IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-6, June 2018 E-ISSN: 2347-2693

Optimization of the Radial Basis Function Neural Networks Using Genetic Algorithm for Stock Index Prediction

P.S. Mishra

Dept. of Computer Science, North Orissa University, Baripada, Odisha, India

*Corresponding Author: parthanou@gmail.com, Tel.: +91-94371-88661

Available online at: www.ijcseonline.org

Accepted: 08/Jun/2018, Published: 30/Jun/2018

Abstract— Stock index prediction is one of the important tasks in the domain of computational finance. A number of tools have been developed by various groups of researchers and are being used by many analysts to identify the future price index. However, due to the high degree of non-linearity of the problem and surrounded by many optimal solutions, this paper proposes Radial Basis Function Neural Networks (RBFNNs) learning using Genetic Algorithm (GA) to predict the stock price index and at the same time the connection weights between the layers and thresholds are optimised using GA. Further, potential indicators are used to make the model robust in terms of its efficiency and accuracy. The accuracy is compared to MLP-BP and GA models. Finally, the experimental results show that the optimized RBFNNs model is the optimum model in comparison to other conventional models.

Keywords— Stock Index Prediction, RBF Neural Network, Genetic Algorithm, Real coding

I. INTRODUCTION

Stock index prediction is a challenging and promising task in the financial domain, where the researchers have their keen interest to solve financial problems using new emerging techniques. Generally these techniques include Neural Networks, Genetic Algorithms, Genetic Programming, Grammatical Evolution, Particle Swarm Optimisation and Ant Colony Models etc. Radial basis function (RBF), emerged as a variant of artificial neural network, have been successfully applied to a large diversity of applications including prediction, classification, interpolation, chaotic time-series modelling, control engineering, image restoration, moreover data fusion etc.[1][2].Genetic algorithms are adaptive and robust computational procedures modelled on the mechanics of natural genetic systems [3]. Due to the extensive global optimization capability, Genetic algorithms have been applied on neural network and so on.

Although numerous theoretical and experimental studies have reported the usefulness of neural networks in financial predictions, still there are several drawbacks in building model. Due to the numerous network architectures, learning methods and parameters, it has become an art to develop an appropriate neural network model to solve the problem. Finally, the user cannot apply the exact rules to build the neural network model due to the 'Black boxes' characteristic of it.

Hence, this paper proposes a high order neural network learning method of RBFNNs using GA for the prediction of stock index. The advantage of this approach offers an exact method of predictions for easy understanding of the users as compared to the other benchmark methods already published by the author[4][5].

The rest of the paper is organised as follows. The Section II provides a brief description of RBFNNs and GA. The Section III describes GA approach for learning RBFNNs. The Section IV reports the model development and the results of the experiments. Finally, Section V concludes research work with future directions.

II. RELATED WORK

Prediction of stock market index, corporate failure, bankruptcy, and as well as bond ratings using past financial data is also a well-documented topic. Early studies of financial domain like index prediction of stock market using artificial neural network [6][7], bankruptcy prediction using statistical techniques such as multiple discriminated analysis [8][9], logit and probit regression models[10][11], neural network[12][13], Corporate failure prediction using grammatical evolution[14], and genetic algorithmic technique[15] are the remarkable ones. The GA's applications in financial domain are growing with successful rate in trading system [16][17], stock selection [18], portfolio selection [19], bankruptcy prediction [20], credit evaluation

[21] and budget allocation [22]. Other methods like intelligent technique of rough set theory for analyzing the imprecise medical data [23]. Although there are some statistical methods have been used for financial prediction, it suffers from the shortcomings. There are a number of reasons to suppose a priori that the use of GA in learning RBFNNs and its optimising capability can prove to be fruitful in the financial problems.

2.1 GA
Genetic algorithm was first introduced by John Holland to model the natural evolution development of highly complex, highly fitted organisms from lower to complex one. GAs are stochastic search techniques that can search large and complicated spaces on the ideas from natural genetics and evolutionary principles [24][25]. These are particularly suitable for multi-parameter optimization problems with an objective function subject to numerous hard and soft constraints. The operation is targeted to individual groups of all through choice of crossover and mutation operators producing new generations of groups, until the results are satisfactory. In the solution space, the genetic algorithm has more random search of solutions to identify the optimal solution. Due to the random genetic algorithm, a search of all solutions is possible, so they can find the global optimum. Therefore, the canonical GA can be described as an algorithm to turns one population of candidate encoding to corresponding solutions into another using a number of stochastic operators [26].

2.1.1Canonical GA
The outline of the operations and the flow chart of the canonical GA is described in the below. The key steps in the algorithms are:

a. Determine how the solution is to be encoded as a string, and determine the definition of the fitness function.
b. Construct an initial population, possibly randomly of n encodings corresponding to the candidate solutions to a problem.
c. Decode each string into a solution, and calculate the fitness of each solution candidate in the population.
d. Implement a selection process to select a pair of encodings corresponding to candidate solutions to a problem (the parents) from the existing population, biasing the selection process in favour of the encodings corresponding to better/fitter solutions.
e. With a probability P_{cross}, perform a crossover process on the encodings of the selected parent solutions, to produce two new (child) solutions.
f. Apply a mutation process, with probability P_{mut}, to each element of the encodings of the two child solutions.
g. Store the encodings corresponding to the child solutions in the new (next generation) population.
h. Repeat steps (d)-(g) until n coding of candidate solutions have been created in new population. Then discard the old population. This constitutes a generation.
i. Go to step (c) and repeat until the desired population fitness level has been reached or until a predetermined number of generations have elapsed.

The flowchart of the canonical GA is shown in Figure 1 below:

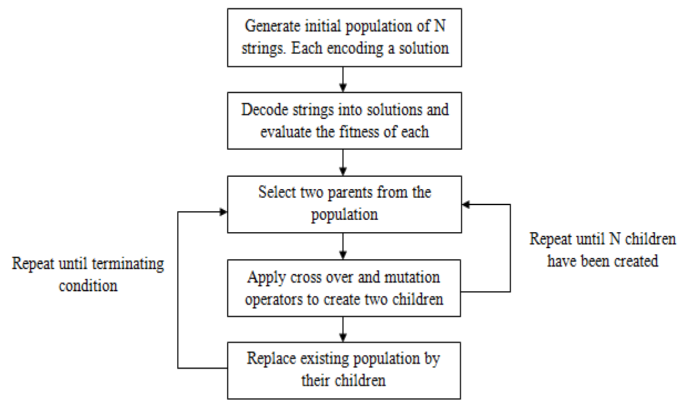

Figure 1. Flowchart of the canonical genetic algorithm

In genetic algorithm, the implicit parallelism plays a vital role finding the optimal solution. Here, this concept is also used for the prediction purpose while it is being used for learning of the RBFNN which has been discussed in the subsequent sections.

2.2 RBFNs
Radial basis function (RBF) neural network was proposed by Broomhead and Lowe [27], which is different from sigmoidal activation function used as basic function in the hidden layer of the neural network, locally responsive to input stimulus. RBF are embedded in a two layer neural network, where each hidden unit implements a radial activated function. The output units implement a weighted sum of hidden unit outputs. The input into a RBF neural network is nonlinear, the output is often linear. Their excellent approximation capabilities have been studied by Park and Sandberg [28]. Owing to their nonlinear approximation properties, RBF networks are able to model complex mappings, indicating that neural networks can only model by means of multiple intermediary layers.

2.2.1 Radial Basis Function Network Model
The RBF network topological structure is shown in Figure 2. The network consists of three layers, namely input layer, radial basis function hidden layer and output layer. The input part does not transform the signals but only dispatches the input vector to the radial basic layer. The function in a hidden layer node (also called nucleus function) responds

partly to the input signals, i.e. when the input function is close to the centre range of the nucleus function, the hidden layer will produce a large output. The output layer makes output values through a linear combination of outputs from the hidden layer.

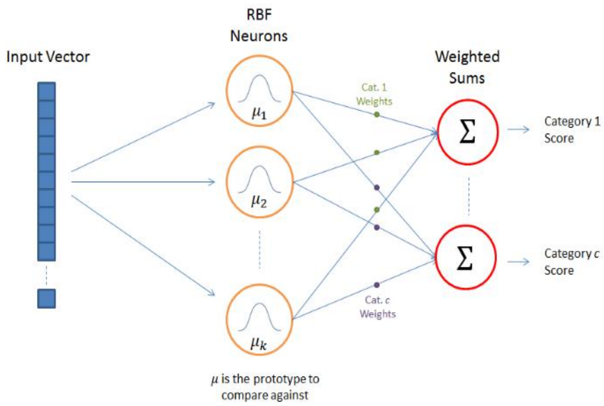

Figure 2. Structure of RBF neural network

A general block diagram of an RBF network is illustrated in Figure. 3 as shown below:

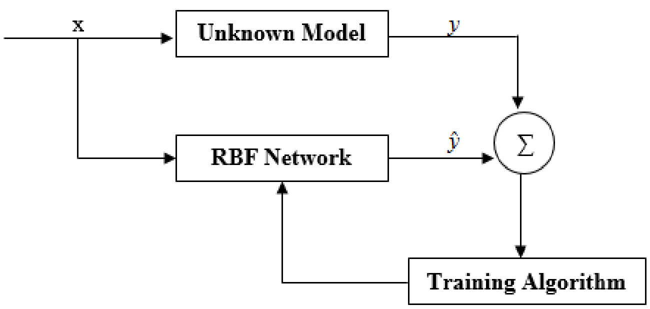

Figure 3. Block diagram of an RBF network

In RBF networks, the outputs of the input layer are determined by calculating the distance between the network inputs and hidden layer centres. The second layer is the linear hidden layer and outputs of this layer are weighted forms of the input layer outputs. Each neuron of the hidden layer has a parameter vector called centre. Here input vector $X=[x_1,x_2,.....x_N]$; C_i -the centre of RBF neural network, a constant vector with the same dimension as X; N - the dimension of the input vector; M -neurons number of the hidden layer; $\varphi(.)$ –radial basis function; $\|X-C_i\|$ - Euclidean distance between X and C_i ; j -output node, $j=1,2.....P$; W_{ij} - the weight value which connected the i^{th} hidden node with the j^{th} output node.

As shown in Figure 2, ideal output $y_j(j=1,2,.....p)$, the actual output $\hat{y}_j$ and the weight value of the output layer w_{ij} can be obtained by the RBF neural network. Choosing Gaussian function $\varphi_i(x)=\exp[-\|x-c_i\|^2/(2\sigma^2)]$ as radial basis function, the actual output $\hat{y}_j$ is calculated by the following formula:

$$\hat{y}_j=\sum_{i=1}^{m} w_{ij}\varphi_i(x)=\sum_{i=1}^{m} w_{ij}\exp[-\|x-c_i\|^2/(2\sigma^2)] \qquad (1)$$

Then, the weight value w_{ij} is adjusted to satisfy the following formula, from which the final result of the RBF neural network can be obtained.

$$\mathrm{E}=\sum_{j=1}^{p}(y_j-\hat{y}_j)^2=\sum_{j=1}^{p}(y_j-\sum_{i=1}^{m} w_{ij}\varphi_i(x))^2 \qquad (2)$$

Where, where m is the number of neurons in the hidden layer ($i\in(1,2,.....m)$) ; p represents he number of neurons in output layer($j\in(1,2,.....p)$) ; w_{ij} is the Weight of the i^{th} neuron and j^{th} output, φ_i is the radial basis function, C_i is the centre vector of the i^{th} neuron, and $\hat{y}_j$ is the network output of j^{th} neuron. Hence the general block diagram of an RBF network is presented in Figure 3.

The basic algorithm for RBFN architecture is as follows:

a. Select the initial number of centres (m).
b. Select the initial location of each of the centres in the data space.
c. For each input data vector/centre pairing, calculate the activation value $\varphi(\|x-y\|)$; where φ is a radial basis function and ‖…‖ is a distance measure between input vector x and a centre y in the data space. As an example, let d=‖x-y‖. The value of a Gaussian RBF is then given by $y=\exp(-d^2/2\sigma^2)$ Where, σ is a modeler selected parameter which determines the size of the region of input space to which a given centre will respond.
d. Once all the activation values for each input vector have been obtained, calculate the weights for the connections between the hidden and output layers using linear regression.
e. Go to step (c) and repeat the above steps until a stopping condition is reached.
f. Improve the fitness of the RBFN to the training data by adjusting some or all of the following: the

number of centres, their location, or the width of the radial basis functions.

From the above steps, it is seen that, substantial modeler involvement is required in order to construct a quality RBFN. Another thing is, selections of a quality set of model inputs are required. Both of these steps represent a combinatorial problem and the solution of this problem can be largely or partly automated through the application of an evolutionary algorithm such as GA.

2.2.2 Learning in RBFN

Training of RBFN requires optimal selection of the parameters vectors c_i and w_i , i=1…h. Both layers are optimized using different techniques and in different time scales. Here, one of the most popular approaches, called gradient descent technique is used to update *c* and *w*. The update rule for center learning is:

$$c_{ij}(t+1) = c_{ij}(t) - \eta_1 \frac{\partial E}{\partial c_{ij}}, \quad \text{for i=1… } p, j\text{=1….h} \tag{3}$$

The weight update law is:

$$w_i(t+1) = w_i(t) - \eta_2 \frac{\partial E}{\partial w_i} \tag{4}$$

where the cost function is,

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

The actual response is

$$y = \sum_{i=1}^{n} \varphi_i w_i \tag{6}$$

Where $z_i = \|X - C_i\|$, σ is the width of the centre. Differentiating E w.r.t w_i , we get

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y} \times \frac{\partial y}{\partial w_i} = -(y^d - y)\varphi_i \tag{7}$$

Differentiating E w.r.t. $C_{ij,}$ we get

$$\frac{\partial E}{\partial c_{ij}} = \frac{\partial E}{\partial y} \times \frac{\partial y}{\partial \varphi_i} \times \frac{\partial \varphi_i}{\partial c_{ij}}$$

$$= -(y^d - y) \times w_i \times \frac{\partial \varphi_i}{\partial z_i} \times \frac{\partial z_i}{\partial c_{ij}}$$

Now, $\frac{\partial \varphi_i}{\partial z_i} = -\frac{z_i}{\sigma^2}\varphi_i$ and $\frac{\partial z_i}{\partial c_{ij}} = \frac{\partial}{\partial c_{ij}}(\sum_j (x_j - c_{ij})^2)^{1/2}$

$$= -(x_j - c_{ij})/z_i$$

After simplification, the update rule for centre learning is:

$$c_{ij}(t+1) = c_{ij}(t) + \eta_1 (y^d - y) w_i \frac{\varphi_i}{\sigma^2}(x_j - c_{ij}) \tag{8}$$

The update rule for the linear weights is:

$$w_i(t+1) = w_i(t) + \eta_2 (y^d - y)\varphi_i \tag{9}$$

III. METHODOLOGY

3. Learning of RBFNs using GA

In this paper GA uses real-coded concept, makes C_i (radial basis function centre, σ_i (variance to RBF), and the right to export unit value of RBF network as multi parameter [27]. If the binary coding concept will be used here, then the coding string become too long and the string will be translated to the real and the fitness is calculated. So, real coding concept is used in this paper as shown in Figure 4. The right to export unit, radial basis function centre and the variance to the RBF of chromosome are assigned to the network structure and is arranged by certain order. The training samples were seen as the input and output of the network.

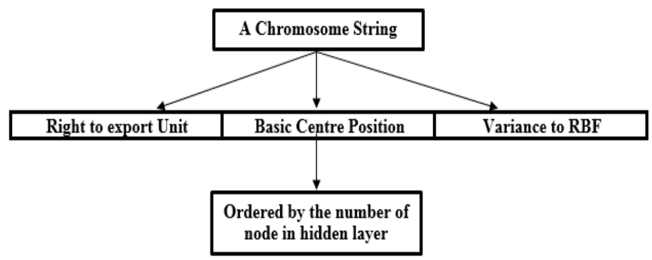

Figure 4. Real coding using Genetic Algorithm

3.1 Proposed Model for Learning of RBFNs using GA

Here, the overall architecture of the proposed model considered for this paper for the development of the GA optimized RBFNs financial modelling is shown in Figure 5. The proposed algorithm of GA-RBFNs consists of three phases.

<u>Phase-I:</u>

Here the, GA searches optimal or near optimal connection weights and thresholds for feature discretization. The populations, the connection weights and the thresholds for feature discretization, are initialized into random values before the search process. The parameters for searching are encoded on chromosomes. This study needs three sets of parameters. The first set is the set of connection weights between the input layer and the hidden layer of the network. The second set is the set of connection weights between the hidden layer and the output layer. The third set represents the thresholds for feature discretization.

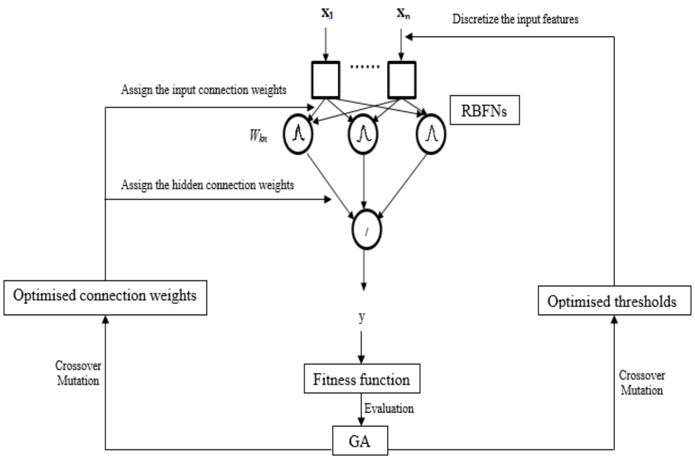

Figure 5. GA optimized RBFNs architecture for financial modeling

Here the following encoding has been done using the strings. This study uses 11 input features and employs 11 processing elements in the hidden layer. Each processing element in the hidden layer receives 11 signals from the input layer. The first 121 bits represent the connection weights between the input layer and the hidden layer. These bits are searched from -5 to 5. Each processing element in the output layer receives a signal from the hidden layer. The next 11 bits indicate the connection weights between the hidden layer and the output layer. These bits also varied between -5 and 5. The following 44 bits are the thresholds for feature discretization. Each feature is discretized into at most five categories and needs four thresholds for discretization. In addition, GA also searches the number of categories to be discretized using these bits. The thresholds are not used if the searched thresholds are more than the maximum value of each feature. The upper limit of the number of categories is five and the lower limit is one. This number is automatically determined by the searching process of GA.

The encoded chromosomes are searched to maximize the fitness function. The fitness function is specific to applications. In this study, the objectives of the model are to approximate connection weights and the thresholds of discretization for the correct solutions. These objectives can be represented by the average prediction accuracy of the training data. This study applies the average prediction accuracy of the training data to the fitness function. The parameters to be searched use only the information about training data. In this phase, GA operates the process of crossover and mutation on initial chromosomes and iterates until the stopping conditions are satisfied. For the controlling parameters of the GA search, the population size is set to 100 organisms and the crossover and mutation rates are varied to prevent RBFNs from falling into a local minimum. The range of the crossover rate is set between 0.5 and 0.7 while the mutation rate ranges from 0.05 to 0.1. As the stopping condition, only 5000 trials are permitted.

Phase –II

The second phase is the process of computation in RBFNs. In this phase, the Gauss function ($\varphi_i(x) = \exp[-\|x - c_i\|^2 / (2\sigma^2)]$) has been chosen as radial basis function or the activation function. Here, one of the most popular approaches called gradient descent technique is used to update c and w. The update rule for the linear weights as shown in equation 9 is used as a combination function for the RBFNs computation with derived connection weights from the first phase.

Phase–III

The derived connection weights and thresholds for feature discretization are applied to the holdout data. This phase is indispensable to validate the generalizability because RBFNs has the eminent ability of learning the known data. If this phase is not carried out, then the model may fall into the problem of over fitting with the training data. So, the algorithm of the GA-RBFNs can be stated as follows:

Algorithm for GA-RBFN

a. Initialize the populations and set evolution generation i.e., gen=0.
b. Give the input signal in the input layer and forwards this signal to all processing elements in the hidden layer.
c. Then sum its weighted input signals in the hidden layer and applies the gauss function to compute its output signal of the hidden processing element and forwards it to the output layer.
d. Then sums its weighted signals from the hidden layer and applies the radial basis function to compute its output signal of the output processing element and computes the difference between the output signal and the target value.
e. Then Calculate fitness function.
f. Then select the individuals to become parents of the next generation. Then perform crossover and mutation of these individual and see that new generation is achieved. Then increment the gen as gen=gen+1.
g. Then stop the execution, If the prediction accuracy is satisfied, otherwise goto step (b).

IV. Results and Discussion

Even though the proposed GA-RBFNs algorithm is primarily intended for prediction of stock price index with large number of records and a moderate number of inputs, it can also be used very well on more conventional datasets. To exhibit this fact we have evaluated our algorithm using a dataset consists of historical prices of the daily closing price, opening price, and lowest value in the day, highest value in the day and the total volume of stocks traded in each day obtained from the yahoo finance website:

http://finance.yahoo.com/. We have collected daily closing price of the index in each day values for DJIA starting from 24 July 2000 to 22 October 2007 for a period of seven years which amounted to about 408 data points (shown in Figure 6). Since the main data structures in GA are chromosomes, phenotypes, objective function values and fitness values, the MATLAB package as a numerical tool has been used in our study for easy implementation. The experiments were carried out on a personal computer having, 2.30 GHz, Core i5-2410M Intel CPU with 4.0 GB RAM in a 64-bit Operating System using MATLAB 2010b.

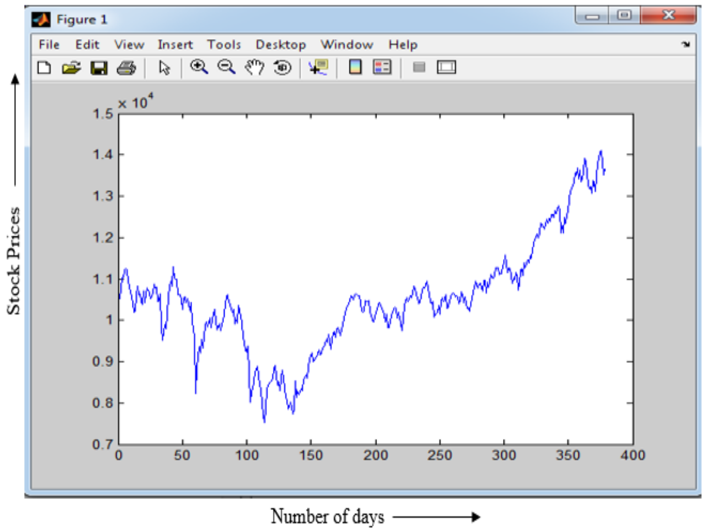

Figure 6. Graph showing daily price change of DJIA from 24 July 2000 to 22 October 2007

Table 1 shows the dataset used for the training and out-of-sample validation for the proposed model of financial modelling. The directions of daily change in the stock price index are categorized as '0' or '1'. '0' means that next day's index is lower than today's index, and '1' means that next day's index is higher than today's index. The 11 potential indicators are taken as the input subsets by the review of domain experts and our previous findings as mentioned earlier. To be specific, every row contains eleven attributes of different potential indicators extracted from their index price as i) 10-dayEMA, ii)20-day EMA, iii)30–day EMA, iv) ADO, v)STI, vi)RSI9,vii)RSI14 ,viii)PROC27,ix)CPACC, x)HPACC, and xi)10-day William's %R.

4.1 Defining Fitness Function

The following fitness function has been used for the development of the GA-RBFNs based financial modelling.

$$F(x) = 1/[1+(\sum_{i=0}^{n}[y_i - \hat{y}_i]^2)/n \quad (10)$$

and

$$\delta = (\sum_{i=0}^{n}[y_i - \hat{y}_i]^2)/n \quad (11)$$

where, y_i is the actual output, $\hat{y}_i$ is the predicted output, and x is the chromosome. The fitness function $F(x)$ and prediction error function δ justifies the optimal output.

4.2 Setting GA Parameters

It is seen that crossover P_{cross} and mutation rate p_{mut} affects the genetic algorithm. If p_{mut} is bigger, the speed of the new entity is faster, causing to the damage of the genetic model. If P_{cross} is bigger, the genetic algorithm will become random search algorithm causing P_{cross} to too small to generate a new individual. Therefore the crossover and mutation rate is adjusted by the following equations.

$$P_{mut1} = 0.99 \times (1 - g_{en}/G) \quad (12)$$

$$P_{mut2} = 0.4 \times (1 - g_{en}/G) \quad (13)$$

$$P_{cross1} = 0.3 \times (1 - g_{en}/G) \quad (14)$$

$$P_{cross2} = 0.01 \times 1 - g_{en}/G) \quad (15)$$

$$P_{mut} = \begin{cases} p_{mut1} - \dfrac{[(P_{mut1} - P_{mut2}) \times (f - f_{avg})]}{[f_{max} - f_{avg}]} & f \geq f_{avg} \\ P_{mut1} \cdots\cdots & f < f_{avg} \end{cases} \quad (16)$$

and,

$$P_{cross} = \begin{cases} p_{cross1} - \dfrac{[(P_{cross1} - P_{cross2}) \times (f_{max} - f)]}{[f_{max} - f_{avg}]} & f \geq f_{avg} \\ P_{cross1} \cdots\cdots & f < f_{avg} \end{cases} \quad (17)$$

where, g_{en} is the present multiply generation and G is a constant $(G \geq g_{en})$, f_{max} is the max fitness of the group and f_{avg} is the average fitness of the group, f is the bigger fitness of the two crossover individual and f is the fitness of the variation individual. It is known that the fitness function determines the crossover and mutation rate for good performance of individual and adaptive adjustment determines the poor performance of the individual having bigger crossover and mutation rate. The summary of the GA characteristics is presented in Table 2.

4.3 Setting GA-RBFNs network parameters

The following network parameters have been set for the proposed hybridized GA-based-RBF neural network for financial modelling (Table 3).

Table 1. Dataset showing calculated closing prices of DJIA using potential indicators

Date	10-day EMA	20-day EMA	30-day EMA	ADO	STI	RSI9	RSI14	PROC27	CPACC	HPACC	14-day William's %R	Target
24-Jul-00	10610.62	10635.57	10671.51	-4172125959	31.67	44.87	44.48	-6.58	-6.58	-7.93	-68.33	10511.17
31-Jul-00	10639.19	10648.16	10677.72	-3605174791	65.52	52.09	48.48	0.27	0.27	-5.45	-34.48	10767.75
7-Aug-00	10709.84	10684.30	10700.30	-3080044746	89.28	58.31	52.24	0.58	0.58	-0.86	-10.72	11027.80
14-Aug-00	10771.04	10718.78	10722.63	-3152317366	82.60	58.75	52.51	5.96	5.96	0.84	-17.40	11046.48
21-Aug-00	10847.68	10763.89	10752.94	-2848911465	90.37	62.20	54.66	9.52	9.52	4.35	-9.63	11192.63
28-Aug-00	10918.79	10809.10	10784.28	-2789847284	85.87	63.30	55.34	13.96	13.96	9.28	-14.13	11238.78
5-Sep-00	10973.66	10848.28	10812.42	-3072360584	78.03	62.50	54.99	8.23	8.23	8.85	-21.97	11220.65
11-Sep-00	10965.18	10855.77	10819.81	-3835525068	56.40	50.82	49.50	10.05	10.05	8.06	-43.60	10927.00
18-Sep-00	10943.76	10854.97	10821.59	-3562077581	50.53	48.08	48.10	2.38	2.38	1.88	-49.47	10847.37
25-Sep-00	10890.52	10835.55	10810.58	-3896228134	36.06	41.82	44.73	-4.16	-4.16	-2.40	-63.94	10650.92
2-Oct-00	10837.08	10812.79	10796.78	-4365883819	24.13	40.19	43.81	-2.98	-2.98	-2.65	-75.87	10596.54
9-Oct-00	10719.84	10753.71	10757.78	-4691896795	19.36	30.31	37.65	-8.27	-8.27	-6.67	-80.64	10192.18
16-Oct-00	10630.16	10703.53	10723.52	-4049375090	33.64	31.91	38.44	-0.77	-0.77	-10.10	-66.36	10226.59
23-Oct-00	10622.97	10692.78	10714.95	-3271854934	52.34	46.54	46.24	-2.34	-2.34	-2.24	-47.66	10590.62
30-Oct-00	10658.42	10704.70	10721.59	-3230829414	64.01	53.55	50.46	0.78	0.78	-1.39	-35.99	10817.95
6-Nov-00	10648.34	10695.01	10713.94	-3880839128	52.97	46.99	46.72	0.24	0.24	1.37	-47.03	10602.95
13-Nov-00	10644.98	10688.81	10708.52	-3658941002	54.35	47.89	47.25	0.19	0.19	0.77	-45.65	10629.87
20-Nov-00	10613.21	10668.00	10693.15	-3692741908	46.15	43.03	44.45	-1.47	-1.47	-3.42	-53.85	10470.23
27-Nov-00	10569.64	10639.97	10672.53	-4063869856	41.19	40.24	42.79	0.72	0.72	0.37	-58.81	10373.54
4-Dec-00	10595.68	10646.91	10675.14	-3842080770	58.62	52.41	49.86	-0.76	-0.76	0.29	-41.38	10712.91

Table 2. Parameters used for Genetic Algorithm

Population Size	40
Maximum Generations	200
Selection Types	Tournament Selection
Crossover Probability	0.9
Mutation Probability	0.1
Fitness Function	$\delta = (\sum_{i=0}^{n}[y_i - \hat{y}_i]^2)/n$

Table 3. Parameters used for GA-RBFNs

Number of Neurons	42
Minimum Radius	0.02809
Maximum Radius	3072.47
Minimum Lambda	0.02059
Maximum Lambda	9.24719
Regularization lambda for final weights= 6.1561e-009	

4.4 Statistical Performance Evaluation

The error measures considered to evaluate the performance of the experiments are mean squared error (MSE) and mean absolute percentage error (MAPE) as shown below.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \qquad (18)$$

where, $\hat{y}_i$ vector of n predictions and y_i is the vector of true values.

$$MAPE = \frac{1}{n}\sum_{j=1}^{n}\left|\frac{Y_j - \hat{Y}_j}{Y_j}\right| \times 100 \qquad (19)$$

where, $\hat{y}_i$ vector of n predictions and y_i is the vector of true values and n is the number of data.

4.5 Results Analysis

Table 4 represents the experimental results carried out in this research and are found to be encouraging. It is observed that the MLP model trained with BP algorithm (MLP-BP) and GA method have reached good results both in training and validating the dataset. However, cross-fertilized GA-RBFN model shows the outstanding performance in all the considered performance measuring factors like R^2 value, Correlation, RMSE, MSE, MAE, and MAPE between training and validating. Hence it is concluded that GA optimised RBFN model is the most acceptable one than the other two models by considering the higher R^2 value and lower MAPE value which indicates the better financial prediction model as compared to others. The graph showing actual vs. predicted values and error rate vs. number of neurons of the outstanding GA-RBFN model is shown in Figures 7 and 8, respectively.

Table 4. Results showing comparison status of different experiments

Sl No	Analysis of Variance	MLP-BP Model (11:5:1)		GA Model		Cross-fertilized GA-RBFNs model	
		Training	Validation	Training	Validation	Training	Validation
1	R^2	0.9937	0.9926	0.9925	0.9910	0.9995	0.9987
2	Correlation	0.9969	0.9951	0.9963	0.9955	0.9997	0.9994
3	RMSE	111.93	123.69	114.95	125.54	30.14	46.60
4	MSE	10698.05	13623.99	13214.02	15762.06	908.32	2177.48
5	MAE	80.57	92.78	81.32	88.35	22.75	30.64
6	MAPE	0.8001	0.8021	0.8060	0.8800	0.2228	0.3028

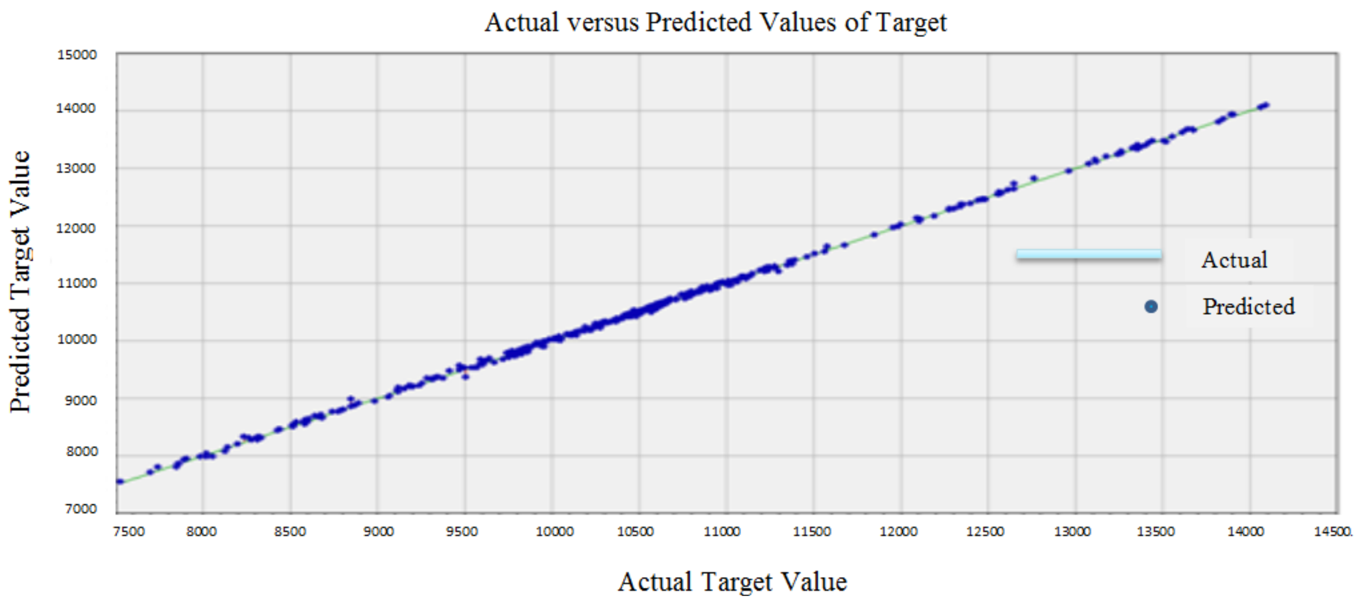

Figure 7. Actual versus predicted values of target by GA-RBFNs model

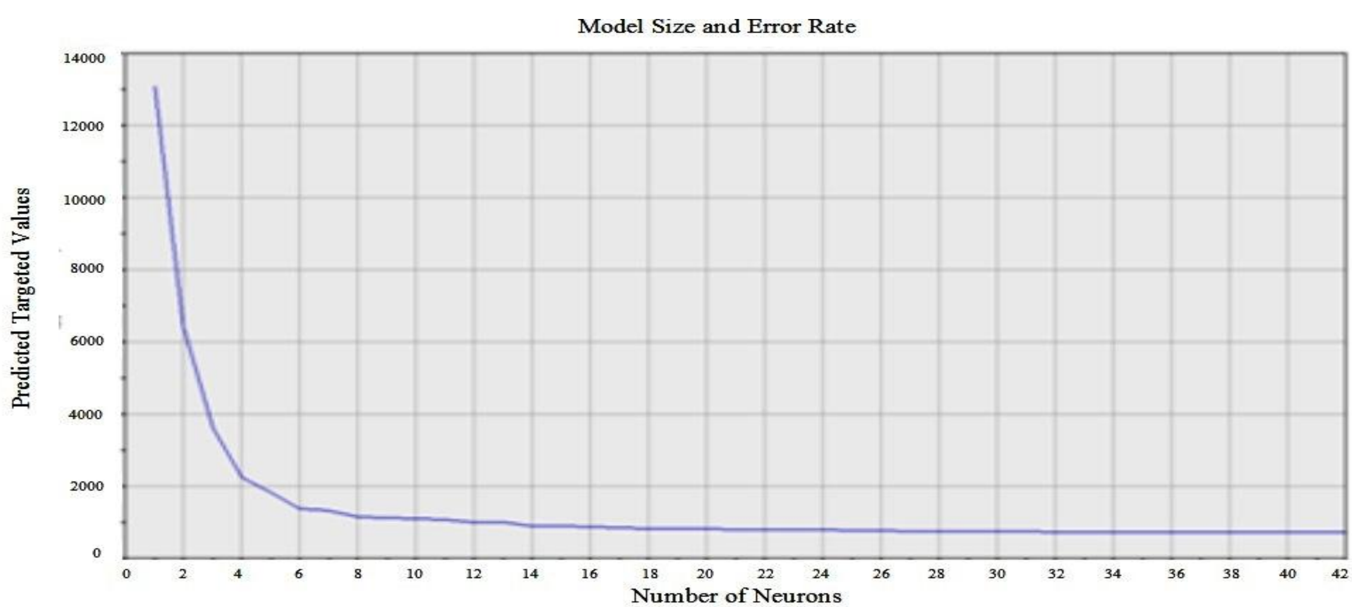

Figure 8. Graph showing error rates vs. number of neurons of proposed GA-RBFN model

V. Conclusion and Future Scope

In this research, the objective was to develop a GA optimised RBFNs model for prediction of stock price index and study its performance as compared to conventional models. It was found that the combination of genetic algorithms with the Radial basis function neural networks model has higher and remarkable accuracy as compared to MLP-BP and GA model. The future research includes extensive study on other factors besides the factors considered in this paper for stock index prediction. The next step in future works is to integrate PSO with RBFNs for index prediction of financial domain. The application of hybrid systems seemed to be well suited for forecasting of financial data.

References

[1] S. Chen, C.F.N. Cowan and P.M. Gant, "*Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks*", J. IEEE Transactions on Neural Networks, Vol. 3, pp.302-308, 1991.

[2] Y. Chao-gang, Y. Yi-bin, W. Jian-ping, N. Jamal and Y. Jia, *"Determining Heating Pipe Temperature in Greenhouse Using Proportional Integral Plus Feed-forward Control and Radial Basis Function Neural Networks*", Journal of Zhejiang University-SCIENCE A, Vol. 6, No.4, pp. 265-269, 2005.

[3] J.H. Holland, "*Adaptation in Natural and Artificial Systems*", University of Michigan Press, Ann Arbor, 1975.

[4] P. S. Mishra and S. Dehuri, "*Potential Indictors for Stock Index Prediction: A Perspective"*, International Journal of Electronic Finance, Vol. 6, No. 2, pp. 157-183, 2012, doi:10.1504/IJEF.2012.048465.

[5] P. S. Mishra and S. Dehuri, "*Potential Indicators based Neural Networks for Cash Forecasting of an ATM*, International Journal of Information Systems and Social Change, Vol. 5, No.4, pp. 41-57, 2014, doi:10.4018/ijissc.2014100103.

[6] A. Brabazon, "*Financial time series modeling using neural networks: An assessment of the utility of a stacking methodology*", in proceedings of AICS 2002, Lecture Notes in Artificial Intelligence(2464),(Eds,) O'Neill et al., Springer,pp.137-144, 2002b.

[7] K. Nygren, "*Stock Prediction—A Neural Network Approach"*, Master's Thesis, Royal Institute of Technology, KTH, Sweden, 2004.

[8] E. Altman, "*Financial ratios, discriminant analysis and the prediction of corporate bankruptcy*", The Journal of Finance, Vol. 23, pp.589-609, 1968.

[9] E. Altman, "*Corporate financial distress-A complete guide to predicting, avoiding and dealing with bankruptcy*", New York: John Wiley, 1983.

[10] J. Ohlson, "*Financial ratios and the probabilistic prediction of bankruptcy*", Journal of Accounting Research, Vol. 18, No.1, pp.109-131, 1980.

[11] M.E. Zmijewski,"*Methodological Issues Related to the Estimated of Financial Distress Prediction Model*", Journal of Accounting Research, Vol.22, No.1, pp.59-82,1984.

[12] R. Barniv, A. Agarwal and R. Leach, "*Predicting the outcome following bankruptcy filing: A three-state classification using neural networks*", Intelligent Systems in Accounting, Finance and management, Vol.6, pp. 177-194, 1997.

[13] J. Boritz and D. Kennedy, "*Effectiveness of neural network types for prediction of business failure*", Expert Systems with Applications, Vol.9, pp.503-512, 1995.

[14] A. Brabazon and M. O'Neill, M, "*Diagnosing Corporate Stability using Grammatical Evolution*", International Journal of Applied Mathematics and Computer Science, Vol. 14, No. 3 pp.363-374,2004.

[15] F. Varetto, "*Genetic algorithms in the analysis of insolvency risk*", Journal of Banking and Finance, Vol. 22, No.10, pp.1421-1439, 1998.

[16] A.M. Colin, "*Genetic algorithms for financial modeling*", In Deboeck, G.J. (Eds.), Trading on the Edge, New York: John Wiley, pp. 148-173, 1994.

[17] G.J. Deboeck, "*Using GAs to optimize a trading system*", In Deboeck, G.J (Eds.), Trading on the Edge, New York: John Wiley, pp. 174-188, 1994.

[18] S. Mahfoud and G. Mani, "*Financial forecasting using genetic algorithms*", Applied Artificial Intelligence, Vol. 10, No.6, pp. 543-565, 1996.

[19] E. Rutan,, "*Experiments with optimal stock screens*", Proceedings of the 3rdInternationalConference on Artificial Intelligence Applications on Wall Street, pp.262-273,1993.

[20] J. Kingdom and K. Feldman, "*Genetic algorithms for bankruptcy prediction",* Search Space Research Report, No.01-95, Search Space Ltd., London, 1995.

[21] R. Walker, E. Haaasdijk and M. Gerrets, "*Credit evaluation using a genetic algorithm*", In Coonatilake, S. and Treleaven, P. (Eds.), Intelligent Systems for Financial and Business, Chichester, John Wiley, pp.35-59, 1995.

[22] N. Packard, "*A genetic learning algorithm for the analysis of complex data*", Complex Systems, Vol. 4, pp.543-572, 1990.

[23] M. Durairaj and T. Sathyavathi, "*Applying Rough Set Theory for Medical Informatics Data Analysis*", International Journal of Scientific Research in Computer Science and Engineering, Vol.1, No.5,pp.1-8, 2013.

[24] L. Davis, "*Handbook of Genetic Algorithms",* Van Nostrand Reinhold, New-York, 1991.

[25] D.E. Goldberg, "*Genetic Algorithms in Search, Optimization and Machine Learning",* MA: Addison-Wesley, 1989.

[26] A. Brabazon and M. O' Neill, "*Biologically Inspired Algorithms for Financial Modelling"*, Springer, Berlin, Germany, 2006.

[27] D.S. Broomhead and D. Lowe, "*Multivariable functional interpolation and adaptive networks*", Complex System, Vol. 2, pp.321-355, 1988.

[28] J. Park and I. W. Sandberg, "*Universal Approximation using Radial-Basis-Function Networks"*, Neural Computation, Vol. 3, pp.246-257, 1991.

Authors Profile

Partha Sarathi Mishra is an Assistant Professor in the Department of Computer Science at North Orissa University, Baripada, Odisha, India. He received his Master degree in Computer Applications from North Orissa University, M. Tech degree in Computer Science from Fakir Mohan University, and Ph.D. degree in Computer Science and Information Technology from North Orissa University respectively. His research interests include Neural Networks, High Order Neural networks, Evolutionary Algorithms and creation of computational intelligent systems that incorporate robustness, adaptation and creativity in their approaches for solving computationally intensive financial problems. He has already published more than 17 research papers in reputed journals and referred conferences, and has published one text book for undergraduate and post graduate students.

