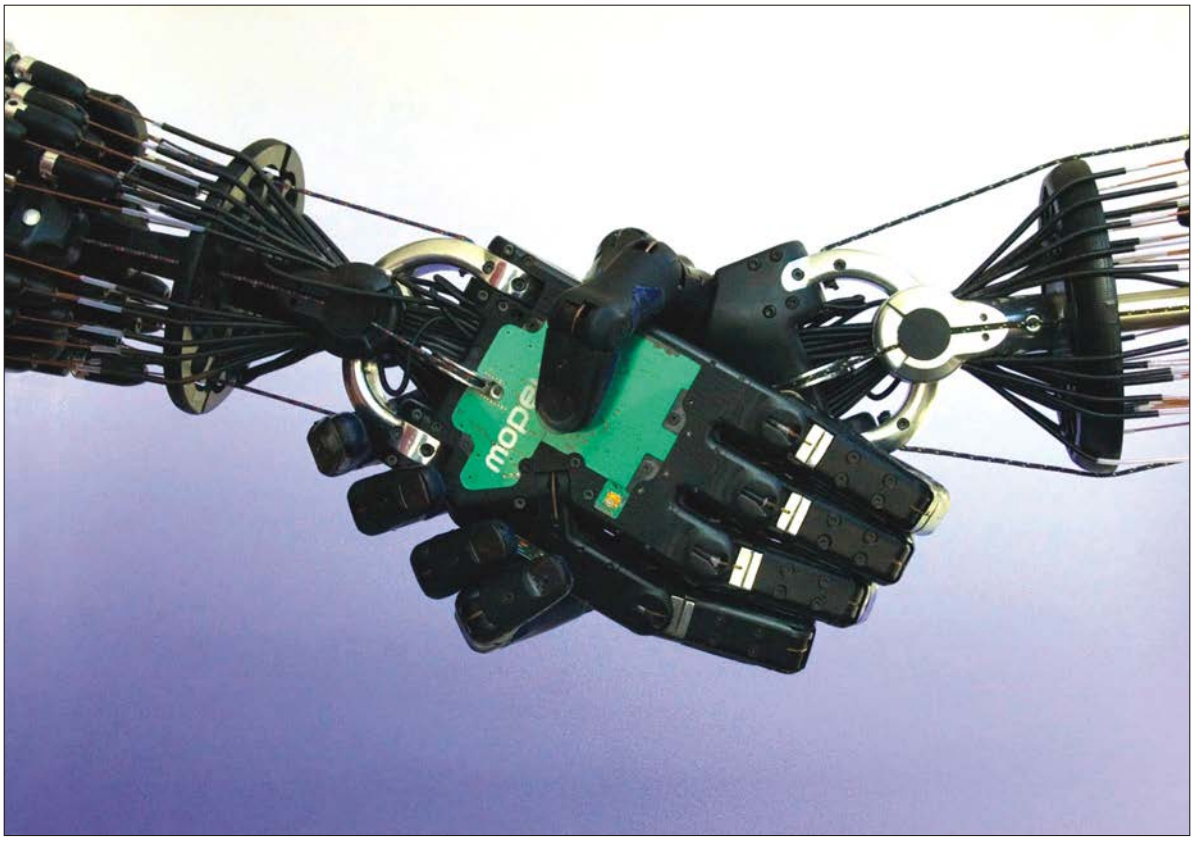

# Journal of Artificial Intelligence

ISSN 1994-5450

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Artificial Intelligence 2 (1): 1-16, 2009
ISSN 1994-5450


# Novel FTLRNN Model for Short Term and Long Term Ahead Prediction of Sun Spots Chaotic Time Series

[1]Sanjay L. Badjate and [2]Sanjay V. Dudul
[1]Jawaharlal Darda of Institute of Engineering and Technology,
Yavatmal (M.S), India
[2]Department of Applied Electronics, Sant Gadgebaba Amravati University,
Amravati (M.S), India

**Abstract:** Multi-Step ahead prediction of a chaotic time series is a difficult task that has attracted increasing interest in recent years. The interest in this study is the development of nonlinear neural network models for the purpose of building short term and long term ahead prediction of monthly sunspots chaotic time series. The solar activity has a measure effect on earth, climate, space weather, satellites and space missions and is a highly nonlinear time series. Such problems exhibit a rich chaotic behavior. In this study the authors compared the performance of three neural network configurations namely a Multilayer Perceptron (MLP), Self Organized Feature Map (SOFM) and Focused Time Lagged Recurrent Neural Network (FTLRNN) for 1, 6, 12,1 8 and 24 months ahead prediction with regards to various performance measures Mean Square Error (MSE), Normalized Mean Square Error (NMSE) and regression (r). The standard back propagation algorithm with momentum term has been used for all the models. The various parameters like number of processing elements, step size, momentum value in hidden layer, in output layer the various transfer functions like tanh, sigmoid, linear-tan-h and linear sigmoid, different error norms $L_1$, $L_2$, $L_3$, $L_4$, $L_5$ to $L_8$, different combination of training and testing samples and epochs variation are exhaustively experimented for obtaining the proposed robust model for the short term and long term ahead prediction. The results suggests that the FTLRNN Model can help in improving the prediction accuracy.

**Key words:** Sunspots chaotic time series, multi- step prediction, focused time lagged neural network, self organizing feature map

## INTRODUCTION

Predicting the future which has been the goal of many research activities in the last century is an important problem for human, arising from the fear of unknown phenomenon and calamities all around the infinitely large world with its many variables showing highly nonlinear and chaotic behavior. Chaotic time series have many applications in various fields of science, e.g., astrophysics, fluid mechanics, medicine, stock market, weather and understand the fundamental feature and processes in system, which are used in every sector of the human life. Recognizing chaotic dynamics is potentially important for understanding and managing real world problems. Neural network have been to the prediction problem (Weigend and Greshenfeld, 1994). Examples range from forecasting the stock market and weather to speech coding (Townshend, 1994), noise cancellation (Leung and Lo, 1993), forecasting bankruptcy, customer service management and industrial forecasting etc. Predicting a chaotic time series using a neural network is of particular interest (Casdagli, 1989). Not only it is an efficient method to reconstruct a dynamical system from an observed time series, but it also has many applications in engineering problems such as speech coding (Leung and Lo, 1993), radar detection, modeling of electromagnetic wave propagation and scattering (Cooper *et al*., 1992).

**Corresponding Author:** Sanjay L. Badjate, Jawaharlal Darda of Institute of Engineering and Technology, Yavatmal (M.S), India

Inspired from the structure of the human brain and the way it is supposed to be operate, neural networks are parallel computational systems capable of solving number of complex problems in such a diverse areas of as pattern recognition, computer vision, robotics, control and medical diagnosis, to name just few (Hykin, 2003). Neural networks are an effective tool to perform any nonlinear input output mappings. It was the (Cybenko, 1989), who first proved that, under appropriate conditions, they are able to uniformly approximate any continuous function to any desired degree of accuracy. It is these fundamental results that allow us to employ neural network as a time series prediction. Since neural networks models do not need any priory assumption about the underlying distribution of the series to be predicted, they are commonly classified as data-driven approach, to contrast them with the model-driven statistical methods. One of the primary reasons for employing neural network was to create a machine that was able to learn from experience. They have the capability to learn the complex nonlinear mappings from a set of observations and predict the future (Dudul, 2007). A hierarchical neural model is reported using Self organizing feature map for short-term load forecasting, where it is compared with MLP (Carpinteiro *et al.*, 2004). Suitability of MLP and Radial basis function NNs is explored in forecasting engine systems reliability (Xu *et al*., 2003). The static MLP network has gained an immense popularity from numerous practical application published over the past decade, there seems to be substantial evidence that multilayer perceptron indeed possesses an impressive ability. There have been some theoretical results that try to explain the reasons for the success (Bahron, 1993) and (Judi, 1995). Most applications are based on feed forward neural networks, such as the Back Propagation (BP) network (Bakker *et al*., 2000), Radial basis function (RBF) network (Zeng and Yurong, 1999) and so on. It has also been shown that modeling capacity of feed forward neural networks can be improved if the iteration of the network is incorporated into the learning process. The static MLP network has gained an immense popularity from numerous practical application published over the past decade, there seems to be substantial evidence that multilayer Perceptron indeed possesses an impressive ability (Dudul, 2005). There have been some theoretical results that try to explain the reasons for the success ( Barron, 1993).

The different chaotic time series attempted with support vector machines and Elman neural network by Thissen *et al*. (2003) for short term ahead prediction of Mackey-Glass chaotic time series. Hong and Wang (2007) used the fuzzy modeling method bases on Singular Value Decomposition (SVD) for prediction of Mackey-glass and Lorenz chaotic time series. A new class of wavelet is developed for learning dynamics for two step ahead prediction of Mackey-glass and the one step ahead of annual sunspots chaotic time series. Min *et al*. (2004) it is described recurrent predictor neural network (Billings and Wei, 2005) structure according to reconstructing phase space and gradient -based and self adaptive algorithm on monthly and yearly sunspots time series. A new class of wavelet network is develop with a standard deviation of 0.0029 for short term ahead prediction o f Mackey-Glass chaotic time series and annual sunspots for 1 step ahead prediction (Billings and Wei, 2005).By using recurrent predictor neural network for monthly sunspots chaotic time series for 6 months ahead prediction with EPA equals to 0.992 and ERMSE equals 4.419, for 10 months ahead prediction with EPA of 0.980 and ERMSE of 7.050, for 15 months ahead prediction of EPA 0.9222 ERMSE equals 13.658 and 20 months ahead prediction EPA of 0.866 and ERMSE of 16.79323 (Min *et al*., 2004).

From the scrupulous review of the related research work, it is noticed that no simple model is available for long term prediction of monthly sunspots chaotic time series so far. It is necessary to develop a simple model that is able to perform short term and long term prediction of monthly sunspots chaotic time series with reasonable accuracy. In view of the remarkable ability of neural network in learning from the instances, it can prove as a potential candidate with a view to design a versatile predictor (forecaster) for the chaotic time series. Hence in this work a novel focused time lagged recurrent neural network model with gamma memory filter is proposed as an intelligent tool for predicting the monthly sun spots chaotic time series.

Defining number of suitable number of hidden nodes is a difficult problem of these network, these networks have limitations to identify the chaotic dynamical systems. It cannot cope up with rapidly changing nonlinear dynamics. Therefore it is necessary a NN configuration that can learn the temporal variation or time structure underlying the data in true sense. In view of this, dynamic modeling will certainly help to improve the learning and generalization ability.

The solar activity has a measure effect on earth, climate, space weather, satellites and space missions and is a highly nonlinear time series. First the optimal static NN based model on MLP is attempted to model the given system. Next on the same parameters the self organizing feature map and best dynamic focused time lagged NN model with built in gamma memory is estimated for prediction for all the steps. Next the comparison between these models are carried out on the basis of the performance measures such as Mean Square Error (MSE), Normalized Mean Square Error (NMSE) and Correlation coefficient (r) on testing data set (The data set which is not used for training) as well as training data set for the short term and long term ahead (K = 1,6,12,18,24) prediction. The various parameters like number of hidden layers, number of processing elements, step size, momentum value in hidden layer, in output layer the various transfer functions like tanh, sigmoid, linear-tan-h and linear sigmoid, different error norms $L_1$, $L_2$, $L_3$, $L_4$, $L_5$ to $L_8$, different combination of training and testing samples and epochs variation during training are exhaustively experimented for obtaining the proposed robust model for the short term and long term ahead prediction.

## MATERIALS AND METHODS

The Sunspot number is a good measure of solar activity which has a period of 11 years, called solar cycle. The solar activity has measure affects on earth, climate, space weather, satellite and space mission. Sunspots are often related to intense magnetic activity such as coronal loops and reconnection. Most solar flares and coronal mass ejection originate in magnetically active regions around sunspot groupings (Huebner *et al*., 1989). Sunspot numbers rise and fall with an irregular cycle with a length of approximately 11 years. In addition to this, there are variations over longer periods. The recent trend is upward from 1900 to the 1960s, then somewhat downward. The Sun was last similarly active over 8,000 years ago. The number of sunspots has been found to correlate with the intensity of solar radiation over the period since 1979 when satellite measurements of radiation are available. Since sunspots are dark it might be expected that more sunspots lead to less solar radiation and a decreased solar constant. However, the surrounding areas are brighter and the overall effect is that more sunspots mean a brighter sun. But due to intrinsic complexity of time behavior and a lack of theoretical model, the prediction of solar cycle is difficult. Many prediction techniques have been examined on the yearly sunspots number is an indicator of solar activity. However, in more recent studies the international monthly smoothed sunspots number time series, which has a better time resolution and accuracy has been used. The monthly smoothed sunspots number time series has been downloaded from the SIDC World data center for the sunspot index (http://sidc.oma.be/index.php3). The study was conducted at solar physics research department of the royal observatory of Belgium.

The data considered the monthly variations from January 1749 to December 2006. The total samples are 3096 considered. The series is normalized in the range of -1 to +1 (Fig. 1).

### Neural Network Models

### Static NN Based Model

Static NNs typically uses MLP as a backbone. They are layered feed forward networks typically trained with static back propagation. MLP solid based model has a solid foundation (Quin *et al*., 1992; Naraendra and Parthasarathy, 1990). The main reason for this is its ability to model simple as well as complex functional relationships. This has been proven through number of practical

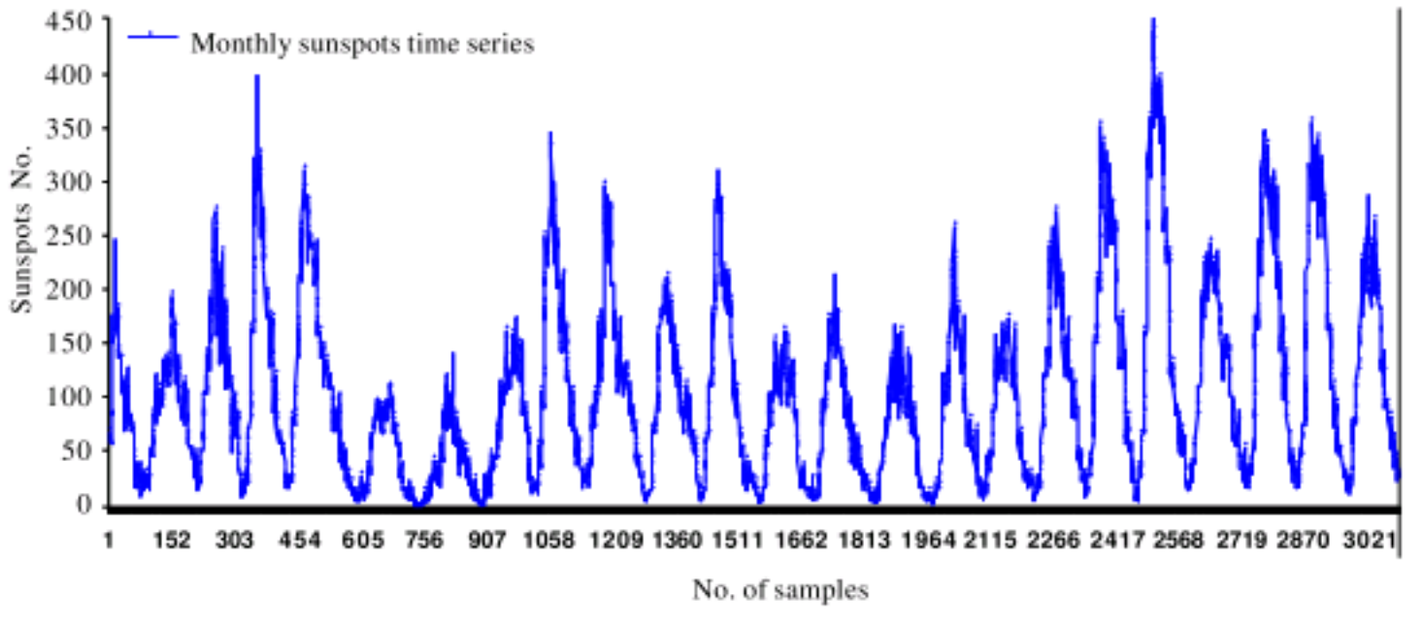


Fig. 1: Monthly sun spots Chaotic time series

applications (Demuth and Beale, 2004). Cybenko (1989) is shown that all continuous functions can be approximated to any desired accuracy, in terms of the uniform norm, with a network of one hidden layer of sigmoid or (hyperbolic tangent) hidden units and a layer of linear or tan h output unit to include in the hidden layer. This study does not explain how many units to include in the hidden layer. Franklin *et al*. (1998) reported a significant result is derived approximation capabilities of two layer perception networks when the function to be approximated shows certain smoothness. The biggest advantage of using MLP NN for approximation of mapping from input to the output of the system resides in its simplicity and the fact that it is well suited for online implementation. The objective of training is then to determine a mapping from a set of training data to the set of possible weights so that the network will produce predictions y (t), which in some sense are close to the true outputs y (t). The prediction error approach is based on the introduction of measure of closeness in terms of Mean Square Error (MSE) criteria:

$$V_N(\theta, Z^N) = (1/2N)\sum_{t=1}^{N}[y(t) - y\hat{}(t|\theta)]^2 = (1/2N)\sum_{t=1}^{N}\varepsilon^2(t,\theta) \quad (1)$$

The weights are then founds as: $\hat{} = \arg\min_0 V_N(,Z^N)$, by some kind of iterative minimization scheme:

$$\theta^{(i+1)} = \theta^{(i)} + \mu^{(i)} f^{(i)}$$

where $\theta^{(i)}$ specifies the current iterate (number "i"), $f^{(i)}$ is the search direction and $\mu^{(i)}$ is the step size.

When NN has been trained, the next step is to evaluate it. This is done by standard method in statistics called independent validation (Fredric and Kostanic, 2002). It is never a good idea to assess the generalization properties of a NN based on training data alone. This method divides the available data sets into two sets namely training data set and testing data set. The training data set are next divide into two partitions: the first partition is used to update the weights in the network and the second partition is used to assess (or cross validate) the training performance. The testing data set are then used to assess how the network has generalized. The learning and generalization ability of the estimated NN based model is assessed on the basis of certain performance measures such as MSE, NMSE and the regression(r) ability of the NN by visual inspection of the regression characteristics for different outputs of system under study.

**FTLRNN Model**

Time Lagged Recurrent Networks (TLRNs) are MLPs extended with short term memory structures. Here, a static NN (e.g., MLP) is augmented with dynamic properties (Dudul, 2007). This, in turn, makes the network reactive to the temporal structure of information bearing signals. For a NN to be dynamic, it must be given memory. This memory may be classified into short-term and long-term memory. Long term memory is built into a NN through supervised learning, whereby the information content of the training data set is stored (in part or in full) in the synaptic weights of the network (Principe *et al*., 2000). However, if the task at hand has a temporal dimension, some form of short-term memory is needed to make the network dynamic. One simple way of building short-term memory into the structure of a NN is through the use of time delays, which can be applied at the input layer of the network (focused). A short-term memory structure transforms a sequence of Samples into a point in the reconstruction space. This memory structure is incorporated inside the learning machine. This means that instead of using a window over the input data, PEs created are dedicated to storing either the history of the input signal or the PE activations.

The input PEs of an MLP are replaced with a tap delay line, which is followed by the MLP NN. This topology is called the focused Time Delay NN (TDNN). The focused topology only includes the memory kernels connected to the input layer. This way, only the past of the input is remembered. The delay line of the focused TDNN stores the past samples of the input. The combination of the tap delay line and the weights that connect the taps to the PEs of the first hidden layer are simply linear combiners followed by a static non-linearity. Typically, a gamma short-term memory mechanism is combined with nonlinear PEs in restricted topologies called focused. Basically, the first layer of the focused TDNN is a filtering layer, with as many adaptive filters as PEs in the first hidden layer. The outputs of the linear combiners are passed through a non linearity (of the hidden-layer PE) and are then further processed by the subsequent layers of the MLP for system identification, where the goal is to find the weights that produce a network output that best matches the present output of the system by combining the information of the present and a predefined number of past samples. (given by the size of the tap delay line) (Bert de and Principe, 1992). Size of the memory layer depends on the number of past samples that are needed to describe the input characteristics in time. This number depends on the characteristics of the input and the task. This focused TDNN can still be trained with static back-propagation, provided that a desired signal is available at each time step. This is because the tap delay line at the input layer doesn't have any free parameters. So, the only adaptive parameters are in the static feed forward path.

The memory PE receives in general many inputs, $x_1(n)$ and produces multiple outputs $y = [y_0(n), ...., y_D(n)]T$, which are delayed versions of $y_0(n)$ the combined input,

$$y_k(n) = g(y_{k-1}(n) \quad y_0(n) = \sum_{j=1}^{P} x_j(n) \tag{2}$$

where, g(.) is a delay function.

These short-term memory structures can be studied by linear adaptive filter theory if g(.) is a linear operator. It is important to emphasize that the memory PE is a short-term memory mechanism, to make clear the distinction from the network weights, which represent the long-term memory of the network.

There are basically two types of memory mechanisms: memory by delay and memory by feedback. It is necessary to find the most general linear delay operator (special case of the Auto Regressive Moving Average model) where the memory traces $y_k(n)$ would be recursively computed from the previous memory trace yk-1(n). This memory PE is the generalized feed forward memory PE. It can be shown that the defining relationship for the generalized feed forward memory PE is mentioned (Bret de and Principie, 1992).

$$g_k(n) = g(n) * g_{k-1}(n) k \geq 1 \tag{3}$$

where, * is the convolution operation, g(n) is a causal time function and k is the tap index. Since this is a recursive equation, $g_0(n)$ should be assigned a value independently. This relationship means that the next memory trace is constructed from the previous memory trace by convolution with the same function g(n), the memory kernel yet unspecified. Different choices of g(n) will provide different choices for the projection space axes. When we apply the input x(n) to the generalized feed forward memory PE, the tap signals $y_k(n)$ become:

$$y_k(n) = g(n)*y_{k-1}(n)\ k \geq\ 1 \tag{4}$$

the convolution of yk – 1(n) with the memory kernel. For k = 0, we have:

$$y_0(n) = g_0(n)*x(n) \tag{5}$$

where, $g_0(n)$ may be specified separately. The projection x(n) of the input signal is obtained by linearly weighting the tap signals according to:

$$x(n) = \sum_{k=0}^{D} w_k y_k(n) \tag{6}$$

The most obvious choice for the basis is to use the past samples of the input signal x(n) directly, that is the kth tap signal becomes $y_k(n) = x(n – k)$. This choice corresponds to:

$$g(n) = \delta(n - 1) \tag{7}$$

In this case $g_0(n)$ is also a delta function δ(n) (delta function operator used in the tap delay line). The memory depth is strictly controlled by D, that is the memory traces store the past D samples of the input. The time delay NN uses exactly this choice of basis.

The gamma memory PE attenuates the signals at each tap because it is a cascade of leaky integrators with the same time constant gamma modal. The gamma memory PE is a special case of the generalized feed forward memory PE where,

$$g(n) = \mu(1-\mu)^n n \geq 1 \tag{8}$$

and $g_0(n) = \delta(n)$. The gamma memory is basically a cascade of low pass filters with the same time constant 1-μ. The over all impulse response of the gamma memory is:

$$g_p(n) = \binom{n-1}{p-1}\mu^p(1-\mu)^{n-p},\ n \geq p \tag{9}$$

where, (:) is a binomial coefficient defined by:

$$\binom{n}{p} = \frac{n(n-1)....(n-p+1)}{p!}$$

For integer values of n and p, the overall impulse response $g_p(n)$ for varying p represents a discrete version of the integrand of the gamma function, hence the name of the memory.

The gamma memory PE has a multiple pole that can be adaptively moved along the real Z-domain axis, that is the gamma memory can implement only low pass ($0<\mu<1$) or high pass ($1<\mu< 2$) transfer functions. The high pass transfer function creates an extra ability to model fast-moving signals by alternating the signs of the samples in the gamma PE (the impulse response for $1<\mu< 2$ has alternating signs). The depth in samples parameters (D) is used to compute the number of taps (T) contained within the memory structure(s) of the network.

**Self Organizing Feature Maps (SOFM)**

These networks are based on the competitive learning; the output neurons of the network compete among themselves to be activated or fired, with the result that only one output neuron, or one neuron per group, is on at any one time. An output neuron that wins the competition is called a winning neuron. The essential parameters of the algorithm are :

- A continuous input space of activation patterns that are generated in accordance with a certain probability distribution
- A topology of the network in the form of a lattice of neurons, which defines a discrete ut space (Haykin, 2003)

The values of weight vectors are updated for the winning node and its neighbors. The weight vectors are calculated as follows:

$$\bar{W}^{1}(t+1) = \bar{W}^{1}(t) + \eta(t)N(c,r)[\bar{X}_{k} - \bar{W}^{1}(t)] \tag{10}$$

where, Input vector X has K patterns and the number of elements of weight vector is equal to the number of the processing elements in the output layer, t-number of iteration, c-number of cycle, $\eta$ (t)-learning rate, N(c, r)-neighborhood function, r-neighborhood radius.

**Performance Evaluation Measures**

Three different types of statistical performance evaluation criteria were employed to evaluate the performance of various models developed in this study. These are as follows:

**Mean Square Error (MSE)**

The mean square error is given by:

$$MSE = \frac{\sum_{j=0}^{p}\sum_{i=0}^{N}(d_{ij} - y_{ij})^2}{N * P} \tag{11}$$

Where:

p = No. of output Pes (processing elements)
N = No. of exemplars in the data set
$y_{ij}$ = Network output for exemplar i at Pej
$d_{ij}$ = Desired output for exemplar i at PEj

**NMSE (Normalized Mean Square Error)**

The normalized mean square error is defined by the following formula:

$$NMSE = \frac{P * N * MSE}{\sum_{j=0}^{P}\left[\frac{N\sum_{i=0}^{N} d_{ij}^{\,2} - \sum_{i=0}^{2} d_{ij}}{N}\right]} \tag{12}$$

Where:

P = No. of output PEs
N = No. of exemplars in data set
MSE = Mean square error
$d_{ij}$ = Desired output for exemplar i at pej

**Correlation Coefficient/Regression Coefficient (r)**

The Mean Square Error (MSE) can be used to determine how well the network output fits the desired output, but it does not necessarily reflect whether the two sets of data move in the same direction. For instance by simply scaling the network output, we can change the MSE without changing the directionality of the data. The correlation coefficient solves this problem. By definition, the correlation coefficient between a network output x and a desired output d is.

$$r = \frac{\frac{\sum_{-i}(xi - \bar{x})(di - \bar{d})}{N}}{\sqrt{\frac{\sum_{i}(d - \bar{d})}{N}}\sqrt{\frac{\sum_{i}(xi - x)}{N}}} \tag{13}$$

Where:

$$\bar{x} = \sum_{i=1}^{N} x_i$$

$$\bar{d} = \frac{1}{N}$$

The correlation coefficient is confined to the range [-1, 1].

## RESULTS AND DISCUSSION

The choice of the number of hidden layers and the number of hidden units in each hidden layers is critical (Turchenko, 2004). It has been established that a MLPNN that has only one hidden layer, with sufficient number of neurons, acts as a universal approximators of nonlinear mappings (Hornik *et al*.,1989; Huang *et al*., 2000). The tradeoff between accuracy and complexity of the model should be resolved accurately (Lee and Lam, 1995). In practice, it is very difficult to determine a sufficient number of neurons necessary to achieve the desired degree of approximation accuracy. Frequently the number of units in the hidden layer is determined by trial and error. To determine the weight values, one must have a set of examples of how the output should relate to the inputs. The task of determining the weights from these examples is called training or learning and is basically a conventional estimation problem. That is, the weights are estimated from the examples in such away that the network, according to metric, models the true relationship as accurately as possible. Since learning is a stochastic process, the learning curve may be drastically different from run to run. In order to compare the performance of a particular search methodology or effect of different parameters have on a system, it is needed to obtain the average learning curve over the number of runs so that the

randomness can be averaged out. An exhaustive and careful experimentations has been carried to determine the configuration of the static MLP Model and the optimal proposed FTLRNN model for all the step (K = 1,6,12,18,24) prediction (Fig. 2).

The comparative study of various training algorithms for one hidden layer with 10 neurons has been carried out for the Monthly sunspots time series for 6 months ahead prediction for the data 60% as a training, 15% as a Cross Validation(CV) and 25% as a testing samples. The maximum number of epochs was set to 2000 in supervised learning model for FTLRNN model for the following Wples = 10, No. of taps = 6, Tap Delay = 1. Trajectory Length = 50. Step learning (Step), Quick Propagation (QP), Momentum learning (MOM), Delta Bar Delta (DBD),and Conjugate Gradient (CG) learning were used for comparison.

In rigorous experimental study, the number of hidden layer with hyperbolic (tanh) activation functions is gradually increased from 2 to 100 and every time the network is trained three times with random weight-initialization till the value of cross-validation error exceeds the training error. All the training process saves automatically the weights at minimum cross-validation error. The average of minimum $MSE_s$ is found to be minimum for the number of hidden layers PEs at 15. Variation of average correlation coefficient and average MSE with number of hidden neurons, for the training, testing and cross-validation data sets is as shown in Fig. 3 and 4.

The variation of average of minimum MSEs (for 3 training runs) versus number of hidden nodes is plotted as shown in Fig. 5 for the series. Maximum value of average correlation coefficient r and minimum value of average MSE for testing data set was obtained for number of hidden PEs equal to 15. Thus increase in the value above 15 did not show any improvement in the performance of the

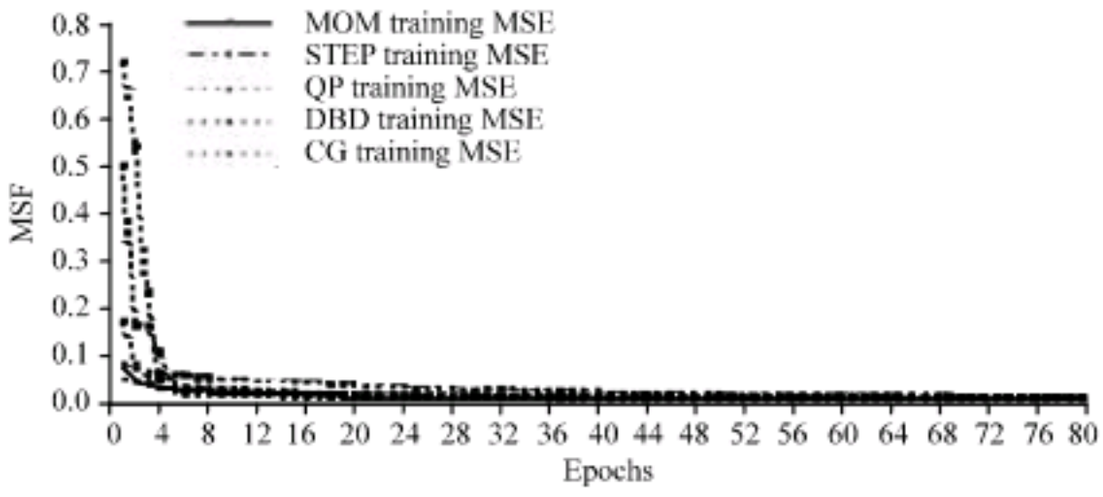


Fig. 2: Comparison of different learning curves for training of FTLRNN for the real time monthly sunspots chaotic time series

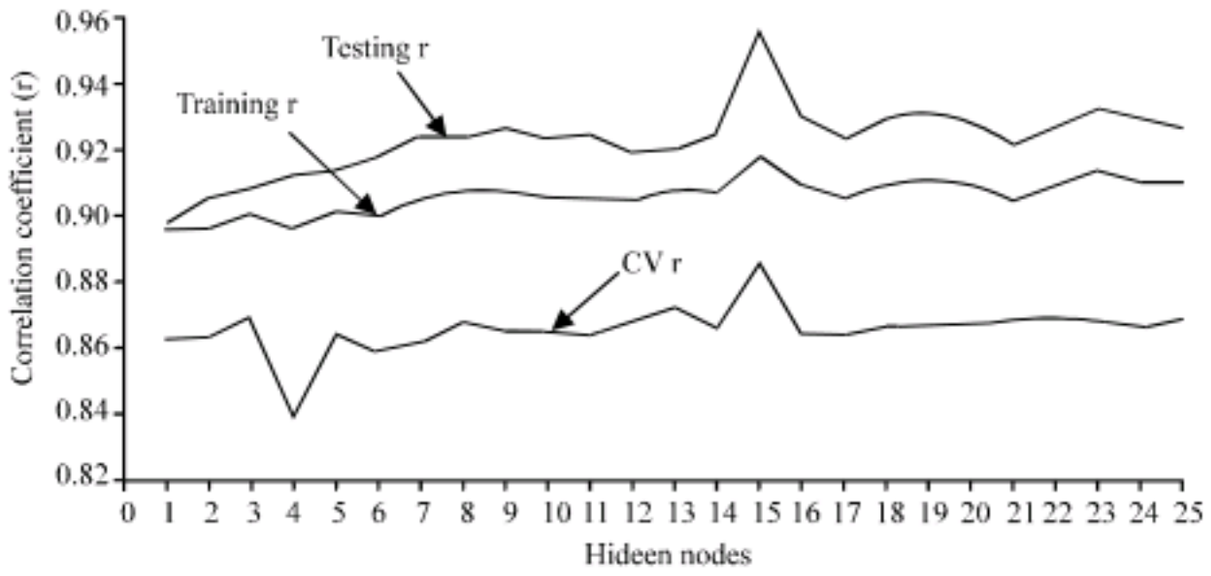


Fig. 3: Variation of correlation coefficient r with number of hidden neurons for training and cross validation and testing data sets

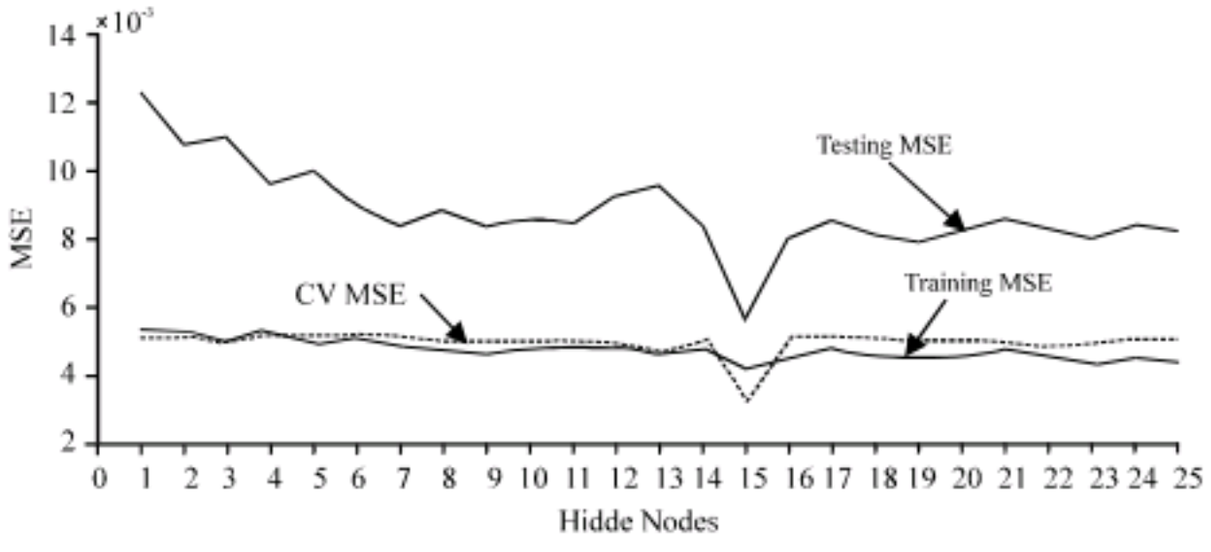


Fig. 4: Variation of average of minimum MSEs with number of hidden neurons for training and cross validation data sets

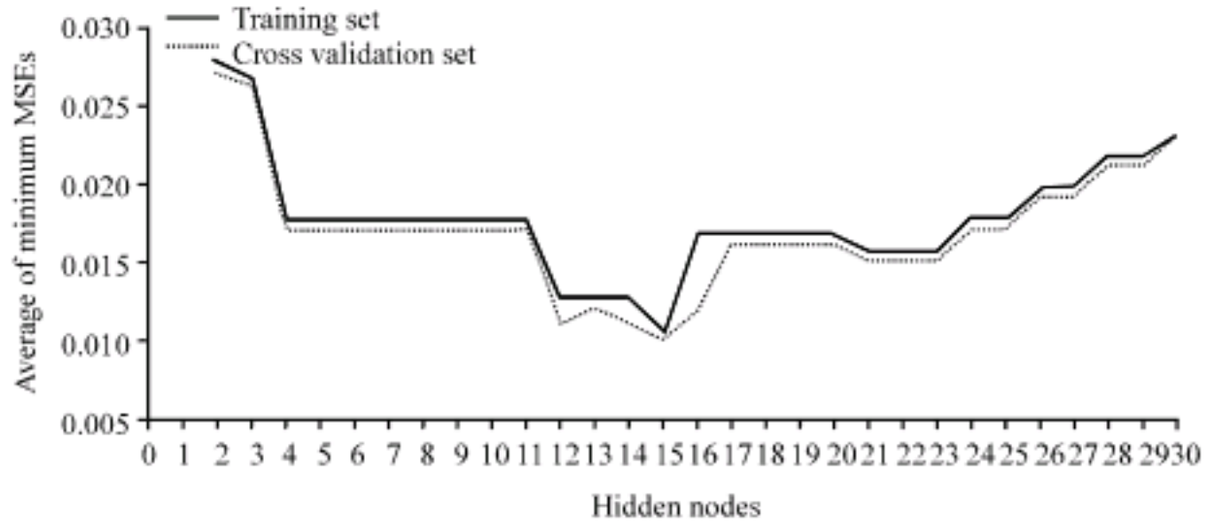


Fig. 5: Variation of average of minimum MSEs with number of hidden neurons for training and cross validation data sets for Monthly Sunspot time series

Table 1: Parameters for the neural network models

| Parameters | Hidden layer | Output layer |
|---|---|---|
| Processing elements | 15 | 1 |
| Transfer function | Tanh | Tanh |
| Learning rule | Momentum | Momentum |
| Step size | 1.0 | 0.1 |
| Momentum | 0.8 | 0.8 |

forecaster. Fifteen PEs in the hidden layer was the optimum choice to develop the FTLRNN with respect to minimum computing resources that further helps in the reduction of time of the neural network. It was found to be considerably high for the other values of hidden PEs. When we attempted to increase the number of hidden layer and the number of processing element in the hidden layer, the performance of the model is not to seen to improve significantly. On the contrary it takes too long time for training because of complexity of the model and the performance of network did not significantly improved.

Then the possible variations for the model such as different transfer functions like tan h, linear tanh, sigmoid, linear sigmoid in output layer, different supervised learning rules like momentum, step, conjugant gradient and quick propagation are investigated in simulation. The step size and momentum are gradually varied from 0.1 to 1 for static back propagation rule. After meticulous examination of the performance measures like MSE, NMSE, Correlation Coefficient and the regression ability, the optimum parameters are found and mentioned in the Table 1 for 60% used as training samples, 25% as testing samples and 15% cross validation samples.

Table 2: Performance of neural network models for testing data set

| | MLP neural network | | | FTLRNN | | | SOFM | | |
|---|---|---|---|---|---|---|---|---|---|
| K (step) | MSE | NMSE | r | MSE | NMSE | r | MSE | NMSE | r |
| 1 | 0.00247 | 0.04533 | 0.97569 | 0.00235 | 0.04319 | 0.97976 | 0.00460 | 0.08426 | 0.96438 |
| 6 | 0.01229 | 0.22340 | 0.89170 | 0.01141 | 0.20730 | 0.90370 | 0.01422 | 0.25840 | 0.87730 |
| 12 | 0.02387 | 0.43132 | 0.77184 | 0.01693 | 0.30599 | 0.85500 | 0.02623 | 0.47398 | 0.75317 |
| 18 | 0.03989 | 0.71740 | 0.57410 | 0.02266 | 0.04075 | 0.79770 | 0.04209 | 0.75710 | 0.55470 |
| 24 | 0.05380 | 0.96580 | 0.35840 | 0.03059 | 0.54899 | 0.71497 | 0.05593 | 1.00360 | 0.33874 |

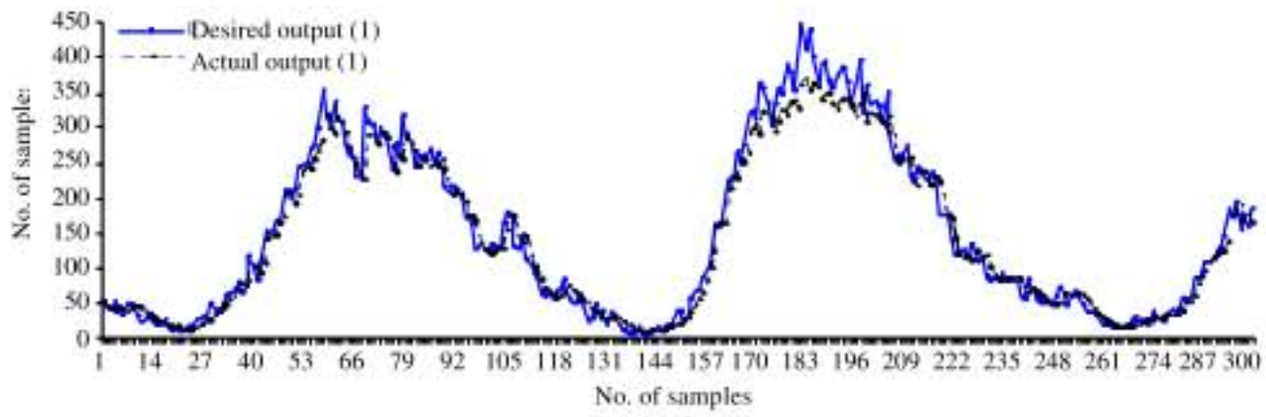


Fig. 6: Plot between Desired Output Vs Actual Network outputs for 1 month ahead prediction for FTLRNN Model

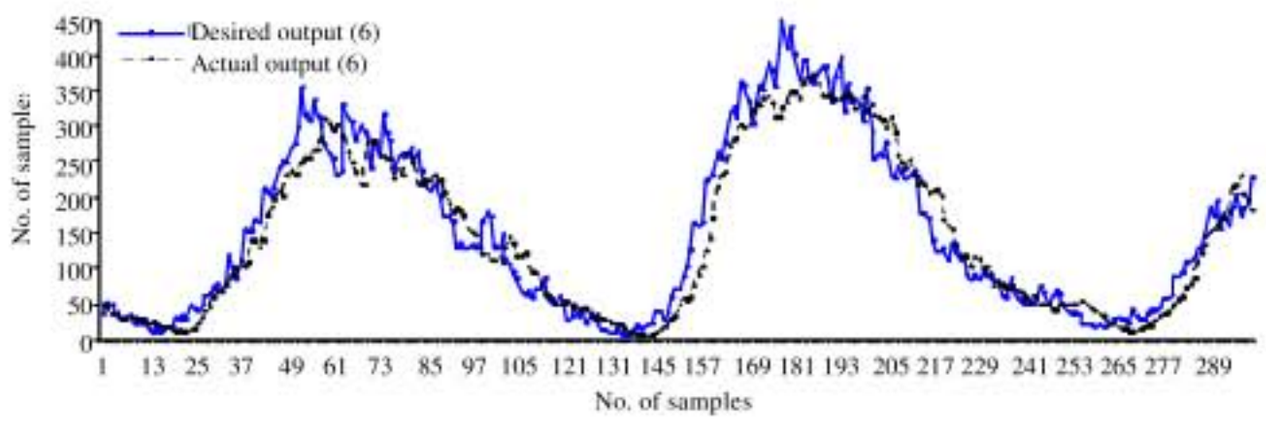


Fig. 7: Plot between Desired Output Vs Actual Network Output for 6 month ahead prediction for FTLRNN Model

It is observed that the performance of the selected model is optimal for 15 neurons in the hidden layer with regard to the MSE, NMSE and the correlation coefficient r for the testing data sets. When we attempted to increase the number of hidden layer and the number of processing element in the hidden layer, the performance of the model is not to seen to improve significantly. On the contrary it takes too long for training because of complexity of the model. As there is single input and single output for the given system, the number of input and output processing elements is chosen as one. Now the NN Model (1:15:1) is trained three times with different weight initialization with 1000 iterations of the static back propagation algorithm with momentum term for all the three models for the optimum parameter resulted in experimentation for all the 1,6,12,18 and 24 months ahead predictions as shown in Table 2.

From the Table 2 it is observed that FTLRNN model is able to predict the monthly sunspot time series elegantly well as compared to Multilayer perceptron (MLP) and Self Organizing Feature Map (SOFM) on testing data set with regards to MSE, NMSE and correlation coefficient (r) not only for 1, 6, 12 months ahead prediction but also for 18 and 24 months ahead prediction. Also the graphs are plotted for desired output and actual network output for 1 month ahead prediction shown in Fig. 6, for 6 months ahead prediction shown in Fig. 7, 12 months ahead prediction shown in Fig. 8, 18 months

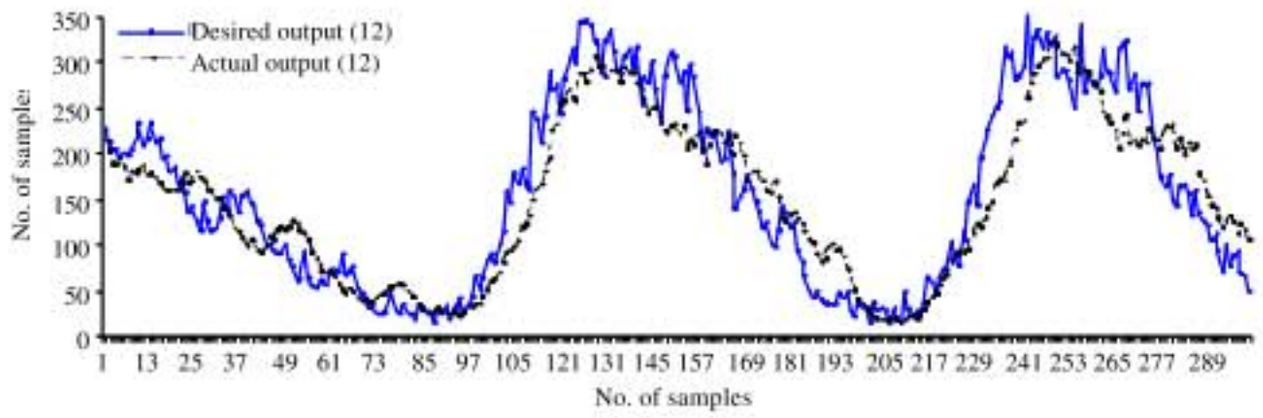


Fig. 8: Plot between Desired Output Vs Actual Network Output for 12 month ahead prediction for FTLRNN Model

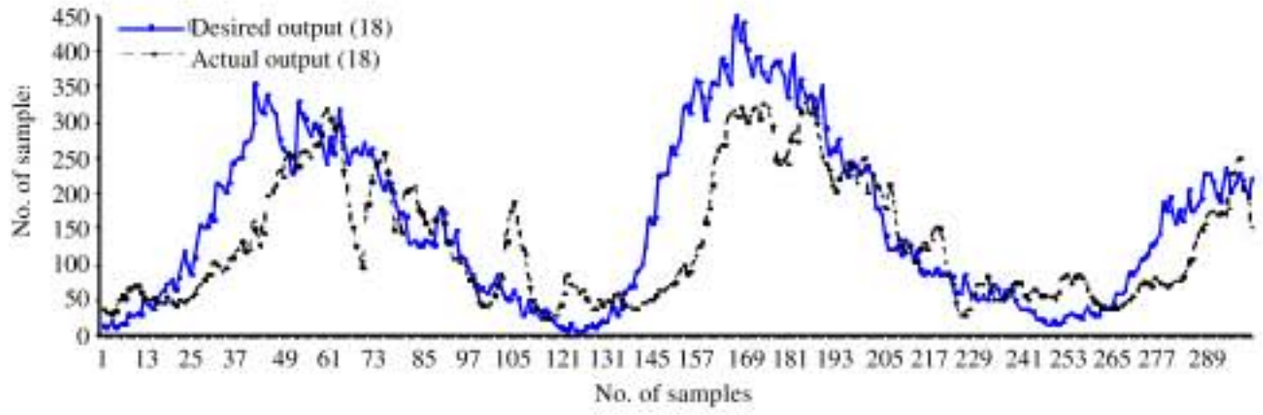


Fig. 9: Plot between Desired Output Vs Actual Network Output for 18 month ahead prediction for FTLRNN Model

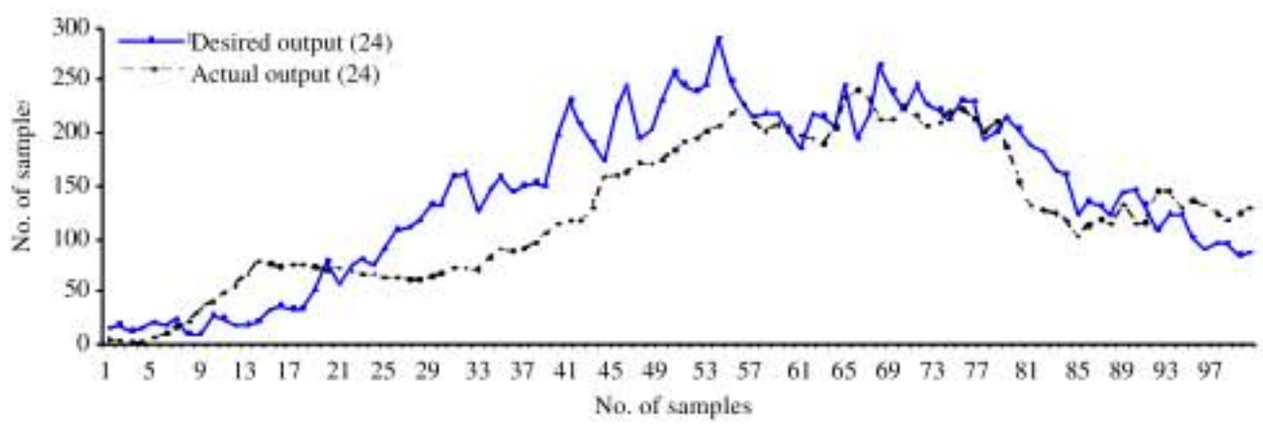


Fig. 10: Plot between Desired Output Vs Actual Network Output for 24 month ahead prediction for FTLRNN Model

ahead prediction shown in Fig. 9 and 24 months ahead prediction shown in Fig. 10 for the FTLRNN model. It is observed from the plots of Fig. 6, 7 and 8 for short term ahead prediction for 1, 6 and 12 months, respectively, that the output of the proposed FTLRNN model closely follows the desired output where as for long term ahead prediction from Fig. 9 for 18 months ahead prediction and Fig. 10 for 24 months ahead prediction the output of the FTLRNN model are slightly deviating.

Similarly for MLP and SOFM model the graphs are plotted for desired and actual output for short term 1, 6 and 12 months ahead prediction as shown in Fig. 11-13, respectively and for long term 18 and 24 months ahead prediction as shown in Fig. 14 and 15. It is inspected from the all the plots that the output of these MLP and SOFM models are not following the desired output well. These neural network models fails to learn the dynamics of the monthly sunspots chaotic time series as compared to the FTLRNN model.

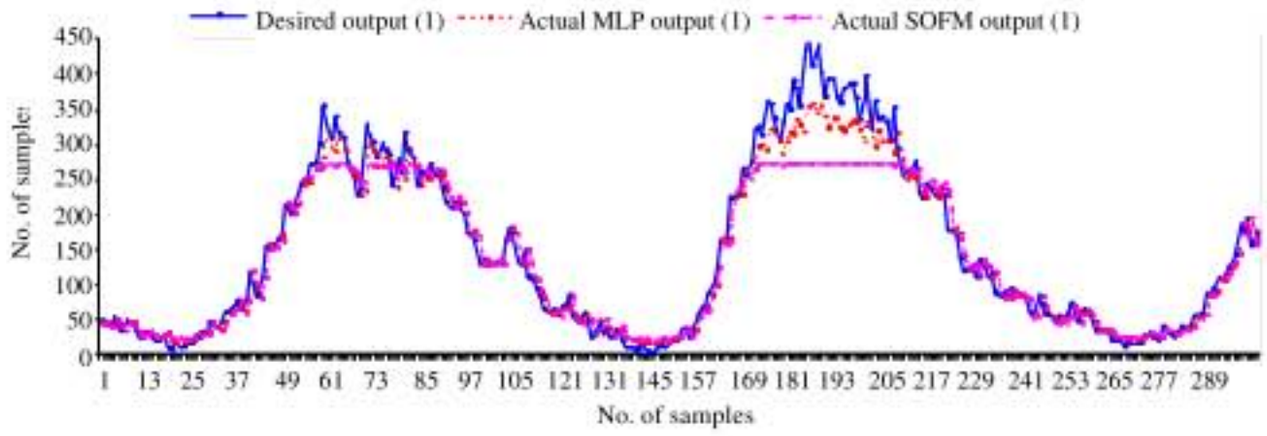


Fig. 11: Plot between desired output Vs Actual output for 1 month ahead prediction for MLP and SOFM model

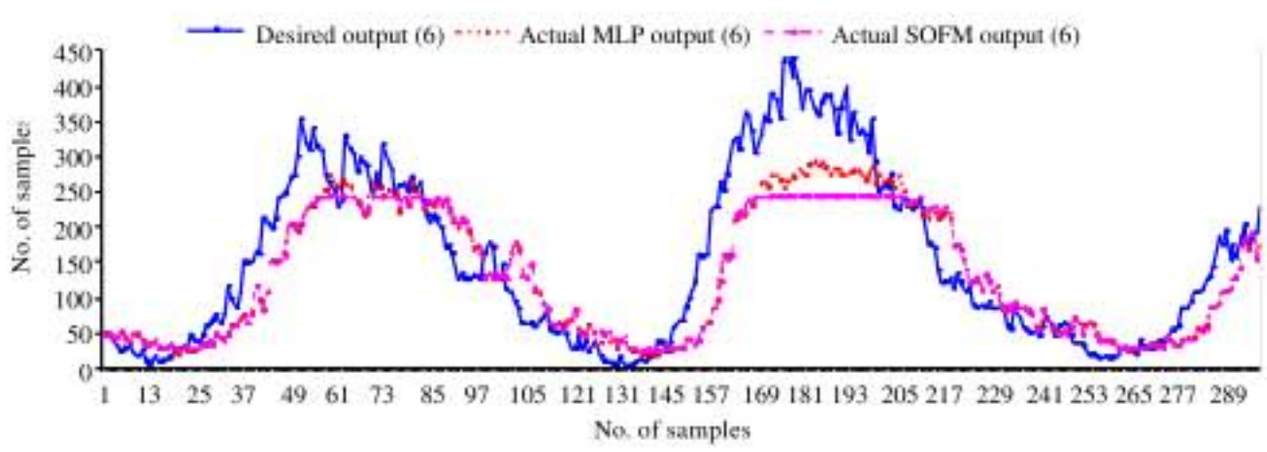


Fig. 12: Plot between desired output Vs Actual output for 6 month ahead prediction for MLP and SOFM model

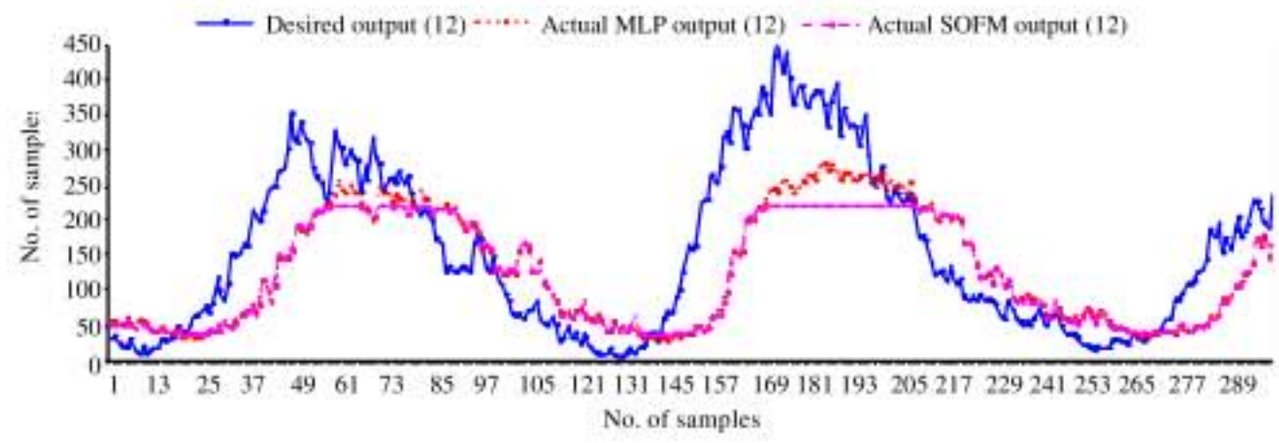


Fig. 13: Plot between desired output Vs Actual output for 12 month ahead prediction for MLP and SOFM model

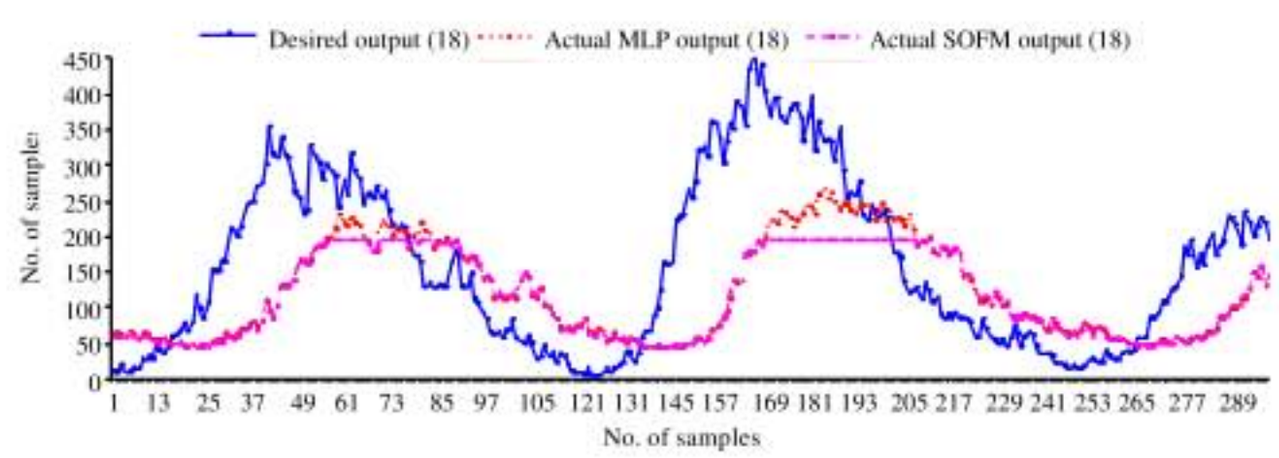


Fig. 14: Plot between desired output Vs Actual output for 18 month ahead prediction for MLP and SOFM model

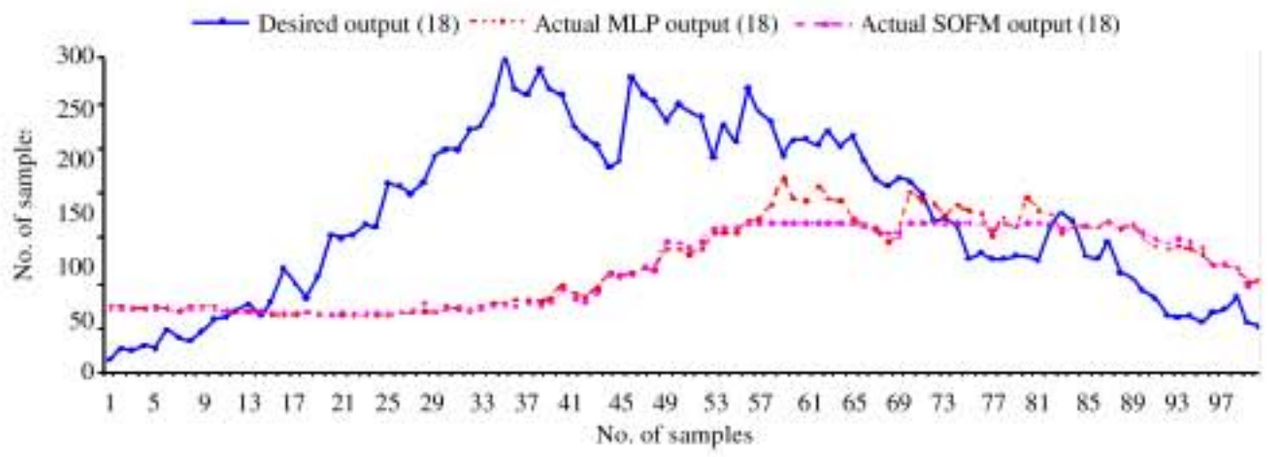


Fig. 15: Plot between desired output Vs Actual output for 24 month ahead prediction for MLP and SOFM model

## CONCLUSION

It is seen that focused time lagged recurrent network with gamma memory is able to predict the sunspots time series quite well in comparison with the Multilayer preceptron (MLP) and self organizing feature map (SOFM). Static NN configuration such as MLP NN based model and self organizing feature map (SOFM) network are failed to cope up with the underlying nonlinear dynamics of the sunspots time series. It is observed from the table 2 that MSE, NMSE of the proposed focused time lagged recurrent neural network (FTLRNN) dynamic model for testing data set as well as for training data set are significant better than those of static MLP NN and SOFM model. In addition it is also observed that the correlation coefficient of this model for testing and training exemplars are much higher than MLP and self organizing feature map neural network for all the month (k=1,6,12,18,24) ahead prediction. From the plots in the Fig the proposed FTLRNN model the network output closely follows the desired output for the 1, 6 and 12 months ahead prediction whereas for 18 and 24 months ahead prediction the proposed FTLRNN model output is slightly deviating from the desired output as shown in Fig. 9 and 10. It is resulted from the experiments that the FTLRNN model learns the dynamics of monthly sunspot time series quite well as compared to Multilayer perceptron (MLP) and self organizing feature map(SOFM). On the contrary, it is observed that static MLP NN and Self Organizing Feature Map (SOFM) performs poorly bad, because on the one hand it yields much higher MSE and NMSE on testing data sets and on the other hand the correlation coefficient for testing data set is far less than unity. This is also confirmed from the desired output Vs actual output plots for all the months ahead prediction for MLP and SOFM model as shown in Fig. 11-15 for the 1, 6, 12, 18 and 24 months ahead prediction respectively. Hence the focused time lagged recurrent neural network with gamma memory filter has out- performed the static MLP based neural network and SOFM better for 1, 6, 12, 18 and 24 months ahead prediction.

## REFERENCES

Bakker, R., J.C. Schouten, C.L. Giles and C.M.V. Bleek, 2000. Learning chaotic attractors by neural network. Neural Comput., 12: 2355-2383.

Barron, A.R., 1993. Universal approximation bounds for super positions of a sigmoidal function. IEEE Trans. Inform. Theory, 39: 930-945.

Bert de, V. and J.C. Principe., 1992. The gamma model. A neural model for temporal processing. Neural Networks, 5: 565-576.

Billings, S.A. and H.L. Wei, 2005. A new class of wavelet network for nonlinear system identification. IEEE Trans. Neural Networks, 16: 862-874.

Carpinteiro, O.A.S., A.J.R. Reis and A.P.A. Silva, 2004. A hierarchical neural model in short-term load forecasting. Applied Soft Comput., 4: 405-412.

Casdagli, M., 1989. Nonlinear prediction of chaotic time series. Physica. D, 35: 335-356.

Cooper, P.G., M.N. Hays and J.E. Whalen, 1992. Neural networks for propagation modeling. Electromagnetic Environmental Test facility, Fort Huachuca, AZ, Atlantic Research Corp Repot, 92-12-002.

Cybenko, G., 1989. Approximations by superposition of a sigmoidal functions. J. Math. Control Signals Syst., 2: 303-314.

Demuth and M. Beale, 2004. Neural Network Tool Box for use with MATLAB Manual Users Guide. Version 4.0, The Math Works Inc., Natick, MA.

Dudul, S.V., 2005. Prediction of a Lorenz chaotic attractor using two layer perceptron network. Elsevier J. Applied Soft Comput., 5: 333-335.

Dudul, S.V., 2007. Identification of a liquid of a liquid saturated steam heat exchanger using focused time lagged recurrent neural network model. IETE J. Res., 53: 69-82.

Franklin, G.F., J.D. Powell and M.L. Workman, 1998. Digital Control of Dynamics Systems. 3rd Edn., Addison Wesley, Reading, MA.

Fredric, M.H. and I. Kostanic, 2002. Principles of Neuro Computing for Science and Engineering. 2nd Edn., Tata McGraw-Hill Publishing Co. Ltd., India, ISBN: 0-07-048663-8.

Haykin, S., 2003. Neural Networks: A Comprehensive Foundation. 2nd Edn., Pearson Education, India, ISBN: 81-7808-300-0, pp: 443.

Hong, G. and H. Wang, 2007. Fuzzy prediction of chaotic time series based on singular value decomposition. Elsevier J. Applied Math. Comput., 185: 1171-1185.

Hornik, K., M. Stinchcombe and H. White, 1989. Multilayer feed forward networks are universal approximators. Neural Networks, 2: 359-366.

Huang, G., Y.Q. Chen and H.A. Babri, 2000. Classification ability of single hidden layer feed forward neural networks. J. IEEE Trans. Neural Networks, 11: 799-801.

Huebner, U., N. Abraham and C. Weiss, 1989. Dimensions and Entropies of chaotic intensity pulsations in a single-mode far infrared NH3 Laser. Phys. Rev. A, 40: 6354-6360.

Judi, S.A., H. Hjalmarson, A. Benefits, B. Delyon, L. Ljung, J. Sjoberg and Q. Zang, 1995. Nonlinear black-box models in system identification: Mathematical foundations. Automatica, 31: 1725-1750.

Lee, K.W. and H.N. Lam, 1995. Optimal sizing of feed forward neural networks: Case studies. Proceedings of the 2nd New Zealand Two Stream International Conference on Artificial Neural Networks and Expert Systems, Nov. 20-23, IEEE Xplore London, 77-82.

Leung, H. and T. Lo, 1993. Chaotic radar signal processing over the sea. IEEE J. Oceanic Eng., 18: 287-285.

Min, H., J. Xi, S. Xu and F.L. Yin, 2004. Prediction of chaotic time series based on the recurrent predictor neural network. IEEE Trans. Signal Process., 52: 3409-3415.

Naraendra, K.S. and K. Parthasarathy, 1990. Identification and control of dynamic systems using neural networks. IEEE Trans. J. Neural. Networks, 1: 4-27.

Principe, J.C., N.R. Euliano and C. Lefebvre, 2000. Neural and Adaptive Systems-Fundamental Through Simulations. 1st Edn., John Wiley and Sons, New York.

Quin, S.Z., H.T. Su and T.J. McAvoy, 1992. Comparison of four neural net learning methods for dynamic system identification. IEEE Trans. Neural Networks, 3: 122-130.

Thissen, U., R.V. Brakel, D.A.P. Weijer, W.J. Melssen and L.M.C. Buydens, 2003. Using support vector machines for time series prediction. Elsevier J. Chemomet. Intell. Lab. Syst., 69: 35-49.

Townshend, B., 1994. Nonlinear Prediction of Speech Signals. In: Nonlinear Modeling and Forecasting, Casdagli and S. Eubank (Eds.). Adison-Wesley, Reading, MA.

Turchenko, I.V., 2004. Simulation modeling of multi-parameter sensor signal identification using neural networks. Proceedings of the 2nd International Conference on Intelligent Systems, June 22-24, Sofia, Bulgaria, 48-53.

Weigend, A.S. and N.A. Greshenfeld, 1994. Time series prediction: Forecasting the future and understanding the past Santa Fe Institute Studies in Science of Complexity, Proceedings Vol. XV, Addison-Wesley, Reading. http://adsabs.harvard.edu/abs/1994tspf.conf.....W.

Xu, K., M. Xie, L.C. Tang and S.L. Ho, 2003. Applications of neural networks in forecasting engine systems reliability. Applied Soft Comput., 2: 255-268.

Zeng, Z. and D. Yurong, 1999. Chaotic time series analysis based on radial basis function network. Chin. J. Chongq. Univ., 22: 113-120.