Australian Journal of Basic and Applied Sciences, 4(5): 932-947, 2010
ISSN 1991-8178

ANN Based Short Term Load Forecasting Paradigms for WAPDA Pakistan

Laiq Khan, Kamran Javed, Sidra Mumtaz

Department of Electrical Engineering, COMSATS Institute of Information Technology, Abbottabad, Pakistan.

Abstract: This paper discusses a comprehensive approach for Short Term Load Forecasting (STLF) for Water and Power Development Authority (WAPDA), Pakistan. Keeping in view, non-linear power systems load characteristics; two different Artificial Neural Network (ANN) based architectures have been devised for STLF. Proposed architectures were trained and tested using previous five years actual load data obtained from WAPDA i.e., year 1991-95. Several back propagation based training algorithms were applied to both architectures. Data engineering approach was applied for thorough comparison and identification of the most appropriate approach that yields accurate forecast.

Key words: Short term load forecasting, artificial neural network, back propagation, data engineering.

INTRODUCTION

One of the major research areas in the field of electrical engineering which has been of great importance for many years is load forecasting. It performs as a core process for effective economic planning as well as operational decisions of electric utility companies. Accurate forecast provides basis for decisions in unit commitment, hydrothermal coordination, hydro scheduling, fuel allocation, economic load dispatch and optimal power flow calculations, which leads to savings for electric utilities. Load forecasting facilitates to maintain a balance between electricity supply and demand. Besides this, system operator utilizes the forecasted results for off-line network analysis (Gross and Galiana, 1987). In terms of planning short term load forecasting (STLF) having prediction period of one day or a week, provides basis for economic purposes (Yalcinoz, T., 2005).

STLF facilitates with necessary information on the basis of power system characteristics to perform daily operational tasks economically. It plays an important role for secure operational strategies and economic optimization. The aim of STLF is to provide basic generation scheduling functions, corrective actions based on future conditions for off-line security and timely dispatcher information.

Forecasting errors result in the form of increased operational costs. Under prediction of STLF leads to a failure for providing the required reserves, thus resulting in increased operational costs by using costly peaking units. On the other hand large reserve capacity and high operational costs are caused due to over prediction of STLF. Thus, lower operational costs due to accurate forecasts yield benefits in the form of savings passed on to the customers (Al-Shareef, A.J., 2008; Sanjib Mishra, 2008).

Various approaches have been applied to the problem of STLF. They may be classified into two types, such as statistical approaches and Artificial Intelligence (AI) based techniques. Statistical methods have limited ability and lack of flexibility to model complexity and nonlinearity. On the other hand AI methods are more flexible as far as complexity and nonlinearity are concerned (Temraz, H.K., 1996; Mcmenamin, J.S. and F.A. Monforte, 1998). Statistical methods include time series (Ibrahim Moghram, 1989; Lru, K., 1996), similar day approach (Eugene A. Feinberg,) and regression methods (Alex, D., C. Timothy, 1990), whereas ANN modeling approach is related to AI (Wasserman, P.D., 1989).

This paper proposes a novel approach for STLF based on different ANN modeling architectures. As ANN has capability to approximate nonlinear functions, different training sets of historical load can be utilized as input variables. When trained ANN model is presented with unseen data, due to good approximation capability it can generalize among different training sets as well as produce corresponding output. Fig. 1.1 shows patterns of hourly mean load from actual load data obtained from WAPDA for year 1991-95 and 2000-04.

Corresponding Author: Laiq Khan, Department of Electrical Engineering, COMSATS Institute of Information Technology, Abbottabad, Pakistan.
E-mail: laiq@ciit.net.pk,

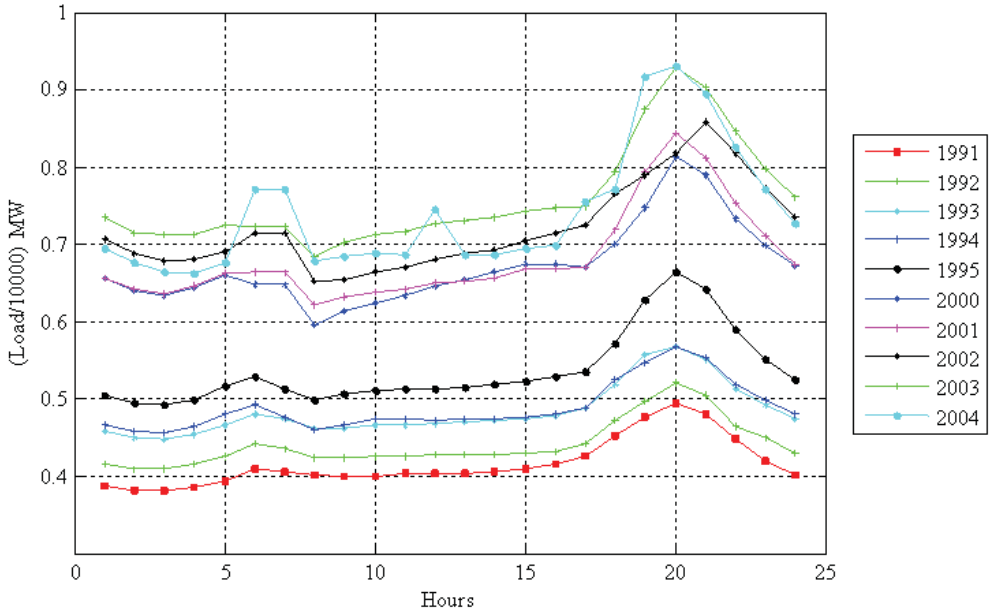


Fig. 1: Hourly mean load (1991-95, 2000-04).

This paper is organized as follows. Section 2 presents an overview of Artificial Neural Network and different types of back propagation algorithms. Section 3 highlights importance of Data Engineering and proposed ANN based architectures. Section 4 deals with simulation results discussion and interactive Graphical User Interface (GUI). Finally Section 5 concludes this research work.

2. Artificial Neural Network:

2.1 ANN Architecture:

The concept of Artificial Neural Network was introduced by McCulloch and Pitts in year 1943. ANN techniques having strong learning ability and good performance on unseen data i.e., generalization can achieve function approximation, classification and pattern recognition etc. ANN is inspired from biological neurons, so neurons provide the information processing ability.

ANN provides the layered approach consisting of three layers namely input layer, hidden layer and output layer. The input layer collects the input, computations are performed by the hidden layer and then the output of the network is produced by the output layer. Each hidden node as well as output node applies an activation function to its net input value. An activation function can be linear or non-linear, as shown in Fig. 2.

Neurons are information processing elements arranged in parallel for performing computational tasks. Connected paths among various neurons provide means for information which also determines the functionality of the network. Each connection provides the weighted strength, in addition a bias is also a weight and its input is usually '1'. For a given data set, adaptive learning of ANN is achieved by adjustment of these weights among different elements of network, comparison between network output and desired output (Lee, K.Y., J.H. Park, 1992).

When there exists an intermediate layer of neurons between input and output layer, with no feedback links, then the structure is said to be multilayer feed forward network (MFNN) as shown in Fig. 3. MFFN can have more than one hidden layers for processing.

2.2 Back Propagation Algorithm:

Back propagation algorithm is a supervised learning method and is of great importance for neural network. It is generalization of Widrow-Hoff rule (1986). It updates weights of ANN model and minimizes mean square error between actual and desired output. Back propagation algorithm is given as follows:

While stopping condition is false

For each input vector $\chi_r, 1 \le r \ge R$

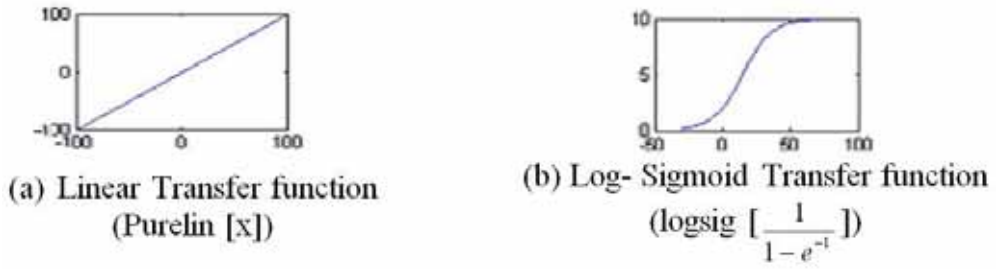


Fig. 2: (a) and (b) show structure plots of linear and nonlinear activation function.

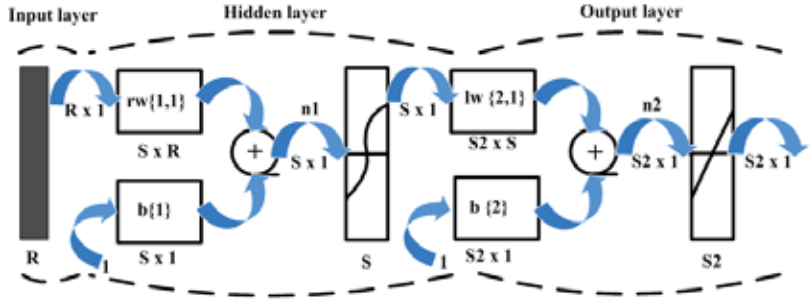


Fig. 3: Multi layer feed forward network.

Calculate hidden neuron inputs

$$\left(\mathrm{Net}_{\mathrm{r,j}}^{(1)}\right)=\sum wt_{j,i}^{(1,0)}*x_{r,i}$$

Calculate hidden neuron outputs

$$\left(x_{\mathrm{r,j}}^{(1)}\right)=s\left(\sum wt_{j,i}^{(1,0)}*x_{r,i}\right)$$

Calculate inputs to the output neurons

$$\left(\mathrm{Net}_{\mathrm{r,k}}^{(2)}\right)=\sum wt_{k,j}^{(2,1)}*x_{r,j}^{(1)}$$

Calculate the network outputs

$$\left(o_{\mathrm{r,k}}\right)=s\left(\sum wt_{k,j}^{(2,1)}*x_{r,j}^{(1)}\right)$$

Calculate the error between $o_{\mathrm{r,k}}$ and desired output $d_{r,k}$

Update the weights between hidden an output neurons

$$\Delta wt_{k,j}^{(2,1)}=\eta*\left(d_{r,k}-o_{r,k}\right)*s'\left(Net_{r,k}^{2}\right)*x_{r,j}^{(1)} \quad (1)$$

Update the weights between hidden an output neurons

$$\Delta wt_{j,i}^{(1,0)} = \eta * \sum \left\{ \left(d_{r,k} - o_{r,k} \right) * s'\left(Net_{r,k}^{2} \right) * wt_{k,j}^{(2,1)} \right\} * s'\left(Net_{r,j}^{1} \right) * x_{r,j}^{1} \quad (2)$$

End For
End While

2.3 Different Back Propagation Algorithms:

There are a variety of back propagation algorithms based on two things, weight up-gradation for minimizing the value of error and variations in learning rate to achieve convergence (Lahiri, S.K., K.C. Ghanta, 2008). Fig. 4 shows structure of different back propagation algorithms.

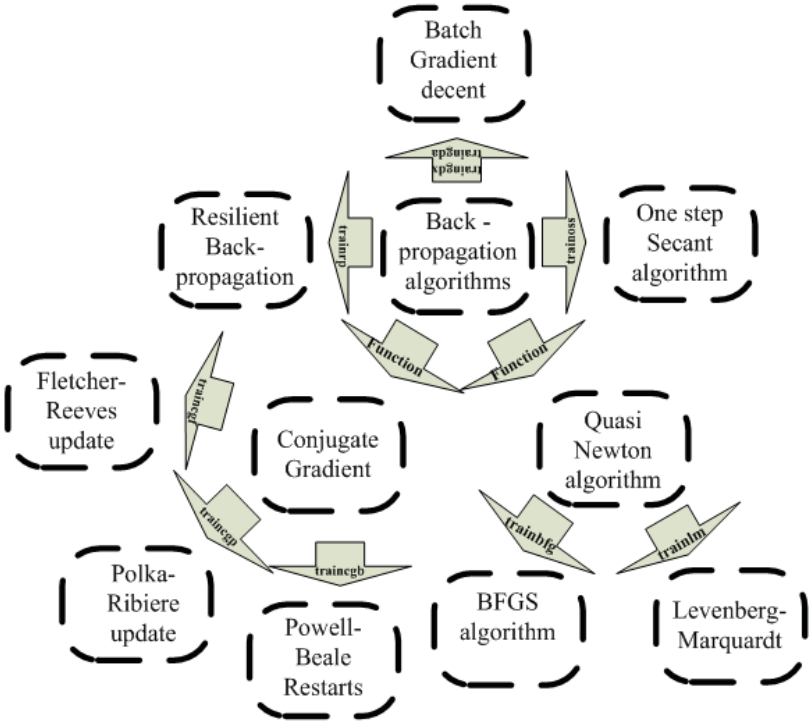


Fig. 4: Structure of different back propagation algorithms.

2.3.1 Gradient Decent with Momentum:

This algorithm functions like a low pass filter as it produces smoothing effect which facilitates the network with the ability to neglect small features that exist in error surface. It has the ability to overcome the problem of local minima. Its functionality is dependant upon the values of learning rate and the momentum constant.

2.3.2 Batch Gradient Decent with Variable Learning Rate:

The enhancement in performance of back propagation algorithm can be achieved by changing learning rate during training session, hence increasing the level of complexity of error surface. Higher values of learning rate causes instability and very small values of learning rate causes increased time factor for convergence. Changes in the values of weights and biases are proportional to learning rate multiplied by the value of negative gradient.

For achieving adaptive learning rate there must be some replacements in training methodology used by gradient decent algorithm. Initially, calculation of network output and error values, after that the value of current learning rate is utilized for calculating new values of weights and biases for every epoch. Then the values of output and error are calculated and if the resulting value of error exceeds previously acquired value of error the learning rate is decreased and vice versa. Higher values of learning rate can lead to instability.

2.3.3 Resilient Back Propagation:

The value of slope in sigmoid functions that are used for multilayer neural networks must minimize to zero value for increased values of input. So, in case of steep decent algorithm this factor causes a problem, as the values of gradient can attain small magnitude and have a very minimum effect on the values of weights and biases.

The purpose of this algorithm is to overcome such phenomena. Therefore, as the direction of the weight update is determined by its sign, so there is no effect produced by the magnitude of the derivative. The change in values of weights and biases is dependant upon three conditions i.e., if for two successive iterations the derivative of the performance function has same sign with respect to changes in weight, then the up-gradation values of biases and weights are increased. Otherwise the values the up- gradation of biases and weights are decreased for opposite scenario, and for zero value of derivative the up gradation value remains unchanged.

2.3.4 Conjugate Gradient Algorithms:

The values of weights in back propagation algorithm are adjusted by negative of gradient, thus showing the direction of performance function decrease. So this approach does not acquire the fast convergence. Therefore, faster convergence is acquired by conjugate gradient algorithms when search is based on the conjugate directions. Fletcher Reeves update, Polka Ribiere update and Powell Beale restarts are the examples of Conjugate gradient algorithm.

2.3.5 Quasi–newton Algorithm:

It is an alternative approach for fast convergence and optimized outcome rather than conjugate gradient algorithm and do not need calculation of second derivatives. For each iteration, up-gradation of an approximate Hessian matrix is achieved. This update is acquired as a function of gradient. BFGS algorithm and Levenberg Marquardt algorithm are examples of Quasi-Newton algorithm.

2.3.6 One Step Secant Algorithm:

This algorithm facilitates in reduction of gap as well as creating a bridge between quasi Newton algorithms and conjugate gradient algorithm. This algorithm has two main advantages i.e., it does not keep Hessian matrix obtained at each iteration and also considers previously obtained Hessian matrix equivalent to identity matrix. Its edge over other algorithms is that it can calculate new search directions without calculating a matrix inverse.

3. Proposed ANN Architectures:

3.1 Data Engineering and Modeling Strategy:

Data engineering is basically an approach for changing data into important information and matching it with the models (Olaf Wolkenhauer, 2001). It is based on the blend of Pattern Recognition approaches with System Theory. Fig. 5 shows the involvement of different levels towards data engineering approach that can lead to prediction.

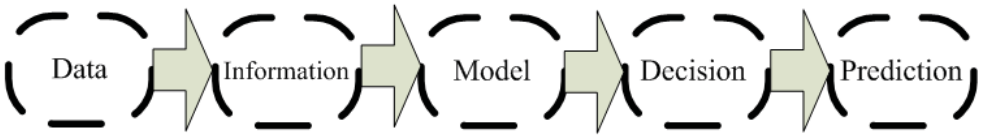


Fig. 5: Data engineering methodology.

Many methodologies have been devised for the purpose of load forecasting using ANN approach [17-18]. In this work, ANN based model development regarding STLF was met by using neural network toolbox. For ANN based STLF architecture modeling, sequence of steps was involved, that are mentioned below:

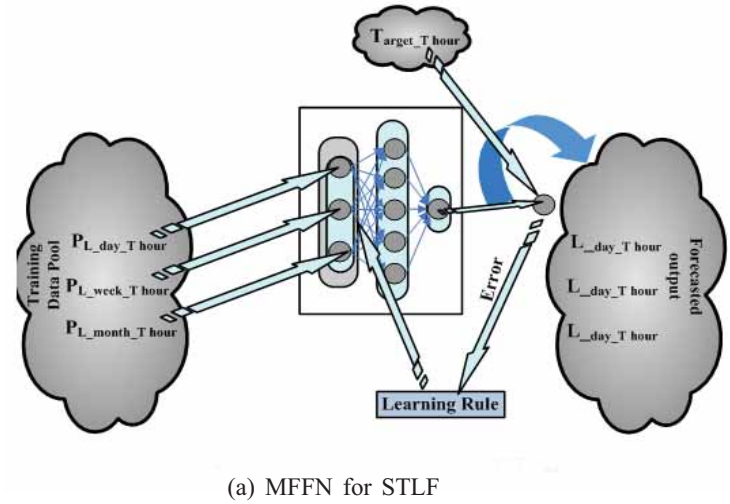
1. Preprocessing hourly load dataset.
2. Neural Network (NN) architecture modeling.
3. Training/ Testing of NN model.
4. Post-training analysis of forecast model.

3.1.2 Pre-processing Load Dataset:

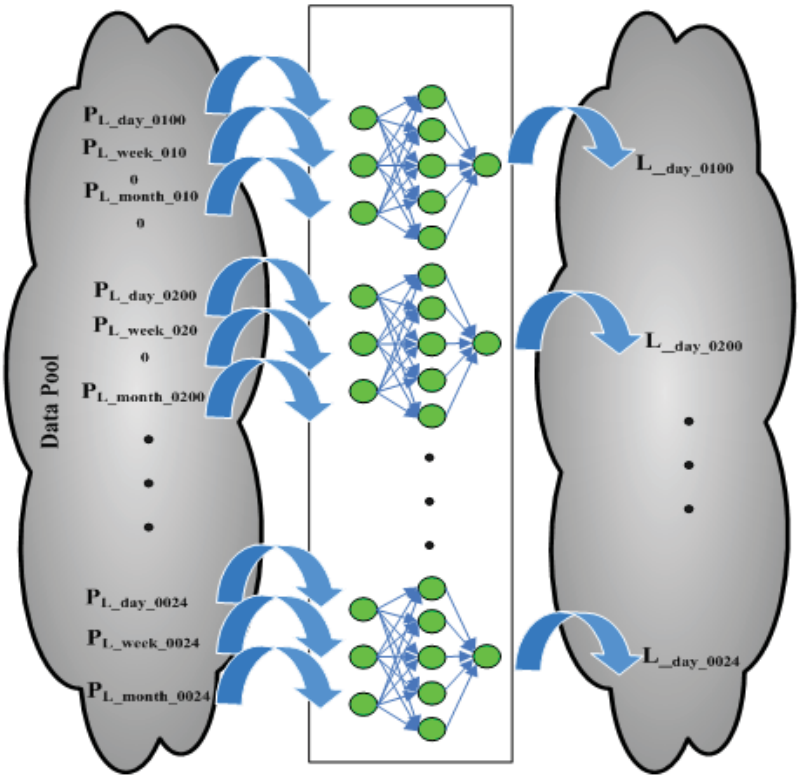
Pre-processing of five years dataset was performed prior to training and testing. Input/output dataset was normalized to lie between 0 and 1. Normalized dataset was divided into training and testing datasets.

3.1.3 ANN Architecture Modeling:

Two different ANN architectures were proposed for a day ahead forecast, namely a Multilayer Feed Forward Network (MFFN) and an ANN engine consisting of twenty four such MFFN architectures. Network topology of MFFN consisted of a three layered structure namely input layer, hidden layer and output layer. The specific load at a given hour does not simply depend upon its previous hour load, but also the actual load at the same hour of previous day, week as well as month. So, keeping in view such technicalities, architectures modeled for a day ahead forecast are shown in Fig. 6 (a) and (b).



(a) MFFN for STLF



(b) ANN engine for STLF

Fig. 6: Different architectures for day ahead forecast.

3.1.4 Training and Testing of ANN Model:

Each forecasting model was evaluated after training. Training dataset consisted of four years of hourly load dataset from year 1991-94. Different types of back propagation based methodologies shown in Fig. 4 were utilized for training as well as evaluating performance of forecasting model. Log-sigmoid activation function was applied to hidden layer neurons, where as linear activation function was utilized for output layer neurons. As far as testing scenario is concerned, both architectures were evaluated upon hourly load dataset of year 1995. Parameters relating forecasting models are summarized in Table. 1.

Table 1: ANN parameters

ANN-Parameters	Settings
Number of input layer neurons	3
Number of hidden layer neurons	5
Number of output layer neurons	1
Activation function for hidden layer	Logsig
Activation function for output layer	Pureline
Epochs	100
Performance function	MSE
Training data	4 years
Test data	1 year

3.5 Post-training Analysis:

For measuring the accuracy of forecast, different performance measures were applied to the output of the model for measuring the forecast accuracy which are mentioned below:
Over fitting condition: Used to check whether data is being memorized or not.

Condition for overfitting:

if $\varepsilon v > \bar{\varepsilon}v + \sigma\varepsilon v$

else

No overfitting

i.e., (validation error > average validation error + standard deviation of validation error).
Curve fitting toolbox was used to check the network's efficiency and goodness of fit based on following criterions**:**
Sum of squares due to error (SSE).

$$SSE=\sum_{i=1}^{n} w_i(y_i-\bar{y}_i)^2$$

R-square.

$$=\frac{SSR=\sum_{i=1}^{n} w_i(\hat{y}_i-\bar{y}_i)^2}{SST=\sum_{i=1}^{n} w_i(y_i-\bar{y}_i)^2}$$

Root mean squared error (RMSE).

$$RMSE = \sqrt{MSE}$$

Where as, y_i represents target, $\bar{y}_i$ represents model output and $\hat{y}_i$ represents average target value.

The quality of forecasting was checked by different error criterions.
Mean square error.

$$MSE=\frac{\sum\left(yt_{pred}\text{-}yt_{act}\right)^2}{N}$$

Mean absolute percentage error.

$$MAPE=\frac{100}{N}\sum\left|\frac{y_{tpred}-y_{tact}}{y_{tact}}\right|$$

Where as, y_{tact} represents actual output and y_{tpred} represents predicted output of the model.

RESULTS AND DISCUSSION

4.1 Comparative Analysis:

To evaluate the performance of ANN based STLF models, different criterions have been devised to identify the specific technique for optimal solution to STLF problem. Therefore, to analyze results in a better way, comparison is done on the basis of applied data engineering approaches and different performance measures. The comparative results obtained from various training methodologies are shown in Table 2.

Table 2: Outcomes of STLF models

(A) MFFN forecast model.

#	ANN Algo	N/W topology			H-Act Ftn	O/p Act Ftn	Epochs	Time (s)	MSE	MAPE	Goodness of fit				
		I L	HL	OL							Correlation Coefficient	Over-fitting	SSE	RSQR	RMSE
1	trainb	3	5	1	logsig	Purelin	100	28.9	0.0015	5.265	0.9342	NO	0.0356	0.0031	0.0385
2	trainbfg	3	5	1	logsig	Purelin	100	67.56	0.0002	2.6482	0.9575	NO	0.0057	0.0113	0.0154
3	trainbr	3	5	1	logsig	Purelin	100	92.1	0.0002	2.4289	0.9594	NO	0.0049	0.01	0.0144
4	traincgb	3	5	1	logsig	Purelin	100	92.1	0.0003	2.7623	0.9538	NO	0.0062	0.0122	0.0161
5	traincgf	3	5	1	logsig	Purelin	100	50.1	0.0002	2.3869	0.9639	NO	0.0054	0.0068	0.015
6	traincgp	3	5	1	logsig	Purelin	100	41.04	0.0002	2.7264	0.9463	NO	0.0091	0.0091	0.0156
7	traingd	3	5	1	logsig	Purelin	100	29.755	0.0012	5.7993	0.7743	NO	0.029	0.0015	0.0347
8	traingda	3	5	1	logsig	Purelin	100	32.86	0.0005	3.4104	0.9097	NO	0.012	0.0137	0.0223
9	traingdm	3	5	1	logsig	Purelin	100	29.39	0.0009	4.9159	0.9161	NO	0.0208	0.0024	0.0294
10	traingdx	3	5	1	logsig	Purelin	100	34.47	0.0008	4.4216	0.9056	NO	0.0193	0.0021	0.0284
11	trainlm	3	5	1	logsig	Purelin	100	92.21	0.0002	2.4659	0.9593	NO	0.005	0.0099	0.0144
12	trainoss	3	5	1	logsig	Purelin	100	102.3	0.0003	2.887	0.9442	NO	0.0063	0.0107	0.0163
13	trainrp	3	5	1	logsig	Purelin	100	42.79	0.0003	2.9257	0.9392	NO	0.0081	0.0132	0.0183
14	trainscg	3	5	1	logsig	Purelin	100	73.4	0.0003	2.9698	0.9551	NO	0.0068	0.0118	0.0168

(B) ANN engine with 24 MFFN

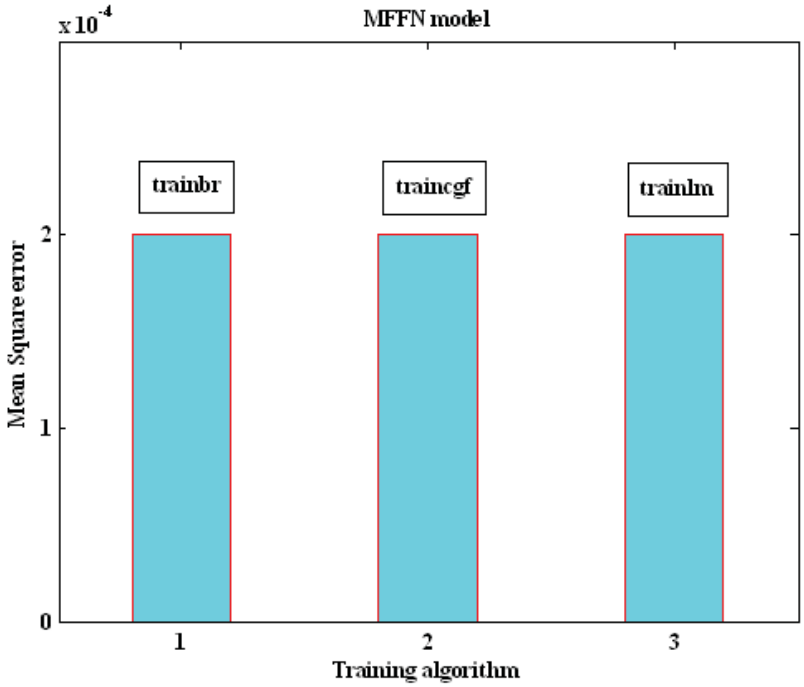
#	ANN Algo	N/W topology			H-Act Ftn	O/p Act Ftn	Epochs	Time (s)	MSE	MAPE	Goodness of fit				
		I L	HL	OL							Correlation Coefficient	Over-fitting	SSE	RSQR	RMSE
1	trainbr	3	5	1	logsig	Purelin	100	104.8	0.0002	2.5152	0.9621	NO	0.0049	0.0105	0.0142
2	traincgf	3	5	1	logsig	Purelin	100	101.89	0.0004	3.5818	0.9371	NO	0.01	0.012	0.0204
3	trainlm	3	5	1	logsig	Purelin	100	166.02	0.0002	2.5086	0.9659	NO	0.0048	0.0105	0.0141

4.2 Speed and Memory Comparison:

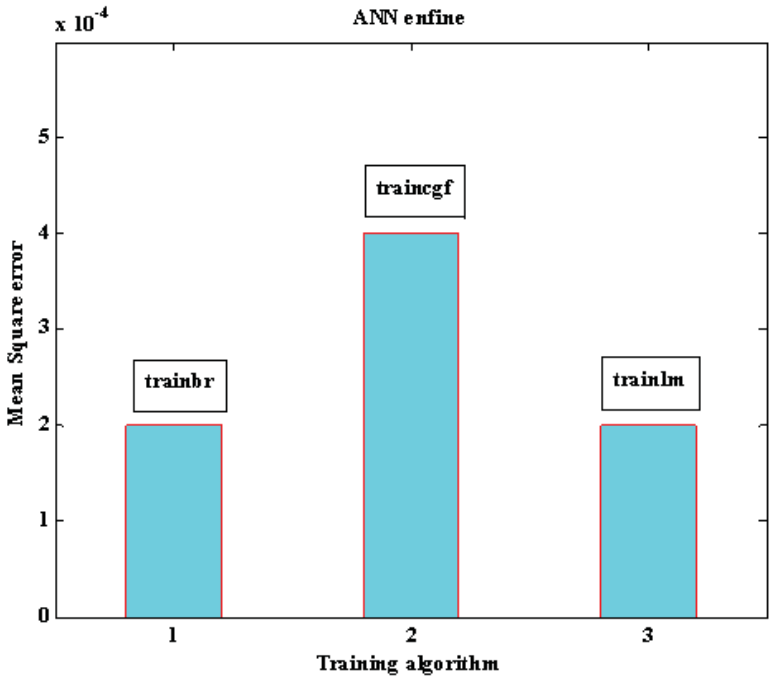
This comparison is done to ensure that, which algorithm produces fastest output response, therefore, minimum utilization of memory. Because, these factors are dependant upon nature of the problem, network topology, training set etc. From Table 2 (a), training algorithms that require minimum training time are *traingd and traingdm with* training time 28.9 sec and 29.3sec.

4.3 Network Accuracy:

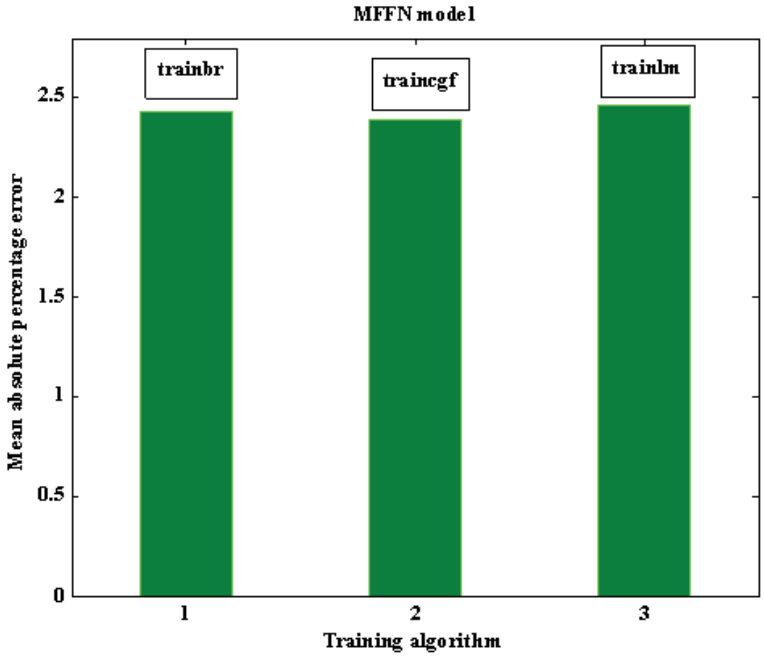
As network accuracy is an important factor to be considered, that is related to different performance measures. Therefore, to measure the performance of network response with certain type of network topology as well as training period, forecasting errors are calculated using MSE and MAPE. From the results shown in Table 2 (a) and (b), training algorithms with more accurate response are *train*cgf, *trainbr* and *trainlm*, with MAPE values 2.3%, 2.4%, 2.4% and MSE values 0.0002 each i.e., for MFFN model. Comparative plots for MAPE and MSE relating both architectures are shown below in Fig. **7**.



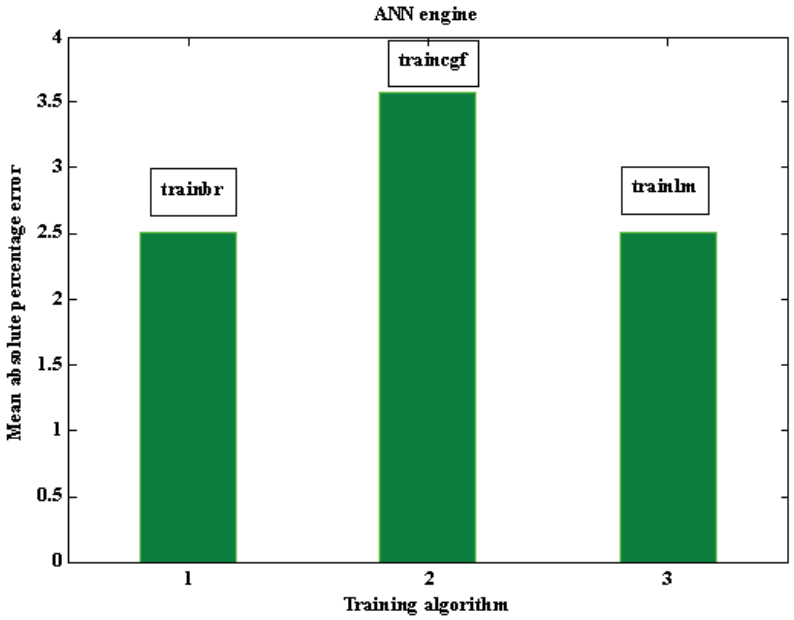
(A) MSE relating MFFN



(B) MSE relating ANN engine



(C) MAPE relating MFFN



(D) MAPE relating ANN engine

Fig. 7: Performance measures for STLF models.

4.4 Goodness of Fit (GOF):

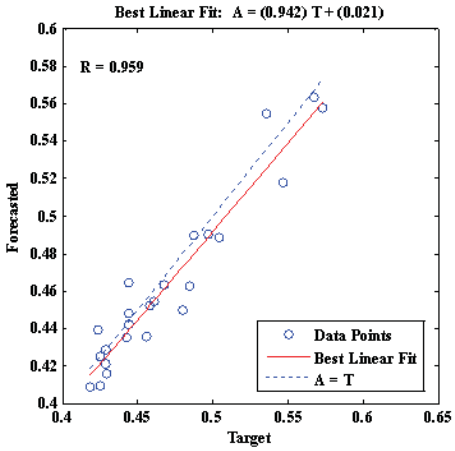
Different algorithms were utilized to train ANN; none of these resulted over fitting. From the results shown in Table 2 (a), the training algorithms with highest correlation are *trainbr, traincgf and trainlm* with correlation coefficient values (R-value) 0.959, 0.963, 0.95 and smaller values for other parameters that relate to goodness of fit.

R-value is computed by performing linear regression between network target and forecasted output. Fig.8 (a-f) shows comparison between actual and forecasted values of electric load using training algorithms with highest correlation values for both architectures.

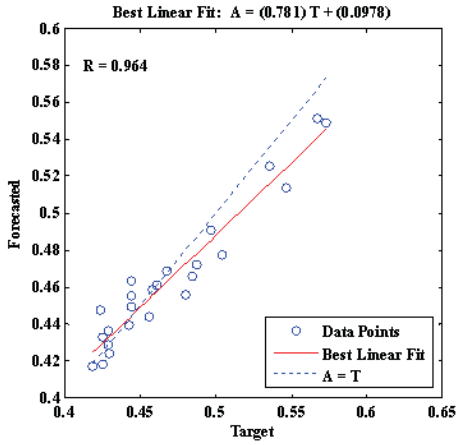
Comparative plots to analyze goodness of fit for both architectures are shown below in Fig. 9 (a) and (b).

4.5 Observations:

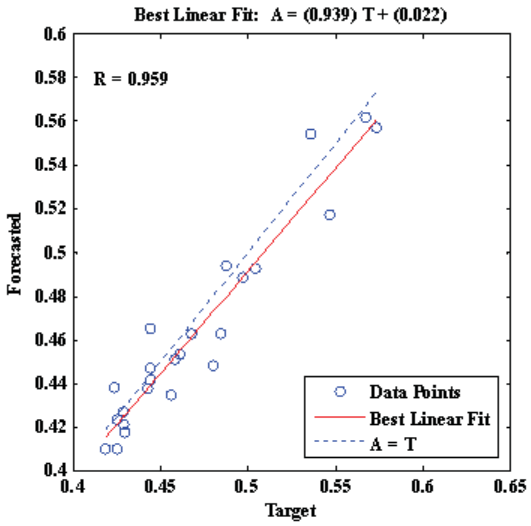
Keeping in view different performance criterians, best results were acquired by three training algorithms, which are *train*br, *traincgf and trainlm* i.e., for MFFN model. These training algorithms were further utilized to train ANN engine (with 24 MFFN). Even with more complex network structure as well as training methodology, ANN engine produced inferior outcomes as compared to MFFN model, as shown in Table 2 (b). Another aspect related to ANN engine was that it took more time for training. Fig. 10 (a) and (b) shows comparative plots relating proposed STLF models considering training algorithms that produced optimum outcomes.



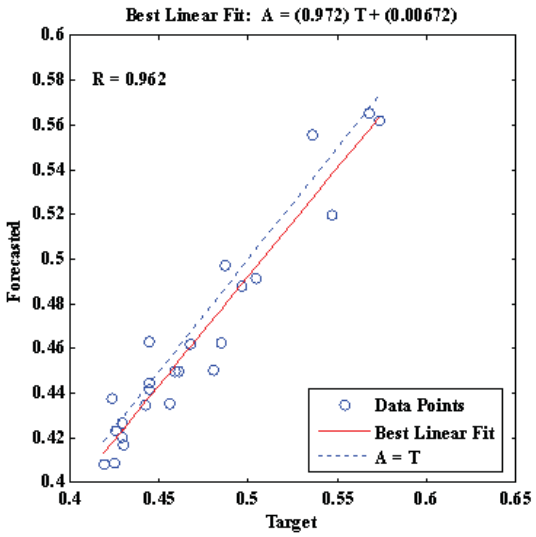
(a) trainbr



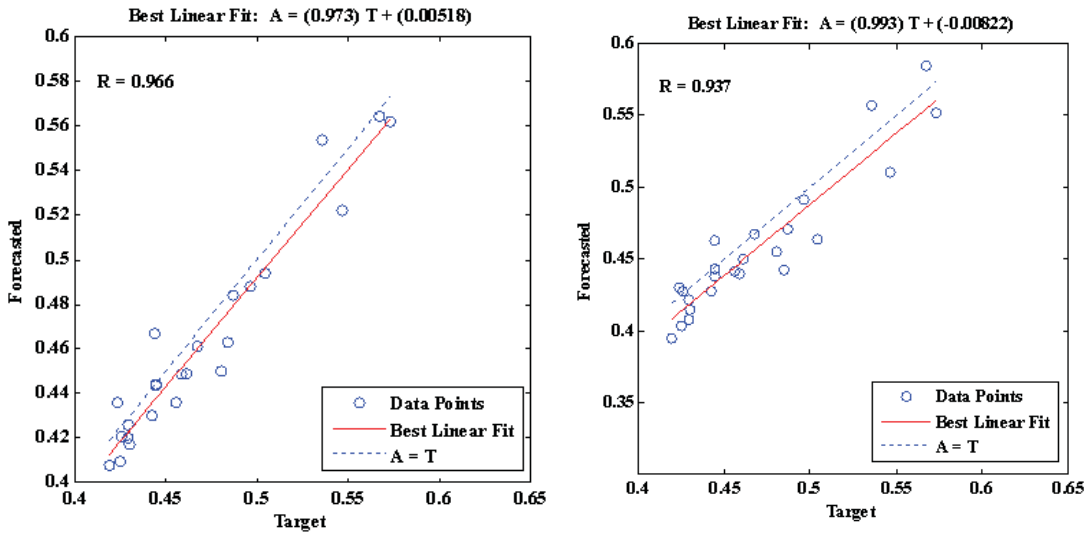
(b) traincgf



(c) trainlm

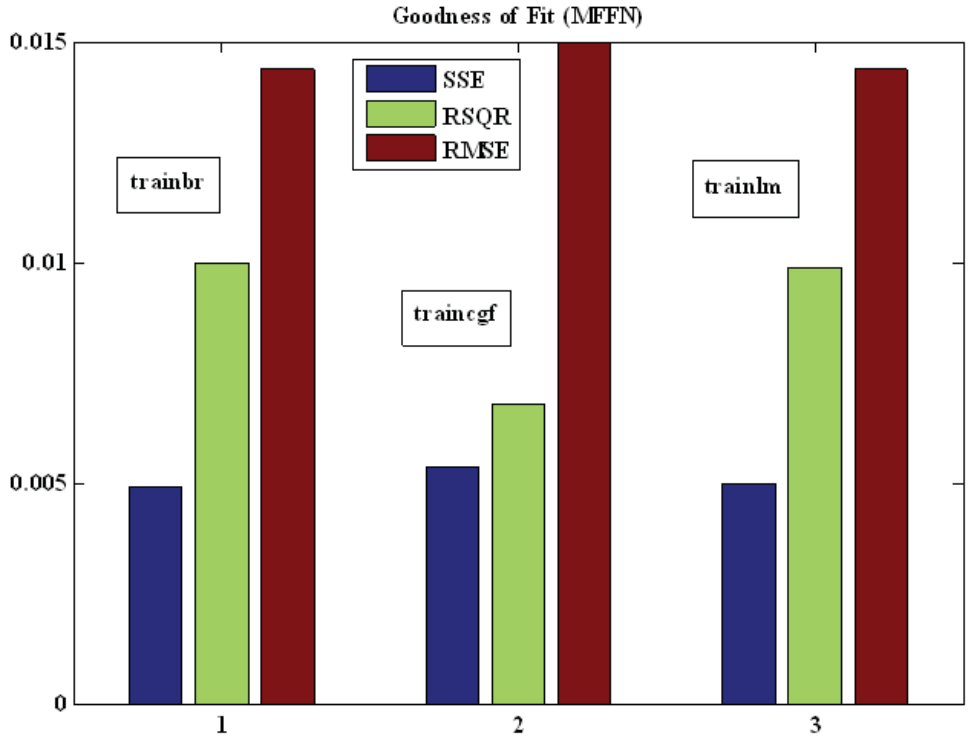


(d) trainbr

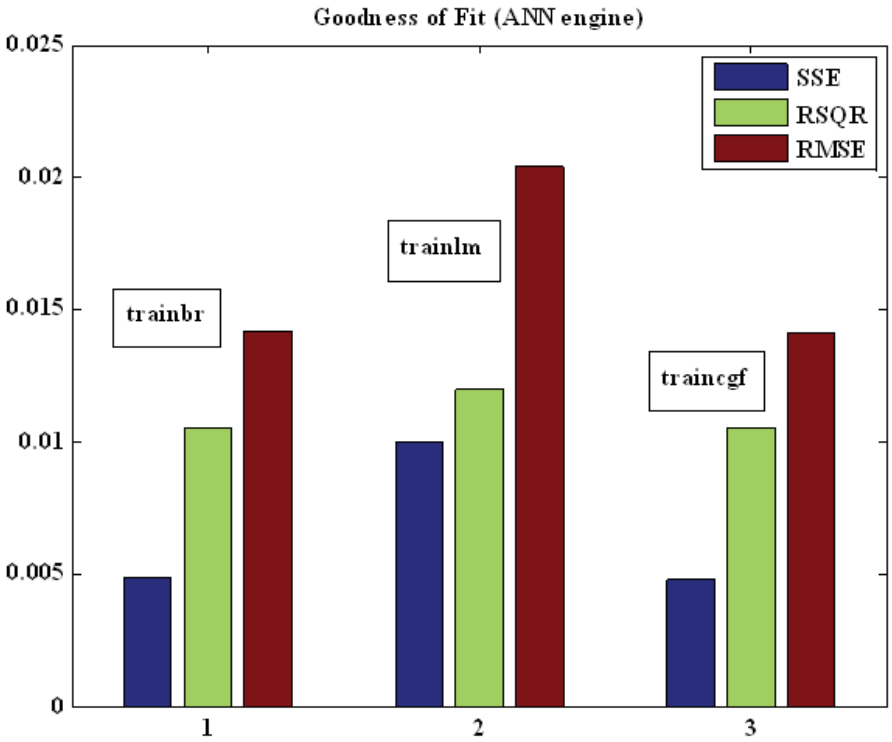


(e) traincgf (f) trainlm

Fig. 8: Comparison between target Vs forecasted electric load using MFFN i.e., (a-c) and ANN engine (d-f).

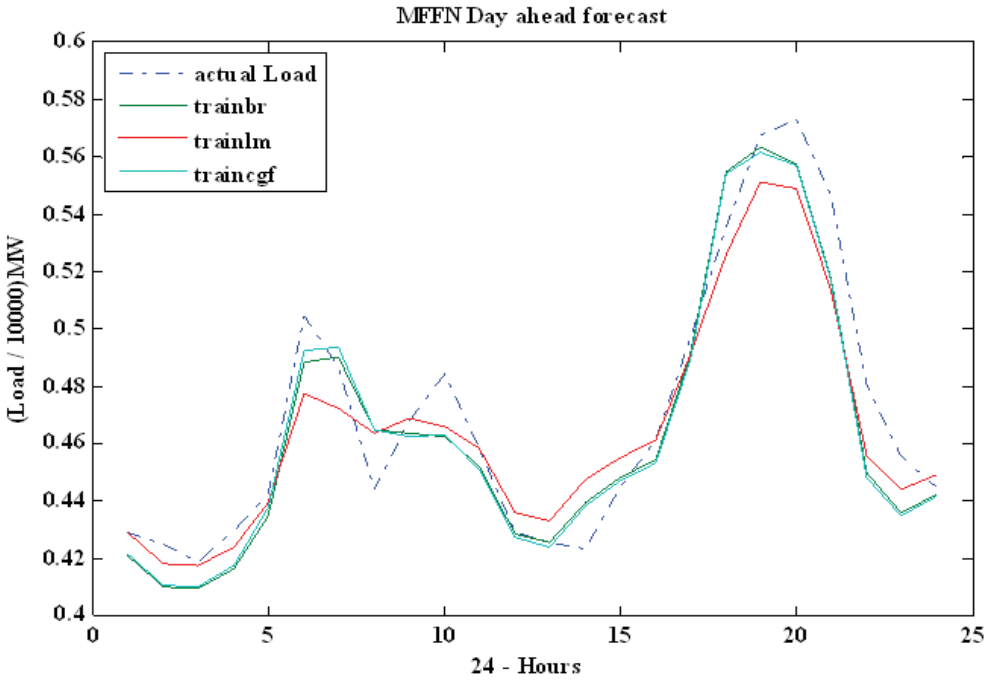


(a) GOF for MFFN

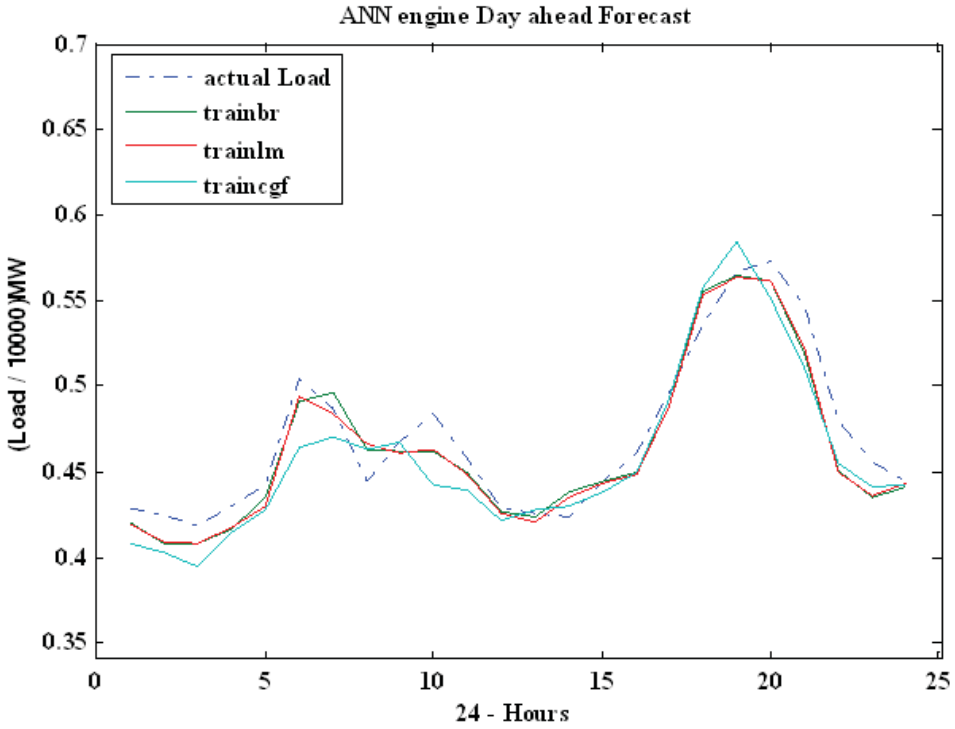


(b) GOF for ANN engine

Fig. 9: Comparative plots relating Goodness of fit.



(a) 24 hr forecast for MFFN



(b) 24 hr forecast for ANN engine

Fig. 10: Comparative plots for Day ahead forecast

Further, its clear from the observations that best outcomes were acquired using MFFN model trained with Levenberg Marquart algorithm.

4.6 Graphical User Interface and Program Structure:

The front-end Graphical User Interface (GUI) for STLF models provides an interactive and user friendly environment. It facilitates the user with different pop-up menus to modify neural network topology, activation function, training algorithm and period of forecast. After execution of the program via load button, resultant plots are displayed in an arranged manner. This provides the user with good understanding of the forecasting model. Layout of GUI is shown in Fig. 11.

This software also permits the user to analyze the predicted output upon different criterions for measuring forecast accuracy. Fig. 12 shows interface to analyze the neural network response through different performance indexes as described in section 3.

5. Conclusion

This problem study incorporates the Short Term Load Forecasting (STLF) using two different strategies for ANN modeling. Out of various back propagation training methodologies some are used for the first time to resolve the problem of STLF. Out of the used techniques faster training algorithms for feed forward training furnished more promising results with superior accuracy rather than traditional methodologies devised for STLF problem on account of different data engineering methods and performance indexes. One of the important aspects of this research work is that the performance of different STLF models is evaluated for improved forecast quality using actual load data of five years and secondly, –these STLF models do not involve much complexity i.e., network topology as well as limited training period to avoid over fitting and acquiring optimum results.

Lastly, an enhanced and interactive Graphical User Interface provides user friendly environment as well as better understanding and analysis ability of different STLF models via front-end interface.

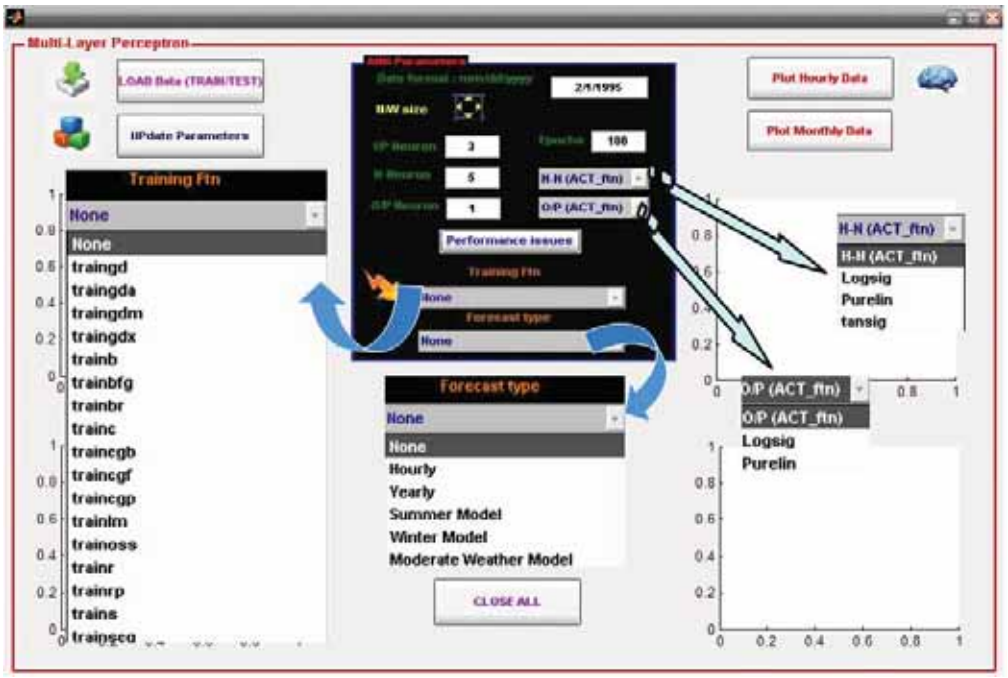


Fig. 11: Interface for artificial neural network based load forecasting model.

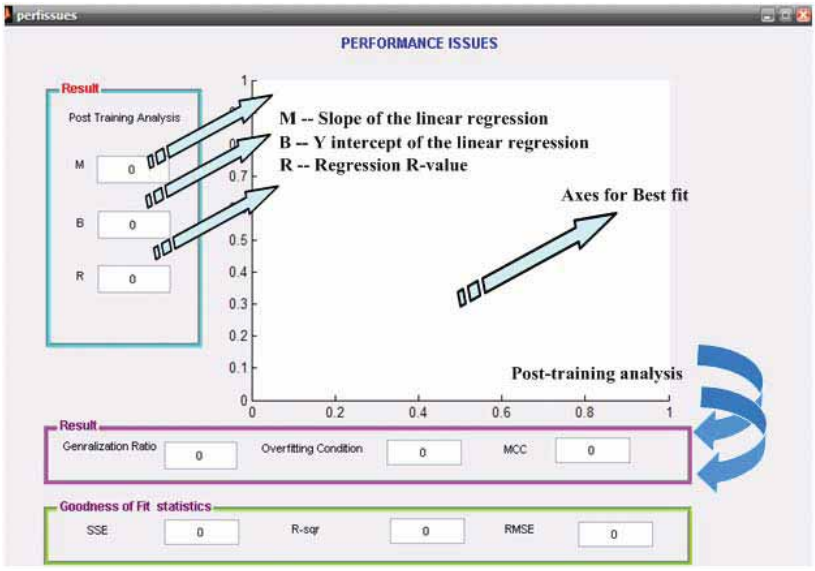


Fig. 12: Interface for performance issues

Notations:

$wt^{(1,0)}$ Hidden layer input weights

$wt^{(2,1)}$ Hidden layer output weights

x_r Input vector

$x^{(1)}$	Hidden neuron output
$d_{r,k}$	Desired output
$Net^{(1)}$	Hidden neuron inputs
$Net^{(2)}$	Inputs to output neurons
$o_{r,k}$	Network outputs
s	Sigmoid function
s'	Sigmoid function (differential)
η	Learning rate

REFERENCES

Alex, D., C. Timothy, 1990. "A regression based approach to short term load forecasting", IEEE transaction on power systems, 5(4): 1535-1550.

Al-Shareef, A.J., E.A. Muhammad and E. Al-Judaibi, 2008. "One Hour Ahead Load Forecasting Using Artificial Neural Network for the Western Area of Saudi Arabia", International Journal of Electrical Systems Science and Engineering, 37: 219-224.

Eugene A. Feinberg, Dora Genethliou, "Applied mathematics for power systems Load Forecasting".

Gross and Galiana, 1987. "Short-Term Load Forecasting", proceedings of the IEEE, 75(12): 1558-1570.

Ibrahim Moghram, Saifure Rahman, 1989. "Analysis Evaluation of five short term load forecasting techniques", IEEE transaction on power systems, 4(4): 1484-1491.

Lahiri, S.K., K.C. Ghanta, 2008. "Development of an artificial neural network correlation for prediction of hold-up of slurry transport in pipelines", chemical engineering science, 63: 1497-1509.

Lru, K., S., Subbarayan, R.R. Shoults, M.T. Manry, C. Kwan, F.L. Lewis, J. Naccarino, 1996. "Comparison of very short term load forecasting techniques", IEEE transactions on power systems, 11(2): 1-8.

Wasserman, P.D., 1989. "Neural computing; Theory and Practice", Van Nostrand Reinhold.

Lee, K.Y., J.H. Park, 1992. "Short term load forecasting using artificial neural network", IEEE transactions on power systems, 7(1): 124-131.

Mcmenamin, J.S. and F.A. Monforte, 1998. "Short term energy forecasting with neural networks", Energy J., 19(4): 43-61.

Olaf Wolkenhauer, 2001. "Data Engineering: Fuzzy Mathematics in Systems Theory and Data Analysis, year.

Sanjib Mishra, 2008. "Short Term Load Forecasting Using Computational Intelligence methods", Masters thesis, National Institute of Technology Rourkela.

Temraz, H.K., M.M.A. Salama and Quintana, 1996. "Application of the decomposition technique for forecasting the load of a large electric power network", IEEE Proc. Gener. Transm. Distrib., 143(1): 13-18.

Widrow, B. and M.E. Hoff, 1986. "Adaptive switching networks", Parallel distributed processing, Cambridge, MA: MIT Press, 1: 123-134.

Yalcinoz, T., 2005. "An educational software package for power systems analysis and operation", International Journal of Electrical Engineering Education, 42(4): 369-382.