ISSN No. 0976-5697

Volume 1, No. 4, Nov-Dec 2010

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

# An Efficient Weather Forecasting System using a Hybrid Neural Network SOFM–MLP

Dr. S. Santhosh Baboo
Reader, PG and Research department of Computer Science,
Dwaraka Doss Goverdhan Doss Vaishnav College
Chennai, India
santhos2001@sify.com

I.Kadar Shereef*
Research Scholar, Dravidian University
Head, Department of Computer Applications
Sree Saraswathi Thyagaraja College,Pollachi.
Coimbatore, Tamil nadu, India
kadarshereef@gmail.com

***Abstract:*** Weather prediction is a challenging task for researchers and has drawn a lot of research interest in the recent years. Literature studies have shown that machine learning techniques achieved better performance than traditional statistical methods. Presently multilayer perceptron networks (MLPs) are used for prediction of the maximum and the minimum temperatures based on past observations on various atmospheric parameters. To capture the seasonality of atmospheric data, with a view to improving the prediction accuracy, a novel weather forecasting system is presented in this paper. The proposed system is based on a neural architecture that combines a selforganizing feature map (SOFM) and MLPs to realize a hybrid network named SOFM–MLP. It is also demonstrated that the use of appropriate features such as temperature gradient can not only reduce the number of features drastically, but also can improve the prediction accuracy. These observations motivated us to use a feature selection MLP (FSMLP) instead of MLP, which can select good features online while learning the prediction task. FSMLP is used as a preprocessor to select good features. The combined use of FSMLP and SOFM–MLP provides better result in a network system that uses only very few inputs but can produce good prediction. The proposed system is experimented using the real time data observations and from which it is found that the proposed system predict the temperature with minimum error.

***Keywords:*** Atmospheric science, back propagation, feature selection, neural networks, self-organizing feature map (SOFM), temperature forecasting.

## I. INTRODUCTION

Weather prediction is a complex process and a challenging task for researchers. It includes expertise in multiple disciplines [1], [2]. The prediction of atmospheric parameters is essential for various applications. Some of them include climate monitoring, drought detection, severe weather prediction, agriculture and production, planning in energy industry, aviation industry, communication, pollution dispersal [3] etc. Accurate prediction of weather parameters is a difficult task due to the dynamic nature of atmosphere. Various techniques like linear regression, auto regression, Multi Layer Perceptron, Radial Basis Function networks are applied to predict atmospheric parameters like temperature, wind speed, rainfall, meteorological pollution etc. Often, it is very much difficult to get an accurate prediction result because of many other factors like topography of a place, surrounding structures, and environmental pollution. The accuracy of a forecasting system may be improved if it can account for all these factors.

Prediction of temperature based on the past measurements of various atmospheric parameters is focused here, and one assumption is made, that short-term changes in the dynamics will be captured in the data available for forecasting. In general, temperatures are measured twice a day at different heights and places using radiosonde (i.e., balloon floating) techniques. Other parameters, such as wind direction and its velocity, are also measured by the meteorologists. Due to chaotic nature [5] of the atmosphere, the massive computational power is required to solve the equations that describe the atmosphere, error involved in measuring the initial conditions, and an incomplete understanding of atmospheric processes. This means that forecasts become less accurate as the difference in current time and the time for which the forecast is being made (the range of the forecast) increases. The use of ensembles and model helps narrow the error and pick the most likely outcome.

Several steps to predict the temperature are

a. Data collection(atmospheric pressure, temperature, wind speed and direction, humidity, precipitation),
b. Data assimilation and analysis,
c. Numerical weather prediction,
d. Model output post processing.

A neural network [4] is a powerful data modeling tool that is able to capture and represent complex input /output relationships. The development motivation of neural network technology stemmed from the desire to implement an artificial system that could perform intelligent tasks similar to those performed by the human brain. Neural network resemble the human brain in the following two ways:

a. A neural network acquires knowledge through learning.
b. A neural network's knowledge is stored within interneuron connection strengths known as synaptic weights

The true power and advantages of neural networks lies in the ability to represent both linear and non linear relationships directly from the data being modeled. Traditional linear models are simply inadequate when it comes for true modeling data that contains non linear characteristics. A neural network model can be adjusted to produce a mapping from a given set of data to features of or relationships among the data. The model is adjusted, or trained, using a collection of data from a given source as input, typically referred to as the training set. After successful training, the neural network will be capable to perform classification, estimation, prediction, or simulation on new data from the same or similar sources.

In this paper, the proposed hybrid network is described primarily, which combines a self-organizing feature map (SOFM) and a multilayer perceptron network (MLP) to realize a much better prediction system. Then, it is demonstrated that the use of appropriate features can not only reduce the number of features, but also can improve the prediction accuracy. Motivated by the results with computed features, a feature selection MLP is used, which can select good features online while learning the prediction task. This, finally, results in a network system that can give good prediction result using very few inputs.

The rest of the paper is organized as follows. The related works on literature for the weather prediction is presented in section II. The SOFM Network and SOFM–MLP Hybrid Network are given in Section III. The experimental results obtained on dataset from a sample dataset from Weather Underground [6] are reported in Sections IV. Finally, conclusions are drawn in Section V.

## II. Related Works

In the literature, several methods have been proposed for the automatic prediction of temperature. Among the most recently published works are those presented as follows

Y.Radhika and M.Shashi [7] presents an application of Support Vector Machines (SVMs) for weather prediction. Time series data of daily maximum temperature at location is studied to predict the maximum temperature of the next day at that location based on the daily maximum temperatures for a span of previous n days referred to as order of the input. Performance of the system is observed for various spans of 2 to 10 days by using optimal values of the kernel.

Mohsen Hayati *et.al,* [8] studied about Artificial Neural Network based on MLP was trained and tested using ten years (1996-2006) meteorological data. The results illustrate that MLP network has the minimum forecasting error and can be considered as a good method to model the short-term temperature forecasting [STTF] systems. Brian A. Smith *et.al,*[16] focused on developing ANN models with reduced average prediction error by increasing the number of distinct observations used in training, adding additional input terms that describe the date of an observation, increasing the duration of prior weather data included in each observation, and reexamining the number of hidden nodes used in the network. Models were created to forecast air temperature at hourly intervals from one to 12 hours ahead. Each ANN model, having a network architecture and set of associated parameters, was evaluated by instantiating and training 30 networks and calculating the mean absolute error (MAE) of the resulting networks for some set of input patterns.

Arvind Sharma *et.al,* [9] presented the details of how the different connectionist paradigms could be formulated using different learning methods and then investigates whether they can provide the required level of performance, which are sufficiently good and robust so as to provide a reliable forecast model for stock market indices. Experiment results exposes that all the connectionist paradigms considered could represent the stock indices behavior very accurately.

Mike O'Neill [10] focus on two major practical considerations: the relationship between the amounts of training data and error rate (corresponding to the effort to collect training data to build a model with given maximum error rate) and the transferability of models' expertise between different datasets (corresponding to the usefulness for general handwritten digit recognition).Henry A. Rowley eliminates the difficult task of manually selecting nonface training examples, which must be chosen to span the entire space of nonface images. Simple heuristics, like using the fact that faces rarely overlap in images, can further improve the accuracy. Comparisons with more than a few other state-of-the-art face detection systems are presented; showing that our system has comparable performance in terms of detection and false-positive rates.

## III. Methodology

### A. *SOFM Network*

In numerous applications such as in [11]–[14] the Kohonen's self-organizing feature map has been successfully used. SOFM [15] possesses the interesting property of achieving a distribution of weight vectors that approximates the distribution of the input data. This property of the SOFM can be used to generate prototypes which in turn can partition the data into homogeneous groups. This property is used in the proposed methodology.

#### *a) Architecture*

The SOFM is an algorithmic transformation $A^D_{SOFM}: R^p \rightarrow V(R^q)$ that is frequently advocated for visualization of metric-topological relationships and distributional density properties of feature vectors (signals) $X = \{x_1, x_2, \dots, x_N\}$ in$R^p$.

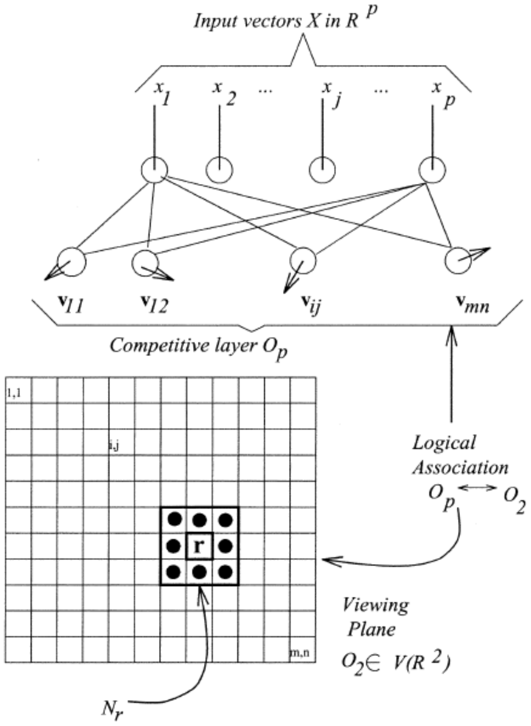

Figure 1. SOFM network architecture

SOFM is deployed through a neural architecture as shown in Fig. 1, and it is believed to be analogous in some ways to the biological neural network. The visual display generated by

SOFM can be used to form hypotheses about topological structure present in $X$. It is concentrated on ($m \times n$) displays in $R^2$, but in principle $X$ can be transformed onto a display lattice in $R^q$ for any q.

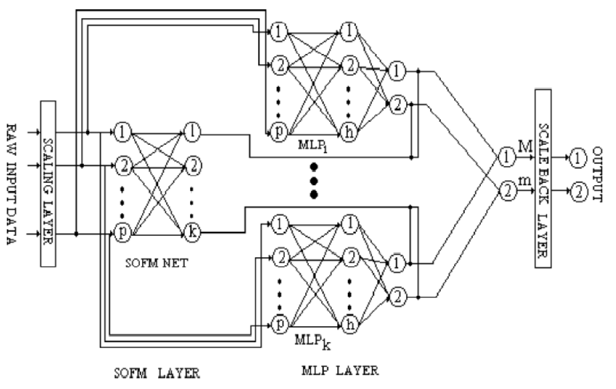

Figure 2. Hybrid neural net for temperature forecasting

Input vectors $x \in R^p$ are distributed by a fan-out layer to each of the ($m \times n$) output nodes in the competitive layer as shown in Fig. 1. Each node in this layer has a weight vector (prototype) $v_{ij}$ attached to it. Let $O_p = \{v_{ij}\} \subset R^p$ denote the set of weight vectors. $O_p$ is (logically) connected to a display grid $O_2 \subset V(R^2)$. ({$i,j$} ) in the index set (1,2,…,m}x(1,2,…,n} is the logical address of a cell. There is a one-to-one correspondence between the $m \times np$-vectors $v_{ij}$ and the $mxn$ cells ({i,j} ), i.e., $O_p \leftrightarrow O_2$.

With a random initialization of the weight vectors $v_{ij}$ the feature mapping algorithm starts. For notational clarity, the double subscripts are suppressed. Now, let x ∈ $R^p$ enter the network, and let denote the current iteration number. Find $v_{r,s-1}$that best matches x in the sense of minimum Euclidean distance in $R^p$. This vector comprises a (logical) "image" that is the cell in $O_2$with subscript r. Next, a topological (spatial) neighborhood $N_r(s)$ centered at is defined in$O_2$, and its display cell neighbors are located. A 3×3 window, N(r), centered at r corresponds to the updating of nine prototypes in $R^p$. Finally, $v_{r,s-1}$ and other weight vectors associated with cells in the spatial neighborhood $N_s(r)$ are updated using the rule as given below

$$v_{i,s} = v_{i,s-1} + H_{ri}(s)(x - v_{i,s-1}) \qquad (1)$$

Here, $r$ is the index of the "winner" prototype and ||*|| is the Euclidean norm on $R^p$.

$$r = \underbrace{\arg min}_{i}\{\left|\left|x - v_{i,s-1}\right|\right|\} \qquad (2)$$

The strength of interaction between cells r and i in $O_2$ usually decreases with s, and for a fixed s it decreases as the distance (in $O_2$) from cell to cell i increases which is expressed by the function $H_{ri}(s)$. $H_{ri}(s)$ is usually expressed as the product of a learning parameter and a lateral feedback function $g_s$(dist(r,i)) . A common choice for $g_s$ is $g_s$(dist(r,i))=$e^{-dist^2(r,i)/\sigma_s^2}$. $\alpha_s$ and $\sigma_s$ both decrease with s. $N_r(s)$ is the topological neighborhood which also decreases with s. This method, when repeated long enough, usually preserves spatial order in the sense that weight vectors which are metrically close in $R^p$ have visually close images in the viewing plane. The SOFM is repeated for (500×m×n) steps.

### *B. SOFM–MLP hybrid network*

The hybrid network's architecture is clearly shown in Figure 2 which has eight layers. The first layer with $p$ nodes scales the data: it is the scaling interface between user and the system at the input side. The second and third layers constitute the SOFM layer. The output of the scaling layer is fed as the input to the SOFM layer. So, the second layer has p nodes. As discussed earlier for the SOFM net, there are complete connections between layers 2 and 3.

The output layer of the SOFM net let possesses K number of nodes. So, there are K MLP networks, each of which receives inputs. Consequently, the fourth layer has $K_p$ nodes. These $K_p$ nodes constitute the input layer of a set of K MLP networks. Without any loss of generality, it is assumed that each of the K MLP networks has only one hidden layer, although it could be more than one and it can vary for different MLP nets. Let the nodes in layer four be numbered as $N_i$ , i=1,2,…,$K_p$. Nodes $N_1$ to $N_p$ will be the input nodes of the first MLP ($M_2$); nodes $N_{p+1}$ to will be input nodes of the second MLP ($M_2$); Similarly, nodes $N_{(K-1)p+1}$ to $N_{np}$ will be the input nodes of Kth MLP, $M_K$. p=9k as mentioned earlier.The jth input node of MLP $M_i$ gets the jth normalized input (say, $x_j$ ) and passes it on to the first hidden layer of $M_i$.

The output of the th node of the SOFM (say, $O_i$) is connected to the output of every node of the last layer of $M_i$. The product of the MLP output and the SOFM output then moves to layer 7. The product can be computed using an additional layer with two neurons for each MLP. Since only one of the SOFM outputs will be one, and the rest will be zero, only one of the MLPs will pass its output unattenuated to layer 7. The remaining (k-1) MLPs will transfer zero to layer 7.Since it is assumed that only one hidden layer, the nodes in layer six are the output nodes of the MLP nets. Each MLP, $M_i$ will have two output nodes. Let us denote these nodes by $O_{ij}^6$ where the index corresponds to the th MLP, $M_i$ and j=1,2, where 1 corresponds to the minimum temperature and 2 corresponds to the maximum temperature. Layers 4–6 together constitute the MLP layer in Fig. 2.

The outputs of this MLP-layer are then aggregated in layer seven which has just two nodes one for the minimum temperature and the other for the maximum temperature. Let us denote these two nodes by and m and M. Now nodes $O_{i1}^6$, ∀ i=1,2,…K are connected to node m and $O_{i2}^6$, ∀ i=1,2,…K are connected to node M. All connection weights between layers 6 and 7 are set to unity and nodes m and M compute the weighted sum of all inputs as the output which is then passed to the scaling layer. Note that the network architecture ensures that the aggregated output that is fed to the scaling layer is nothing but the output of the MLP corresponding to the winning node of the SOFM net.

It must be analyzed why not a clustering algorithm used instead of SOM. The prototypes generated by SOFM not only preserve topology but also density. This density preservation property must be exploited. Because of the density matching property of SOFM, if a particular region of the input space contains frequently occurring stimuli, it will be represented by a larger area in the feature map than a region of the input space where the stimuli occur less frequently. Consequently, if there is a dense area in the input space SOFM will place more prototypes there. So, there will be more competitive MLPs for dense regions. Consequently finer details of the process can be modeled better and this will result in an enhancement of the overall performance. Density matching property is not available in K-means type of clustering algorithms.

*a) Training the SOFM–MLP Hybrid Net*

Input normalization (i.e., scaling) layer is used to normalize $X_{tr}$ firstly. Then the SOFM net is trained with the normalized $X_{tr}$. Once the SOFM training is over, $X_{tr}$ is partitioned into K subsets, $X_{tr}^{(l)}$, l=1,2,…,K as follows:

$$X_{tr}^{(l)} = \{x_i \in R^p \,|\, ||x_i - v_l|| = \min_j ||x_i - v_j||\} \qquad (3)$$

$X_{tr}^{(l)}$ Can also be said as the set of input vectors for which the *l*th prototype, of the SOFM becomes the winner. Let$Y_{tr}^{(l)}$ be the set of output vectors associated with vectors in$X_{tr}^{(l)}$. Now multilayer perceptron nets $M_1$, $M_2$,…, $M_K$, where is trained with ($X_{tr}^{(l)}$,$Y_{tr}^{(l)}$). Note that each of $M_l$, l=1,2,…,K will have the same number of nodes in the input layer, i.e.,p=9k and the same number of nodes in the output layer, i.e., 2. But the number of nodes in the hidden layer for different $M_l$ could be different. This training is done offline and during training; the output of the SOFM is not considered. In fact, the input to SOFM is not feed for training the MLP. Once the training of both SOFM and K MLPs is over, the hybrid net for prediction of temperatures must be used.

x(t) is applied to the first layer suppose an input vector $x(t) \in R^{9k}$ comes (this will be generated based on nine observations on each of the past k days). The first layer normalizes it, and the normalized input then goes to the SOFM layer. x(t) makes the output of only one of the K SOFM output nodes (say, of the l th node) high (1) and sets the rest ( K-1 outputs) to zero. The normalized x(t) and output of the i th SOFM node are now fed to the th MLP $M_i$, i=1,2,…,K . Consequently, only the *l* th MLP will be active, and rest of the MLPs will be inactive. The integrated output from the MLP layer will be nothing but the output of the *l* th MLP, which will then be scaled back to the original scale by the output scaling layer—and the prediction for the maximum and the minimum temperatures of day t+1 are obtained.

## C. Prediction with computed features

In all the work explained previously in this paper, the entire information available for the past three days is used to predict the temperatures for the next day. As a result, the number of input features becomes 27, which made the learning task a difficult one. The use of suitable features is a key factor toward determining the success of the learning process. In this case, also, if some derived features can be used that are better suited for the task at hand, then better prediction is expected to get. With a view to achieving this, used local gradients of the temperature sequences are used as features. The local gradient is computed as follows. Suppose, $T^{max}(t-4)$, $T^{max}(t-3)$, $T^{max}(t-2)$ and $T^{max}(t-1)$ are the maximum temperatures recorded for the last four days. Then, the temperature gradients or changes in temperature are $T^{max}(t-4)$- $T^{max}(t-3)$, $T^{max}(t-3)$-$T^{max}(t-2)$, and $T^{max}(t-2)$-$T^{max}(t-1)$. Similarly, three such components can be computed for the minimum temperature. Here, 15 features are used which comprises of nine features giving the atmospheric conditions of today (day ) and six temperature gradients as discussed above. The advantage with this scheme is that: 1) it reduces the number of input features, and 2) it gives an idea to the MLP network about the changes in the maximum and the minimum temperatures. This can make the learning task simpler.

## D. Online feature selection and hybrid network

This is the final part of the proposed system. In the previous part two things were observed: the hybrid network works better than MLP, and the choice of good features improves the prediction accuracy. Therefore, if online feature selection can be done, i.e., select the good features while learning the prediction task, the performance of the network can be probably further improved, and this can also tell us about various important features responsible for temperature variations. This may help us to get a better insight into the temperature variation process. This process is given as follows

*a) Online Feature Selection Net*

Several attempts have been made previously to use neural networks for feature selection. These methods are offline in nature. In this paper, an online feature selection method is used due to Pal and Chintalapudi [17]. The acronym FSMLP (Feature Selection MLP) is used for the Pal–Chintalapudi modification of the standard MLP that can select features. In a standard MLP, the effect of some features (inputs) can be eliminated by not allowing them into the network. If "partially useful" features can be identified, then they can be attenuated according to their relative usefulness. This can be realized by associating an adaptive gate to each input node. The gate should be modeled in such a manner that for a good feature, it is completely opened, and the feature is passed unattenuated into the net; while for a bad feature, the gate should be closed tightly. Conversely, for a partially important feature, the gate could be partially opened. Mathematically, the gate is modeled by a function with a tunable parameter. The degree to which the gate is opened finds the goodness of the feature. An input feature value is multiplied by its gate function value, and the modulated feature value is then passed into the network. The gate functions attenuate the features before they propagate through the net, so these gate functions can be called as attenuator functions. The usage of s-type (or sigmoidal) functions with a tunable parameter is a simple way of finding useful gate functions, which can be learned using the training data.

An attenuation (gate) function associated with an input node let be $F:R \rightarrow [0,1]$. If x is the node input, then $xF(\gamma)$ is the node output. Thus, $xF(\gamma_i)$ can be viewed as the activation function of the *i* th input node, where $\gamma_i$ is a parameter (not a connection weight) of the activation function. Thus, the input layer nodes act as "neurons" (i.e., have internal calculations). Notice that once $\gamma_i$ is known, $F(\gamma_i)$ acts as a fixed multiplier for all input values of the *i* th feature. The function F can have various forms. In the experiments described below, the attenuation function $F(\gamma)=1/(1+e^{-\gamma})$ is used. Thus, the i th input node attenuates $x_i$ by an amount $F(\gamma_i)\in(0,1)$, where $\gamma_i$ is a parameter to be learned during training. If $F(\gamma_i)$ is close to zero, it may chosen to eliminate input feature $x_i$: this is how the FSMLP accomplishes feature selection. The backpropagation formulas for the MLP can simply be extended backward into this modified input layer to adjust the $\gamma_i$s during training.

Number of nodes in the first hidden (not input) layer let be $n_h$; Learning rate for the parameters of the attenuator membership functions be $\mu$; Learning rate for the connection weights be $\eta$; $F:R\rightarrow(0,1)$=attenuator function with argument $\gamma_i$ for input node i; $F'(\gamma_i)$ derivative of F at $\gamma_i$; $w_{ji}^{ih}(t)$=weight connecting th node of the input layer to the th node of the first hidden layer for the *t* th iteration; and $\delta_j^1$=error term for the *j* th node of the first hidden layer.

The learning rule for connection weights can be easily shown that it remains the same for all layers except for $w_{ji}^{ih}(t)$. The update rules for $w_{ji}^{ih}(t)$ and $\gamma_i$ are

$$w_{ji}^{ih}(t) = w_{ji}^{ih}(t-1) - \eta x_i \delta_j^1 F(\gamma_i(t-1)) \qquad (4)$$

$$\gamma_i(t) = \gamma_i(t-1) + \mu x_i \Big(\sum_{j=1}^{n_h} w_{ji}^{ho}\delta_j^1\Big) F'(\gamma_i(t-1)) \qquad (5)$$

The $\gamma_i$, i=1,2,…,p are initialized with values that make $F(\gamma_i)$close to zero for all i. Accordingly, $x_iF(\gamma_i)$ is small at the beginning of training. So, a very small “fraction” of each input feature value is only allowed by FSMLP to pass into the standard part of the MLP. As the network trains, it selectively allows only the important features to be active by increasing their attenuator weights (and, hence, increasing the multipliers of $\{x_i\}$ associated with these weights) as dictated by the gradient descent. The training can be stopped when the mean squared error is low or when the number of iterations reaches a maximum limit. Features with low attenuator weights are then eliminated from the feature set. In this feature selection process, only those features are considered whose attenuation values less than 90% at the end of the training.

## IV. EXPERIMENTAL RESULTS

To experiment the proposed system a sample dataset is taken from Weather Underground [6]. This dataset contains the real time observation of the weather for a particular period of time. For this experiment, an observation of the complete previous year from January 2009 to December 2009 is taken. The dataset contains many attributes such as Temp. (°C), Dew Point (°C), Humidity (%), Sea Level Pressure (hPa), Visibility (km), Wind (km/h), Gust Speed (km/h) and Precip (cm). Table I gives the cumulative percentage frequencies for the MLP Networks.

Table I: Cumulative percentage frequency for MLP networks

| **Range in °C** | **% Frequency of Temperature for test data** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **$n_h$=10** | | **$n_h$=15** | | **$n_h$=20** | | **$n_h$=25** | |
| | **Max** | **Min** | **Max** | **Min** | **Max** | **Min** | **Max** | **Min** |
| ±0.5 | 31.9 | 30.4 | 32.1 | 31.1 | 31.9 | 31.1 | 30.9 | 29.9 |
| ±1.0 | 58.3 | 57.3 | 58.8 | 57.9 | 58.4 | 57.9 | 57.6 | 56.7 |
| ±1.5 | 73.1 | 73.4 | 73.8 | 74.3 | 73.4 | 73.1 | 72.6 | 73.1 |
| ±2.0 | 85.5 | 84.4 | 85.8 | 84.8 | 85.7 | 84.4 | 84.6 | 83.6 |
| ±2.5 | 90.3 | 90.2 | 90.8 | 91 | 90.4 | 89.6 | 89.6 | 89.8 |
| ±3.0 | 94.2 | 92.2 | 94.6 | 91.9 | 94 | 93.9 | 93.4 | 90.7 |
| Max. Dev. | 5.71 | 6.64 | 5.82 | 5.85 | 5.61 | 5.64 | 4.62 | 4.65 |
| Avg. Dev. | 1.2 | 1.2 | 1.1 | 1.1 | 1.1 | 1.2 | 1.1 | 1.2 |

Table II depicts the performance of the SOFM–MLP network on the test data when each of the K(=8) MLPs uses $n_h$=10, $n_h$=15, and $n_h$=20 nodes in the hidden layer. For the SOFM layer, eight nodes are used, and thereby the training data were partitioned into eight homogeneous subgroups. For this dataset, the choice of eight was made based on a few experiments. In this case, use of more than eight nodes results in some clusters with very few data points. Each MLP is trained ten times with different random initialization, and Table II represents the average prediction accuracy over these runs. Comparing Table II values with observed correct value, it is found that within ±1° C error, the SOFM–MLP shows an improvement between 2.7% to 7.8% over the direct use of MLP. This improvement is reduced to about 2.2% to 7.4% within ±2° C error. If the maximum deviation and the average deviation are considered, and also better results for SOFM–MLP are found consistently. Comparing SOFM–MLP (Table I) with another local predictor, the RBF network, It is found that all three architectures of SOFM–MLP significantly outperform the Radial Basis Function (RBF) net for all deviations less than equal to ±2° C.

As explained in the methodology the combination of the MLP and the SOFM-MLP on the test data produces the accurate results. Table III shows the performance of the MLP and SOFM–MLP on the test data with the new features.

Table II: Cumulative percentage frequency table for SOFM–MLP when observations from the past three days are used as input

| **Range in °C** | **% Frequency of Temperature for test data** | | | | | |
|---|---|---|---|---|---|---|
| | **$n_h$=10** | | **$n_h$=15** | | **$n_h$=20** | |
| | **Max** | **Min** | **Max** | **Min** | **Max** | **Min** |
| ±0.5 | 34.9 | 33.4 | 35.1 | 34.1 | 34.9 | 34.1 |
| ±1.0 | 62.3 | 61.3 | 62.8 | 61.9 | 62.4 | 61.9 |
| ±1.5 | 78.2 | 78.5 | 78.9 | 79.4 | 78.5 | 78.2 |
| ±2.0 | 90.6 | 89.5 | 90.9 | 89.9 | 90.8 | 89.5 |
| ±2.5 | 93.5 | 93.4 | 94.0 | 94.2 | 93.6 | 92.8 |
| ±3.0 | 97.4 | 95.4 | 97.8 | 95.1 | 97.2 | 97.1 |
| Max. Dev. | 5.71 | 6.64 | 5.82 | 5.85 | 5.61 | 5.64 |
| Avg. Dev. | 1.05 | 1.06 | 1.05 | 1.08 | 1.07 | 1.08 |

Comparing Table III with Table I, it is found that with a smaller architecture (small number of input units), the performance of the ordinary MLP is consistently better. For deviations less than or equal to ±2° C, the performance of the

MLP network with gradients as features is consistently better than the corresponding MLP using 27 features.

Table III: Cumulative percentage frequency table for MLP and SOFM–MLP using temperature gradients as features

| Range in °C | % Frequency of Temperature for test data | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | MLP | | | | SOFM-MLP | | | |
| | $n_h$=10 | | $n_h$=15 | | $n_h$=10 | | $n_h$=15 | |
| | Max | Min | Max | Min | Max | Min | Max | Min |
| ±0.5 | 35 | 33.5 | 35.2 | 34.2 | 39.9 | 38.4 | 40.1 | 39.1 |
| ±1.0 | 61.4 | 60.4 | 61.9 | 61 | 67.3 | 66.3 | 67.8 | 66.9 |
| ±1.5 | 76.2 | 76.5 | 76.9 | 77.4 | 83.2 | 83.5 | 83.9 | 84.4 |
| ±2.0 | 88.6 | 87.5 | 88.9 | 87.9 | 93.8 | 92.7 | 94.1 | 93.1 |
| ±2.5 | 93.4 | 93.3 | 93.9 | 94.1 | 96.7 | 96.6 | 97.2 | 97.4 |
| ±3.0 | 97.3 | 95.3 | 97.7 | 95 | 98.9 | 96.9 | 99.3 | 96.6 |
| Max. Dev. | 5.41 | 6.34 | 5.52 | 5.55 | 4.91 | 5.84 | 5.02 | 5.05 |
| Avg. Dev. | 1.04 | 1.02 | 1.05 | 1.06 | 1.04 | 1.07 | 0.99 | 1.03 |

This is clearly a significant improvement because with the new features a much smaller network is used. Comparing columns of SOFM–MLP in Table III with Table II, it is found that for deviations less than or equal to ±1.5°C, SOFM–MLP using gradient as features exhibits consistently better performance than the corresponding SOFM–MLP using all 27 features. For deviations less than or equal to ±2°C, the performance of SOFM–MLP more or less remains the same. Table III also reveals that the maximum deviation and average deviation are better for SOFM–MLP than direct use of MLP.

Instead of using all the features some features are selected and the use of selected features increases the accuracy. In order to select the good features, the FSMLP is trained using the entire dataset. And after the features are selected, the SOFM–MLP is trained with the selected set of features. Table IV displays the attenuation factors of the 15 features after training. Table IV reveals that only eight of the 15 features are important for prediction of the next day's temperature [$T^{max}(t)$ or $T^{min}(t)$]. The network rejects the maximum pressure, minimum pressure, and minimum relative humidity, but not the maximum relative humidity. It is very reasonable to expect that the maximum relative humidity can have influence on the temperature variation but not the minimum relative humidity. The network can capture this information. Similarly, of the six temperature gradients, the network picks up only the two most recent temperature gradients, i.e., the difference of the maximum temperatures of today and yesterday [$T^{max}(t)$- $T^{max}(t-1)$] and the difference of the minimum temperatures of today and yesterday[$T^{min}(t)$- $T^{min}(t-1)$]. This tells us that only very local (with respect to time) variation of temperature has effect on predicting the temperature. Of the accepted features, FSMLP has given the maximum importance to the minimum temperature of today [$T^{min}(t)$]; the next important feature is the maximum temperature of today $T^{max}(t)$. This is also very logical, as the maximum and minimum temperatures are predicted. The third most important feature selected by the network, as a meteorologist will expect, is the maximum vapor pressure of today. In previous experiments, validation set is not used to guard against overtraining or memorization by the network. In order to demonstrate the fact that our earlier results do not suffer from overtraining and memorization, and to see the effect of validation, a validation set is used now. 70 points is used for validation and 200 points for testing.

Table IV: FSMLP attenuation factors for network with 15 inputs

| No. | Attenuation (%) | Feature | Remark |
|---|---|---|---|
| 1 | 99.37 | Max. pressure | Reject |
| 2 | 99.28 | Min. pressure | Reject |
| 3 | 62.06 | Max. vapor pressure | Accept |
| 4 | 79.18 | Min. vapor pressure | Accept |
| 5 | 73.80 | Max. relative humidity | Accept |
| 6 | 98.79 | Min. relative humidity | Reject |
| 7 | 59.67 | $T^{max}(t)$ | Accept |
| 8 | 22.19 | $T^{min}(t)$ | Accept |
| 9 | 99.53 | $T^{max}(t-2)$-$T^{max}(t-3)$ | Reject |
| 10 | 99.54 | $T^{min}(t-2)$-$T^{min}(t-3)$ | Reject |
| 11 | 99.71 | $T^{max}(t-1)$-$T^{max}(t-2)$ | Reject |
| 12 | 99.62 | $T^{min}(t-1)$- $T^{min}(t-2)$ | Reject |
| 13 | 81.13 | $T^{max}(t)$- $T^{max}(t-1)$ | Accept |
| 14 | 70.02 | $T^{min}(t)$- $T^{min}(t-1)$ | Accept |
| 15 | 62.24 | Rainfall | Accept |

Table V: Cumulative percentage frequency table for MLP and SOFM–MLP using selected features

| Range in °C | % Frequency of Temperature for test data | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | MLP | | | | SOFM-MLP | | | |
| | $n_h$=10 | | $n_h$=12 | | $n_h$=10 | | $n_h$=12 | |
| | Max | Min | Max | Min | Max | Min | Max | Min |
| ±0.5 | 39.1 | 37.6 | 39.3 | 38.3 | 44 | 42.5 | 44.2 | 43.2 |
| ±1.0 | 65.5 | 64.5 | 66 | 65.1 | 71.4 | 70.4 | 71.9 | 71 |
| ±1.5 | 80.3 | 80.6 | 81 | 81.5 | 87.3 | 87.6 | 88 | 88.5 |
| ±2.0 | 90.3 | 89.2 | 90.6 | 89.6 | 95.5 | 94.4 | 95.8 | 94.8 |
| ±2.5 | 95.1 | 95 | 95.6 | 95.8 | 98.4 | 98.3 | 98.9 | 99.1 |
| ±3.0 | 97.7 | 95.7 | 98.1 | 95.4 | 99.3 | 97.3 | 99.7 | 97 |
| Max. Dev. | 5.57 | 6.34 | 5.52 | 5.55 | 4.91 | 5.84 | 4.81 | 4.85 |
| Avg. Dev. | 1.04 | 1.02 | 1.05 | 1.06 | 1.04 | 1.07 | 0.94 | 1.01 |

For each network, the training is stopped when the prediction error on the validation set starts increasing. Ten experiments have been made each with MLP and SOFM–MLP. Interestingly, in all but two cases, the training error and validation error exhibited identical behavior. Table V depicts the average performance of MLP and SOFM–MLP using the selected features in conjunction with a validation set. Since in these cases only eight input features are used, the maximum number of nodes is restricted in the hidden layer to 12 only. Comparing Table V with Table III, It is found that in this case, too, there is a marginal improvement in performance for SOFM–MLP with the selected features. Comparing the columns for SOFM–MLP with those of MLP in Table V, again it is found that SOFM–MLP outperforms the conventional MLP. The most important point is that only a few features can be used to get good results.

Table VI: Exact and predicted values of maximum temperature for unseen days

| Unseen days | Predicted Temperature | Exact Temperature |
|---|---|---|
| 02-Jan-2009 | 30 | 31 |
| 01-mar-2009 | 36 | 35 |
| 27-Aug-2009 | 29 | 30 |
| 09-Jun-2009 | 32 | 32 |
| 29-Nov-2009 | 29 | 31 |

Fig. 3 depicts the plot of predicted maximum temperature averaged over ten runs and the accurate temperature on the particular day measured before. The minimum and the maximum error can be viewed with the use of the trend line. The figure is plotted against the values in Table VI.

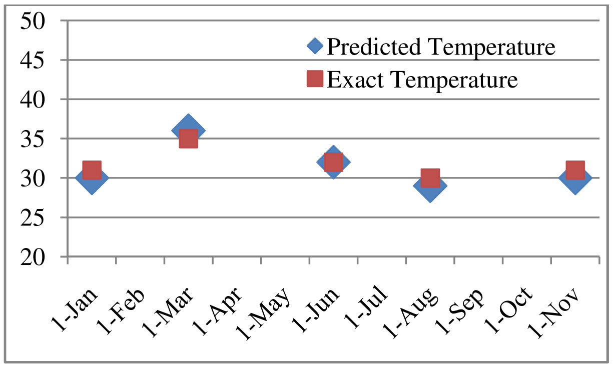

Figure 3. Values plotted for exact and predicted temperature

The value in the graph shows that the predicted values are much closer to the exact values which illustrate the accuracy in temperature prediction of the proposed system.

## V. Conclusion

In this paper a hybrid neural network based temperature prediction system is proposed. The hybrid neural network model combines the self-organizing feature map and the multilayer perceptron network for temperature prediction. In this context, some derived features are used to enhance the prediction accuracy. The importance of feature analysis is further demonstrated using an online feature selection technique. The proposed technique has been compared with both local and global predictors and has been found to produce a much better prediction than the temperature prediction system which uses the MLP, radial basis function (RBF) networks, autoregressive (AR) and linear regressive (LR) models. From the experimental results it can be stated that the Derived features could be more effective than raw features. For example, use of gradient as features instead of the raw observations can reduce the required size of the network and make the training task simpler yet achieving better performance over the raw features. Feature selection is an important factor for better prediction of atmospheric parameters. In this regard, the FSMLP turns out to be an excellent tool that can select good features while learning the prediction task. The combined use of FSMLP and SOFM–MLP results in an excellent paradigm for prediction of atmospheric parameters.

## VI. References

[1] Denis Riordan, and Bjarne K Hansen, "A fuzzy case-based system for weather prediction." Engineering Intelligent Systems, Vol .10, No.3. 2002

[2] Guhathakurtha, P., "Long-Range monsoon rainfall prediction of 2005 for the districts and sub-division kerala with artificial neural network." Current Science, Vol.90, No.6, 25. 2006

[3] Pal, N.R., Srimanta Pal, Jyotirmoy Das, and Kausik Majumdar, "SOFM-MLP: A Hybrid Neural Network for Atmospheric Temperature Prediction." IEEE Transactions on Geoscience and Remote Sensing, Vol.41, No, 12, pp.2783-2791. 2003

[4] Xinghuo Yu, M. Onder Efe, and Okyay Kaynak," A General Back propagation Algorithm for Feedforward Neural Networks Learning,"

[5] Ajith Abraham, Ninan Sajith Philip, Baikunth Nath, P. Saratchandran," Performance Analysis of Connectionist Paradigms for Modeling Chaotic Behavior of Stock Indices,"

[6] http://www.wunderground.com/history/airport/VABB/2005/3/27/CustomHistory.html?dayend=27&monthend=3&yearend=2006&req_city=NA&req_state=NA&req_statename=NA

[7] Y.Radhika and M.Shashi," Atmospheric Temperature Prediction using Support Vector Machines," International Journal of Computer Theory and Engineering, Vol. 1, No. 1, April 2009 1793-8201.

[8] Mohsen Hayati, and Zahra Mohebi," Application of Artificial Neural Networks for Temperature Forecasting," World Academy of Science, Engineering and Technology 28 2007.

[9] Arvind Sharma, Prof. Manish Manoria," A Weather Forecasting System using concept of Soft Computing," pp.12-20 (2006)

[10] Mike O'Neill," Neural Network for Recognition of Handwritten Digits," Standard Reference Data Program National Institute of Standards and Technology.

[11] T. Kohonen, K. Torkkola, M. Shozakai, J. Kangas, and O. Venta, "Microprocessor implementation of a large vocabulary speech recognizer and phonetic typewriter for Finnish and Japanese," in Proc. Eur. Conf. Speech Technology, Edinburg, U.K., 1987, pp. 377–380.

[12] N. M. Nasarabadi and Y. Feng, "Vector quantization of images based on kohonen self-organizing feature maps," Proc. IEEE Int. Conf. on Neural Networks, pp. I-101–I-108, 1988.

[13] K. M. Marks and K. F. Goser, "Analysis of VLSI process data based on self-organizing feature maps," in Proc. Neuro-Nimes, Nimes, France, pp. 337–347.

[14] Z. Chi, J. Wu, and H. Yan, "Handwritten numeral character recognition using self-organizing maps and fuzzy rules," Pattern Recognit., vol. 28, no. 1, pp. 59–66, 1995.

[15] T. Kohonen, The Self-Organizing Maps, 2nd ed. Berlin, Germany: Springer-Verlag, 1997.

[16] Brian A. Smith, Ronald W. McClendon, and Gerrit Hoogenboom," Improving Air Temperature Prediction with Artificial Neural Networks" International Journal of Computational Intelligence 3;3 2007.

[17] N. R. Pal and K. Chintalapudi, "A connectionist system for feature selection," Neural, Parallel Sci. Computat., vol. 5, no. 3, pp. 359–381, 1997.

**AUTHORS**

**Lt. Dr. S. Santhosh Baboo**, aged forty two, has around Nineteen years of postgraduate teaching experience in Computer Science, which includes Six years of administrative experience. He is a member, board of studies, in several autonomous colleges, and designs the curriculum of undergraduate and postgraduate programmes. He is a consultant for starting new courses, setting up computer labs, and recruiting lecturers for many colleges. Equipped with a Masters degree in Computer Science and a Doctorate in Computer Science, he is a visiting faculty to IT companies. It is customary to see him at several national/international conferences and training programmes, both as a participant and as a resource person. He has been keenly involved in organizing training programmes for students and faculty members. His good rapport with the IT companies has been instrumental in on/off campus interviews, and has helped the post graduate students to get real time projects. He has also guided many such live projects. Lt. Dr. Santhosh Baboo has authored a commendable number of research papers in international/national Conference/journals and also guides research scholars in Computer Science. Currently he is Senior Lecturer in the Postgraduate and Research department of Computer Science at Dwaraka Doss Goverdhan Doss Vaishnav College (accredited at 'A' grade by NAAC), one of the premier institutions in Chennai.

**I. Kadar Shereef,** done his Under-Graduation (B.Sc., Mathematics) in NGM College, Post-Graduation in Trichy Jamal Mohamed college and Master of Philosophy Degree in Periyar University (distance education). He is currently pursuing his Ph.D., in Computer Science in Dravidian University, Kuppam, and Andhra Pradesh. Also, he is working as a Lecturer, Department of BCA, Sree Saraswathi Thyagaraja College of Arts and Science, Pollachi. He is having more than one year of research experience and more than five years of teaching experience. His research interest includes Data mining, Climate Prediction, Neural Network and Soft Computing.