IMPACT: International Journal of Research in Engineering & Technology (IMPACT: IJRET)
ISSN(E): 2321-8843; ISSN(P): 2347-4599
Vol. 2, Issue 4, Apr 2014, 111-122

AN ARCHITECTURAL FRAME WORK OF ANN BASED SHORT TERM ELECTRICITY PRICE FORECAST ENGINE FOR INDIAN ENERGY EXCHANGE USING SIMILAR DAY APPROACH

SMITHA ELSA PETER[1], I. JACOB RAGLEND[2] & SISHAJ P SIMON[3]

[1]Department of ECE, PRIST University, Thanjavur, Tamil Nadu, India

[2]Department of EEE, Noorul Islam University, Kumarakovil, Tamil Nadu, India

[3]Department of EEE, National Institute of Technology, Tamil Nadu, India

ABSTRACT

In a deregulated power market, generating companies (Gencos) evaluate bidding strategies to maximize their profit. A Genco has to make a decision based on limited information available, since it does not know the actual system Market Clearing Price (MCP) beforehand. Thus, an optimal bidding strategy is a challenging task for GenCos. Accurately forecasted MCP will aid as vital information in enhancing the chances of winning bids in today's competitive electricity markets. Based on the literatures, neural networks are used in most of the forecasting applications. This paper proposes a near optimal ANN architecture based electricity price forecast engine using the available historical data for forecasting MCP in Indian Energy exchange (IEX). This paper uses a similar-day approach for forecasting the MCP. The recent available historical data from 1st January 2014 to 16th March 2014 is used in this research work. This paper also investigates the performance related issues with the various ANN architecture models.

KEYWORDS: Error Variance, Feed Forward Back Propagation Neural Network, Market Clearing Price, Mean Absolute Percentage Error, Normalized Mean Square Error

1. INTRODUCTION

A decisive issue for all market participants in today's restructured electricity power industry has been the electricity price forecasting. A precise price forecasting helps suppliers to set up bidding strategies, make investment decisions and be cautious against risks. Conversely, consumers can use price forecasting to exploit appropriate power purchasing strategies for maximum utility utilization. Electricity market clearing price (MCP) is the price that exists when an electric market is clear of shortage and surplus [1]. It is the final outcome of market bidding price. When electricity MCP is determined, every supplier whose offering price is below or equal to the electricity MCP will be picked up to supply electricity at that hour. They will be paid at the same price, the electricity MCP, not the price they offered. The reason for this is to keep fairness of the market and to avoid market manipulation. The accuracy of the forecast depends on the availability of the data and further depends on other influential price drivers such as volatility in fuel price, load uncertainty, fluctuations in hydroelectricity production, generation uncertainties, transmission congestion, behaviour of market participants etc…

Owing to the significance and intricacies of the electricity price forecasting, several methods have been proposed by researchers for short-term price forecasting. Among these methods, two extensively used approaches are time series [2]

models and artificial neural networks (ANNs)[3]. Time series models such as dynamic regression and transfer function, ARIMA [1], EGARCH (exponential GARCH) [4,5], WT-ARIMA model [6] have been proposed for this purpose. However, most time series models are linear predictors, which have difficulties in predicting the hard nonlinear behaviour of electricity price.

ANNs have also been used by many researchers for price forecasting. Yamin et al. [7] have proposed a comprehensive model using ANN for short-term electricity price forecasting. Zhang et al. [8] have applied the cascaded architecture of multiple ANN to forecast the market clearing price (MCP) in New England to improve the prediction accuracy, other approaches considering hybrid model have been proposed. Rodriguez and Anders [9] have proposed a combination of neural networks and fuzzy logic for MCP prediction in the Ontario electricity market. Li et al. [10] have presented the fuzzy inference system and least-squares estimation for price forecasting. Though ANN based forecast engines are developed, the network architecture and the manner in which the available historical data being used will be different for different electricity markets or energy exchanges. Therefore, with the available data, designing the near optimal ANN architecture for a typical exchange is always challenging.

2. PROPOSED WORK

The proposed work is carried out for forecasting market clearing price of the Indian Energy Exchange. Not so many literatures are available for the forecast of MCP in the Indian Energy Exchange (IEX). IEX is one of the India's electricity power trading platform, Over 2600 participants across utilities from 27 states, 5 Union Territories, more than 500 private generators and more than 2300 open access consumers are doing business with IEX to manage power portfolio in the most competitive and reliable way. Day-Ahead and Term-Ahead market is followed in the IEX. Day-Ahead-Market (DAM) is a physical electricity trading market for deliveries for any/some/all 15 minute time blocks in 24 hours of next day starting from midnight. The prices and quantum of electricity to be traded are determined through a double sided closed auction bidding process. Term-Ahead-Market (TAM) provides a range of products allowing participants to buy/sell electricity for contracts beyond day-ahead market, besides intraday contracts [www.iexindia.com]. The proposed work concentrates in forecasting the hourly Weak-Ahead Market Clearing price which is the part of TAM using a similar day approach using feed forward back propagation neural network (FFBPNN).

The activities of the consumers are found to be similar on the same week days. So, in this case study, MCP of similar days is correlated for training the historical MCP data. For example, the MCP profile on Monday of the previous week is correlated to Monday of the present week. So when a test input is fed into the forecast model, a week-ahead MCP profile is forecasted. Various architectures of FFBPNN are tried out and the best one is proposed. The data is pre-processed by normalizing the load between 0.1 and 0.9 and is used in this work.

3. HISTORICAL DATA OF IEX

The historical data reports that are available in the IEX website as market snapshot are considered in the proposed work. The market snap shot consist of the hourly Purchase Bid (MW), Sell Bid (MW), Market Clearing Volume (MW), Cleared Volume (MW) and Market Clearing Price (MCP). Market Clearing Volume (MCV) is carried out before transmission congestion, whereas Cleared Volume (CV) is carried out after transmission congestion. It is very important to understand the nature of the recorded data which may be vital or very much related to the MCP. Sometimes,

the performance of the forecast largely varies due to the homogeneity of the data used. It should be noted that the Market Clearing Price is non-homogenous in nature. Therefore, understanding the shape of the historical data, it will be easier to choose the right data for the development of the proposed forecast engine. The market snapshot data for first 75 days is presented in the Figure 1.

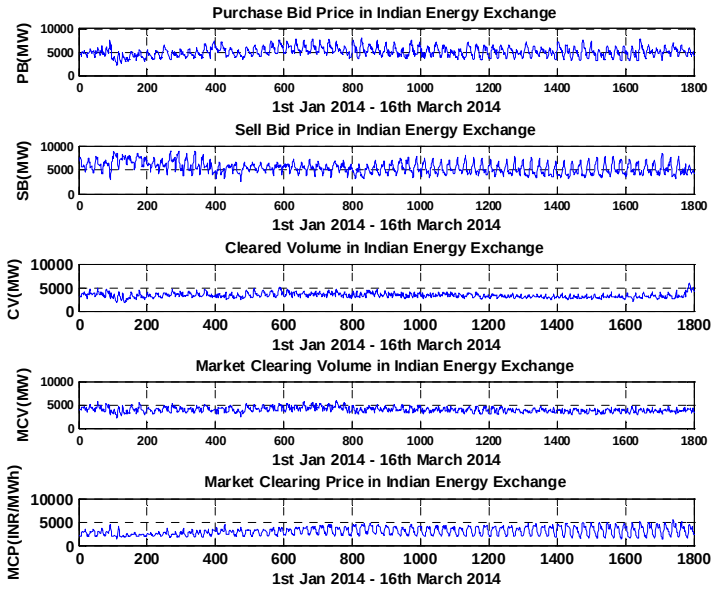

Figure 1: Market Snapshot of Historical Data (1st 75 Days of the Year 2014)

The total number of samples for the 75 days is 19320. The wave form profile of Purchase Bid (PB), Sell Bid (SB), Market Clearing Volume (MCV), Cleared Volume (CV) and Market Clearing Price (MCP) are found to be homogenous in nature. Since the all data is non- homogenous in nature, the correlation of any combination of the 5 waveforms (PB, SB, MCV, CV and MCP) with that of the Market Clearing Price (MCP) need to be explored in the forecast engine. However, all possible architectures will be tried out in the following section before a near optimal ANN model is proposed for the IEX. The training data, validation data and the testing data for the FFBNN is considered only from the 19320 samples. The source and the target training data for FFBNN training is taken from 1st Jan 2014 to 26th Feb 2014, and from 8th Jan 2014 to 5th Mar 2014, respectively. The validation data is taken from 12th Feb 2014 to 19th Feb 2014 and is compared with the actual data from 20th Feb 2014 to 26th Feb 2014. The testing or verification data is taken from 26th Feb 2014 to 5th Mar 2014, and is compared with the actual data from 6th Mar 2014 to 12th Mar 2014. It should be noted that the testing data is not used in the training set whereas the validation data is used in the training set. Validation is carried out while training to check that the network do not over train, thereby the forecast accuracy will not deteriorate.

4. PROPOSED METHODOLOGY

4.1 Architecture

The architecture of the feed forward back propagation neural network is given in Figure 2. This ANN model consists of 'M' input nodes and 'O' output nodes with 'H' hidden nodes in the hidden layer. The hidden layer and the output layer nodes consist of log-sigmoid transfer function whose output value will in the range between 0 and 1.

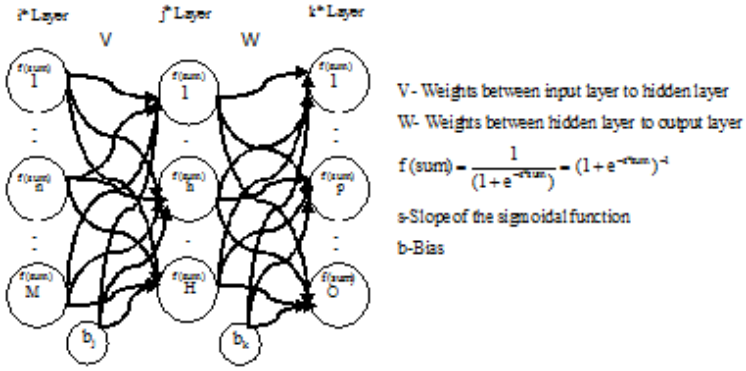

Figure 2: Architecture of Feed Forward Back Propagation Neural Network (FFBPNN)

The historical dataset is usually not used directly in process modelling of ANNs due to the difference in magnitude of the process variables. Therefore, the data needs to be scaled to a fixed range to prevent unnecessary domination of certain variables, and to prevent data with larger magnitude from overriding the smaller and impede the premature learning process. The choice of range depends on transfer function of the output nodes in ANN. Typically, [0, 1] for sigmoid function and [-1, 1] for hyperbolic tangent function. However, due to nonlinear transfer function has asymptotic limits; the range of dataset is always set slightly less than the lower and upper limits. In this work, since the sigmoid function is adopted, the data is normalized in the range of [0.1-0.9]. i.e., If x_1 and x_2 is the maximum and minimum value of the training set, respectively, then the normalised data is given by $N(x)$ as in (4.1).

$$N(x) = \left(\frac{(x - x_1) \times (0.1 - 0.9)}{(x_2 - x_1)} \right) + 0.9 \tag{4.1}$$

Based on the data being sent in the forecast engine the following cases of various architectures are proposed and the performance related to training error and forecast accuracy are discussed. In all possible architecture the output node remains one and the number of hidden nodes is set based on trial and error. To understand the relationship of all the 5 waveforms with that of the MCP, the number of input nodes varies from 1 to 5. The training data set, validation data and testing data is created based on the similar day approach.

Case-I (5-H-1 FFBNN Architecture)

The network architecture consists of 5 input nodes and 1 output node. All the 5 input waveforms are given as input for the training set.

Case-II (4-H-1 FFBNN Architecture)

The network architecture consists of 4 input nodes and 1 output node. If the 5 waveforms are represented as (PB,-1 SB-2, MCV-3, CV-4 and MCP-5), then the following 5 combination of input data need to be evaluated for the ANN model. They are 4(1)-H-1, 4(2)-H-1, 4(3)-H-1, 4(4)-H-1 and 4(5)-H-1. The number within the bracket is the waveform which is not considered. For example, in 4(2)-H-1, the 2nd waveform (Sell Bid Price) is not considered.

Case-III (3-H-1 FFBNN Architecture)

The network architecture consists of 3 input nodes and 1 output node. There will be 10 possible combinations.

They are 3(1-2)-H-1, 3(1-3)-H-1, 3(1-4)-H-1, 3(1-5)-H-1, 3(2-3)-H-1, 3(2-4)-H-1, 3(2-5)-H-1, 3(3-4)-H-1, 3(3-5)-H-1 and 3(4-5)-H-1. The numbers within the bracket are the waveform which are not considered

Case-IV (2-H-1 FFBNN Architecture)

The network architecture consists of 2 input nodes and 1 output node. There will be 9 possible combinations. They are 2(1-2-3)-H-1, 2(1-3-4)-H-1, 2(1-4-5)-H-1, 2(2-3-4)-H-1, 2(2-4-5)-H-1, 2(2-5-1)-H-1, 2(3-4-5)-H-1, 2(3-5-1)-H-1 and 2(3-5-2)-H-1.

Case-V (1-H-1 FFBNN Architecture)

The network architecture consists of 2 input nodes and 1 output node. There will be 5 possible combinations. They are 1(2-3-4-5)-H-1, 1(1-3-4-5)-H-1, 1(1-2-4-5)-H-1, 1(1-2-3-5)-H-1 and 1(1-2-3-4)-H-1.

4.2 Step by Step Algorithm of FFBPNN Architecture

Nomenclature

I Input training vector

$$I = (i_1,...,i_n,...,i_M)$$

T Output target vector

$$T = (t_1,...,t_y,...,t_O)$$

δ_y Error correction weight adjustment for w_{hy} due to an error at output unit K_y

δ_h Error correction weight adjustment for v_{nh} due to an error at hidden unit J_h

α Learning rate

$f(\text{sum}) = \dfrac{1}{1+\exp(-\text{sum})}$ Activation function or Threshold function

Step 1: Set the trial number tr =1

Step 2: Set the epoch ep =1

Step 3: Generate the weights randomly to small random values between 0 and 1 to ensure that the network is not saturated by large values of weights. Let I and T be the normalized input and target training vector from set of P number of training patterns.

Step 4: Choose a training pair from the training set.

Step 5: For each training pair, do steps 6 -11

Step 6: Each input unit receives input signal i_n and broadcasts this signal to all units in the hidden layer J.

Step 7: Each hidden unit J_h sums its weighted input signals and the net input to the hidden unit is given as in (4.2) and the output at the hidden layer (J) is given as in (4.3). Send the output of the hidden layer signals to all units in the output units.

$$sum_{Jh} = b_J + \sum_{n=1}^{M} i_n \times V_{nh}, \tag{4.2}$$

$$f(sum_{Jh}) = \frac{1}{1+\exp(-sum_{Jh})} \tag{4.3}$$

Step 8: Each output unit K_y sums its weighted input signals and the net input to the output unit is given as in (4.4) and the output at the output layer (K) is given as in (4.5).

$$sum_{Ky} = b_K + \sum_{h=1}^{H} J_h \times W_{hy}, \tag{4.4}$$

$$f(sum_{Ky}) = \frac{1}{1+\exp(-sum_{Ky})} \tag{4.5}$$

Back Propagation of Error

Step 9: Each output unit K_y receives a target pattern corresponding to the input training pattern, computes its error information term as in (4.6) and calculates its weight correction term as in (4.7) which is used to update W_{hy} later.

$$\delta_y = (t_y - K_y) \times f'(sum_{Ky}) \tag{4.6}$$

$$\Delta w_{hy} = \alpha \times \delta_y \times f(sum_{Jh}) \tag{4.7}$$

The bias correction term is given in (4.8)

$$\Delta b_K = \alpha \times \delta_y \tag{4.8}$$

Step 10: Each hidden unit J_h sums its delta inputs as in (4.9), multiplies by the derivative of its activation function to calculate its error information term as in (4.10) and calculates its weight correction term as in (4.11)

$$sum_{\delta J} = \sum_{n=1}^{O} \delta_y \times W_{hy}, \tag{4.9}$$

$$\delta_h = sum_{\delta J} \times f'(sum_{Jh}) \tag{4.10}$$

$$\Delta v_{nh} = \alpha \times \delta_h \times i_n \tag{4.11}$$

The bias correction term is given in (4.12)

$$\Delta b_J = \alpha x \delta_h \tag{4.12}$$

Update Weights and Biases

Step 11: Each output unit K_y updates its weights and bias as in (4.13) and (4.14). Also each hidden unit J_h updates its weights and bias as in (4.15) and (4.16).

$$w_{hy}(new) = w_{hy}(old) + \Delta w_{hy} \tag{4.13}$$

$$b_K(new) = b_K(old) + \Delta b_K \tag{4.14}$$

$$w_{nh}(new) = w_{nh}(old) + \Delta w_{nh} \tag{4.15}$$

$$b_J(new) = b_J(old) + \Delta b_J \tag{4.16}$$

Go to Step 5, till all the training pairs in the training set are sent into the input layer I (one epoch is over). Otherwise go to Step 12.

Step 12: Do again Step 4 to Step 8 till all the training pairs in the training set are sent into the input layer I. Calculate the error (ε), the difference between the network output and the desired output, for all the training pairs as in (4.17) and then the average mean squared error (AMSE) as in (4.18), which is calculated for every epoch. Update ep=ep+1.

$$\varepsilon_p^y = T_p^y - K_p^y \tag{4.17}$$

$$\text{AMSE} = \frac{\sum_{p=1}^{P}\left(\frac{\sum_{y=1}^{O}\varepsilon_p^y}{O}\right)}{P} \tag{4.18}$$

Step 13: Repeat steps 2-12, if ep<TE (total number epochs), else go to step 14. The total number of epochs is fixed based on trial and error approach such that the AMSE obtained is the least. Record the final weights and biases obtained for the trial number tr =1. Update tr = tr+1. Also if the validation error is increasing and if the number validation checks are greater than the validation count (VC), then stop the training for the current trial and update tr = tr+1.

Step 14: Do sufficient numbers of trials (TR) and record the final weights obtained in each of the trials. If tr < TR, go to step 1, else stop the execution.

4.3 Performance Evaluation

The accuracy of the results in this case study is evaluated based on three error indices. They are: Mean Absolute Percentage Error (MAPE), Normalized Mean Square Error (NMSE) and Error Variance (EV). The Mean Absolute Percentage Error (MAPE) is defined by the following equation (4.19).

$$MAPE = \frac{1}{NH}\sum_{i=1}^{NH}\left|\frac{P_i - A_i}{A_i}\right| \tag{4.19}$$

NMSE (Nima Amjady *et al*, 2011) [11] is defined as

$$NMSE = \left[\frac{1}{\Delta^2 NH}\sum_{i=1}^{NH}\left(P_i - A_i\right)^2\right] \tag{4.20}$$

$$\text{Where, } \Delta = \frac{1}{NH-1}\sum_{i=1}^{NH}\left(A_i - A_{Ave}\right)^2$$

EV (Nima Amjady *et al*, 2011) [11] is defined as

$$\sigma^2 = \frac{1}{NH}\sum_{i=1}^{NH}\left(\left|\frac{P_i - A_i}{A_i}\right| - MAPE\right)^2 \tag{4.21}$$

Where, P_i and A_i are the i[th] predicted and actual values respectively, A_{Ave} is the mean of the actual value and NH is the total number of predictions.

5. RESULTS AND DISCUSSIONS

The five types of architectures mentioned in section 4.1 is simulated for ten number of trials. A statistical analysis considering the average of the performance indices for all the trials is evaluated. The parameter settings in all the five architectures such as learning rate (0.9), momentum factor (0.9), slope factor (0.05) and validation count (VC=10) are kept same so as to have a fair comparison on the same reference among the architectures. The weights and bias are initialized randomly between zero to one. The number of epochs is kept same for all the architectures as 1000. The number of nodes in the hidden layer is kept as H=20.

Tables 1-5 give the best and average of all the performance indices for all the cases. From the results (Table 1-5), five best performing architectures are grouped based on the lowest average error and are given five ranks according to their performance in Table 6 below. Table 6 gives the details of the five best architectures. Here, the network which consists of two input nodes with Purchase Bid and Market Clearing Price data as input is ranked I as the best performing architecture with an average Training Error=4.7774E-05, Validation Error=1.1072E+01, MAPE=1.4428E+01, NMSE=1.4654E-07 and EV=2.0406E+02.

Table 1: Case-I (5-H-1)

Architecture		Training Error (AMSE)	Validation Error (MAPE)	Performance Indices (Testing/Verification)		
				MAPE	NMSE	EV
5-H-1	Best	4.0499E-05	8.2242E+00	1.4681E+01	1.3885E-07	2.1127E+02
	Average	4.0543E-05	8.2406E+00	1.4689E+01	1.3894E-07	2.1149E+02

Table 2: Case-II (4-H-1)

Architecture		Training Error (AMSE)	Validation Error (MAPE)	Performance Indices (Testing/Verification)		
				MAPE	NMSE	EV
4(1)-H-1	Best	4.0828E-05	7.5049E+00	1.5701E+01	1.6034E-07	2.4163E+02
	Average	4.0901E-05	7.5264E+00	1.5772E+01	1.6137E-07	2.4383E+02
4(2)-H-1	Best	4.7406E-05	1.0906E+01	1.4421E+01	1.4131E-07	2.0383E+02
	Average	4.7583E-05	1.0937E+01	1.4474E+01	1.4244E-07	2.0533E+02
4(3)-H-1	Best	4.2252E-05	8.5164E+00	1.4889E+01	1.4046E-07	2.1730E+02
	Average	4.2263E-05	8.5186E+00	1.4905E+01	1.4063E-07	2.1774E+02
4(4)-H-1	Best	4.0448E-05	8.1838E+00	1.4548E+01	1.3752E-07	2.0745E+02
	Average	4.0483E-05	8.2066E+00	1.4572E+01	1.3787E-07	2.0812E+02
4(5)-H-1	Best	4.7194E-05	9.2728E+00	1.6891E+01	1.8534E-07	2.7964E+02
	Average	4.7280E-05	9.2766E+00	1.6902E+01	1.8582E-07	2.8003E+02

Table 3: Case-III (3-H-1)

Architecture		Training Error (AMSE)	Validation Error (MAPE)	Performance Indices (Testing/Verification)		
				MAPE	NMSE	EV
3(1-2)-H-1	Best	6.1538E-05	1.4202E+01	2.1478E+01	2.9060E-07	4.5216E+02
	Average	6.2337E-05	1.4333E+01	2.1689E+01	2.9536E-07	4.6109E+02
3(1-3)-H-1	Best	4.6265E-05	8.2380E+00	1.6989E+01	1.7785E-07	2.8291E+02
	Average	4.6409E-05	8.2643E+00	1.7067E+01	1.7906E-07	2.8551E+02
3(1-4)-H-1	Best	4.0168E-05	7.3845E+00	1.5777E+01	1.5679E-07	2.4400E+02
	Average	4.0397E-05	7.4171E+00	1.5895E+01	1.5850E-07	2.4766E+02
3(1-5)-H-1	Best	5.2585E-05	9.2638E+00	2.0495E+01	2.7967E-07	4.1172E+02
	Average	5.3010E-05	9.4481E+00	2.0744E+01	2.8455E-07	4.2184E+02
3(2-3)-H-1	Best	4.7303E-05	1.0974E+01	1.4325E+01	1.4119E-07	2.0112E+02
	Average	4.8002E-05	1.1118E+01	1.4593E+01	1.4650E-07	2.0876E+02
3(2-4)-H-1	Best	4.8068E-05	1.0962E+01	1.4412E+01	1.4603E-07	2.0358E+02
	Average	4.8278E-05	1.1010E+01	1.4495E+01	1.4744E-07	2.0595E+02
3(2-5)-H-1	Best	8.6617E-05	1.6717E+01	2.4366E+01	3.8160E-07	5.8191E+02
	Average	8.6632E-05	1.6779E+01	2.4474E+01	3.8511E-07	5.8711E+02
3(3-4)-H-1	Best	4.2215E-05	8.4959E+00	1.4847E+01	1.3919E-07	2.1606E+02
	Average	4.2196E-05	8.5052E+00	1.4860E+01	1.3918E-07	2.1645E+02
3(3-5)-H-1	Best	4.9472E-05	9.7010E+00	1.7193E+01	1.8833E-07	2.8972E+02
	Average	4.9520E-05	9.7126E+00	1.7207E+01	1.8871E-07	2.9020E+02
3(4-5)-H-1	Best	4.6918E-05	9.2394E+00	1.6725E+01	1.8273E-07	2.7419E+02
	Average	4.6843E-05	9.2562E+00	1.6752E+01	1.8299E-07	2.7506E+02

Table 4: Case-IV (2-H-1)

Architecture		Training Error (AMSE)	Validation Error (MAPE)	Performance Indices (During Testing)		
				MAPE	NMSE	EV
2(1-2-3)-H-1	Best	7.7646E-05	1.7006E+01	2.5066E+01	3.8325E-07	6.1585E+02
	Average	8.6740E-05	1.8319E+01	2.6790E+01	1.5502E-05	7.0496E+02
2(1-3-4)-H-1	Best	4.8678E-05	8.4985E+00	1.8032E+01	1.8559E-07	3.1870E+02
	Average	4.8842E-05	8.5286E+00	1.8109E+01	1.8707E-07	3.2145E+02
2(1-4-5)-H-1	Best	5.2865E-05	9.4558E+00	2.1145E+01	2.8151E-07	4.3824E+02
	Average	5.3746E-05	9.8048E+00	2.1530E+01	2.8967E-07	4.5443E+02
2(2-3-4)-H-1	Best	4.7416E-05	1.0978E+01	1.4254E+01	1.4361E-07	1.9916E+02
	Average	4.7774E-05	1.1072E+01	1.4428E+01	1.4654E-07	2.0406E+02
2(2-4-5)-H-1	Best	8.4882E-05	1.6979E+01	2.4678E+01	3.9372E-07	5.9691E+02
	Average	8.6256E-05	1.7380E+01	2.5341E+01	4.1136E-07	6.2975E+02
2(2-5-1)-H-1	Best	1.5446E-04	2.6004E+01	3.6551E+01	8.1991E-07	1.3095E+03
	Average	1.5444E-04	2.6002E+01	3.5553E+01	8.1979E-07	1.3096E+03
2(3-4-5)-H-1	Best	4.92E-05	9.7131E+00	1.7074E+01	1.84E-07	2.8575E+02
	Average	4.9196E-05	9.7170E+00	1.7085E+01	1.8394E-07	2.8611E+02
2(3-5-1)-H-1	Best	6.4777E-05	1.1174E+01	2.3325E+01	3.3285E-07	5.3325E+02
	Average	6.5308E-05	1.1460E+01	2.3589E+01	3.3917E-07	5.4546E+02
2(3-5-2)-H-1	Best	8.6256E-05	1.7580E+01	2.5400E+01	4.0591E-07	6.3235E+02
	Average	8.6302E-05	1.7629E+01	2.5478E+01	4.0963E-07	6.3623E+02

Table 5: Case-V (1-H-1)

Architecture		Training Error (AMSE)	Validation Error (MAPE)	Performance Indices (During Testing)		
				MAPE	NMSE	EV
1(2-3-4-5)-H-1	Best	8.5530E-05	1.7874E+01	2.5704E+01	4.1545E-07	6.4760E+02
	Average	8.6380E-05	1.8046E+01	2.5974E+01	4.2340E-07	6.6135E+02
1(1-3-4-5)-H-1	Best	7.3056E-05	1.2929E+01	2.5396E+01	3.6849E-07	6.3218E+02
	Average	7.3360E-05	1.3092E+01	2.5503E+01	3.7202E-07	6.3753E+02

Table 5: Contd.,

1(1-2-4-5)-H-1	Best	1.5460E-04	2.6015E+01	3.6557E+01	8.1981E-07	1.3099E+03
	Average	1.5471E-04	2.6024E+01	3.6565E+01	8.2008E-07	1.3105E+03
1(1-2-3-5)-H-1	Best	1.5511E-04	2.6058E+01	3.6599E+01	8.2131E-07	1.3129E+03
	Average	1.5514E-04	2.6060E+01	3.6602E+01	8.2128E-07	1.3131E+03
1(1-2-3-4)-H-1	Best	9.6083E-05	1.9662E+01	2.8310E+01	4.8280E-07	7.8555E+02
	Average	9.8895E-05	2.0012E+01	2.8778E+01	4.9848E-07	8.1198E+02

Table 6: Average Ranking of Best Architectures

Rank No.	Architecture	Purchase Bid (MW)	Sell Bid (MW)	Market Clearing Vol (MW)	Cleared Volume (MW)	Market Clearing Price (Rs/MWh)
I	2(2-3-4)-H-1	√				√
II	4(2)-H-1	√		√	√	√
III	3(2-4)-H-1	√		√		√
IV	4(4)-H-1	√	√	√		√
V	3(2-3)-H-1	√			√	√

From Table 6, it is observed that in all the five best categories, both Purchase Bid and Market Clearing Price is available as input data which indicates a good correlation with Market Clearing Price as the target data in the training set. Therefore, Purchase Bid data is found to be more suitable with MCP when training is carried out using FFBNN.

Since the architecture 2(2-3-4)-H-1 is found to be the best among all the architectures considered for performance evaluation, instead of stopping at 1000th epoch, the training for the same architecture is carried out for 5000 epochs. The resultant plots for the training error convergence and validation error convergence is given in Figures 3 and 4, respectively. The final results of the performance indices after 5000 epochs are Training Error=3.8409E-05, Validation Error=8.5732E+00, MAPE=1.1853E+01, NMSE=9.4847E-08 and EV=1.3771E+02.

The forecasted MCP and the actual MCP from March 6th to March 12th, 2014 is shown in Figure 5. The forecasted price for the future week using similar day approach will enable the generating companies to carefully participate in bidding process of the electricity price in the week-ahead market.

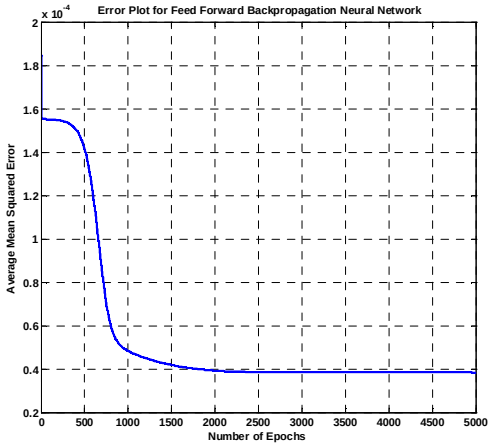

Figure 3: Average Mean Square Error Convergence Plot (Training)

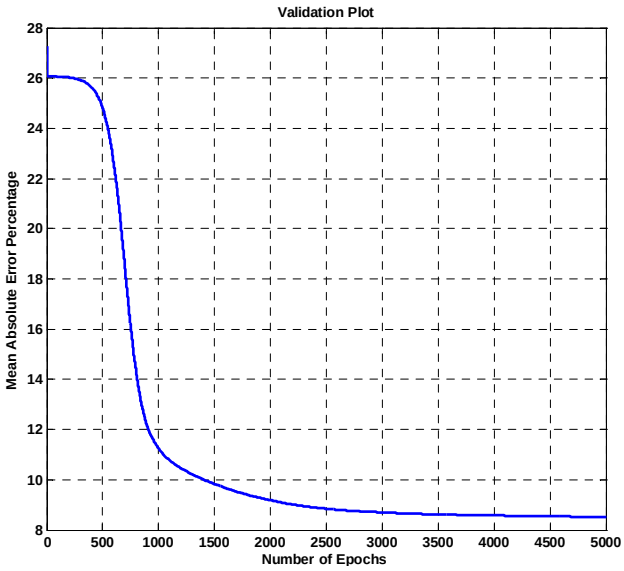

Figure 4: Mean Absolute Error Percentage Convergence Plot (Validation)

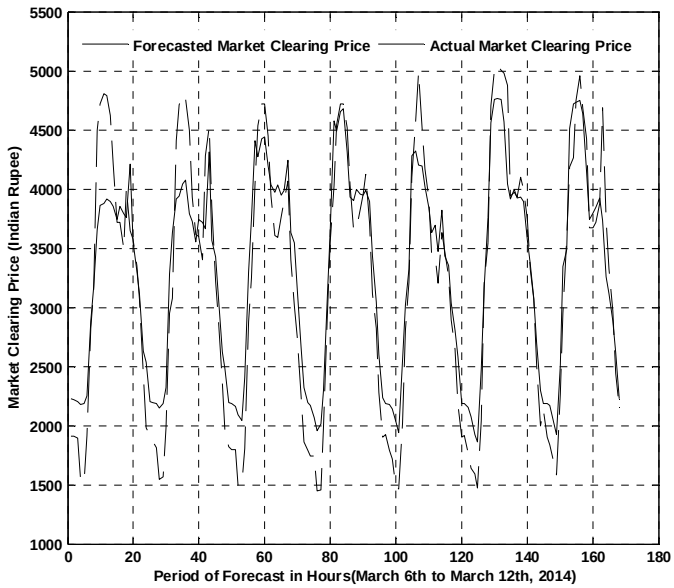

Figure 5: Forecasted MCP and Actual MCP from March 6th to March 12th, 2014

6. CONCLUSIONS

The statistical analysis with the available data in the Indian Energy Exchange shows the importance of Purchase Bid data closely related to MCP even with the non-homogenous nature of the data profile. Among 30 various combinations of architectures, the architecture with two input nodes with Purchase Bid and MCP is found to be successful in minimizing the Mean Absolute Error between the forecasted MCP and actual MCP. This architecture can be used by generating companies in deciding the bidding strategy in the highly competitive Indian Energy Exchange.

REFERENCES

1. Antonio J. Conejo, Javier Contreras, Rosa Espı´nola, Miguel A. Plazas, Forecasting electricity prices for a day-ahead pool-based electric energy market, *International Journal of Forecasting 21,* 2004, 435– 462.

2. M.Kheshei, S.Reza Hejazi, and M. Bijari, A new hybrid artificial neural networks and fuzzy regression model for time series forecasting, *Fuzzy sets and systems, Vol. 159*, 2008, 769-786.
3. Deepak Singhal, K.S. Swarup, Electricity price forecasting using artificial neural networks, *Electrical Power and Energy Systems 33*, 2011, 550–555.
4. Jinliang Zhang, Zhongfu Tan, Day-ahead electricity price forecasting using WT, CLSSVM and EGARCH model, *Electrical Power and Energy Systems 45*, 2013, 362–368.
5. M. Zhou, Z. Yan, Y.X. Ni, G. Li and Y. Nie, Electricity price forecasting with confidence-interval estimation through an extended ARIMA approach, *IEE Proceedings on Generation, Transmission and Distribution, Vol. 153, No. 2*, March 2006, 187-195.
6. N. M. Pindoriya, S. N. Singh, S. K. Singh, An Adaptive Wavelet Neural Network-Based Energy Price Forecasting in Electricity Markets, *IEEE Transactions on Power Systems, Vol. 23, No. 3*, Aug 2008, 1423-1432.
7. Yamin H. Y., Shahidehpour S. M., Li Z., Adaptive short-term electricity price forecasting using artificial neural networks in the restructured power markets, *Electrical Power Energy Systems 26*, 2004, 571–581.
8. Zhang L., Luh P. B., Neural network-based market clearing price prediction and confidence interval estimation with an improved extended Kalman filter method, *IEEE Transactions on Power Systems 20(1)*, 2005, 59–66.
9. Rodriguez C.P., Anders G. J., Energy price forecasting in the Ontario competitive power system market, *IEEE Transactions on Power Systems 19 (3)*, 2004, 366–374.
10. Li G., Liu C. C., Mattson C., Lawarrée J., Day-ahead electricity price forecasting in a grid environment, *IEEE Transactions on Power Systems 22*, 2007, 266–274.
11. Nima Amjady, Farshid Keynia and Hamidreza Zareipour, Short term wind forecasting using Ridgelet neural network, *Electrical Power Systems Research, Vol. 81*, 2011, 2099-2107.