neuro fuzzy system by hybrid soft computing approaches for system identification. Int J Adv Comput Intell Intell Inform 5(4):229–238
18. Senapati MR, Vijaya I, Dash PK (2007) Rule extraction from radial basis functional neural networks by using particle swarm optimization. J Comput Sci 3:592–599
19. Senapati MR, Panda G, Dash PK (2012) Hybrid approach using KPSO and RLS for RBFNN design for breast cancer detection. Neural Comput Appl. doi:10.1007/s00521-012-1286-6
20. www.finance.yahoo.inc
21. http://archive.ics.uci.edu
22. Tomczak JM, Gonczarek A (2013) Decision rules extraction from data stream in the presence of changing context for diabetes treatment. Knowl Inf Syst 34(3):521–546