I.J. Intelligent Systems and Applications, **2017, 8, 71-85**
Published Online August 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2017.08.08

Development and Performance Evaluation of Adaptive Hybrid Higher Order Neural Networks for Exchange Rate Prediction

Sarat Chandra Nayak
Kommuri Pratap Reddy Institute of technology, Department of Computer Science & Engineering,
Ghatkesar, R.R. Dist.-500088, Hyderabad, India
E-mail: saratnayak234@gmail.com

Received: 28 February 2017; Accepted: 10 April 2017; Published: 08 August 2017

***Abstract*—**Higher Order Neural Networks (HONN) are characterized with fast learning abilities, stronger approximation, greater storage capacity, higher fault tolerance capability and powerful mapping of single layer trainable weights. Since higher order terms are introduced, they provide nonlinear decision boundaries, hence offering better classification capability as compared to linear neuron. Nature-inspired optimization algorithms are capable of searching better than gradient descent-based search techniques. This paper develops some hybrid models by considering four HONNs such as Pi-Sigma, Sigma-Pi, Jordan Pi-Sigma neural network and Functional link artificial neural network as the base model. The optimal parameters of these neural nets are searched by a Particle swarm optimization, and a Genetic Algorithm. The models are employed to capture the extreme volatility, nonlinearity and uncertainty associated with stock data. Performance of these hybrid models is evaluated through prediction of one-step-ahead exchange rates of some real stock market. The efficiency of the models is compared with that of a Radial basis functional neural network, a multilayer perceptron, and a multi linear regression method and established their superiority. Friedman's test and Nemenyi post-hoc test are conducted for statistical significance of the results.

***Index Terms*—**Higher Order Neural Network, Jordan Pi-Sigma Neural Network, Radial Basis Function, Pi-Sigma Neural Network, Functional Link Artificial Neural Network, Genetic Algorithm, Particle Swarm Optimization, Exchange Rate Prediction.

I. INTRODUCTION

Tremendous improvement in computational intelligence capabilities since last few decades has been enhanced the modeling of complex, dynamic and multivariate nonlinear systems. Soft computing methodologies which include Artificial Neural Network (ANN), Evolutionary Algorithms (EA), Genetic Algorithms (GA), and fuzzy systems have been applied successfully to the area data classification, financial forecasting, credit scoring, portfolio management, risk level evaluation and are found to be producing better performance. Artificial Neural Network has the analogy with the thinking capacity of human brain and thus mimicking it [1, 2]. The ANN can imitate the process of human behavior and solve nonlinear complex problems. These properties have made it popular and are commonly used in calculating and predicting complicated systems.

Multilayer perceptron (MLP) is a feed forward error back propagation ANN which contains at least one hidden layer of neurons. It has been established as most frequent used neural tool for solving many real applications. The ability of MLP to perform complex nonlinear mappings and tolerance to noise in financial time series has been well established. However, suffering from slow convergence, sticking to local minima are the two well known lacunas associated with it. In order to overcome the local minima, more number of nodes can be added to the hidden layers. Multiple hidden layers and more number of neurons in each layer also add more computational complexity to the network. Also, various feed forward and multilayer neural networks are found to be characterized with several drawbacks such as poor generalization, nonlinear input-output mapping capability as well as slow rate of learning capacity.

Higher Order Neural Networks (HONN) have fast learning properties, stronger approximation, greater storage capacity, higher fault tolerance capability and powerful mapping single layer trainable weights [3]. Higher order terms in HONN can increase the information capacity of the network. This representational power of higher order terms can help solving complex nonlinear problems with small networks as well as maintaining fast convergence capabilities. With the increase in order of the network, there may exponential growth in tunable weights in HONN and hence more computation time. However, there is a special type of HONN called as Pi-Sigma neural network (PSNN) using less number of weights has been introduced by Shin and Ghosh in 1992 [4].

Ghazali developed a HONN based financial forecasting model which performed superior as compared to conventional multilayer neural network based models

[5]. Knowles et al. in 2009 used HONNs with Bayesian confidence measure for prediction of EUR/USD exchange rates and observed that the simulation results are much better than multilayer approach based models [6]. However, there is exponential increase in the required higher order terms which may affect the network performance. But the PSNN developed by Shin and Ghosh in 1992 is able to avoid such increase in number of weight vectors along with the processing units [7]. Some recent applications of PSNN are found in [8-10]. The PSNN has been successfully employed solving several difficult problems including polynomial factorization [11], zeroing polynomials [12], classification [7, 13], time series forecasting [14, 15]. Yong Nie and Wei Deng proposed a hybrid genetic algorithm trained PSNN to resolve the function optimization problem [16]. They concluded that the hybrid method can have better search capability and faster than genetic algorithm. Epitropakis et al. has proposed a PSNN trained by distributed evolutionary algorithm which has superior performance [13]. Another novel hybrid HONN has been proposed by authors in [17] for the classification problem. This hybrid model exhibits good generalization capability and also improved classification accuracy. A Neuron-Adaptive Higher Order Neural-Network Model for automated financial data modeling has been suggested by Zhang et al. in 2002 [18]. Their model shown to be "open box" and as such is more acceptable to financial experts than classical closed box neural networks. This model is further shown to be capable of automatically finding not only the optimum model, but also the appropriate order for specific financial data.

Due to the lower complexity in computation, Functional Link Artificial Neural Network (FLANN) is also used in various fields of research. A research work led by Majhi et al. in 2012 demonstrates the implementation of Wilcoxon FLANN (WFLANN) model to predict FOREX rate [19]. Another investigation by Sahu et al. in 2014 used a model of FLANN-CRO for FOREX prediction and the experimental results depict the dominance of the model as compared to other models [20]. Yu et al. in 2005 used a non linear ensemble forecasting model, which is a combination of Generalized Linear Auto-Regression (GLAR) and ANN, to predict US Dollar exchange rate against German Marks, British Pound and Japanese Yen [21]. They concluded that the combined model outperformed each of the individual models i.e. GLAR and ANN.

Jordan Pi-Sigma Neural Network (JPSNN) is a HONN which posses characteristics from a Jordan Neural Network architecture and PSNN [22]. The JPSNN that managed to incorporates feedback connections in their structure and having the finer properties of PSNN is mapped to function variable and coefficient related to the research area. Some applications of JPSNN for prediction of temperature time series signals and data classification were found in articles [23-25].

Exchange rate prediction is relevant to all sorts of firms and interesting for international companies which want to decrease exchange exposure. The foreign exchange rates have an important role in the financial market as well as economy of a country. Its area of influence includes not only interest rate and inflation but also the economic stability of any country. While deciding the monetary policies of any country FOREX rates acts as a vital factor. Global economy also comes under the influence of FOREX rate. Various massive economic crises such as The Asian crisis of 1997-98, China's undervalued Yuan (1994-2004) and Japanese yen's gyrations from 2008 to mid-2013, portrays the influence of FOREX rate on global economy. Hence to maintain the national as well as international economic stability numerous research activities have been carried out in this area. Till now it is one of the most demanding fields of research due to the highly volatile nature of FOREX rate. The dependency of FOREX rates on various fundamental and technical factors such as, inflation, interest rate differentials, capital flows, technical support and resistance levels, and so on, is the root cause of its dynamic nature.

Traditional techniques such as autoregressive moving average (ARMA), autoregressive conditional heteroscedastic (ARCH), and generalized autoregressive conditional Heteroskedasticity (GARCH) models are pioneers in this field of research. But these models assumed that the time series data is static in nature and performance of these models in predicting the time series were very low.

In the expedition of enhancing the accuracy of the prediction evolutionary algorithms or optimization techniques such as GA, Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and so on, were used along with ANNs. Work found by Nenortaite and Simutis in 2004 and Zhao and Yang in 2009 implemented the Particle swarm optimization (PSO) algorithm to forecast the stock market index [26, 27]. Sermpinis et al. in 2013 investigated the performance of an adaptive radial basis neural network along with PSO in the field of FOREX rate forecasting [28]. The experimental result shows that the hybrid model performs better than other traditional models in terms of both statistical and trading efficiency. Another investigation lead by Chen and Zhang in 2013 used genetic algorithm based on Mendel's principle of evolution to predict the FOREX rate [29]. The outcome shows that the Mendel's-GA model works as a valuable tool for FOREX rate prediction in case of high-frequency data. A back propagation neural network was employed by Chang et al. in 2009 to predict the buy/sell points for a stock [30]. The study applied a case based dynamic window for improved forecast accuracy. Some compressive reviews of data mining applications in stock market forecasting were found in articles [31, 32]. It was observed that neural networks, neuro-fuzzy models and other soft computing techniques outperform conventional models in most of cases. The prediction of stock closing prices of NSE and SENSEX using hybrid ANN model of functional link fuzzy logic neural models was found in the article [33]. Forecasting currency exchange rates using an adaptive ARMA model with differential evolution based training was proposed by Rout et al. in 2013 [34]. A novel knowledge guided artificial neural network (KGANN) was proposed by Jena et al. in 2015 for exchange rate prediction [35]. The prediction results of their proposed models were compared with that of the individual FLANN and LMS based models and found better. Performance evaluation of ANN based model for exchange rate prediction was conducted by Svitlana Galeshchuk in 2016 [36]. The author claims the superiority of a MLP based model for one-step-ahead prediction of daily, monthly, and quarterly collected samples.

There also found some applications of ANN based models toward other areas of data mining. Nanda et al. proposed ARIMA, MLP, FLANN, and Legendre polynomial equation based models for prediction of rainfall in India and found that ANN based models were superior [37]. An evolving cascade model based on the neo-fuzzy nodes is proposed in [38]. The model can adjust both its architecture and parameters in an online mode, has simple computational implementation and can process data sets with a high speed.

The objective of this work is to develop hybrid adaptive models by considering various HONNs such as PSNN, SPNN, JPSNN, and FLANN as the base model. To circumvent the demerits of gradient descent based back propagation learning, authors of this study use two evolutionary leaning techniques such as PSO and GA to find out the optimal parameters of the HONN based forecasting models. Their performance are evaluated and analyzed separately for GA and PSO learning. These models are experimented on five real exchange series to see whether they are able to capture the nonlinearity and uncertainties associated with the stock data. Adaptive models are used which reduces the computational time significantly. Friedman’s test and Nemenyi post-hoc test are conducted for statistical significance of the results.

The article is organized as follows. A brief introduction about the necessities as well as applicability of HONNs was presented in section I. Also some related researches have been documented in this section. Section II describes about the neural networks used in this study such as PSNN, SPNN, JPSNN, FLANN, MLP, and RBFN. The methods and methodologies are presented by section III. The experimental setup, input design, result analysis and statistical significance test are presented in section IV. Section V gives the concluding remarks followed by a list of references.

II. Description of used Neural Network Models and Learning Techniques

This section gives a small introduction about the different neural models and learning techniques used in this article. The neural models include PSNN, SPNN, JPSNN, FLANN, and RBFN. Two learning methods used are GA and PSO.

A. Pi-Sigma Neural Network (PSNN)

The PSNN is a class of HONN and has architecture of fully connected two-layered feed forward network [39]. The input layers are connected to the first layer, i.e. summing layer and the output of this layer is feed to the second layer, i.e. product unit. The weight set between input and summing layer is adjustable and the weight set between summing and product unit is non-trainable and set to unity. Hence, this network having only one tunable weight set reduces the training time of the network substantially. The summing units use a linear activation where as the product unit uses a nonlinear activation function. Incorporation of each extra summing unit increases the order by one. The product units give the networks higher order capabilities by expanding input space into higher dimension space offering greater nonlinear separability without suffering the exponential increase in weights. A PSNN based forecasting model is shown by Fig. 1.

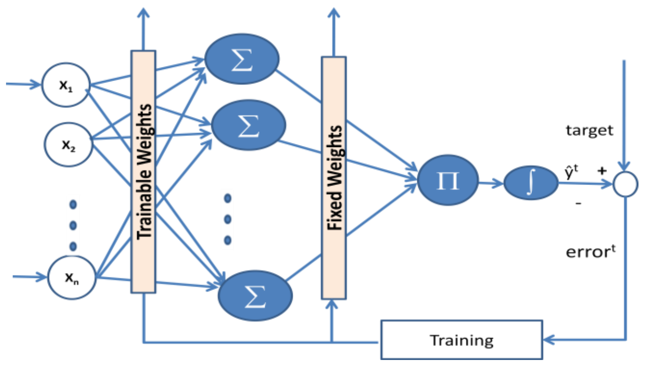

Fig.1. Pi-Sigma Neural Network based Forecasting Model

B. Sigma-Pi Neural Network (SPNN)

As compared to PSNN, this network use *sum of products* of input signals. It also contains one layer of tunable weights. A weight is applied not only to each input signal, but also to the possibly higher-order products or conjuncts of the input signals. In contrast to PSNN, here at each hidden unit, product of weighted signals are computed. However, the number of terms, and therefore the weights, increase rapidly which may results a combinatorial explosion in the number of weights. This problem can be avoided by restricting number of units which is sufficient to achieve the desired degree of accuracy using a prior knowledge about the task. A SPNN based forecasting model is shown by Fig. 2.

C. Jordan Pi-Sigma Neural Network (JPSNN)

The architecture of JPSNN is quite similar to PSNN. It combines the architecture of a recurrent neural net as well as PSNN. It has a special recurrent link from the output layer back to the input layer. This structure gives the temporal dynamics of the time-series process that allows the network to compute in a more parsimonious way [40]. The architecture of JPSNN based forecasting model is presented by Fig. 3.

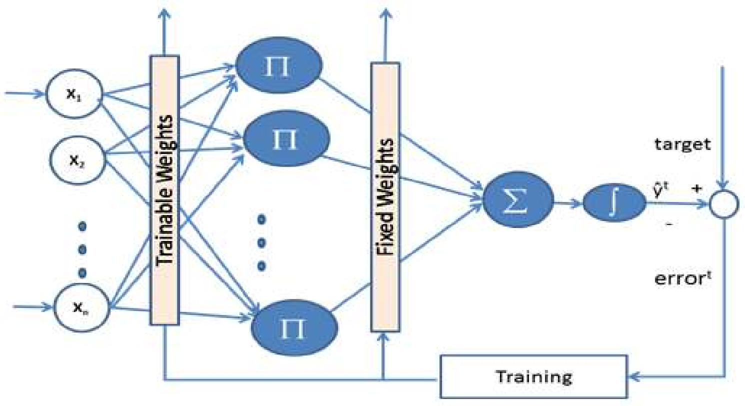

Fig.2. Sigma-Pi Neural Network based Forecasting Model

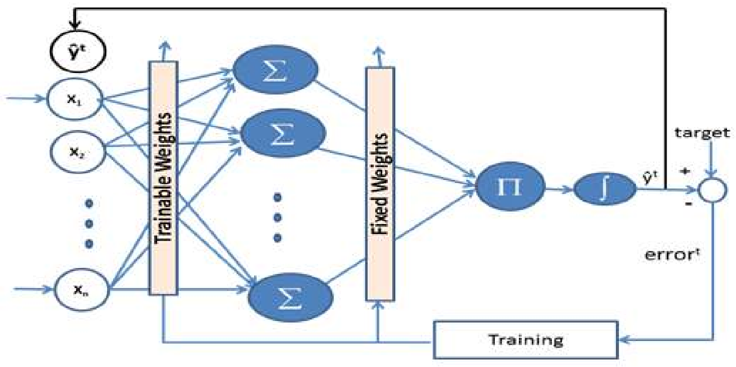

Fig.3. Jordan Pi-sigma Neural Network based Forecasting Model

D. Functional Link Artificial Neural Network (FLANN)

The FLANN architecture uses a single layer feed forward network without hidden layers originally proposed by Y. H. Pao in 1989 [41]. The input signal first goes through a functional expansion unit. The functional expansion effectively increases the dimensionality of the input vector and hence the hyper plane generated by the FLANN provides greater discrimination capability in the input pattern space. The FLANN based forecasting model is represented by Fig. 4. The adaptive algorithms are more easily used to train the network and have lower complexity because of the absence of any hidden layers. The functional expansion effectively increases the dimensionality of the input signal and generates hyper planes. These hyper planes provide greater discrimination capability in the input pattern space. FLANN has simpler structure, faster convergence, lower computational complexity and better nonlinear approximation capacity. The set of functions considered for functional expansion may not be always suitable for mapping the nonlinearity of the financial time series. In such cases few more functions may be incorporated. However dimensionality of many problems itself are very high and further increasing the dimensionality by to a very large extent may not be appropriate choice. So a small set of appropriate functions should be used in order to map the functions to the desired extent.

E. Radial Basis Function Network (RBFN)

Radial basis function network can be used for approximating functions and recognizing patterns. The network is a two layered network. In this network, each hidden unit of hidden layer implements a radial activation function and each output neuron of output layer implements a weighted sum of hidden units' output. This network is a special class of neural network in which the activation of a hidden neuron is determined by the distance between the input vector and a prototype vector. Prototype vectors refer to centers of clusters formed by the patterns or vectors in the input space. The interconnection between the hidden and output layer are made through a weighted connections. The output layer, a summation unit, supplies the response of the network to the outside world. The basic structure of a RBF neural network is shown in Fig. 5.

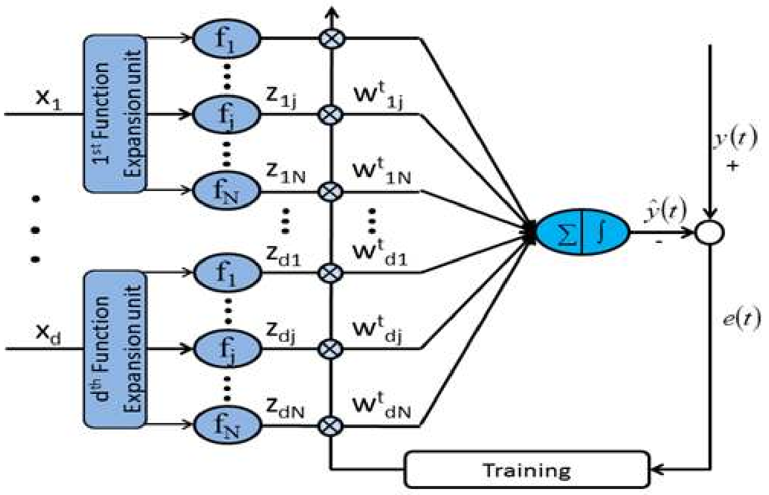

Fig.4. FLANN based Forecasting Model

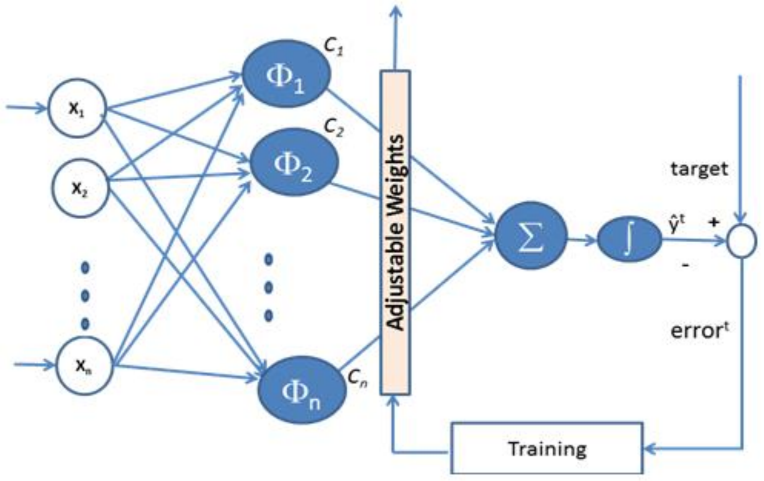

Fig.5. RBFN based Forecasting Model

F. Genetic Algorithm (GA)

GA is considered as a popular global search optimization, works on a population of probable solutions in the form of chromosomes attempt to trace the optimal solution all the way through the process of artificial evolution [42]. GAs are based on biological evolutionary theory and used to solve optimization problems which work with encoding parameter instead of parameter itself. It consists of repeated artificial genetic operations such as *evaluation*, *selection*, *crossover*, and *mutation*. In general the genetic evolution process consist the following basic steps:

- Initialization of the search space.
- Evaluation of fitness of individuals.
- Application of genetic operators for exploration as well as exploitation
- Repetition of the above steps until convergence.

The suitability of the best and average individual in each generation increases towards a global optimum.

G. Particle Swarm Optimization (PSO)

Particle swarm optimization (PSO) is a nature inspired metaheuristic technique [43] designed by mimicking the simulated social behavior of bird flocking, insects, fish schooling etc. which is capable to find the global best solution. Like other population based algorithm, PSO starts with a set of randomly generated initial swarms or particles each represents a candidate solution in the search space. PSO is similar to evolutionary computing techniques in that, a population of candidate/potential solutions to the problem under consideration is used to probe the search space. Each particle is associated with an adaptable velocity (position change) according to which it moves in the search space and has a memory, remembering the best position of the search space it has ever visited [44]. The PSO finds the best solution by adjusting the trajectory of each particle towards its best location and also towards the best particle of the population at each generation. The PSO algorithm is simple to implement, has the ability of quickly converging to an optimal solution and becoming very popular to solve large multidimensional problems. In PSO, individuals of a swarm communicate their information and adjusting position and velocity using their group information according to the best information appeared in the current movement of the swarm [45]. In this way, the initial solution propagates through the search space and gradually moves towards the global optimum over a number of generations. The standard PSO algorithm consist mainly three computational steps as follows:

- Initialization of particles' positions and velocities
- Updating the position of each particle
- Updating the velocity of each particle

Suppose for a multidimensional problem under consideration, at k^{th}instant the i^{th}particle is moving in a multidimensional search space associated with a position P_i and velocity V_i represented as follows:

$$P_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$$

$$V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$$

Where: D represents the dimension of the search space. The position and velocity of the particle at $(k+1)$ instant can be manipulated as follows:

$$v_i(k+1) = w_i v_i(k) + c_1 * rand * (pbest_i - P_i(k) \\ + c_2 * rand * \left(gbest_i - P_i(k)\right)$$

$$P_i(k+1) = P_i(k) + V_i(k+1)$$

Where c_1 and c_2 are two constants and called as the acceleration coefficients. Particularly, c_1 is the cognitive parameter and c_2 is the social parameter. The *rand* generates a random number in the range [0, 1] and w_i is the inertia weight for the i^{th} particle, $pbest_i$ and $gbest_i$ are the local best and global best of i^{th} particle respectively.

III. Methods And Methodologies

This section describes the architecture of all the HONNs used in this experimental work. The models include a PSNN, SPNN, JPSNN, FLANN, RBFN, and MLP. These models are trained with a GA and a PSO learning technique separately, thus forming ten different hybrid forecasting models. The model training is presented at the last of this section.

A. Hybrid Forecasting Models

The architecture of Pi-Sigma neural network based forecasting model is shown by Fig. 1.

The Sigma-Pi neural network based forecasting model is shown by Fig. 2. In contrast to PSNN model, here the network use *sum of products* of input signals. The output of the network is computed as in Eq.1.

$$Y = f\left(\sum_{j=1}^{k}\left(\prod_{i=1}^{n} w_{ij} * x_i\right)\right) \tag{1}$$

The optimal weight values are adjusted by the training algorithm.

Fig. 3 represents the architecture of a Jordan Pi-Sigma neural network based forecasting model. This model combines the advantages of a recurrent neural network and PSNN to achieve better performance. As shown in Fig.3, a recurrent link from output layer back to the input layer. Since network with recurrent connection holds numerous advantages over ordinary feed forward MLP especially in dealing with time-series, therefore, by adding the dynamic properties to the PSNN, this network may outperform the ordinary feed forward MLP and also the ordinary PSNN [40]. Additionally, the unique architecture of JPSNN may also stay away from the combinatorial explosion of higher-order terms as the network order increases. The optimal weight vectors are searched by the training algorithm.

Fig. 4 shows the structure of a FLANN based forecasting model. For designing the network, at the first instance a functional expansion unit (FE) expands each input attribute of the input data. The simple trigonometric basis functions of sine and cosine are used here to expand the original input value into higher dimensions. For example an input value x_i expanded to several terms by using the trigonometric expansion functions such as in Eq. 2.

$$\left.\begin{aligned} c_1(x_i) &= (x_i), \\ c_2(x_i) &= \sin(x_i), \\ c_3(x_i) &= \cos(x_i), \\ c_4(x_i) &= \sin(\pi x_i), \\ c_5(x_i) &= \cos(\pi x_i), \\ c_6(x_i) &= \sin(2\pi x_i), \\ c_7(x_i) &= \cos(2\pi x_i). \end{aligned}\right\} \tag{2}$$

The attributes of each input pattern is passed through the functional expansion unit. The sum of the output signals of the functional expansion units multiplied with weights is passed on to the sigmoidal activation function of the output unit. The estimated output is compared with the target output and error signal is obtained. This error signal is used to train the model.

As shown in the Fig. 5, in the input layer, the number of input neurons is determined based on the input signals that connect the network to the environment. The hidden layer consists of a set of kernels units that carry out a nonlinear transformation from the input space to the hidden space. Two parameters, the center and the width are associated with each RBF node. The centers are determined during RBF training. Some of the commonly used kernel functions are Gaussian function, cubic function, linear function, generalized multi quadratic function etc. we used the Gaussian function which is represented in Eq. 3.

$$\emptyset_i(x) = exp\left(-\frac{\|x-\mu_i\|^2}{2\sigma_i^2}\right) \qquad (3)$$

Where:
$\|\cdots\|$ represents the Euclidean norm, x is the input vector, μ_i is the center, σ_i is the spread and $\emptyset_i(x)$ represents the output of the i^{th} hidden node. The output of the RBF network is calculated as in Eq. 4.

$$\hat{y} = f(x) = \sum_{k=1}^{N} w_k \emptyset_k(\|x - c_k\|) \qquad (4)$$

Where:
$\hat{y}$ is the network output, x is an input vector signal, $w = [w_1, w_2, \cdots, w_N]^T$ is the weight vector in the output layer, N is the number of hidden neurons, $\emptyset_k(\cdot)$ is the basis function, k is the bandwidth of the basis function, x is the input vector and $c_k = (c_{k1}, c_{k2}, \cdots, c_{km})^T$ is the center vector for k^{th} node, m is the number of input.

B. Model training

Each individual of GA or PSO represents a potential solution for a model (i.e. possible weight vector for a model). A set of such solution forms a search space and the optimal solution can be obtained by applying the search operators of respective algorithm.

The major steps of the GA based training can be summarized and described by Algorithm 1. This is a generalized training algorithm and here the *Model* can be any one of the aforesaid forecasting model.

Algorithm 1: GA Learning

```
1. Generate Initial Population P
2. Set MaxGen  /* initially set maximum iteration number with a bigger value for first training */
3. Select the next trainData and testData and normalize
4. fitness = EvaluateFitness (P, trainData)
       /*Use Algorithm2*/
5. P = ApplyGeneticOperator(P, trainData)
6. BestSoln = Choose the chromosome from P with minimum error
7. err = EvaluateFitness (BestSoln, testData)
       /*Use Algorithm2*/
8. store err as error for the respective testData
9. set MaxGen /* set maximum iteration number with a small value for subsequent training*/
10. Repeat step 3–8 till end of trainData and testData sets
```

Algorithm 2: EvaluateFitness (*P*, DataSet)

```
1. Error = 0;
2. For i =1: number of chromosome in P
3. For each input-output pair in the DataSet
       i. Estimate the output of the Model for the input pattern and weight.
       ii. Compare the desired output with estimated value and obtain error.
       iii. Error (i) = Error (i) + |error|.
   End
4. End
```

For each category of the aforesaid models, the PSO maintains a set of model. Each model can be encoded as a particle. A set of such models represent as a swarm of particles, compete among themselves to obtain better parameter set in the search space for designing the global best. The error signals are calculated by comparing the estimated signal with the desired signal presented at the output neuron for each model separately. These error values are considered as the fitness which is to be minimized by the PSO.

Algorithm 3: PSO Learning

```
1. Initialization of search spaceP /*Initialize each particle randomly with small values, i.e. particles with values from the domain [-1, 1] */
2. Set MaxGen  /* initially set maximum iteration number with a bigger value for first training */
3. Select the next trainData and testData and normalize
4. fitness = EvaluateFitness (P, trainData)
   /*Use Algorithm 4*/
5. P = UpdatePositionVelocity (P, trainData)
6. GBest = Choose the particle from P with best fitness value
7. err = EvaluateFitness (GBest, testData)
    /*Use Algorithm 4*/
8. store err as error for the respective testData
9. set MaxGen /* set maximum iteration number with a small value for subsequent training*/
10. Repeat step 3–8 till end of trainData and testData sets
```

Algorithm 4: EvaluateFitness (*P*, DataSet)

```
1. Error = 0;
2. For $i$ =1: number of particles in $P$
       For each input-output pair in the DataSet
           i.   Estimate the output of the Model for the input pattern and weight.
           ii.  Compare the desired output with estimated value and obtain error.
           iii. Error ($i$) = Error ($i$) + |error|.
       End
3. End
```

IV. Experimental Setup and Results Analysis

This section briefly explains about the data set used for experimentation, performance metrics, data normalization and input signal design, and experimental setup. Special attention has been given to train the model with minimal number of input signals and making the model adaptive. Results obtained from GA learning and PSO learning are summarized and analyzed separately. Statistical significance test has been conducted thoroughly for the models.

A. Data Set Description

For experimental purpose real data from five exchange Rates have been collected from the website www.forecasts.org. The data set consists of exchange rates of European Euro, British Pound, Indian Rupees, Japanese Yen, and Australian Dollar. The data show the average of daily figures on the 1st day of each month and collected for the period of 1999 to 2016. The number of data in each set is 214. The descriptive statistics of the exchange rate series are summarized in Table 1. It can be observed that except Yen to Dollar and Pound to Dollar dataset all other datasets show positive skewness value. This means, these datasets are spread out more toward right and suggest investment opportunities. The kurtosis analysis implies that exchange rate of all datasets are less outlier prone than the normal distribution. Again from the Jarque-Bera test statistics, it can be observed that all the stock price datasets are non-normal distributed. For the sake of convince, let the financial time series are denoted as follows:

D1: Rupees/Dollar
D2: Euro/Dollar
D3: Yen/Dollar
D4: Australian Dollar/Dollar
D5: Pound/Dollar

B. Performance Metric

The Mean Absolute Percentage Errors (MAPE) has been considered for performance metric in order to have a comparable measure across experiments with different datasets. The formula for MAPE is represented by Eq. 5.

$$\mathrm{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|x_i - \hat{x}_i|}{x_i} \times 100\% \tag{5}$$

The nest performance index considered here for evaluation of the performance of models is the Average Relative Variance (ARV) calculated as in Eq. 6.

$$\mathrm{ARV} = \frac{\sum_{i=1}^{N}(\hat{x}_i - x_i)^2}{\sum_{i=1}^{N}(\hat{x}_i - \bar{X})^2} \tag{6}$$

The model is considered as performing the worst as compared to mean if the ARV value is greater than 1. The model is considered performing better than simply calculating the mean if its ARV value is less than 1. The closer the value to 0, the forecasting model tends to be more accurate.

The next measure considered for evaluation of the models is U of Theil (UT) metric which compare the model performance with a random walk model. The metric can be calculated by using Eq. 7.

$$U\ of\ Theil = \frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{N}(x_i - x_{i+1})^2} \tag{7}$$

If the value of this statistics is equal to 1, then the model has the same performance of the random walk model, which suggests that stock prices change randomly, and it is impossible to predict the stock prices. If it is greater than 1, the model is considered performing the worst as compared to a random walk model. The model is performing better than a random walk model if its U of Theil statistics is less than 1. Therefore, the closer the UT value to 0, the better the model is.

Where:
x_i = actual closing price,
$\hat{x}_i$ = estimated price,
$\bar{X}$ = mean of dataset,
x_{i+1}= actual closing price of the next day and
N = total training patterns.

C. Data Normalization And Design Of Input Signals

To establish that the suggested models are unbiased and they can work for different types of trend in different economic/political scenario without much deviation in the capabilities of prediction, the exchange rate prices are used for a period from January 1999 to December 2016.

Again to significantly reduce the computation time consumed for training to a significant extent the following two steps are considered:

- Minimal data is used for training i.e. few input neuron with minimal patterns presented in each epoch
- Adaptive models are used

It is a common practice that, to train a model a large number of patterns are presented with large number of epochs to train the model for prediction. Here, though the objective is to design a generalized model, but very often it fails to follow the market trend in general. Further the number of neurons in the input layer is also kept relatively high, which also adds to the computation time. The Training and Test patterns generated for one-month-ahead exchange rate forecasting by sliding window technique is presented below. Here the bed length (window size) is written as *blen* and training length is represented as *l*. In general, the Training data with window size = *blen* and training length *l* is represented by Figure 6.

For this experiment, a sliding window takes only five values for the input layer and only three patterns are presented to build a model. Three patterns per epoch keeps the computation time per epoch significantly low.

It is a fact that as each time the sliding window moves one step ahead, exchange rate data at the beginning is dropped and one new data at the end is included. Therefore two consecutive training sets possibly possess minimal change in the nonlinear behavior of the input-output mapping. To incorporate the minor change in input-output mapping in the new model, the optimized weight set of the predecessor model for the same dataset is considered and minimal epochs help in capturing the change in nonlinear mapping [46].

Further considering the requirements of the neural network, the input data needs to be normalized. Here sigmoidal function is used for this purpose as in Eq. 8.

$$x_{norm} = \frac{1}{1+e^{-\lambda x_i}} \quad (8)$$

Where x_{norm} is the normalized price, x_i is the current day closing price, $\lambda = 1/x_{max}$ and $x_{max}$is the maximum price of the respective training set [47, 48].

D. Experimental Setup

Each model is simulated for 10 times for each training set, in order to reduce the stochastic behavior of the model and the average error is considered for comparative analysis of results. Since each time the sliding window moves one step ahead, only one new closing price data has been included into the training set. So there may not be significant change in nonlinearity behavior of the training data set. For that reason, instead of considering another random weight set, we have used the previously optimized weight set for the successive training. In this way, after the first training set, the number of iteration has been fixed to a small value, hence significant reduction in training time. For comparison purpose the same set of training and testing data are fed to all the models. Also another statistical based model, i.e. Multi linear regression (MLR) has been considered for comparative study.

The experiment was carried out by a system with Intel ® core TM i3 CPU, 2.27 GHz and 2.42 GB memory. The programming language used MATLAB-2009 Version-7.8.0.347.

E. Result Analysis From GA Learning

The MAPE values obtained from PSO learning are summarized in Table 2. For each data set Di, the performance of model is recorded. The models are ranked according to their MAPE values. Smaller the MAPE value better is the rank. For example, in case of data set D1, model JPSNN is assigned with rank 1 with minimum MAPE 0.003471, FLANN is assigned rank 2 with next lowest MAPE of 0.006857 and so on. The respective rank of each model/data is mentioned right to its value in parenthesis. The average rank over all data sets is calculated and presented in column 7 of Table 2. Another second level rank has been assigned to the models according to their average rank, which is shown on the last column of Table 2 as re-ranking which is used for subsequent analysis.

For the performance metrics MAPE, ARV and UT the lower the value the better it is, therefore the same process as discussed above is applied to these performance metrics. The ARV and UT values obtained from different models on different financial time series are summarized in Table 3 and Table 4 respectively.

Now the re-ranked values obtained by the models in Tables 2-4 for all five performance metrics are presented in Table 5. The null-hypothesis of Friedman's test, states that all the algorithms are equivalent and so their ranks Rj should be equal. To nullify the null-hypothesis, the Friedman statistic $\chi_F^2 = \left[\frac{12}{[N*k*(k+1)]}\right] * \sum R^2 - [3*N*(k+1)]$ is compared with the F-distribution$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}$, if the Friedman statistic χ_F^2 is greater than the F-distribution $F_F$then the null-hypothesis is rejected.

Here the number of models, k is 7 and number of performance metrics N is 3. It is found that the Friedman statistic for Table 5, $\chi_F^2 = \left[\frac{12}{[3*7*8)]}\right] * \sum(4.16^2 + 3^2 + 1.5^2 + 2^2 + 5.66^2 + 6.66^2 + 4.33^2) - [3*3*(7+1)] = 3.9783$ is greater than the F-distribution $F_F = \frac{(3-1)*3.9783}{3(7-1)-3.9783} = 0.5674$, therefore the null-hypothesis is rejected.

Since the data used is not guaranteed to be normally distributed, Nemenyi post-hoc test is suggested [49]. Nemenyi post hoc test is a nonparametric test to find out the significant difference among the model. It works on the null hypothesis that there exist no significant difference between a pair of models and the null hypothesis is rejected if significant difference exists. In this test pair wise comparison of model is done to find whether a significant difference exists between two models or not. If grade of the models differ by the critical difference (CD), then a significant difference exists. The critical difference is defined as in Eq.9.

$$CD = q_a \sqrt{\frac{k(k+1)}{6N}} \quad (9)$$

where the value of q_a is the critical value of Q for a multiple non-parametric comparison with a control (Table B.16 in [50]), k is the number of models and N is number of performance metrics.

Considering $\alpha = 0.01$, q_a is found to be 0.4643 and the corresponding CD is $0.4643 * \sqrt{\frac{7*(7+1)}{6*3}} = 3.1111$. Nemenyi post hoc test is used for pair wise comparison. If the grade difference between the worst performing model and model under consideration is larger than the CD value then the post-hoc test is powerful enough to detect any significant differences between the models. In this case study, the significant difference is obtained by the difference of average grade of any model from the average grade of worst average ranked model. In Table 4, MLR has the lowest level of performance with maximum average rank of 6.66. On comparison the following models are found to have significant difference in performance form MLR.

1. AverageRank(MLR)-AverageRank(SPNN)=6.66-3.0= 3.66> 3.1111 CD value
2. AverageRank(MLR)-AverageRank(JPSNN)=6.66-1.5=5.16> 3.1111 CD value
3. AverageRank(MLR)-AverageRank(FLANN)=6.66-2.0=4.66> 3.1111 CD value

SPNN, JPSNN, FLANN models satisfies the Nemenyi post hoc test for significant difference. Apart from these three models, all other models do not satisfy Nemenyi post hoc test. Therefore these three models can be treated as consistent models. These models may not perform the best always but their results will be competitive with other better performing models

$$\begin{array}{ccccccc}
 & Train\ data & & & & Train\ target \\
x(i) & x(i+1) & \cdots & x(i+blen) & \vdots & x(i+blen+1) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
x(i+l) & x(i+l+1) & \cdots & x(i+l+blen) & \vdots & x(i+l+blen+1)
\end{array}$$

And the respective Test data is

$$\begin{array}{cccccc}
 & Test\ data & & & & Test\ target \\
x(i+l+1) & x(i+l+2) & \cdots & x(i+l+blen+1) & \vdots & x(i+l+blen+2)
\end{array}$$

Fig.6. Training and Test Dataset Generation for Forecasting Model

Table 1. Descriptive statistics from all financial time series

Dataset	Descriptive statistics						
	Minimum	**Maximum**	**Mean**	**Standard deviation**	**Skewness**	**Kurtosis**	**Jarque-Bera test statistics**
D1	39.2680	68.2400	49.5192	7.5903	1.0952	2.9765	42.7875(h=1)
D2	0.6345	1.1723	0.8430	0.1344	0.8812	2.8473	27.9051(h=1)
D3	76.6430	133.6430	106.2796	14.0317	-0.5481	2.3795	14.1480(h=1)
D4	0.9276	1.9920	1.3393	0.2888	0.6141	2.4885	15.7831(h=1)
D5	0.4831	0.7886	0.6161	0.0619	-0.2110	2.6432	2.7234(h=0)

Table 2. MAPE of the models from different exchange rate time series from GA learning

MODEL	MAPE from exchange rate series					Avg. Rank	Re-ranking
	D1	D2	D3	D4	D5		
PSNN	0.014039 (3)	0.026301 (1.5)	0.011617 (1)	0.030515 (5)	0.048330 (5)	3.1	4
SPNN	0.025269 (4)	0.026301 (1.5)	0.033496 (3)	0.003380 (1)	0.007442 (2)	2.3	1.5
JPSNN	0.003471 (1)	0.043026 (4.5)	0.030954 (2)	0.004854 (3)	0.007159 (1)	2.3	1.5
FLANN	0.006857 (2)	0.026583 (3)	0.041241 (4)	0.003737 (2)	0.025681 (3)	2.8	3
MLP	0.151640 (6)	0.250201 (5)	0.247855 (7)	0.023652 (4.5)	0.025985 (4)	5.3	6
MLR	0.761849 (7)	0.579331 (6)	0.233385 (6)	0.038932 (6)	0.102556 (7)	6.4	7
RBFN	0.151004(5)	0.043026(4.5)	0.211452(5)	0.023652(4.5)	0.049773(6)	5	5

Table 3. ARV of the models from different exchange rate time series from GA learning

MODEL	ARV from exchange rate series					Avg. Rank	Re-ranking
	D1	D2	D3	D4	D5		
PSNN	0.015524 (2)	0.031764 (3)	0.044255 (5)	0.162428 (5)	0.026125 (4)	3.8	4
SPNN	0.017282 (3.5)	0.055792 (4)	0.031769 (3)	0.077379 (4)	0.068802 (2)	3.3	3
JPSNN	0.015106 (1)	0.021880 (1)	0.018008 (1)	0.028305 (1)	0.062900 (1)	1	1
FLANN	0.017282 (3.5)	0.031097 (2)	0.024542 (2)	0.034845 (2)	0.023685 (3)	2.5	2
MLP	0.045002 (6)	0.064112 (6)	0.322530 (6)	0.500321 (6)	0.270608 (6)	6	6
MLR	0.127503 (7)	0.072843 (7)	0.621800 (7)	0.532601 (7)	0.291933 (7)	7	7
RBFN	0.041224(5)	0.047022(5)	0.041003(4)	0.071336(3)	0.210045(5)	4.4	5

Table 4. UT of the models from different exchange rate time series from GA learning

MODEL	UT from exchange rate series					Avg. Rank	Re-ranking
	D1	D2	D3	D4	D5		
PSNN	0.035142 (3)	0.023821 (5)	0.042527 (5)	0.004994 (4)	0.008053 (3)	4	4.5
SPNN	0.036268 (4)	0.016527 (4)	0.006379 (3)	0.012040 (5)	0.008181 (4)	4	4.5
JPSNN	0.014436 (2)	0.006573 (2)	0.004873 (4)	0.001470 (1)	0.007153 (2)	2.2	2
FLANN	0.003642 (1)	0.005584 (1)	0.003628 (1)	0.002161 (2)	0.005922 (1)	1.2	1
MLP	0.057351 (7)	0.053271 (7)	0.043826 (6)	0.017302 (6)	0.056624 (6)	6.2	5
MLR	0.023144 (6)	0.052932 (6)	0.049936 (7)	0.135209 (7)	0.059956 (7)	6.6	6
RBFN	0.038843(5)	0.003273(3)	0.004015(2)	0.003624(3)	0.009442(5)	3.6	3

Table 5. Re-ranked values of models for the performance metrics

MODEL	MAPE	ARV	UT	Average of Ranks
PSNN	4	4	4.5	4.16
SPNN	1.5	3	4.5	3
JPSNN	1.5	1	2	1.5
FLANN	3	2	1	2
MLP	6	6	5	5.66
MLR	7	7	6	6.66
RBFN	5	5	3	4.33

F. Result Analysis From PSO Training

Similarly, the MAPE, ARV, and UT values obtained from PSO learning are summarized in Table 6-8 respectively. Now the re-ranked values obtained by the models in Tables 6-8 for all three performance metrics are presented in Table 9.

Again here the number of models, k is 7 and number of performance metrics N is 3. It is found that the Friedman statistic for Table 9, $\chi_F^2 = \left[\frac{12}{[3*7*8)]}\right] * \sum(3.33^2 + 4^2 + 1.5^2 + 1.5 + 6^2 + 7^2 + 4.66^2) - [3*3*(7+1)] = 4.7360$ is greater than the F-distribution $F_F = \frac{(3-1)*4.7360}{3(7-1)-4.7360} = 0.7141$, therefore the null-hypothesis is rejected.

Further Nemenyi post hoc test is conducted for pair wise comparison with CD $= 4.0205$, for k=7, N=3 and $\alpha = 0.01$. It is found that PSNN, JPSNN, and FLANN models satisfy the Nemenyi post hoc test for significant difference. Apart from these three models, all other models do not satisfy Nemenyi post hoc test.

It can be observed from the above result analysis that HONN based forecasting models outperforming over other models such as MLP, RBFN, and MLR. Now the performance of HONN based models can be compared. The average rankings from GA and PSO learning of such models are presented by Table 10. Lower the rank values, better is the model. Table 10 shows the superiority of JPSNN model followed by FLANN based models. However, all the HONN based forecasting models have shown consistent performance. For example, the estimated exchange rates by the JPSNN against the actual exchange rates are plotted and shown by Fig. 7-11 for all data sets.

Table 6. MAPE of the models from different exchange rate time series from PSO learning

MODEL	MAPE from exchange rate series					Avg. Rank	Re-ranking
	D1	D2	D3	D4	D5		
PSNN	0.031439 (3)	0.022635 (1)	0.025161 (2)	0.032544 (3)	0.041133 (3.5)	3.1	3
SPNN	0.035261 (4)	0.023377 (3)	0.033406 (5)	0.033822 (4)	0.041133 (3.5)	3.9	4
JPSNN	0.005476 (2)	0.023583 (4)	0.030954 (3)	0.007585 (1)	0.007445 (2)	2.4	2
FLANN	0.004087 (1)	0.026506 (5)	0.021277 (1)	0.008673 (2)	0.005568 (1)	2	1
MLP	0.225164 (6)	0.754233 (6)	0.204789 (6)	0.063651 (7)	0.075982 (6)	6.2	6
MLR	0.782842 (7)	0.859332 (7)	0.238720 (7)	0.048978 (6)	0.102556 (7)	6.8	7
RBFN	0.205114(5)	0.023125(2)	0.032145 (4)	0.039652(5)	0.049712(5)	4.2	5

Table 7. ARV of the models from different exchange rate time series from PSO learning

MODEL	ARV from exchange rate series					Avg. Rank	Re-ranking
	D1	D2	D3	D4	D5		
PSNN	0.026524 (4)	0.031764 (3.5)	0.034256 (3)	0.116242 (5)	0.026695 (1)	3.3	3
SPNN	0.027284 (5)	0.035702 (5)	0.034769 (4)	0.054376 (4)	0.028877 (2)	4	5
JPSNN	0.022516 (1)	0.021885 (1.5)	0.026108 (2)	0.028385 (1)	0.062955 (4)	1.9	1.5
FLANN	0.023282 (2)	0.021885 (1.5)	0.024544 (1)	0.034045 (2)	0.029647 (3)	1.9	1.5
MLP	0.045682 (6)	0.084432 (6)	0.321532 (6)	0.350325 (6)	0.277638 (6)	6	6
MLR	0.227533 (7)	0.088847 (7)	0.621835 (7)	0.503261 (7)	0.295938 (7)	7	7
RBFN	0.024227(3)	0.031764 (3.5)	0.041133(5)	0.051036(3)	0.200047(5)	3.9	4

Table 8. UT of the models from different exchange rate time series from PSO learning

MODEL	UT from exchange rate series					Avg. Rank	Re-ranking
	D1	D2	D3	D4	D5		
PSNN	0.023866 (5)	0.035040 (3)	0.040017(4)	0.008113 (3.5)	0.004997 (4)	3.9	4
SPNN	0.016527 (3)	0.037278 (4)	0.008369 (3)	0.008113 (3.5)	0.012045 (5)	3.7	3
JPSNN	0.008573 (2)	0.014538 (2)	0.007674 (1)	0.007250 (1)	0.001478 (1)	1.4	1
FLANN	0.008589 (1)	0.003699 (1)	0.007828 (2)	0.007948 (2)	0.002265 (2)	1.6	2
MLP	0.053776 (6)	0.076355 (7)	0.043866 (6)	0.058626 (6)	0.017392 (6)	6.2	6
MLR	0.152902 (7)	0.253142 (6)	0.249956 (7)	0.159256 (7)	0.135217 (7)	6.8	7
RBFN	0.021173(4)	0.038245(5)	0.040528 (5)	0.019442(5)	0.003923(3)	4.4	5

Table 9. Re-ranked values of models for the performance metrics

MODEL	MAPE	ARV	UT	Average of Ranks
PSNN	3	3	4	3.33
SPNN	4	5	3	4.00
JPSNN	2	1.5	1	1.50
FLANN	1	1.5	2	1.50
MLP	6	6	6	6
MLR	7	7	7	7
RBFN	5	4	5	4.66

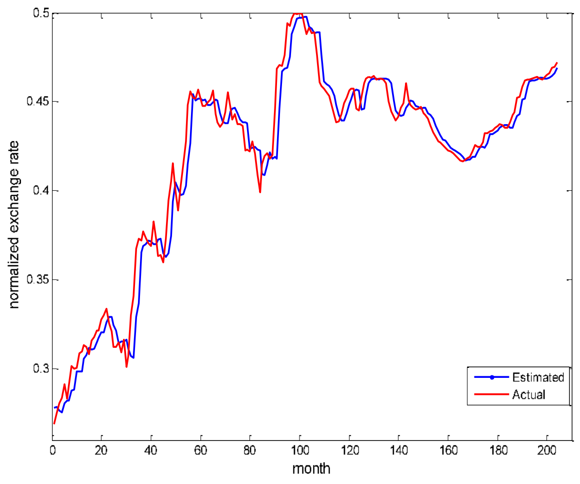

Fig.7. Actual v/s estimated exchange rate prices for Rupees/Dollar by JPSNN forecasting model

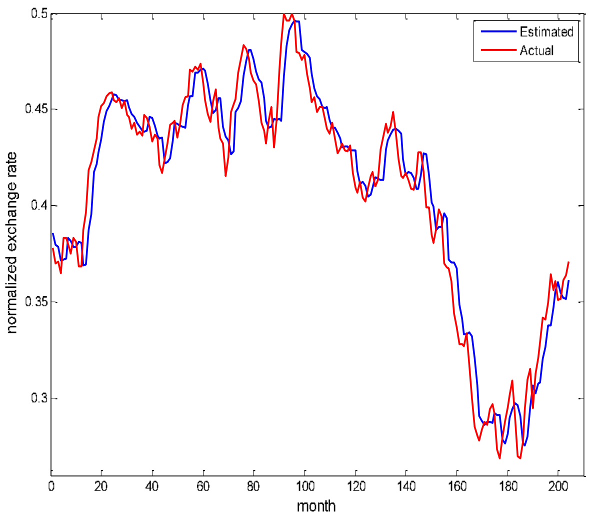

Fig.8. Actual v/s estimated exchange rate prices for Euro/Dollar by JPSNN forecasting model

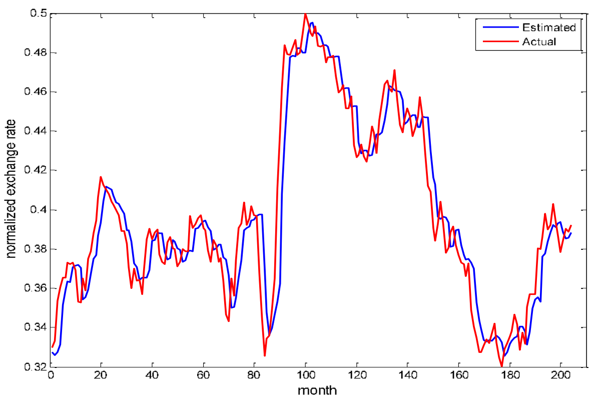

Fig.9. Actual v/s estimated exchange rate prices for Pound/Dollar by JPSNN forecasting model

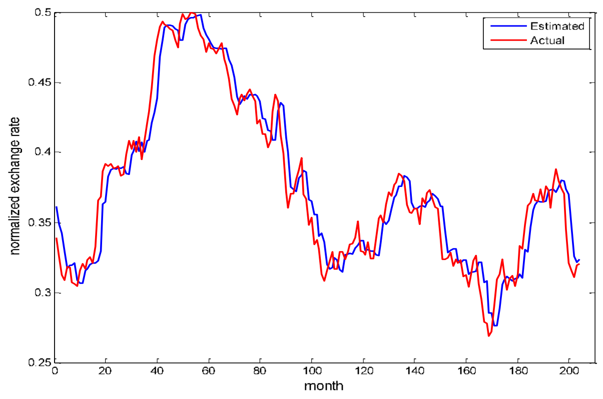

Fig.10. Actual v/s estimated exchange rate prices for Yen/Dollar by JPSNN forecasting model

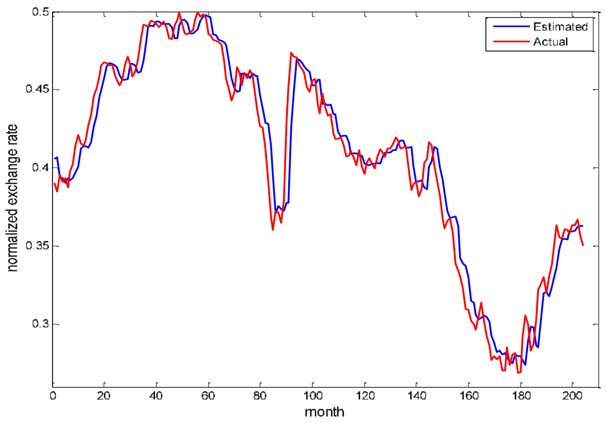

Fig.11. Actual v/s estimated exchange rate prices for Australian dollar/Dollar by JPSNN forecasting model

Table 10. Overall performances comparison of PSNN, SPNN, JPSNN, FLANN from GA and PSO learning.

MODEL	Ranks from GA learning	Ranks from PSO learning	Average ranking
PSNN	4.16	3.33	3,75
SPNN	3	4.00	3.5
JPSNN	1.5	1.50	1.5
FLANN	2	1.50	1.75

V. Conclusions

Soft and evolutionary computing based techniques have been introduced in the literature in order to overcome the drawbacks of statistical based methods of forecasting of foreign exchange rates. Several limitations such as the complexity and accuracy of such models make the existing system less desirable. Ordinary feed forward neural network i.e. the MLP, is prone to over fitting and easily get stuck into local minima. Thus, to overcome the drawbacks, hybrid higher order neural network based models are proposed as an alternative mechanism to predict the one-step-ahead exchange rate. To evaluate the performance of these models, GA and PSO learning have been conducted. Models are constructed from minimal training data and trained adaptively, resulted significant reduction of time. Four HONNs such as PSNN, SPNN, JPSNN, and trigonometric FLANN are used to build the forecasting models. The representational power of higher order terms helped solving the problems with small networks and maintaining fast convergence capabilities. Simulations for the comprehensive evaluation of the models were presented. The evaluation covering several performance criteria such as MAPE, ARV, and U of Thiels were discussed. The outcomes of the hybrid models are compared with that of the MLP, RBFN, and MLR and found superior. Friedman's test and Nemenyi post-hoc test are conducted for statistical significance of the results. Exploring other promising adaptive models as well as conducting long range prediction will be our further consideration. Use of other additional hidden features of the financial time series as input to the model to achieve enhanced forecasting accuracy can be suggested.

Acknowledgement

The author likes to thank anonymous reviewer(s) and the Chief Editor for their valuable comments and suggestions which have certainly improved the quality of this paper.

References

[1] Haykin, S., Neural Networks and Learning Machine, PHI, ISBN -978-81-203-4000-8, 2010.

[2] Rajasekaran, S. and Vijayalakshmi Pai, G. A., Neural Networks, Fuzzy Logic and Genetic Algorithms Synthesis and Application, PHI, ISBN-978-81-203-2186-1, 2007.

[3] Wang, Z., Fang, J., and Liu, X., 'Global stability of stochastic high-order neural networks with discrete and distributed delays', Chaos, Solutions and Fractals, Vol. 36, No. 2, pp.388–396, 2008.

[4] Shin, Y. and Ghosh, J., 'Efficient higher-order neural networks for classification and function approximation', International Journal on Neural Systems, Vol. 3, pp. 323–350, 1992.

[5] Ghazali, R., Hussain, A., and El-Deredy, W., "Application of ridge polynomial neural networks to financial time series prediction", International Joint Conference on Neural Networks, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, pp. 913–920, (July, 2006).

[6] Knowles, A., Hussain, A., El Deredy, W., Lisboa, P. G., and Dunis, C. L., "Higher order neural networks with Bayesian confidence measure for the prediction of the EUR/USD exchange rate" In Artificial higher order neural networks for economics and business (pp. 48-59), IGI Global , 2009.

[7] Shin, Y., and Ghosh, J., "Efficient higher-order neural networks for classification and function approximation", International Journal on Neural Systems, Vol. 3(4), pp. 323–350, 1992.

[8] Nayak, S. C., Misra, B. B., and Behera, H. S. , "A Pi-Sigma Higher Order Neural Network for Stock Index Forecasting", . In Computational Intelligence in Data Mining, Vol. 2, pp 311-319, Springer India, (2015).

[9] Nayak, S. C., Misra, B. B., and Behera, H. S., "Fluctuation prediction of stock market index by adaptive evolutionary higher order neural networks", International Journal of Swarm Intelligence, Vol. 2(2-4), pp. 229-253, Inderscience, (2016).

[10] Nayak, J., Naik, B., and Behera, H. S., "A novel

Chemical Reaction Optimization based Higher order Neural Network (CRO-HONN) for nonlinear classification", Ain Shams Engineering Journal, Vol. 6(3), pp. 1069-1091, (2015).

[11] Perantonis, S., Ampazis, N., Varoufakis S., and Antoniou, G. (1998) 'Constrained learning in neural networks: Application to stable factorization of 2-d polynomials', *Neural Processing Letter*, Vol.7, No. 1, pp. 5–14.

[12] Huang, D. S., Ip, H. H. S., Law K. C. K., and Chi, Z. (2005) 'Zeroing polynomials using modified constrained neural network approach', IEEE Transactions on Neural Networks, Vol. 16, No. 3, pp. 721–732.

[13] Epitropakis, M. G., Plagianakos, V. P. and Vrahatis, M. N. (2010), 'Hardwarefriendly higher-order neural network training using distributed evolutionary algorithms', Appl Soft Comput, Vol. 10, pp.398–408.

[14] Ghazali, R., Hussain, A. J. and Liatsis, P. (2011) 'Dynamic Ridge Polynomial Neural Network: Forecasting the univariate non-stationary and stationary trading signals', Expert Systems with Applications, Vol. 38, pp. 3765-3776.

[15] Ghazali, R., Husaini, N. A., Ismail, L. H. and Samsuddin, N. A. (2012) 'An Application of Jordan Pi-Sigma Neural Network for the Prediction of Temperature Time Series Signal', Recurrent Neural Networks and Soft Computing, Dr. Mahmoud ElHefnawi (Ed.), ISBN: 978-953-51-0409-4.

[16] Yong, N., Wei, D. (2008) 'A hybrid genetic learning algorithm for Pi–Sigma neural network and the analysis of its convergence', In: IEEE fourth international conference on natural computation, pp. 19–23.

[17] Fallahnezhad, M., Moradi, M. H., Zaferanlouei, S. (2011), 'A hybrid higher order neural classifier for handling classification problems', Expert Systems with Appl., Vol. 38, pp.386–393.

[18] Zhang, M., Xu, S., and Fulcher, J. (2002), 'Neuron-Adaptive Higher Order Neural-Network Models for Automated Financial Data Modeling', IEEE Transactions on neural networks, Vol. 13, No. 1.

[19] Majhi B., Rout M., Majhi R., Panda G. and Fleming P. J. (2012), 'New robust forecasting models for exchange rates prediction', Expert Systems with Applications, Vol. 39, pp.12658–12670.

[20] Sahu K. K., Biswal G.R., Sahu P.K., Sahu S.R., and Behera H. S. (2014) 'A CRO based FLANN for Forecasting Foreign Exchange Rates', 2014 International Conference on Computational Intelligence in Data Mining (ICCIDM), Smart Innovation, System and Technology, Vol. 31, pp. 647-664.

[21] Yu. Lean, Wang S. and Lai K. K. (2005), 'A novel nonlinear ensemble forecasting model incorporating GLAR and ANN for foreign exchange rates', Computers and Operations Research 32, Elsevier, 2523-2541.

[22] Jordan, M. I. (1986). Attractor Dynamics and Parallelism in a Connectionist Sequential Machine. Paper presented at the Proceedings of the Eighth Conference of the Cognitive Science Society, New Jersey, USA.

[23] Husaini, N.A., Ghazali, R., Nawi, N.M. and Ismail, L.H., 2011, April. Jordan pi-sigma neural network for temperature prediction. In *International Conference on Ubiquitous Computing and Multimedia Applications* (pp. 547-558). Springer Berlin Heidelberg.

[24] Husaini, N. A., Ghazali, R., Ismail, L. H., & Herawan, T. (2014). A jordan pi-sigma neural network for temperature forecasting in batu pahat region. In*Recent Advances on Soft Computing and Data Mining* (pp. 11-24). Springer International Publishing.

[25] Nayak, J., Kanungo, D. P., Naik, B., & Behera, H. S. (2014, December). A higher order evolutionary Jordan Pi-Sigma neural network with gradient descent learning for classification. In *High Performance Computing and Applications (ICHPCA), 2014 International Conference on* (pp. 1-6). IEEE.

[26] Nenortaite J., Simutis R.(2004), 'Stocks' Trading System Based on the Particle Swarm Optimization Algorithm', Workshop on Computational Methods in Finance and Insurance, Lecture Notes in Computer Science, Springer, vol. 3039/2004.

[27] Zhao L., Yang Y. (2009), 'Expert systems with applications: PSO-based single multiplicative neuron model for time series prediction', Expert Systems with Applications, Vol. 36, pp. 2805–2812.

[28] Sermpinis, G., Theofilatos, K., Karathanasopoulos, A., Georgopoulos, E. F., and Dunis, C. L. (2013) 'Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and Particle Swarm Optimization', European Journal of Operational Research, Vol. 225, pp. 528–540.

[29] Chen Y., and Zhang G. (2013), 'Exchange rates determination based on genetic algorithms using Mendel's principles: Investigation and estimation under uncertainty', Information Fusion, Vol. 14, pp. 327–333.

[30] Chang, P.-C., Liu, C.-H., Lin, J.-L., Fan, C.-Y., Ng, Celeste S.P., 2009. A neural network with a case based dynamic window for stock trading prediction. Expert Syst. Appl. 36, 6889–6898.

[31] Atsalakis, G.S., Valavanis, K.P., 2009. Surveying stock market forecasting techniques – part II: soft computing methods. Expert Syst. Appl. 36, 5932–5941.

[32] Venugopal Setty, D., Rangaswamy, T.M., Subramanya, K.N., 2010. A review on data mining applications to the performance of stock marketing. Int. J. Comput. Appl. 1 (3), 33–43.

[33] Kumaran Kumar, J., Kailas, A., 2012. Prediction of future stock close price using proposed hybrid ANN model of functional link fuzzy logic neural model (FLFNM). Int. J. Comput. Appl. Eng. Sci. II (1).

[34] Rout, Minakhi, et al. "Forecasting of currency exchange rates using an adaptive ARMA model with differential evolution based training." Journal of King Saud University-Computer and Information Sciences 26.1 (2014): 7-18.

[35] Jena, Pradyot Ranjan, Ritanjali Majhi, and Babita Majhi. "Development and performance evaluation of a novel knowledge guided artificial neural network (KGANN) model for exchange rate prediction." Journal of King Saud University-Computer and Information Sciences 27.4 (2015): 450-457.

[36] Galeshchuk, Svitlana. "Neural networks performance in exchange rate prediction." Neurocomputing 172 (2016): 446-452.

[37] Nanda, S. K., Tripathy, D. P., Nayak, S. K., & Mohapatra, S. (2013). Prediction of rainfall in India using Artificial Neural Network (ANN) models.International Journal of Intelligent Systems and Applications, 5(12), 1.

[38] Hu, Z., Bodyanskiy, Y. V., Tyshchenko, O. K., & Boiko, O. O. (2016). An Evolving Cascade System Based on A Set Of Neo Fuzzy Nodes. International Journal of Intelligent Systems and Applications, 9, 1-7.

[39] Y. Shin, and J. Ghosh, 'Efficient higher-order neural networks for classification and function approximation', International Journal on Neural Systems, Vol. 3, pp. 323–350, 1992.

[40] Hussain, A. J. & Liatsis, P. (2002). Recurrent Pi-Sigma

Networks for DPCM Image Coding. Neurocomputing, 55, pp. 363-382.

[41] Pao, Y. H., Takefuji, Y., Functional-link net computing: thory, system architecture, and functionalities. Computer, Vol. 25, 76-79, 1992.

[42] Goldberg, D. E., Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, Reading, MA, USA, 1989.

[43] Kennedy, J., Eberhart, R.C.,. "Particle swarm optimization", In: Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948, 1995.

[44] Eberhart, R. C., Simpson, P. and Dobbins, R., "Computational intelligence PC tools," Academic Press, 1996.

[45] Babaei M, A general approach to approximate solutions of nonlinear differential equations using particle swarm optimization. Appl. Soft. Comput 2013(13): 3354-65.

[46] Nayak, S. C., Misra, B. B., and Behera, H. S. (2015), 'Artificial Chemical Reaction Optimization of Neural Networks for Efficient Prediction of Stock Market Index', Ain Shams Engineering Journal

[47] Nayak, S. C., Misra, B. B., and Behera, H. S. (2014), 'Impact of data normalization on stock index forecasting', International Journal of Computer Information Systems and Industrial Management, Vol. 6, pp. 357-369.

[48] Nayak, S. C., Misra, B. B., and Behera, H. S.," Evaluation of Normalization Methods on Neuro-Genetic Models for Stock Index Forecasting," IEEE World Congress on Information and Communication Technologies, (WICT 2012), doi: 10.1109/WICT.2012.6409147.

[49] Demsar, J., "Statistical Comparisons of Classifiers over Multiple Data Sets," Journal of Machine Learning Research, 7(2006)1–30.

[50] Zar, J. H. (1999). More on dichotomous variables. Biostatistical analysis. 4th ed. Upper Saddle River: Prentice Hall, 516-65.

Authors' Profiles

Dr. Sarat Chandra Nayak holds a Ph.D. degree in Computer Engineering from VSSUT, Burla, India and M. Tech. in Computer Science from Utkal University, Bhubaneswar, India. His research interests are Data Mining, Soft Computing, Predictive Systems, Financial Time Series Forecasting, Computational Intelligence, Evolutionary Computation, and Classification. He has more than 25 research articles in reputed International journals and conferences, and 4 book chapters in his credit. He has 10 years of experience in teaching and research. Dr. Nayak currently associated with computer science and engineering department as a Professor at Kommuri Pratap Reddy Institute of Technology, Hyderabad, India.

How to cite this paper: Sarat Chandra Nayak, "Development and Performance Evaluation of Adaptive Hybrid Higher Order Neural Networks for Exchange Rate Prediction", International Journal of Intelligent Systems and Applications(IJISA), Vol.9, No.8, pp.71-85, 2017. DOI: 10.5815/ijisa.2017.08.08