Volume 6, Issue 2, February 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Prediction of Stock Rates using PSO Hybridized BPNN Model

Prashant Swami*
PG Scholar, TIT-College,
Madhya Pradesh, India

Rohit Vyas
Assistant Professor, TIT- College,
Madhya Pradesh, India

Abstract*— The main aim for this research work is to forecast the next day's closing stock prices of the companies with better accuracy. Forecasting the future data points according to the past values keeping in mind that the series can be linear as well as non-linear an intelligent approach is required that can handle both at the same time. Earlier the researchers were designing the model assuming the series to be linear. Autoregressive models, Moving Average models, and the combination of two ARIMA model were developed for the analysis of time series data. In this research work, ANN based model is used that uses Particle Swarm Optimization (PSO) for training the model and then optimizes it further with traditional gradient descent method. The advantage of PSO is fast learning and obtaining the global optimum quickly but it lacks when it reaches the optimum location. The traditional BPNN can be effective in this case as it has a tendency to acquire local optima very quickly. These advantages of both techniques are utilized in this research work and the model "PSO hybridized BPNN" is proposed for forecasting next day's closing price of the company.

Keywords*— PSO, BPN, Feature Selection, ANN, ARIMA

I. INTRODUCTION

Forecasting stock prices are more of a challenge these days where every single activity in any sector affects the stock market heavily. Stock market is growing day by day with many companies going public one after another stock market prediction is a special case of time series prediction. An overview about time series is required for proper understanding of the stock data.Prices are monitored at each day's closing, making it continuous and well defined.
Time series data can be classified into two types:

1. **Continuous:** The data measured at every instance of time e.g. ECG
2. **Discrete:** When the observations are taken at a regular interval of time e.g. Stock market data.

A time series data differs from the usual data in the fact that it is composed of few components forming its internal structure Time series representation as additive model:

$$Y_t = T_t + S_t + C_t + I_t$$

Where,
Y_t Represents the original time series
T_t Represents trend component
S_t Represents seasonality component
C_t Represents cyclic component
I_t Represents Irregular component

All these components may or may not present at once in a time series data. But most of the time these components are present all together and they are needed to be removed for the proper analysis of data.Earlier the model proposed was based on the assumption that the data will always be linear and used ARIMA models and its derivatives for forecasting [1][2]. A new hybrid model was then proposed to solve this problem. It worked on the principle that time series is the combination of linear and nonlinear components and both needs to be analysed by a different model. ARIMA model was then hybridized with artificial neural networks (ANN) to produce better results [3].

1.1. Time series analysis

Time series analysis requires the observations to be analysed at each and every interval of time to extract meaningful information and other characteristics from the data. The information extracted can further be used for designing forecasting models in different application areas. While regression analysis is often used to verify if one or more past time series value affects the current value of the time series, but time series analysis completely differs from it. A time series has certain temporal ordering that cannot be analysed just by judging past values and obtaining the current one using that. Various components that are discussed above like trend, seasonality etc. present in a time series makes it rather complex to analyse using regression analysis [6].

1.2. Time Series Models

A sample data of time series is taken indicating precipitation per day this data may be used to predict the next days' precipitation measure by using proper time series models. Time period used here is in the form of days, it may vary according to the time series opted for forecasting.

1.3. Forecasting Models

Once the time series data has been studied and a model is developed, forecasting can be employed using a proper forecasting model. The term prediction is very closely related to forecasting. Forecast refers to announcing the future values after analysing the varying factors and different parameters in a model and prediction is getting to the outcome based on logic rather than an approach.

II. LITERATURE SURVEY

G. Armano et al. (2005)[16]in this paper, the proposed forecasting model consists of a population of experts with each of them integrated with genetic and neural technologies. To control the activation of ffnn a genetic classifier is used. Both the components are provided with different inputs. Genetic component is given inputs through technical analysis of the stock data and neural components are provided with other relevant inputs. The performance is measured by testing the model with two different stock market data and it shows that it clearly outperforms "*buy and hold*" strategy.

M. R. Hassan et al. (2007)[17]a fusion model has been proposed that combines hidden markov model (hmm), artificial neural network (ann), and genetic algorithms (ga) for forecasting stock market. In this model ann is used to make individual sets of similar patterns in the time series which is later given input to hmm. Ga was used to optimize initial parameters of hmm. After training hmm, the test input is matched with the similar sets. The difference of the matched days and the next day is found and returned. The performance analysis shows that this model outperforms the arima model.

L. Yu et al. (2009)[21]in this paper a meta-learning model is proposed on the basis of supervised ann for forecasting financial time series. The data sampling techniques are used in this system to divide the data into different subsets. These subsets are trained by ann with different initial parameters. Later principal component analysis (pca) is used to get optimal base models by pruning the unwanted models. At last a nonlinear metamodel neural network is developed that learns from the optimal models achieved through pca. The experimental analysis shows that it outperforms the single models like arima, fnn, and svm in prediction accuracy in terms of directional statistics.

V.nekoukar, m. T.hamidibeheshti (2009)[22]in this paper a variant of neural network model is proposed where each connection weight is replaced by a local linear model. The llrbfn model is trained with an evolutionary algorithm particle swarm optimization (pso). The pso algorithm is modified to be used in this model. It is tested with the stock market data that shows its feasibility and effectiveness in forecasting future values.

Z. Fangwen et al. (2010)[27]neural networks are quite popular in the area of forecasting a time series. In this paper a recurrent neural network is used called as echo state networks. This model reduces the risk of local convergence and is effective in short term prediction. Principal component analysis (pca) is used along with this model for feature selection and technical analysis of time series. It is tested with six companies' data like s&p 500, nasdaq, etc. The results show that the model has better prediction accuracy than the conventional model.

E. Guresen et al. (2011)[28]this paper states the effectiveness of neural networks and its hybrid models used in the field of time series forecasting. The analysed models include multi-layer perceptron (mlp), dan2, garch, and different hybrid models like neuro-fuzzy model, cbr-ann, etc. The dataset chosen for common analysis was nasdaq stock data. The results shows that nonlinear mlp model outperforms garch or hybrid garch-ann models.

J. L. Ticknor (2013)[29]a variant of artificial neural network is used in this model that assigns probabilistic weights initially to ann and also avoids the overfitting and overtraining problem of ann. The model called as "*bayesian regularized artificial neural network*" improves prediction accuracy without the need of preprocessing and smoothening of data. The probabilistic nature of weights makes the network safe to expand and the regularization avoids overfitting of the network. This model outperforms the model given by hassan et al. (2007) [ ] in performance analysis. It also states that the model can be further optimized by addition or substitution of other indicators of stock market.

C. S. Vui et al. (2013)[30]this paper provides a review of ann models used in the area of stock market prediction. The stock market is quite volatile, non-linear and is very largely affected by the external factors. Ann models are very good in generalization of complex non-linear functions. This paper states the hybrid ann models used for this purpose and also indicated their effectiveness and feasibility.

III. PROBLEM IDENTIFICATION AND SIMULATION

3.1 Particle Swarm Optimization

Particle swarm optimization (PSO) is a robust stochastic computational method that optimizes a problem iteratively based on the movement of its particles and intelligence of swarm. Swarm intelligence refers to a group of small organisms like flocking birds, fish school etc. that live intact and help each other in finding their food. The task is accomplished by sharing each individual's intelligence collected so far with each other. [33]

3.2 Artificial Neural Network

Back in 1943 Warren McCulloch and Walter Pittsdeveloped the first computational model for artificial neural network. It was called threshold logic. It caused the research in neural network to split in two fields. First, the study of animal brain and biological processes taking place in the brain; and second, field focused on the study of applications of neural network [40].

3.3 Multilayer Neural Network

A multilayer neural network contains multiple layers of nodes having multiple numbers of nodes too. These are connected in the form of a directed graph. Each layer is fully connected with the other layer. Every node in this network

is a neuron (with a nonlinear activation function) except the input layer. MLP utilizes a back-propagation algorithm which is a supervised learning technique for training purpose. MLP is capable of distinguishing data which is not linearly separable. It is highly robust.

3.4 The Back-propagation Training Algorithm:

Step 1: Initialization: The values of weights and threshold values are initialized to some uniform random values in a small range. The weight initialization is done on a neuron-by-neuron basis.

Step 2: Activation: The back-propagation neural network is activated by applying input and desired values to it. First the input data values are combined with the weights of hidden layer and output of hidden layer is calculated. It is given by the following equation:

$$y_j(p) = sigmoid\left[\sum_{i=1}^{n} x_i(p).w_{ij}(p) - \theta_j\right] \quad (3.14)$$

Where 'n' is the number of inputs, *'sigmoid'* is the activation function, 'x_i' are the inputs, 'y_j' are the outputs of 'j_{th}' neuron of hidden layer. 'w_{ij}' are the weights from input to hidden layer and 'p' is the iteration number.

IV. PROPOSED METHODOLOGY

Particle Swarm Optimization and Backpropagation algorithm have their own merits and demerits. Although PSO trained neural network takes very less time in training in comparison to BPNN but after reaching near global optimum the particles many a times overshoot the optimum and get trapped in that region. Beyond that time, the training stops and the solution which may further be refined to get to the global optimum is compromised. On the other hand, backpropagation neural network is quite effective when it comes to the network to find the local optimum.. So, in this research work we have hybridized PSO-NN with backpropagation algorithm and the results were very much improved.

There are two steps involved in this process:

1. Training neural network with PSO learning algorithm, and
2. Weight optimization using BPNN.

4.1 Training neural network with PSO learning algorithm

PSO as a training method for neural network has been in research for many years as it is a faster learning approach that involves particles in the form of learners. Each particle learns from itself and their neighbours. Each particle first tries to explore its nearest optimum location and share the knowledge with all other particles on the search space. With each iteration passes the global knowledge gets better and better as each of them updates the global best location once its fitness value is better than the older one. The fitness function or cost function is taken to be:

$$f(y) = \sum_{i=0}^{n}\left(X_{Target} - X_{Predicted}\right)^2 \quad (4.1)$$

Here, $f(y)$is the cost function that needs to be minimized so that the error in prediction lowers and the network learns better. "n" is the total input size that is provided at the time of learning. The above equation is basically the mean square error (MSE) obtained after squaring the difference of predicted value and target value.A two-layered network (8-30-1) has been used with 30 neurons in the hidden layer and a single neuron in the output layer. The activation function or transfer function used is sigmoid function that keeps the values in range [0, 1]. The sigmoid function used is:

$$Sin(x) = \frac{1}{(1+e^{-x})} \quad (4.2)$$

The connection weights in the neural network are arranged in the form of a vector such that it will be easy for PSO algorithm to work on. Every particle in PSO algorithm has three parameters: a position, its velocity and a fitness value. The position of a particle in the search space is represented by the connection weights of the neural network. Initially all the positions and velocities are randomly distributed to every particle. Following is the example of a single layer neural network (3-5-1) with 3 input neurons, 5 hidden neurons and a single output neuron.

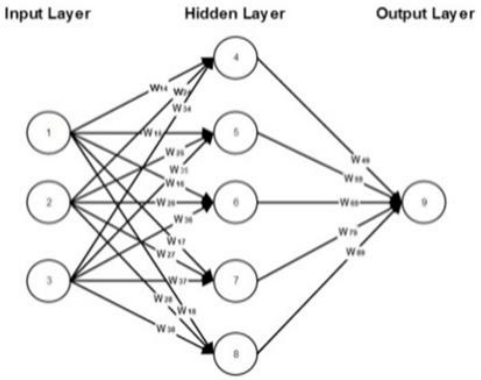

Figure 4-1: Representation of connection weights in neural network

Here, in the traditional BPNN hidden layer and output layer connection weights are represented as $\begin{bmatrix} W_{14} W_{15} W_{16} W_{17} W_{18} \\ W_{24} W_{25} W_{26} W_{27} W_{28} \\ W_{34} W_{35} W_{36} W_{37} W_{38} \end{bmatrix}$ and $[W_{49} W_{59} W_{69}\ W_{79} W_{89}]$ respectively. Analogous to this representation, the arrangement is changed for easy computation in PSO that is shown by the following representation:

$[W_{14}W_{24}W_{34}W_{15}W_{25}W_{35}W_{16}W_{26}W_{36}W_{17}W_{27}W_{89}]$ Thus the size of the position vector is defined by: $V_{size} = (No.of\ input\ neurons\ +\ No.of\ hidden\ neurons)\ +\ (No.of\ hidden\ neurons + No.of\ output\ neurons)$ (4.3)
The above two matrices are converted into a single vector so that the particle using the weights as a position vector in the search space can be easily updated and manipulated according to the algorithm. There are many other ways that can also be employed for the very same purpose [5].The connection weights are then trained through PSO algorithm discussed in above section. The weights being the position vector is updated according to the velocity of the particle. The velocity is affected by the personal best position of the particle and the global best location obtained so far. The training is stopped when a stopping criteria is fulfilled. In this thesis work the stopping criteria is given by the maximum number of iterations, I_{max} achieved or 400 iterations without updating in the global best location, whichever is earlier. When PSO training stops, the weight vector is passed to the backpropagation algorithm that will further optimize the weight vector for better accuracy. Following is the block diagram for PSO as a training method in neural network with stock market data as input (Figure 4-2).

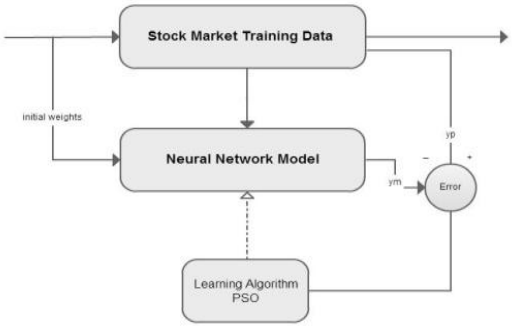

Figure 4-2: PSO trained neural network

The stock market data is given as input to the neural network model with randomly initialized weights to every particle that acts as a position vector for the particle. Similarly the velocity vector of same size is initialized randomly. The weights are updated in such a way that the error is minimised and the optimum solution reaches to the global optimum.

4.2 Weight optimization using BPNN

The traditional backpropagation network is used with the initials weights returned by the PSO-NN model. The background idea is if the initial weight is already optimized to a certain extent, it needs lesser than and computational resource to process further and reach a global optimum. The activation function and the fitness function used in BPNN are same as used in PSO-NN model. The advantage of BPNN network is that it converges well to the local optima. Thus, the resultant output of PSO-NN is given as initial weights to the backpropagation network so that the model would optimize the weights obtained so far through PSO and minimize mean square error. This algorithm is run until MSE is lesser than a fixed value or it exceeds 50000 iterations. Other steps are same as in traditional BPNN model to minimise the error in the result. Following are the basic steps involved in this model:

a) Fetch pattern x and input it to the input layer of BPNN
b) The input is then feed-forwarded to the output layer by multiplying it with respective weights.
c) Error, E is calculated at the output layer i.e. the difference between the target output and the predicted value.
d) The error gradient is calculated and the output layer weights are updated. It is propagated back to the hidden layer.
e) Error gradient for the hidden layer neurons are calculated and the hidden layer weights are updated.
f) The process is repeated till the stopping criterion is satisfied.

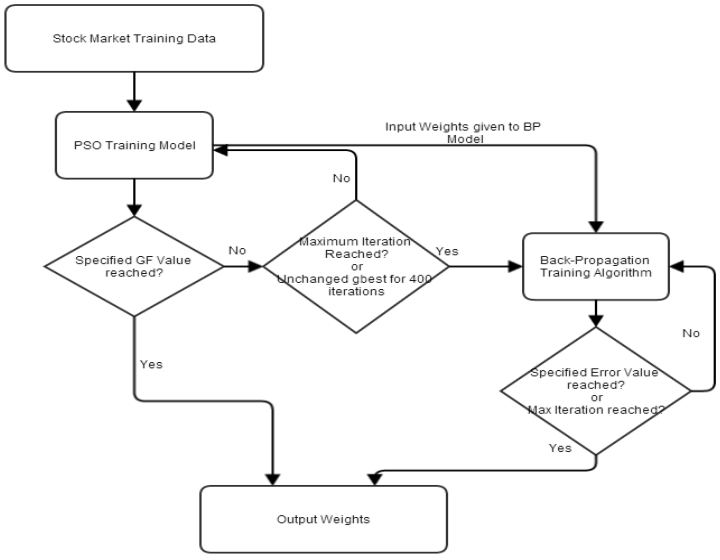

Figure 4-3: Block Diagram of working of the Hybridized PSO-NN and BPNN Model

The block diagram of the whole model is shown in figure 4.3. The stock market data is given as input in the form of a excel file to the PSO-NN model and after pre-processing and normalization of the data, the network is trained as per the given algorithm in section 4.1. After a certain stopping criterion is reached the weights obtained by the network is given as input to the BPNN model.

The backpropagation neural network model again processes the data to optimize the output until the mean square error *E*is reduced to *1e-4* or number of iteration exceeds 50000. After the network is trained it is tested with the next day parameters.

The normalized next days' parameters are given as input to the network which is then multiplied by the weights obtained as a result of the training and feed-forwarded to the output layer. The result here obtained is converted into the original form by de-normalizing it and then it is compared with the target output.

$Prediction\ error = |Target\ output - predicted\ output|$(4.4)

4.3 Algorithm of Hybridized PSO-NN and BPNN Model

Step 1: Set the initial parameters of ANN, number of hidden layers, number of neurons required in each layer. Define an activation function and cost function.

Step 2: Set the number of particles, *P* to be deployed in search space for PSO algorithm and set the initial parameters like$c_1\ and\ c_2$.

Step 3: Initialize the particles' position and velocity randomly in the range [0, 1] and calculate the initial fitness value for each particle and a global fitness value.

Step 4: While stopping criteria is reached i.e. 1500 iterations is completed or 400 iterations is completed without significant change in g_{best}

I. While the particles is lesser than *P*.

a) Calculate velocity of each particle by the given formula:

$$V^i(t+1) = V^i(t) + c_1 * rand_1 * [P^i{}_{best} - P^i(t)] + c_2 * rand_2 * [\, g_{best} - P^i(t)]$$

b) Update each particle's position by adding the velocity obtained by the above equation.

$$P^i(t+1) = P^i(t) + V^i(t)$$

c) Compute the local best position of each particle and reset$P^i{}_{best}$ for each particle with better fitness value is found.

d) Go to step I

II. Compute the best position among all the particles by comparing their own best location found so far. Comparison is done by evaluating the fitness value of each particle.

III. Update the global fitness value and the global best location to be utilized in next iteration.

a)If$f(\, g_{new}) \geq f(\, g_{best})$: Update $g_{best} = g_{new}$.

b)Else g_{best} remains constant.

IV. Go to Step 4.

Step 5: Initialize weights of the neural network in BPNN with the output of PSO-NN. Set the default parameters like learning rate, number of epochs, etc.

Step 6: While the number of iterations reaches 50000 or MSE <= *1e-4,* Repeat

a. Weight training is done to reduce the error in each iteration. The error is backpropagated in the networks to update its weight**.** Error is calculated as

$$e_k(p) = y_{d,k}(p) - y_k(p)$$

b. The error gradient at the output layer is calculated by the giveequation:

$$\delta_k(p) = y_k(p).[1 - y_k(p)]\,.e_k(p)$$

The correction in weights is computed as:

$$\Delta w_{jk}(p) = \propto .\, y_j(p).\delta_k(p)$$

The output layer weights are updated using

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p).$$

a. Compute the error gradient in the hidden layer by the equation:

$$\delta_j(p) = y_j(p).\big[1 - y_j(p)\big]\,.\sum_{k=1}^{l} \delta_k(p).w_{jk}(p)$$

Correction of weights is done by:

$$\Delta w_{ij}(p) = \alpha\,.x_i(p)\,.\delta_j(p)$$

Update the hidden layer weights:

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

b. Go to step 6.

Note: Refer to figure 4-4 for flow chart of the algorithm.

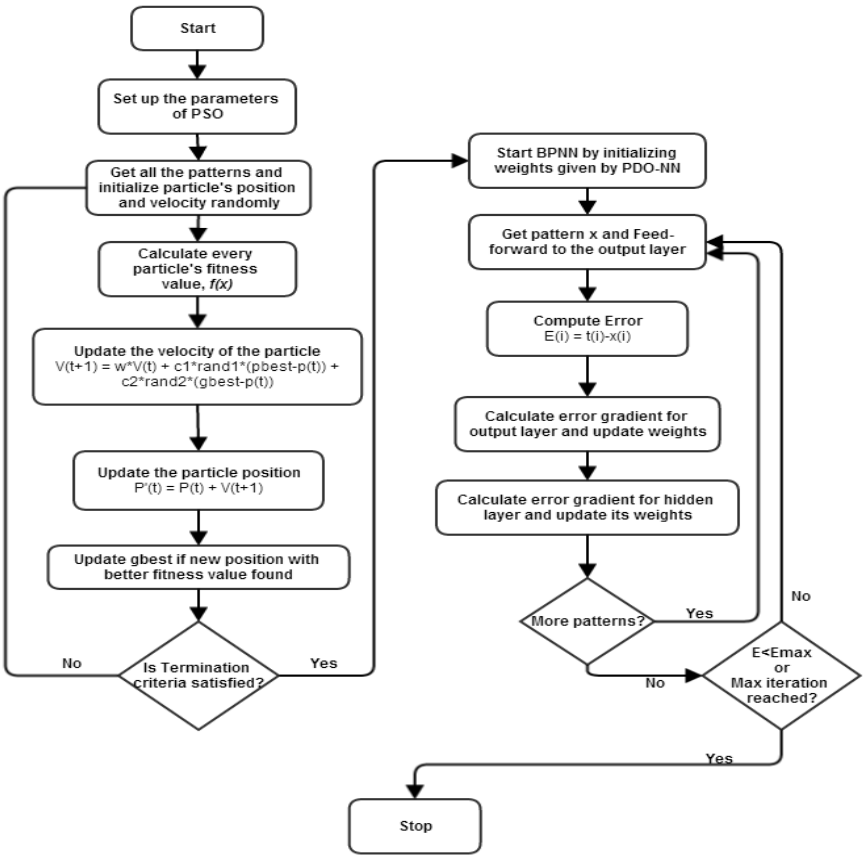

Figure 4-4: Flow-chart of PSO hybridized BPNN Model

V. IMPLEMENTATION

5.1 Tools and Technologies used

5.1.1. Hardware used

The hardware configuration used for the development of the proposed model is:
A core i3 CPU @ 3.30 GHz
6 GB RAM
The developed model is converted to binary and can run on any system that can run an executable file with a minimum of 128 MB of RAM.

5.1.2. Software used

CodeBlocks 12.11 IDE for C++

5.2 Analysis and development

The developed model is the combination of two different models PSONN and BPNN. PSONN and BPNN models are first developed and tested with the same sets of input and the comparison of the output is recorded. After analysing the advantages and learning the hybridization technique the proposed model is developed. Following are the steps explained in detail:

Pre-processing: The stock market data of 13 different companies from four different sectors are downloaded from http://www.finance.yahoo.com. The original data file contains the attributes as date, open, high, low, close, volume used, and adj. close. All the attributes are not required for the analysis and prediction of the future values. For better prediction some additional attributes need to be attached with the original attribute set. Thus, the date attribute, volume used, and adj. close attributes are ignored in the pre-processed data and also the data is aligned serially. The attributes like SMA 5 and EMA 5 are further added to the attribute set after computation from the observed data.

Development of PSO-NN model: After the dataset is finalised and is stored in a vector, the initial parameters of PSO is decided according to the algorithm.A PSO-NN model is developed where gradient descent algorithm is replaced by PSO for training the neural network. The connection weights in the neural network are arranged as the dimensions of the particles in PSO algorithm.

Development of BPNN model: The traditional model is also developed for the comparison of the results of PSO-NN model. A function is developed for the backpropagation algorithm to create the model that could comply with the outputs of PSO-NN model. As said earlier, the connection weights in PSO-NN model is arranged in vector form it is further used in the same form to train the single layer neural network using backpropagation algorithm

Hybridizing PSONN and BPNN to form PSO hybridized BPNN model: The disadvantage of PSO algorithm is that it doesn't guarantee global optima. It is verified with the results of the PSO-NN model that shows the uneven nature of the algorithm. For overcoming its shortcomings and utilizing the advantage of backpropagation algorithm of quickly

attaining local minima, a hybridised algorithm is developed involving both models PSO-NN and BPNN. The connection weights trained by PSO-NN model are given back to the BPNN model as input and again the result is optimized. The weights obtained as a result are used for testing the next days' prediction.

VI. RESULT ANALYSIS S AND DISCUSSION

6.1 Pre-processing

The original data source is http://finance.yahoo.com from where company's daily stock status is obtained in the form of a *.csv file (Table 6-1). The original dataset contains the parameters date of transaction (as Date), open price (as Open), highest price (as High), lowest price (as Low), closing price (as Close), Volume used (as Volume), Adjacent close (as Adj Close). The data here is arranged in reversed form from recent to the oldest data. This dataset is taken input as a whole to the model that then pre-processes it and converts it to the format used by the developed computational model for prediction. The historical raw dataset of any stock market contains attributes date of trade (Date), date wise open price (Price), date wise low price (Low), date wise high price (High), date wise closing price (Close), date wise volume (Volume), date wise adjacent close (Adj. Close).the data set of any stock market is arranged in inverse order from present date wise date to oldest date wise

Table 6-1: Original input dataset format

Date	Open	High	Low	Close	Volume	Adj Close
23/05/2015	2276	2317.8	2276	2308.7	545900	2308.7
22/05/2015	2316.5	2325	2280	2296.1	726100	2296.1
21/05/2015	2320	2328.45	2280	2300.05	687100	2300.05
20/05/2015	2284.9	2354.95	2266.15	2323.25	647000	2323.25
19/05/2015	2410	2421	2246.5	2286.15	1047300	2286.15
18/05/2015	2464	2495	2400	2406.9	1089100	2406.9
17/05/2015	2515	2543.85	2466.6	2481.6	663800	2481.6
16/05/2015	2600	2604.5	2513.75	2518.35	1040200	2518.35
15/05/2015	2740	2740.55	2600.75	2610.75	995500	2610.75
14/05/2015	2720	2742	2711.5	2722.9	365900	2722.9
13/05/2015	2738	2738	2694.7	2718	631800	2718
12/05/2015	2743.9	2747.95	2725.55	2737.15	306700	2737.15
11/05/2015	2744.35	2754	2727.55	2742.2	350300	2742.2
10/05/2015	2740	2755	2727.15	2744.35	163500	2744.35
09/05/2015	2750	2750	2720.2	2729.15	232700	2729.15
08/05/2015	2692.1	2749.9	2692.1	2737.35	289700	2737.35
07/05/2015	2706.15	2706.15	2706.15	2706.15	0	2706.15
06/05/2015	2665	2781.75	2664.65	2706.15	471300	2706.15
05/05/2015	2680	2689.35	2640	2664.65	205500	2664.65
04/05/2015	2646	2684	2642	2676.75	382500	2676.75
03/05/2015	2590	2638.8	2590	2629.25	617600	2629.25
02/05/2015	2565.9	2565.9	2565.9	2565.9	0	2565.9

The pre-processed data is now arranged in sequence fromthe older transaction to the recent transaction so that the model learns serially to predict the next day's closing price. The dataset that is fed input to the model contains the parameters like Open, High, Low, close, SMA5, EMA5, and Target (Table 6-2). Target is given along with the input to simultaneously check the predicted output and compute the error.

Table 6-2: Pre-processed input dataset format

Open	High	Low	Close	SMA5	EMA5	Target
1922	1959	1922	1945.3	1934.5	1933.9	1912.35
1961	1961	1906	1912.35	1928.82	1926.71	1899.85
1912.35	1912.35	1895.05	1899.85	1921.22	1917.76	1899.25
1908	1919.5	1888.1	1899.25	1914.53	1911.59	1895.2
1895.1	1913.5	1883	1895.2	1910.39	1906.13	1855.4
1884.1	1900	1847.4	1855.4	1892.41	1889.22	1898.85
1873.8	1909.9	1871.25	1898.85	1889.71	1892.43	1915.05
1921.1	1921.2	1903.2	1915.05	1892.75	1899.97	1905.2
1920	1920	1873.05	1905.2	1893.94	1901.71	1874.35
1906	1908	1852.45	1874.35	1889.77	1892.59	1818.9
1870	1878.8	1812	1818.9	1882.47	1868.03	1802.2
1820.95	1830.65	1792.3	1802.2	1863.14	1846.09	1823.5
1803	1828.8	1788	1823.5	1844.83	1838.56	1825.8
1827.95	1836.9	1812.3	1825.8	1828.95	1834.3	1821.2
1823.8	1831.9	1800	1821.2	1818.32	1829.94	1816.2
1815	1824.95	1790.1	1816.2	1817.78	1825.36	1804.35
1805	1814	1791.55	1804.35	1818.21	1818.36	1769.15
1803.1	1808.3	1758	1769.15	1807.34	1801.95	1755.85
1752.5	1768.95	1733.45	1755.85	1793.35	1786.59	1755.45
1768	1779.9	1725.1	1755.45	1780.2	1776.21	1741.85
1752	1772	1720.4	1741.85	1765.33	1764.75	1765.1
1741.1	1772	1738.9	1765.1	1757.48	1764.87	1793.95
1765	1804.5	1751.3	1793.95	1762.44	1774.56	1809.8
1799.9	1814.9	1780	1809.8	1773.23	1786.31	1808.7
1802	1824	1793.55	1808.7	1783.88	1793.77	1819.2
1817	1829.4	1782.05	1819.2	1799.35	1802.25	1813.15
1819	1819	1801	1813.15	1808.96	1805.88	1817.15
1810	1828.5	1808.9	1817.15	1813.6	1809.64	1807.95
1816	1822.9	1799.9	1807.95	1813.23	1809.08	1817
1808	1822.95	1801.05	1817	1814.89	1811.72	1822.65
1822	1834	1806	1822.65	1815.58	1815.36	1802.9
1791	1822.75	1785.65	1802.9	1813.53	1811.21	1797.85
1345	1349.6	1303.9	1309.55	1318.55	1319.94	1315.7
1309.55	1337.8	1306.85	1315.7	1324.03	1318.52	1315.7

6.2 Choice of single hidden layer

A two layered artificial neural network has been used to frame this model and maximize its accuracy of prediction. The choice of single hidden layer is made by comparing the training results of a single hidden layer model and double hidden layer model. The single hidden layer model outperforms the double layered while training the model. Following is the comparison in the form of graphical representation that indicates the single layered model learns faster than the double layered model and reached a global optimum quickly (Figure 6-1).

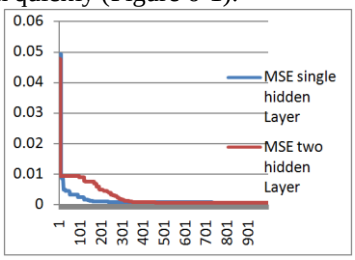

Figure 6-5: Comparison of error among single and double hidden layer model

6.3 Comparison of Models

The single hidden layer with 30 neurons is decided by trial and error that is used for developing the model. Three models were developed to compare the effectiveness and accuracy of the hybridized model. PSO-NN involved artificial neural network trained using particle swarm optimization technique. BPNN is the traditional artificial neural network that uses gradient descent for backpropagation of error. At last these two models were hybridized to form hybridized PSO-BPNN model that utilizes both of their advantages and cancels out their demerits. The measures used for comparison of the models are Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), and Mean Absolute Error (MAE). In all the cases the hybridized model outperforms the other models. Following table 6-3 shows the comparison of Mean Absolute Percentage Error (MAPE) in PSO-NN, BPNN and PSO hybridized BPNN while testing.

Table 6-3: Comparison of models in terms of Mean Absolute Percentage Error (MAPE)

Following table 6-4 shows the comparison of Root MeanSquare Error (RMSE)in PSO-NN, BPNN and PSOhybridized BPNN while testing.

Table 6-4: Comparison of models in terms of Root Mean Square Error (RMSE)

Company	PSO-NN	BPNN	PSO Hybridized BPNN Model
SBI	146.58050	44.49194	46.87563
PNB	89.44634	20.62597	21.42487
ICICI	83.47983	62.42907	26.81232
YES	40.03723	24.01010	12.45148
TCS	102.32437	42.53773	38.77169
WIPRO	33.36311	17.42810	10.46840
INFI	200.46194	154.44814	80.22282
MARUTI	136.90134	43.20709	43.94092
TATAMOTOR	23.73897	11.78173	7.91796
RANBAXY	24.30631	16.81222	13.33102
SUN	26.96102	18.31653	11.51637
DRREDDY	133.40312	79.73334	49.15982
NIFTY	253.14092	274.39281	86.75966

Following table 6-5 shows the comparison of Mean Absolute Error (MAE)in PSO-NN, BPNN and PSO hybridized BPNN while testing.

Table 6-4: Comparison of models in terms of Mean Absolute Error (MAE)

Company	PSO-NN	BPNN	PSO Hybridized BPNN Model
SBI	94.54793	29.89327	31.42840
PNB	54.06489	14.11423	14.39975
ICICI	56.98154	25.45804	19.74856
YES	26.53033	10.37455	9.06231
TCS	85.28113	27.75480	27.16687
WIPRO	27.87754	9.54793	8.07447
INFI	171.27080	81.69380	59.42493
MARUTI	99.74407	27.45647	27.07647
TATAMOTOR	19.86101	6.66486	5.81215
RANBAXY	18.90712	10.66785	9.21199
SUN	22.95667	12.21205	9.03547
DRREDDY	116.91100	45.42780	37.70973
NIFTY	203.71540	109.87280	64.72813

The graphical representation of error makes difference sclear about which model is better in the representation shown below, all the error measures used shows lesser rise in the hybridized model in comparison to the other models where the peak is very high. The model PSO-NN and BPNN shows different behaviour in different dataset. They cannot be relied onto which dataset will perform better in which condition. Thus, we concentrated on the hybridized model that shows consistent performance throughout the dataset. Following is the graphical representation of comparison of the models PSO-NN, BPNN and PSO hybridized BPNN while testing in terms of Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), and Mean Absolute Error (MAE).

VII. CONCLUSION AND FUTURE WORKS

In this research work, we have proposed a hybridized model combining PSO algorithm and backpropagation algorithm that utilizes the PSO algorithm's ability of quickly searching global optimum and backpropagation algorithm's ability to effectively search its local optima. The weights of the neural network are manipulated to be used in PSO-NN as well as BPNN simultaneously. Initially weights are assigned random values and passed to the PSO-NN model that updates it and reaches to the global optimum and then it is given as input to BPNN model that optimizes it further. PSO makes the learning faster in comparison to gradient descent algorithm. Thus it reduces CPU time and makes the overall process faster.The "PSO hybridized BPNN" model shows great improvement in the prediction accuracy. Also PSO is very effective in training the model quickly and further it is optimized very well by BPNN to reach global optimum. This model is tested on different companies' data belonging to different sectors like banking, IT, automobile, etc. which shows a promising 98.25% accuracy in prediction. The performance comparison of PSO-NN, BPNN and PSO hybridized BPNN indicates that the developed model outperforms the other models in terms of mean absolute percentage error.

The proposed model is applied in different sectors where IT sector shows least accuracy and can be further analysed for better performance. This model can be further improved by providing dynamic input parameters that affect stock market in real time. In this research work we have used only Indian stock market data for training and testing the model. It can further be extended to use different types of data too.

REFERENCES

[1] Contreras et al., "ARIMA models to predict next-day electricity prices," *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1014–1020, Aug. 2003.

[2] W. W. S. Wei, *Time series analysis: univariate and multivariate methods*. Pearson Addison Wesley, 2006.

[3] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, Jan. 2003.

[4] Y. Chen et al., "How external factors influence stock market: A model based SVM," in *2010 IEEE International Conference on Service Operations and Logistics and Informatics (SOLI)*, 2010, pp. 325–329.

[5] J.R. Zhang et al., "A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training," *Applied Mathematics and Computation*, vol. 185, no. 2, pp. 1026–1037, Feb. 2007.

[6] "Time series," *Wikipedia, the free encyclopedia*. 02-March-2014

[7] "NIST/SEMATECH e-Handbook of Statistical Methods," *NIST*. [Online]. Available: http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc42.htm, 08-March-2014

[8] "Exponential smoothing," *Wikipedia, the free encyclopedia*. 11-March-2014

[9] Paul A. Jensen and Jonathan F. Bard, "Time Series and Forecasting," in *Operations Research Models and Methods*, 2003.

[10] A. Pankratz, *Forecasting with Univariate Box - Jenkins Models: Concepts and Cases*. John Wiley & Sons, 2009.

[11] Ajoy K. Palit and DobrivojePopovic, *Computational Intelligence in Time Series Forecasting - Theory and Engineering Applications*. 2005.

[12] K. Kohara et al., "Stock Price Prediction Using Prior Knowledge and Neural Networks," *Int. J. Intell. Syst. Acc. Fin. Mgmt.*, vol. 6, no. 1, pp. 11–22, Mar. 1997.

[13] G. Zhang et al., "Forecasting with artificial neural networks:: The state of the art," *International Journal of Forecasting*, vol. 14, no. 1, pp. 35–62, Mar. 1998.

[14] X. Wang et al., "Stock market prediction using neural networks: Does trading volume help in short-term prediction?," in *Proceedings of the International Joint Conference on Neural Networks, 2003*, 2003, vol. 4, pp. 2438–2442 vol.4.

[15] K. Kim and W. B. Lee, "Stock market prediction using artificial neural networks with optimal feature transformation," *Neural Comput&Applic*, vol. 13, no. 3, pp. 255–260, Sep. 2004.

[16] G. Armano et al., "A hybrid genetic-neural architecture for stock indexes forecasting," *Information Sciences*, vol. 170, no. 1, pp. 3–33, Feb. 2005.

[17] M. R. Hassan et al., "A fusion model of HMM, ANN and GA for stock market forecasting," *Expert Systems with Applications*, vol. 33, no. 1, pp. 171–180, Jul. 2007.

[18] B. Junyou, "Stock Price forecasting using PSO-trained neural networks," in *IEEE Congress on Evolutionary Computation, 2007. CEC 2007*, 2007, pp. 2879–2885.

[19] X. Zhang et al., "Stock Index Forecasting Using PSO Based Selective Neural Network Ensemble.," in *IC-AI*, 2007, pp. 260–264.

[20] R. Majhi et al., "Prediction of S&P 500 and DJIA stock indices using Particle Swarm Optimization technique," in *IEEE Congress on Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence)*, 2008, pp. 1276–1282.

[21] L. Yu et al., "A neural-network-based nonlinear metamodeling approach to financial time series forecasting," *Applied Soft Computing*, vol. 9, no. 2, pp. 563–574, Mar. 2009.

[22] V. Nekoukar and M. T. Hamidi Beheshti, "A local linear radial basis function neural network for financial time-series forecasting," *ApplIntell*, vol. 33, no. 3, pp. 352–356, Dec. 2010.

[23] P.C. Chang et al., "A neural network with a case based dynamic window for stock trading prediction," *Expert Systems with Applications*, vol. 36, no. 3, Part 2, pp. 6889–6898, Apr. 2009.

[24] G. S. Atsalakis and K. P. Valavanis, "Forecasting stock market short-term trends using a neuro-fuzzy based methodology," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10696–10707, Sep. 2009.

[25] Y. Zhang and L. Wu, "Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network," *Expert Systems with Applications*, vol. 36, no. 5, pp. 8849–8854, Jul. 2009.

[26] W. K. Wong et al., "Adaptive neural network model for time-series forecasting," *European Journal of Operational Research*, vol. 207, no. 2, pp. 807–816, Dec. 2010.

[27] Z. Fangwen et al., "Financial time series prediction based on Echo State Network.," pp. 3983–3987, 2010.

[28] E. Guresen et al., "Using artificial neural network models in stock market index prediction," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10389–10397, Aug. 2011.

[29] J. L. Ticknor, "A Bayesian regularized artificial neural network for stock market forecasting," *Expert Systems with Applications*, vol. 40, no. 14, pp. 5501–5506, Oct. 2013.

[30] C. S. Vui et al., "A review of stock market prediction with Artificial neural network (ANN)," in *2013 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, 2013, pp. 477–482.

[31] G. Sermpinis et al., "Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and Particle Swarm Optimization," *European Journal of Operational Research*, vol. 225, no. 3, pp. 528–540, Mar. 2013.

[32] B. Lu, "A stock predition method based on PSO and BP hybrid algorithm," in *2011 International Conference on E -Business and E -Government (ICEE)*, 2011, pp. 1–4.

[33] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *, IEEE International Conference on Neural Networks, 1995. Proceedings*, 1995, vol. 4, pp. 1942–1948.

[34] DiptamDutt et al., "Training Artificial Neural Network using Particle Swarm Optimization Algorithm," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 3, pp. 430–434, Mar. 2013.

[35] J. Liu and X. Qiu, "A Novel Hybrid PSO-BP Algorithm for Neural Network Training," in *International Joint Conference on Computational Sciences and Optimization, 2009. CSO 2009*, 2009, vol. 1, pp. 300–303.

[36] M. Reyes-sierra and C. A. C. Coello, "Multi-Objective particle swarm optimizers: A survey of the state-of-the-art," International Journal of Computational Intelligence Research, vol. 2, no. 3, pp. 287–308, 2006.

[37] D. PalupiRini et al., "Particle Swarm Optimization: Technique, System and Challenges," *International Journal of Computer Applications*, vol. 14, no. 1, pp. 19–27, Jan. 2011.

[38] X.S. Yang, *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, 2008.

[39] Z.-H. Zhan et al., "Adaptive Particle Swarm Optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 6, pp. 1362–1381, Dec. 2009.

[40] "Artificial neural network," *Wikipedia, the free encyclopedia*. 21-March-2014.

