Chapter 2

Developed Feed Forward Neural Network models for Foreign Exchange Rate Prediction

2.1 Introduction

Over the past few decades, the scenario of foreign exchange markets has reached its height in the financial time series prediction and analysis. Exchange rates are very essential for traders in respect of purchasing the needed international goods. Since, the currency rates are traded twenty four hours a day over the globe, traders are highly interested in predicting the exchange rates at which the currencies need to be exchanged. Understanding the daily movement of exchange rates becomes a challenging task for most of the researchers and as well for business people. Also, the exchange rate predictions are unpredictable due to noise and high ambiguity that exist in the growing markets. Hence, numerous researches are carried out in the field of predicting the foreign exchange market rates.

Artificial Neural Networks (ANN) can predict the future foreign exchange rates that are more complex to be predicted by the traditional models. Amidst, Neural Networks (NN) possessing several advantages, they have some limitations as well. The disadvantage of neural networks is that the entire process is time consuming and the outputs rely on trial and error method. At this juncture, the developed Radial Basis Function (RBF) neural network for forecasting application in this thesis is a powerful, fast learning, and self-organized neural network model. It is better than other neural network models with respect to approximation, classification and learning speed, with more focus to processing application.

In this chapter, a Radial Basis Function Neural Network (RBFNN) model is developed to forecast the daily foreign exchange rate of United States Dollars (USD) in terms of Indian rupees in India during the period 2009-2014. Here, seven technical indicators which
include - simple moving average of one week, two week, momentum, price rate of change, disparity 7, disparity 14, and price oscillator are proposed as inputs for forecasting the time series model. Further, this chapter compares the four models namely Pattern Recognition Networks, Feed Forward Back Propagation Networks, Feed Forward Networks with no feedback, and modeled Radial Basis Function Network to forecast the daily currency exchange rate during the considered period 2009 - 2014. The performances of all the said models are analyzed based on the accuracy measures namely Mean Square Error, Mean Absolute Error, Sum Square Error and Root Mean Square Error. The computed simulation results prove that the average performance of developed Radial Basis Function neural network for this prediction application is considerably better than that of the other networks considered for comparison.

2.2 Description of Datasets and Technical Indicators

The developed radial basis function neural model and the other feed forward models are applied in this chapter for predicting the forthcoming foreign exchange rates. The datasets employed to test the proposed model along with the technical indicators (performance indices) are presented in this section.

2.2.1 Foreign Exchange Rate Datasets

The datasets employed in this research work is collected from the Indian Reserve Bank website (www.rbi.org.in). The data of foreign exchange rate (Indian Rupee equivalent of USD) is considered for the period from (01.01.2009 to 30.04.2014). This covers total number of 1285 trading days between 01.01.2009 to 30.04.2014. The developed ANN models in this thesis aim at predicting one day ahead i.e., next day’s exchange rate of USD in Indian rupees. Table 2.1 presents the sample datasets of foreign exchange rate used in this thesis. This dataset with 1285 samples is used during the implementation of the proposed model and are divided into the required training dataset, validation dataset and testing dataset.
<table>
<thead>
<tr>
<th>Date</th>
<th>USD in Indian Rupees</th>
<th>Date</th>
<th>USD in Indian Rupees</th>
<th>Date</th>
<th>USD in Indian Rupees</th>
</tr>
</thead>
<tbody>
<tr>
<td>30/04/2014</td>
<td>60.3375</td>
<td>9/12/2013</td>
<td>61.1785</td>
<td>31/07/2012</td>
<td>55.807</td>
</tr>
<tr>
<td>29/04/2014</td>
<td>60.5253</td>
<td>29/11/2013</td>
<td>62.3948</td>
<td>22/05/2012</td>
<td>54.8845</td>
</tr>
<tr>
<td>28/04/2014</td>
<td>60.5041</td>
<td>30/10/2013</td>
<td>61.4871</td>
<td>16/04/2012</td>
<td>51.659</td>
</tr>
<tr>
<td>25/04/2014</td>
<td>61.1163</td>
<td>24/09/2013</td>
<td>62.6585</td>
<td>15/02/2012</td>
<td>49.252</td>
</tr>
<tr>
<td>27/03/2014</td>
<td>60.1295</td>
<td>12/8/2013</td>
<td>60.8025</td>
<td>20/10/2011</td>
<td>49.711</td>
</tr>
<tr>
<td>26/03/2014</td>
<td>60.1725</td>
<td>26/07/2013</td>
<td>58.9133</td>
<td>12/7/2011</td>
<td>44.6878</td>
</tr>
<tr>
<td>25/03/2014</td>
<td>60.4935</td>
<td>10/6/2013</td>
<td>57.782</td>
<td>16/12/2010</td>
<td>45.39</td>
</tr>
<tr>
<td>24/03/2014</td>
<td>60.703</td>
<td>14/05/2013</td>
<td>54.6275</td>
<td>8/9/2010</td>
<td>46.7</td>
</tr>
<tr>
<td>4/2/2014</td>
<td>62.6815</td>
<td>9/4/2013</td>
<td>54.4613</td>
<td>20/05/2010</td>
<td>46.75</td>
</tr>
<tr>
<td>3/2/2014</td>
<td>62.6891</td>
<td>21/03/2013</td>
<td>54.281</td>
<td>10/2/2010</td>
<td>46.56</td>
</tr>
<tr>
<td>30/01/2014</td>
<td>62.7335</td>
<td>1/1/2013</td>
<td>54.832</td>
<td>1/1/2009</td>
<td>48.73</td>
</tr>
</tbody>
</table>

Table 2.1 Sample foreign exchange rate datasets

2.2.2 Technical indicators employed in the developed RBFNN model

In order to predict the future markets numerous technical indicators (performance indices or metrics) are available. Many investors use different indicators for forecasting the future foreign exchange rates. In this research thesis, seven technical indicators are used as inputs to the network for forecasting the daily change in the foreign exchange rate based on the considered time period and dataset as suggested by prior researchers [4], [20] and [162]. The descriptions of the employed technical indicators are presented in Table 2.2.
<table>
<thead>
<tr>
<th>Sl.No.</th>
<th>Technical Indicators</th>
<th>Description</th>
<th>Formula for Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Simple moving average for 1 week [20]</td>
<td>Average of currency rates for 1 week</td>
<td>[ \frac{C_t + C_{t-1} + C_{t-2} + C_{t-3} + C_{t-4} + C_{t-5} + C_{t-6} + C_{t-7}}{7} ]</td>
</tr>
<tr>
<td>2.</td>
<td>Simple moving average for 2 week [20]</td>
<td>Average of currency rates for 2 weeks</td>
<td>[ \frac{C_t + C_{t-1} + C_{t-2} + \ldots + C_{t-14}}{14} ]</td>
</tr>
<tr>
<td>3.</td>
<td>Momentum [4]</td>
<td>Gives the quantity that the currency rates have altered over a time t</td>
<td>[ C_t - C_{t-4} ]</td>
</tr>
<tr>
<td>4.</td>
<td>Price rate of change (ROC) [20]</td>
<td>Gives the difference between the current rate and the rate at 4 days ago.</td>
<td>[ \frac{C_t \times 100}{C_{t-4}} ]</td>
</tr>
<tr>
<td>5.</td>
<td>Disparity 7 (7 day disparity) [4]</td>
<td>It displays the distance of current rate and the moving average of 7 days</td>
<td>[ \frac{C_t \times 100}{MA7} ]</td>
</tr>
<tr>
<td>6.</td>
<td>Disparity 14 (14 day disparity) [4]</td>
<td>It displays the distance between the current rate and the moving average of 14 days</td>
<td>[ \frac{C_t \times 100}{MA14} ]</td>
</tr>
<tr>
<td>7.</td>
<td>Price Oscillator [4]</td>
<td>Distance between moving average of 7 days and moving average of 14 days</td>
<td>[ \frac{MA7 - MA14}{MA7} ]</td>
</tr>
</tbody>
</table>

\( C_t \) is the closing price at time \( t \) and \( MA \) is the moving average.

Table 2.2 Description of technical indicators
To implement the developed NN models, normalization process is applied for all the data samples. The input data samples are converted between the range (0.0 to 1.0) in order to have zero mean and unity standard deviation. On evolving the output, then those samples are converted back to normal units. The statistical values of the considered data samples calculated for the technical indicators given in Table 2.2 is presented in Table 2.3.

<table>
<thead>
<tr>
<th>Technical Indicator</th>
<th>Maximum value</th>
<th>Minimum value</th>
<th>Mean value</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple moving average for 1 week</td>
<td>66.923</td>
<td>6.961</td>
<td>51.091</td>
<td>6.190</td>
</tr>
<tr>
<td>Simple moving average for 2 week</td>
<td>65.896</td>
<td>3.480</td>
<td>50.92661</td>
<td>6.546</td>
</tr>
<tr>
<td>Momentum</td>
<td>3.673</td>
<td>-2.812</td>
<td>0.0275</td>
<td>0.553</td>
</tr>
<tr>
<td>Price rate of change (ROC)</td>
<td>105.678</td>
<td>95.304</td>
<td>99.822</td>
<td>4.938</td>
</tr>
<tr>
<td>Disparity 7 (7 day disparity)</td>
<td>700</td>
<td>96.861</td>
<td>100.919</td>
<td>18.667</td>
</tr>
<tr>
<td>Disparity 14 (14 day disparity)</td>
<td>1400</td>
<td>95.276</td>
<td>102.566</td>
<td>42.431</td>
</tr>
<tr>
<td>Price Oscillator</td>
<td>0.5</td>
<td>-0.0241</td>
<td>0.00443</td>
<td>0.0418</td>
</tr>
</tbody>
</table>

Table 2.3 Statistical values of the considered dataset with respect to the technical indicators

2.3 Developed Neural Network Models for foreign exchange rate prediction

This thesis evolved four neural network models – pattern recognition model, feed forward back propagation model, feed forward network with no feedback and finally the radial basis function neural network model for predicting the foreign exchange rate equivalent to Indian rupees for the real time datasets considered over the period from 2009 to 2014.
Fundamentally, artificial neural networks are a kind of artificial intelligence models that are designed based on the processing that takes place in a human brain. The artificial neural model comprises of highly interconnected processing elements called neurons which process the information to solve the specific problems. The artificial neurons are connected by means of connection links. Each connection link possesses a weight. Each input signal of the neuron is multiplied with the corresponding weights. Bias is added to the net to improve the performance of the neural network. Then these products are summed and fed through an activation function to compute the result. Several activation functions like identity function, binary sigmoidal function, bipolar sigmoidal function, linear activation function, discrete linear and continuous linear function and so on are available. Basically, there are two basic types of networks, feedback networks and feed forward networks. In feedback networks, the output values are propagated back to input values, whereas in feed forward networks, for every input value given to the net, the corresponding output value is calculated which can be read from output neurons. Here feedback is not done.

### 2.3.1 Pattern Recognition model

The ultimate aim of pattern recognition machine learning model is to construct a neuronal model which makes predictions based on the earlier expertise values in the existence of uncertainty. Conventional adaptive algorithms tend to identify pattern in data and a machine or a computer learns from the observations. When the machine model is subject to numerous observations, then the learning improves the predictive performance. Generally, a pattern recognition model considers a known set of input data and the known responses from the previous instances, and then trains a model for generating suitable predictions in response to the new data. The pattern recognition model splits into two main modules: classification and regression.

In classification module, the aim is to assign a class or label from a given finite set of class modules pertaining to an observation. The response of these will be categorical variables. These classification modules tend to apply nominal response values. In case of regression process, the aim is to predict a continuous measurement for an observation. Here, the responses are real numbers.
2.3.1.1 Steps involved in pattern recognition model

The various steps involved in the pattern recognition process are as follows:

Step 1: Prepare the data based on the considered datasets. All the pattern recognition networks start with an input data matrix and the row of this matrix is called as the observation and the column of this matrix represents a variable or a predictor. For representing the output response data, in case of regression this response data must be a numeric entity with the same number of elements as that of the number of rows in input data matrix. In case of classification process, the output response data can be of any data type. In the proposed foreign exchange rate prediction application, this is of numeric type.

Step 2: Apply the classification tree and regression tree module to perform the operational process. This process is to ensure the speed of training, memory usage, predictive accuracy of new data and interpretability.

Step 3: Fit the predictive model. The fitting function depends on the classification algorithm.

Step 4: Perform validation by choosing a validation model. This process examines the resubstitution error and then cross validates a regression tree.

Step 5: Modify the fitting parameters to obtain a more accurate model until the error reaches a minimal value.

Step 6: Employ the fitted model for predictions.

Thus, adopting the above mentioned steps the pattern recognition model is employed for predicting the foreign exchange rate in this thesis work.

2.3.2 Feed Forward Back Propagation Neural Network (FFBPNN) model

Feed forward back propagation neuronal model is a systematic method to train multilayer artificial neural networks. FFBPNN multi-layer forward network is employed in this work that uses gradient – descent learning rule and is as well known as back propagation (of errors) rule. Back propagation model is noted to be a computationally efficient
method for changing the weights in a feed forward network, with differentiable activation function units, to learn a training set of input-output examples. The most commonly used activation function in FFBPNN is sigmoidal activation function and is of continuous form. Being a gradient descent method this network minimizes the total squared error of the output computed by the neural net. The network is a supervised learning network and the ultimate aim is to train the net to achieve a balance between the ability to respond correctly to the input patterns that are used for training and the ability to provide good responses to the input that are similar.

2.3.2.1 Architecture of FFBPNN

A multi layer feed forward back propagation network with one layer of z-hidden units is shown in Figure 8.1. The Y output unit has $W_{ok}$ bias and Z hidden unit has $V_{ok}$ as bias. In FFBPNN model both the output units and the hidden units have bias. The bias term acts like weights on connection from units whose output is always 1. In Figure 2.1, it is observed that the network has one input layer, one hidden layer and one output layer. There can be any number of hidden layers, but increase in the number of hidden layers tends to increase the complexity of the network. The input layer gets connected to the hidden layer and the hidden layer in turn gets connected to the output layer by means of interconnection weights. Figure 2.1 shows only the feed forward phase of operation and during the back propagation phase of learning, the signals are transferred in the reverse direction. There exists bias in the layers to act upon the net input to be calculated.

Fig 2.1 Architecture of Feed Forward Back Propagation Neural Network
2.3.2.2 Training Algorithm of FFBPNN model

The training Algorithm of FFBPNN model involves four phases,

- Initialization of weights
- Feed forward phase
- Error Back Propagation phase and
- Weight and bias updation phase.

At the beginning, the initialization of weights is carried out i.e., some small random values are assigned to all the interconnected weighted links. During feed forward phase, each input unit \( X_i \) receives an input signal and the signals are transmitted to each of the hidden units \( z_1 \ldots z_p \). Each hidden unit then computes the activation function and sends its signal \( z_j \) to each of the output unit. Then the output unit computes the activation function to form the response of the net for the given input pattern.

At the error back propagation phase, each output unit compares its computed activation \( y_k \) with its target value \( t_k \) for determining the associated error for that pattern with that unit. Based on the error computed, the factor \( \delta_k \ (k = 1, \ldots, m) \) is computed and is used to distribute the error at output unit \( y_k \) back to all units in the previous layer. In a similar manner, the factor \( \delta_j \ (j=1, \ldots, p) \) is computed for each hidden unit \( z_j \). At the final stage, the weight and biases are updated using the \( \delta \) factor and the activations obtained. The parameters and the training algorithm employed in the FFBPNN is as follows. The algorithm is given with the various phases. The various parameters used in the training algorithm are as follows:

\[
x: \quad \text{Input training vector:} \\
x: \quad (x_1, \ldots, x_i, \ldots, x_n) \\
t: \quad \text{Output target vector} \\
t: \quad (t_1, \ldots, t_k, \ldots, t_m) \\
\delta_k: \quad \text{error at output unit } y_k \\
\delta_j: \quad \text{error at hidden unit } z_j
\]
\( \infty \): learning rate

\( V_{oj} \): Bias on hidden unit j

\( z_j \): Hidden unit J

\( w_{ok} \): Bias on output unit k

\( \gamma_k \): output unit k.

Step 1: Initialize weight to small random values.

Step 2: While stopping condition is false do steps 3-10

Step 3: For each of the input: output training pair do steps 4-9

Step 4: Each input unit receives the input signals \( x_i \) and transmit this signals to all units in the layer above i.e. hidden units

Step 5: Each hidden unit \( (z_j, j=1, \ldots, p) \) sums its weighted input signals

\[
 z_{-inj} = v_{oj} + \sum_{i=1}^{n} x_i \cdot v_{ij} \quad (2.1)
\]

Applying activation function,

\[
 Z_j = f(z_{inj}) \quad (2.2)
\]

and sends this signal to all units in the layer above i.e. output units.

Step 6: Each output unit \( (y_k, k=1, \ldots, m) \)

\[
 y_{-inj} = w_{ok} + \sum_{j=1}^{p} z_j \cdot w_{jk} \quad (2.3)
\]

and applies its activation function is used to calculate the output signals.

\[
 Y_k = f(y_{-ink}) \quad (2.4)
\]

Step 7: Each output unit \( (y_k, k=1, \ldots, m) \) receives a target pattern corresponding to an input pattern, error information term is calculated as
\( \delta_k = (t_k-y_k) f'(y-ink) \)  \hspace{1cm} (2.5)

Step 8: Each hidden unit \( (z_j, j=1, \ldots, n) \) sums its delta inputs from units in the layer above

\[ \delta_{-inj} = \sum_{k=1}^{m} \delta_j w_{jk} \]  \hspace{1cm} (2.6)

error information term is calculated as,

\[ \delta_j = \delta_{-inj} f'(z_{-inj}) \]  \hspace{1cm} (2.7)

Step 9: Each output unit \( (y_k, k=1, \ldots, m) \) updates its bias and weights \( (j=0, \ldots, p) \) and the weight correction term is given by,

\[ \Delta W_{jk} = \alpha \delta_k z_j \]  \hspace{1cm} (2.8)

And the bias correction term is given by,

\[ \Delta W_{ok} = \alpha \delta_k \]  \hspace{1cm} (2.9)

The weight updation between output and hidden is given by,

\[ W_{jk}(new) = W_{jk}(old) + \Delta W_{jk} \]  \hspace{1cm} (2.10)

Each hidden unit \( (z_j, j=1, \ldots, p) \) updates its bias and weights \( (i=0, \ldots, n) \) and the weight correction term,

\[ \Delta V_{ij} = \alpha \delta_j x_i \]  \hspace{1cm} (2.11)

And the bias correction term is given by,

\[ \Delta V_{oj} = \alpha \delta_j \]  \hspace{1cm} (2.12)

The weight updation between hidden and input unit is given by,

\[ V_{ij}(new) = V_{ij}(old) + \Delta V_{ij} \]  \hspace{1cm} (2.13)

Step 10: Test for stopping condition of the network.
The stopping condition may be the minimization of the errors, number of epochs, and reduction of learning rate and so on. In this thesis work, the stopping condition is chosen to be the minimization of errors.

### 2.3.3 Feed Forward Neural Network Model without Feedback

Feed forward networks are networks where only forward flow of information is present. In this model, unlike the case of error back propagation feedback being accomplished by a separate feedback to achieve the specified topology and synapse strength; a new feed forward neural network learning algorithm accomplishes the same without using a separate feedback network. The elimination of the feedback network enables the system more likely for a biological neural system to achieve same adaptation with some means of regulatory mechanisms. Data enters at the inputs and passes through the network, layer by layer, until it arrives at the outputs. During normal operation, that is when it acts as a classifier, there is no feedback between layers. Hence they are termed as feed forward neural networks with no feedback.

This model uses one input layer, two hidden layers and one output layer. The first layer receives the information from the technical indicators. It associates weights to each signal and sent it to the hidden layer. The hidden layer calculates the activation function and associates weights and sent it to the last layer. The output layer calculates the network output. The hidden and output layer has biases. The algorithm to compute the predicted output using this model is as follows:

Step 1: Initialize the weights to be a set of small random values

Step 2: For each input vector do steps 3 – 6.

Step 3: For i=1, ..., n; set activation of input unit $x_i$;

Step 4: For j=1,..,p;

\[
\begin{align*}
    z_{-inj} &= v_{oj} + \sum_{i=1}^{n} x_i v_{ij} \\
    z_j &= f(z_{-inj}) = \frac{1}{1 + e^{-z_{-inj}}} 
\end{align*}
\]  
(2.14)

Step 5: For k=1,..,m; set activation of output unit $x_k$;

Step 6: Output layer computes the network output.
Step 5: For \( l=1, \ldots, q \):

\[
\begin{align*}
  h_{-in}^q &= u_{oq} + \sum_{i=1}^{q} z_i u_{jq} \\
  h_q &= f( h_{-in} ) = \frac{1}{1 + e^{-h_{-in}}} 
\end{align*}
\] (2.15)

Step 6: For \( k=1, \ldots, m \)

\[
\begin{align*}
  y_{-ink} &= w_{ok} + \sum_{j=1}^{q} h_q w_{qk} \\
  y_k &= f( y_{-ink} ) = \frac{1}{1 + e^{-y_{-ink}}} 
\end{align*}
\] (2.16)

Step 7: Compute the errors based on the value of \( y_k \) and given target \( t_k \).

Step 8: Update the weights using the gradient descent learning rule as in equation (2.10) and (2.13) and proceed.

Step 9: Test for stopping condition.

The stopping condition may be the minimization of the errors, number of epochs, and reduction of learning rate and so on. In this thesis work, the stopping condition is chosen to be the minimization of errors.

### 2.3.4 Radial Basis Function Neural Network model

Radial basis function neural network (RBFNN) model uses Gaussian function and is found to have a long history in the applications of recognition and approximating function. RBFNN possess three layers - the input layer, the hidden layer and the output layer where, the hidden layer functions as layer of RBF units. The interconnection between input and hidden layer form hypothetical connection and between the hidden and output layers form weighted connections. Each hidden layer unit represents a single radial basis function, with associated center position and width. Each neuron on the hidden layer employs a radial basis function as a nonlinear transfer function to operate on the input data. The widely used Radial Basis Function is a Gaussian function that is characterized by a center and width and is as shown in Fig 2.2. Radial Basis Function...
Neural Network functions by measuring the Euclidean distance between input vector and the radial basis function center. The Gaussian RBF may be tuned by adjusting spread. The Gaussian function curve which has a peak at zero distance and it decreases as the distance from the centre increases. The network architecture of RBFNN is as given in Fig 2.3.

![Figure 2.2 Representation of Gaussian radial basis activation function](image)

The Radial Basis Gaussian function is generally defined as,

\[ f(\theta_{in}) = e^{-\theta_{in}^2} \quad (2.17) \]

Where, \( \theta_{in} \) is the net input. The advantages of RBFNN model is its compact nature; possessing less training time, while eliminating local minima phenomena. The selection of the centers for the Gaussian function is important for nonlinear approximation. The RBFN network design includes structural and parameter design. The structural design involves finding the number of neurons required. The parameter design involves spread and weight of output mode. The RBFNN model is noted to be an effective neural network architecture models which gains the advantage of both generalizing and refining local features over the traditional Back Propagation Neural Network.
The various steps involved in the algorithmic flow of RBFNN model for predicting foreign exchange rate is as follows:

Step 1: Initialize the weight between the input layer to hidden layer and between hidden layers to output layer to small random values.

Step 2: Initialize the momentum factor and learning rate parameter.

Step 3: When the stopping condition is false do steps 4-11

Step 4: For each the training dataset pair do steps 5-10

Step 5: Each input unit belonging to the input layer receives the input signals $x_i$ and transmits this signals to all units in the hidden layer above i.e. to the hidden units

Step 6: Each hidden layer units ($z_j$, $j=1, \ldots, p$) sums the received weighted input signals

$$z_{-inj} = v_{oj} + \sum_{i=1}^{n} x_i v_{ij}$$  \hspace{1cm} (2.18)

Applying the continuous Gaussian activation function at this point,
\[ Z_j = f(z_{inj}) \text{ i.e, } f(Z_{inj}) = e^{-Z_{inj}^2} \]  

(2.19)

and sends this signal to all units in the layer above i.e output units.

Step 7: For each of the output unit \((y_k, k=1, \ldots, m)\), compute its net input,

\[ y_{-inj} = w_{ok} + \sum_{j=1}^{n} z_j w_{jk} \]  

(2.20)

and apply Gaussian activation function to the net input to calculate the output signals.

\[ Y_k = f(y_{-inj}) \text{ i.e, } f(Y_{inj}) = e^{-Y_{inj}^2} \]  

(2.21)

Step 8: Each output unit \((y_k, k=1, \ldots, m)\) receives a target pattern corresponding to an input pattern, error information term is calculated as,

\[ \delta_k = (t_k - y_k) f'(y_{-inj}) \]  

(2.22)

Step 9: Each hidden unit \((z_j, j=1, \ldots, n)\) sums its delta inputs from units in the layer above above

\[ \delta_{-inj} = \sum_{k=1}^{m} \delta_j w_{jk} \]  

(2.23)

Error information term is calculated as,

\[ \delta_j = \delta_{-inj} f'(z_{-inj}) \]  

(2.24)

Step 10: Compute the weight correction term between the output unit and hidden unit and is given by, \(\Delta w_{jk} = \alpha \delta_k z_j + \mu \Delta w_{jk} (old)\)

(2.25)

And the bias correction term is given by,

\[ \Delta w_{ok} = \alpha \delta_k + \mu \Delta w_{ok} (old) \]  

(2.26)
Step 11: Compute the weight correction term between the hidden unit and input unit and is given by,

\[ \Delta v_{ij} = \alpha \delta_j x_i + \mu \Delta v_{ij} \text{ (old) } \]  

(2.27)

And the bias correction term is given by,

\[ \Delta v_{oj} = \alpha \delta_j + \mu \Delta v_{ok} \text{ (old) } \]  

(2.28)

Step 12: Each output unit \((y_k, k=1, \ldots, m)\) updates its bias and weights \((j=0,\ldots,p)\) and are given by,

\[ w_{jk} \text{ (new) } = w_{jk} \text{ (old) } + \Delta w_{jk} \]
\[ w_{ok} \text{ (new) } = w_{ok} \text{ (old) } + \Delta w_{ok} \]  

(2.29)

Step 13: Each hidden unit \((z_j, j=1, \ldots,p)\) updates its bias and weights \((i=0, \ldots,n)\) and are given by,

\[ v_{ij} \text{ (new) } = v_{ij} \text{ (old) } + \Delta v_{ij} \]
\[ v_{oj} \text{ (new) } = v_{oj} \text{ (old) } + \Delta v_{oj} \]  

(2.30)

Step 14: Test for the stopping condition of the RBFNN model.

The stopping condition can be number of iterations reached; minimization of the MSE value and till the learning rate is decreased to a particular value. Figure 2.4 shows the flowchart depicting the process flow of developed RBFNN model for foreign exchange rate prediction.
Figure 2.4 Flowchart of the developed RBFNN model for foreign exchange rate prediction
2.4 Numerical Simulation of ANN models for foreign exchange rate prediction

The presented four ANN models in section 2.3 are applied for the reserve bank datasets as given in Table 2.1. The datasets considered includes 1285 real time datasets over a period from 2009 to 2014. From the considered data, 70% of the data are used for training, 15% are used for validation and 15% of the data are used for testing process. Based on this split up, the number of data employed for the various processes is as tabulated in Table 2.4.

<table>
<thead>
<tr>
<th>Details of the dataset</th>
<th>Total number of trading days</th>
<th>Training samples</th>
<th>Validation samples</th>
<th>Testing samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>USD in Indian Rupees (01/01/2009 - 30/04/2014)</td>
<td>1285</td>
<td>899</td>
<td>193</td>
<td>193</td>
</tr>
</tbody>
</table>

Table 2.4 Number of datasets for training, validation and testing processes

Artificial neural network models are designed based on the seven technical indicators as given in Table 2.2 - Simple moving average for 1 week, Simple moving average of 2 weeks, Momentum, Rate of change, Disparity 7, Disparity 14, Price oscillator. These seven technical indicators act as inputs to the Input layer to the ANN models. Hence 7 artificial neurons are used for Input layer. The input layer passes the information to the hidden layer which is then fed to one neuron on the output layer. This neuron decides the next day’s exchange rate of USD in Indian rupees.
This section presents the results of the numerical simulations carried out with respect to the considered datasets for all the four ANN models - Pattern Recognition models, Feed Forward Back Propagation Networks, Feed forward networks with no feedback, and Radial Basis Function network. These models are implemented for the financial time series problem i.e., foreign exchange rate prediction considered in this thesis. The entire proposed ANN models for foreign rate prediction are run in MATLABR2009 environment and executed in Intel Core2 Duo Processor with 2.27GHz speed and 2.00 GB RAM.

2.4.1 Implementation of pattern recognition ANN model

Pattern recognition model belonging to the category of feed forward neural network employs tan sigmoidal transfer function in both hidden and output layer. As in Figure 2.5, the seven technical indicators are presented to the inputs to input layer. Here the default scaled conjugate gradient algorithm in MATLAB ‘\texttt{trainscg}’ is used for training and 15
hidden neurons are used in the hidden layer. The scaled conjugate gradient algorithm works on the principle of conjugate directions. Weights are adjusted between the outputs of the output unit and the targets of the corresponding inputs such that the errors are minimized [163]. The training, confusion matrix and the regression analysis carried out for the proposed pattern recognition model to predict foreign exchange rate prediction is presented below.

2.4.1.1 Training the Pattern Recognition model for foreign exchange rate prediction

On initiating the training process, pattern recognition network algorithm starts its processing and the network structure, algorithms used, and performance of the network are as given in Fig 2.6. The training process continued for 57 iterations and the best average performance is noted at 0.29841 at the 51st iteration. The performance of the pattern recognition neural network model for training of foreign exchange rate datasets is shown in Fig 2.7. The performance curve gives the training, validation and testing errors during the iterative process.

2.4.1.2 Computed confusion matrix for the pattern recognition model

The confusion matrix gives the exact information on predicted unit and target unit. The diagonal elements in the confusion matrix show the correctly classified labels. Each element (i,j) in the matrix represents number of samples where the known target samples is ‘i’, and predicted samples is ‘j’. The confusion matrix for training set, validation set and testing set are found and shown in Fig 2.8. The confusion matrix is calculated by the comparing the outputs of the network with the targets.

From Fig 2.8, it is clearly noted that during the training process, 450 samples are classified correctly among 899 and validation process has classified 97 samples among 193 and testing process has correctly classified 97 samples among 193. On a whole, 643 samples were correctly classified among 1285 samples and this has resulted in 50% accuracy rate.
Fig 2.6 Training process of pattern recognition neural network model
Fig 2.7 Performance curve of the pattern recognition network model for foreign exchange rate prediction

Fig 2.8 Confusion matrix of the pattern recognition neural model
2.4.1.3 Regression Analysis of the pattern recognition model

Regression plots are linear regression between network outputs and the corresponding targets. The regression plots are drawn for training, validation and testing of the considered datasets. The good fit curve will have value closer to 1. When the values are closer to ‘0’, it represents that the outputs and the targets are not fit. Regression plot is drawn for the proposed pattern recognition network and is shown in Fig 2.9. It is clearly understood from Fig 2.9, that in the training stage, R-value has 0.99477 and in the validation stage, R-value has 0.99618 and in the testing stage, R-value has 0.99336. On a whole, the R-value is noted to be 0.99461 which is closer to 1. This proves that the pattern recognition network model tracks the targets well and performs prediction in a better manner.

![Regression plots for pattern recognition neural network model](image)

**Fig 2.9 Regression plots for pattern recognition neural network model**
2.4.2 Implementation of feed forward back—propagation neural network model

This section implements the feed forward back propagation neural network model using one input layer, two hidden layers and one output layer. During feed forward stage, each input unit \( (I_i) \) receives an input signal from seven technical indicators as given in Fig 2.5 and transmits these signals to hidden units \( (H_j) \). As presented in section 2.3.2, the errors are calculated and back propagated to the previous layers. Simulated results on the training performance and regression analysis of this feed forward back propagation network model are elucidated in this section for the foreign exchange rate dataset.

2.4.2.1 Training the Feed Forward back propagation NN model

The training algorithm of the feed forward back propagation network is initiated and carried out employing its four basic steps. In the first step, the weights and bias are initialized randomly with some small values and in the second step, each input unit receives signal from corresponding technical indicators and transmits this signal to hidden layer which in turn calculates the activation function and sends it to the output layer. Each output layer in turn applies activation function to get the response of the net.

In the third step, each output unit compares the output response with the target value and determines the errors associated with each unit. From the error, the error information term \( (\delta) \) is calculated and back propagated to all units in the hidden layer. Similarly, each hidden unit computes the error information term \( (\delta) \) and finally, the weights and bias are updated using this term \( (\delta) \) as given in the training algorithm of section 2.3.2.2 in this chapter.

In this FFBPNN model, 15 neurons are used in the hidden layer and Levenberg-Marquardt algorithm (‘trainlm’) is used for neural network training. Levenberg-Marquardt algorithm is the default training algorithm for all networks. It updates the weights and bias according to the Levenberg-Marquardt algorithm. The training stage of the proposed model is shown in Fig 2.10. This network runs for 20 iterations to achieve the best performance goal as given in Fig 2.10. A plot of training errors, validation errors and testing errors are achieved using this model and it is shown in Figure 2.11. From Fig
2.11, it is noted that the best performance accuracy is achieved at 0.3582 during 14th iteration.

Fig 2.10 Training process of the feed forward back propagation neural network model
Fig 2.11 Performance curve obtained during the simulation of feed forward back propagation neural network model for the foreign exchange rate datasets

2.4.2.2 Regression Analysis

The Regression plot of the proposed FFBPNN model is drawn for the training, validation and testing stages and it is given in Fig 2.12. The training stage possesses R-value of 0.99803 and validation stage has R-value of 0.99487 and testing stage has 0.9978. As a whole, this feed forward back propagation network has 0.9975 which is closer to 1 proving the prediction model tracks the target well. This implies that output samples matches closely with the target samples and results in better prediction rate.
Fig 2.12 Regression plot for Feed Forward Back Propagation Neural Network model with respect to foreign exchange rate prediction

2.4.3 Implementation of feed forward network without feedback

Feed forward networks without feedback is realized for foreign exchange rate prediction and is simulated in this section. This model is constructed with one input layer, two hidden layers and one output layer. The first layer receives the information from the seven technical indicators as given in Table 2.2 of this chapter. It associates weights to each signal and will send it to the hidden layer. The hidden layer applies the sigmoidal activation function to compute the hidden layer output and associates weights and sends
it to the output layer. The output layer calculates the final output. The hidden and output layer possesses bias inputs and associated bias weights.

The activation transfer functions used are differentiable transfer function such as ‘TANSIG’ and ‘PURELIN’. In this model, ‘TANSIG’ function is used between input layers and hidden layers and as well between hidden layers and output layer. ‘PURELIN’ function is used in the input layer when presenting the input samples to the network. Every layer is initialized with ‘INITNW’ function and ‘TRAINS’ function is used for adaption and this updates the weights. The network is trained employing the Levenberg-Marquardt algorithm (‘TRAINLM’). This subsection discusses the training and regression analysis of simulated feed forward networks without feedback.

2.4.3.1 Training the Feed Forward Neural Network without Feedback

During training process, this model runs up to 127 iterations and achieves the best performance for the considered datasets. The training process of the feed forward network without feedback is as shown in Fig 2.13.
Fig 2.13 Training feed forward neural network without feedback for the foreign exchange rate datasets

Fig 2.13 displays the structure of the network, training function used, and the performance of the network. When performance button is clicked, performance curve is viewed. The confusion matrix curve can be viewed from confusion button and similarly regression curves are viewed from regression button. The performance curve for this
model is shown in Fig 2.14. This model achieves 0.14734 at 121\textsuperscript{th} iteration which is better than the previous models.

![Performance curve of Feed Forward Network without feedback](image)

**Fig 2.14 Performance curve of Feed Forward Network without feedback**

### 2.4.3.2 Regression Analysis of feed forward neural network without feedback

From the Regression plots obtained and as shown in Fig 2.15, it is stated that this model achieves 0.99841 R-values on a whole, which are closer to 1. This proves that the model closely tracks the target samples and enables for better prediction.
Fig 2.15 Regression Analysis of Feed Forward Neural Network without feedback for foreign exchange rate datasets from Reserve Bank of India

2.4.4 Implementation of radial basis function neural network model

Radial Basis Function Neural Network is a particular type of feed forward neural network employed for approximating the functions and recognizing the patterns and follows the training algorithm given in section 2.3.4 of this chapter.
2.4.4.1 Developed RBFNN model in MATLAB environment

The architecture of the RBFNN consists of three layers namely, the input layer, hidden layer and the output layer. Between the input layer and the hidden layer, hypothetical connection is used and between the hidden layer and the output layer the weighted connections are used [161], [164]. The developed RBFNN model architecture for foreign exchange rate prediction is as shown in Fig 2.16. This model uses two layer networks. The first layer has ‘RADBAS’ neurons and the second layer has ‘PURELIN’ neurons. Both the layers have bias inputs and weights. Radial Basis Function adds neurons to the hidden layer until it reaches the specified goal i.e., the target level.

Fig 2.16 Developed architecture model of Radial Basis Function Neural Network

2.4.4.2 Training Algorithm

The training process for the radial basis function neural network is carried out and its progress is as given below. Using this radial basis function model, the performance curve for the Radial Basis Function is as shown in Fig 2.17. This curve shows that proposed work achieves error accuracy of 0.0000220398 at 1275 epochs. The simulated radial basis function neural network model output is given in Table 2.5.
<table>
<thead>
<tr>
<th>neurons</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>34.6872</td>
</tr>
<tr>
<td>25</td>
<td>7.39566</td>
</tr>
<tr>
<td>50</td>
<td>4.73948</td>
</tr>
<tr>
<td>75</td>
<td>2.90158</td>
</tr>
<tr>
<td>100</td>
<td>1.43965</td>
</tr>
<tr>
<td>125</td>
<td>0.957251</td>
</tr>
<tr>
<td>150</td>
<td>0.602601</td>
</tr>
<tr>
<td>175</td>
<td>0.420282</td>
</tr>
<tr>
<td>200</td>
<td>0.282647</td>
</tr>
<tr>
<td>225</td>
<td>0.174406</td>
</tr>
<tr>
<td>250</td>
<td>0.156688</td>
</tr>
<tr>
<td>1275</td>
<td>2.20398e-005</td>
</tr>
</tbody>
</table>

Table 2.5 Simulated RBFNN model output for foreign exchange rate dataset

The error value of the Radial Basis Function Neural Network is obtained as 0.0000220398 and this error value is noted to be much smaller than the other three ANN models - Pattern Recognition network model with an error of 0.29841, Feed Forward back propagation network with an error of 0.3582, Feed forward networks without feedback with an error of 0.14734.
Fig 2.17 Performance curve of simulated radial basis function neural network model for foreign exchange rate prediction

2.5 Performance Comparison of ANN models for foreign exchange rate prediction

This section presents the performance comparison of the developed four artificial neural network models - Pattern recognition network model, Feed forward back propagation network model, Feed forward network without feedback and Radial basis function neural network model. The performance indicators are calculated in order to predict the performance of the proposed models and are as listed in Table 2.6.
<table>
<thead>
<tr>
<th><strong>Performance Indicators</strong></th>
<th><strong>Definition</strong></th>
<th><strong>Formulae Employed</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean Square Error (MSE)</td>
<td>It measures the performance with respect to the mean of squared errors.</td>
<td>( \sum \frac{(x-t)^2}{n} )</td>
</tr>
<tr>
<td>Mean Absolute Error (MAE)</td>
<td>It measures the performance with respect to the mean of absolute errors.</td>
<td>( \sum \frac{</td>
</tr>
<tr>
<td>Sum Squared Error (SSE)</td>
<td>It measures the performance with respect to the sum of squared errors.</td>
<td>( \sum (x-t)^2 )</td>
</tr>
<tr>
<td>Root mean square error (RMSE)</td>
<td>It is the square root of the mean square error.</td>
<td>( \sqrt{\sum \frac{(x-t)^2}{n}} )</td>
</tr>
</tbody>
</table>

Here ‘x’ represents the input data, ‘t’ represents the target data and ‘n’ represents the number of observations.

**Table 2.6 Performance Indicators for comparison of the proposed ANN models**

Table 2.7 gives the comparison of all the neural network models simulated for foreign exchange rate prediction with respect to the performance indicators given in Table 2.6. From Table 2.7, it is inferred that the Radial Basis Function neural network possess minimal performance indicator value for MSE, MAE, SSE and RMSE to be 2.5217e-017, 2.8380e-009, 3.2404e-014 and 5.0190e-017 respectively which is lesser than all the other ANN models simulated and considered for comparison. Further to this, Mehdi Khashei et al. [19] has implemented a neural network model for foreign exchange rate prediction and the proposed RBFNN model is noted to have minimal error values even in comparison with this model as well. In the model proposed and implemented by Mehdi Khashei et al., it is noted that its mean square error, mean absolute error, sum squared error and root mean square errors is higher than the proposed Radial basis function network proving the Radial Basis function neural network gives better results in foreign exchange rate prediction.
Table 2.7 Performance comparison of the developed ANN models with that of the other model from the literature for foreign exchange rate prediction

<table>
<thead>
<tr>
<th>Developed ANN Models</th>
<th>Performance Indicators</th>
<th>Pattern Recognition Neural Network model</th>
<th>Feed forward Back Propagation Network</th>
<th>Feed forward networks without feedback</th>
<th>Radial Basis Function Neural Network model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mehdi Khashei et al. model [19]</td>
<td>Mean Square Error</td>
<td>0.020254</td>
<td>0.4084</td>
<td>0.1721</td>
<td>0.1103</td>
</tr>
<tr>
<td></td>
<td>Mean Absolute Error</td>
<td>0.010626</td>
<td>0.4820</td>
<td>0.2653</td>
<td>0.2353</td>
</tr>
<tr>
<td></td>
<td>Sum Squared Error</td>
<td>0.089561</td>
<td>524.7545</td>
<td>221.1428</td>
<td>141.7423</td>
</tr>
<tr>
<td></td>
<td>Root Mean Square Error</td>
<td>0.054479</td>
<td>0.6390</td>
<td>0.4148</td>
<td>0.3321</td>
</tr>
</tbody>
</table>

2.6 Summary

In this chapter four novel artificial neural network models which includes pattern recognition network model, feed forward back propagation neural network model, feed forward network without feedback and radial basis function neural network model were proposed for predicting the exchange rate of USD in Indian rupees during the time period 2009-2014. Based on the foreign exchange datasets, seven technical indicators were used as inputs for all these proposed models and outputs were computed in line with the specified target samples. Performance criteria used for validating the proposed ANN models with that of the earlier methods proposed in the literature includes mean square error, mean absolute error, sum square error and root mean square error. It is inferred from the obtained simulation results, that the developed radial basis function neural network for foreign exchange rate prediction results in minimal error and better convergence that of the other three models employed and the other methods considered for comparison from the literature.