Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Technology 4 (2012) 311 – 318

Procedia Technology

C3IT-2012

Weather forecasting model using Artificial Neural Network

Kumar Abhishek[a], M.P.Singh[a], Saswata Ghosh[b], Abhishek Anand[c]

[a]*Dept of CSE, NIT Patna -800005,India*
[b]*Mphasis an HP Company, Mangalore -575001,India*
[c]*Accenture , Bangalore ,India*

Abstract

Weather forecasting has become an important field of research in the last few decades. In most of the cases the researcher had attempted to establish a linear relationship between the input weather data and the corresponding target data. But with the discovery of nonlinearity in the nature of weather data, the focus has shifted towards the nonlinear prediction of the weather data. Although, there are many literatures in nonlinear statistics for the weather forecasting, most of them required that the nonlinear model be specified before the estimation is done. But since the weather data is nonlinear and follows a very irregular trend, Artificial Neural Network (ANN) has evolved out to be a better technique to bring out the structural relationship between the various entities. The paper examines the applicability of ANN approach by developing effective and reliable nonlinear predictive models for weather analysis also compare and evaluate the performance of the developed models using different transfer functions, hidden layers and neurons to forecast maximum, temperature for 365 days of the year.

Keywords-ANN, hidden layer; artificial neurons; MSE; generalization; validation; over-fitting

1. Introduction

Weather forecasting is the application of science and technology to predict the state of the atmosphere for a given location. Weather forecasts are made by collecting quantitative data about the current state of the atmosphere and using scientific_understanding of atmospheric processes to project how the atmosphere will evolve. The chaotic nature of the atmosphere, the massive computational power required to solve the equations that describe the atmosphere, error involved in measuring the initial conditions, and an incomplete understanding of atmospheric processes mean that forecasts become less accurate as the difference in current time and the time for which the forecast is being made increases.
There are a variety of end uses to weather forecasts. Weather warnings are important forecasts because they are used to protect life and property. Forecasts based on temperature and precipitation are important to agriculture, and therefore to traders within commodity markets. Temperature forecasts are used by utility companies to estimate demand over coming days. On an everyday basis, people use weather forecasts to determine what to wear on a given day. Since outdoor activities are severely curtailed by heavy rain, snow and the wind chill, forecasts can be used to plan activities around these events, and to plan ahead and survive them. In order to predict weather in a very effective way and to help overcome all such problems we have proposed a weather forecasting model using Artificial Neural Network. The

advantage which ANN has over other weather forecasting method is that the ANN minimizes the error using various algorithms and gives us a predicted value which is nearly equal to the actual value. Such network is simulated over newer data to find out the weather trend in future course.

2. Related Work

This section focuses on survey that investigates the work that has been done on weather forecasting using artificial neural networks. Special interest is taken on temperature forecast.

In temperature forecasting one has to distinguish between the times the forecast goes ahead, for example temperature one hour ahead or minimum and maximum temperature of a given day. Several works has been done and different artificial neural networks (ANN) models have been tested.

Kaur[8] and Maqsood[1] describes a model that predicts the hourly temperature, wind speed and relative humidity 24 hour ahead. Training and Testing is done separately for winter, spring, and summer and fall season. The authors have made a comparison of Multilayer Perceptron Networks (MLP), Elman Recurrent Neural Network (ERNN), Radial Basis Function Network (RBFN) and the Hopfield Model (HFM) and ensembles of these networks. MLP was trained by back propagation. RBFN has natural unsupervised learning. The authors have suggested one hidden layer and 72 neurons for the MLP network and 2 hidden layers with 180 neurons for RBFN as the optimal architecture. The log-sigmoid is the activation function for the hidden layer unit of MLP network. In RBFN they use a Gaussian activation function. The output is pure line in both cases. The accuracy measure used is the mean absolute percentage error (MAPE). RBFN has the best performance. RBFN and MLP have about the same accuracy, but the MLP learning process is more time consuming. For winter and spring humidity prediction has the lowest MAPE, for summer and fall temperature forecast performs best. However, the performance of ensembles outperformed all single networks. Two different types of ensembles were used: weighted average (WA) and winner takes all (WTA). WTA had the lowest MAPE and therefore the highest accuracy. Unfortunately no information about input parameters for the ANNs is provided.

The work described by Sanjay Mathur[11] focuses on maximum and minimum temperature forecasting and relative humidity prediction using time series analysis. The network model used is a Multilayer feed-forward ANN with back propagation learning. Direct and statistical input parameters and the period are compared. For minimum/maximum temperature forecasting the optimum seems to be a 15 week period of input data. Input features were features of maximum and minimum, respectively. Namely these features are moving average, exponential moving average, oscillator, rate of change and the third moment. For the 15 week period the error was less than 3%. The main result is that in general statistical parameters can be used to extract trends. Good parameters are moving average, exponential moving average, oscillator, rate of change and moments. Skewness and kurtosis did not perform well.

Another short term temperature forecasting system is described by Hayati[10]. A three layer MLP network with 6 hidden neurons, a sigmoid transfer function for the hidden layer and a pure linear function for the output layer was found to yield the best performance. The scaled conjugate gradient algorithm was used for training. The following input parameters were measured every three hours: wind speed, wind direction, dry bulb, temperature, wet bulb temperature, relative humidity, dew point, pressure, visibility, amount of cloud. The other input parameters were measured daily: gust wind, mean temperature, maximum temperature, minimum temperature, precipitation, mean humidity, mean pressure, sunshine, radiation, evaporation.

A fully connected, feed forward 3 layer MLP network for temperature prediction is also presented by Santhosh Babu [7]. The error is said to be "very less". The set of input differs. Atmospheric pressure, atmospheric temperature, relative humidity, wind velocity and wind direction are chosen. The training is done by back propagation. The predictions are restricted by an upper bound, which can be considered as reducing the transferability to other locations.

Most of the approaches mentioned above use MLP networks. Sergio[12] uses evolutionary neural networks in combination with generic algorithms to predict the maximum temperature per day. The suggested input parameters are month, day, daily precipitation, max temperature, min temperature, max

soil temperature, min soil temperature, max relative humidity, min relative humidity, solar radiation, and wind speed. The accuracy is 79.49 for a 2-degree error bound. Training was done by back propagation. The author assumes that accuracy could be increased by using a larger training set, different training algorithms and more atmospheric values.
The most often used architecture is MLP. Also a recurrent architecture (RBFN) has been used and yielded good results. Ensembles of ANNs also seem to be promising. The parameters differ slightly between the approaches.

3. Experimental Set up

The Neural Network Fitting Tool GUI *nntool* available in MATLAB 7.6.0 (R2008a) is used to carry out the analysis on the weather data using Artificial Feed-Forward Neural Network with back-propagation principles [6].
The paper of Maqsood and Khan[1] does a comparative study of MLPN, ERNN etc. However, we have begun our research with a simple feed-forward back propagation model and have explored the importance of the ANN at its unit level that is the artificial neurons. These artificial neurons are the building blocks of all such ANNs and to understand their potential has been our main study. The questions we have attempted to answer are:

1. How and to what extent, increasing the number of artificial neurons increases the performance?
2. How does increasing the number of hidden layers affect the ANN performance?
3. The conclusions from 1 and 2 are integrated and again experimented to help the ANN designer tune the delicate balance between choosing the number of hidden layers, and how many neurons he should put in each layer for optimal performance. In all the papers that we have referenced, only 1 hidden layer was taken. We have experimented with more hidden layers.

3.1 Data Specification

The main aim is to create and train a network that can predict the individual weather components for e.g. maximum temperature, minimum temperature, wind speed etc. for a particular station and a particular day given the weather for the previous day (*target data*) and the historical 10 year data of that particular day *(input data*). Our study has examined only the maximum temperature because our main focus is on the design of the Artificial Neural Network which itself has numerous parameters to vary and optimize during experimentation. The model once created can then be fed with other weather factors in a similar fashion.
We used data available for the station Toronto Lester B. Pearson Int'l A, Ontario, Canada (Latitude 43.68, Longitude -79.63, elevation 173.4 m above msl) from period 1999-2009. [4]

3.2 Input Variables

The input dataset consists of 365 samples corresponding to the 365 days of a year arranged column-wise in an Excel sheet which is later imported into the MATLAB workspace. February 29th in leap years has not been considered for the sake of uniformity. Each sample (column) i.e. day of the year in turn has 10 rows corresponding to the maximum temperature on that day recorded in the past 10 years. Out of the 365 samples, 60% samples called the *training data* are randomly selected by *nntool* for training the neural network. 20% samples called the *validation dat*a measure the generalization of the network by feeding it with data it has not seen before. The remaining 20% samples called the *test data* give an independent measure of the performance of the neural network in terms of MSE (Mean squared error). It is the square of the difference of the predicted value and the target, hence always positive.

It is known that Neural Network training is better with larger dataset. However, our aim restricts us to 365 samples. Therefore, we have augmented our dataset to twice and four times its size (730 and 1460 samples respectively) by simply appending it with itself that many times. Although it does not provide the variety needed for better generalization, it increases the learning rate as will be illustrated in our observations.

3.3 Neural Network Model

A typical feed forward with back propagation network should have at least three layers- an input layer, a hidden layer, and an output layer. Appropriate selection of number of hidden layers and the number of neurons in each of them needs experimentation. We train the ANN using the Levenberg-Marquardt algorithm, a standard training algorithm from the literature. The algorithm is terminated according to the early stopping procedure. The validation set used in the early stopping procedure is chosen in a somewhat unusual manner. Finally, the training function produce forecast results on the basis of MSE (Mean square error) minimization criteria. In one complete cycle of the training process, a set of input data of maximum temperature {Year1, Year 2, and Year3…Year10} is presented to the input node. The corresponding target output is the maximum temperature of the previous day. It is presented to the output node in order to show the network what type of behavior is expected. The output signal is compared with the desired response or target output and consequently an error signal is produced. In each step of iterative process, the error signal activates a control mechanism which applies a sequence of corrective adjustments of the weights and biases of the neuron. The corrective adjustments continue until the training data attains the desired mapping to obtain the target output as closely as possible. After a number of iterations the neural network is trained and the weights are saved. The test set of data is presented to the trained neural network to test the performance of the neural network. The result is recorded to see how well the network is able to predict the output using the adjusted weights of the network.

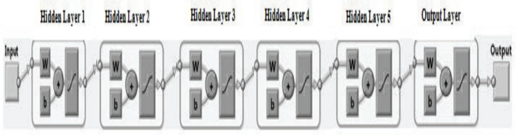

Fig-2 Five hidden layer model

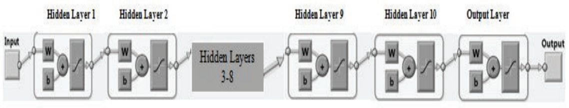

Fig- 3 Ten hidden layer model

3.4 Observation Recorded

We have first analyzed the effect of number of neurons (20, 50 and 80) and the transfer function (tan-sigmoid or pure-linear) on the MSE in case of a single hidden layer. Sometimes a very low MSE can be mistaken as good accuracy when in fact it points to a serious problem called '*overfitting'* which is a characteristic of many learning algorithms. *Overfitting* occurs when a model begins to memorize the training data rather than learning to generalize from the model. We then seek to demonstrate if increasing the number of hidden layers and distributing the number of neurons in them can contribute in any way to overcome this problem without compromising on performance. The following six distributions are therefore proposed:-

20 Neurons:- 4/Layer and 2/Layer for 5 and 10 hidden layers respectively
50 Neurons:- 10/Layer and 5/Layer for 5 and 10 hidden layers respectively
80 Neurons:- 16/Layer and 8/Layer for 5 and 10 hidden layers respectively

The readings as shown in Table 1, Table 2 & Table 3 have been taken considering the best of 20 trials and re-trainings. The stopping criteria tabulated above is simply the message displayed when the network stops training and the resulting MSE is generated. The three most common things that we encountered are:

Validation Stop:- It is most frequently observed when the transfer function is tan-sigmoid. Validation vectors are used to stop training early if the network performance on the validation vectors fails to improve or remains the same for max_fail epochs in a row.
Maximum Mu reached:- Mu, is a parameter of the trainlm algorithm and measures the adapting/learning rate of the network. Maximum Mu reached means the learning rate has reached its maximum and further trainings will lead only to a validation stop or mostly, a minimum gradient.
Minimum Gradient reached:- Gradient is the direction of change of the values. Our algorithms implement gradient descent to find local minimum, and hence the term 'minimum gradient' which is pre-defined for the network.

Table 1- Single Hidden Layer

Samples	Neurons /Layer	Transfer Function of HL	MSE	Stopping Criteria
365	20	TANSIG	15.4	Validation Stop
730	20	TANSIG	10.6	Validation Stop
1460	20	TANSIG	2.79	Validation Stop
365	20	PURELIN	19.3	Min Gradient
730	20	PURELIN	18.8	Min Gradient
1460	20	PURELIN	16.1	Max MU Reached
365	50	TANSIG	10.3	Validation Stop
730	50	TANSIG	3.65	Validation Stop
1460	50	TANSIG	0.885	Validation Stop
365	50	PURELIN	18	Max MU Reached
730	50	PURELIN	17.6	Max MU Reached
1460	50	PURELIN	17.4	Max MU Reached
365	80	TANSIG	7.14	Validation Stop
730	80	TANSIG	2.71	Validation Stop
1460	80	TANSIG	0.211	Validation Stop
365	80	PURELIN	16.5	Min Gradient
730	80	PURELIN	17	Min Gradient
1460	80	PURELIN	18.2	Min Gradient

Table 2- Five Hidden Layer

Samples	Neurons /Layer	Transfer Function of HL	MSE	Stopping Criteria
365	4	TANSIG	16.1	Validation Stop
730	4	TANSIG	12.5	Validation Stop
1460	4	TANSIG	7.23	Min Gradient
365	4	PURELIN	18.8	Min Gradient
730	4	PURELIN	18.6	Min Gradient
1460	4	PURELIN	18.2	Min Gradient
365	10	TANSIG	13.4	Validation Stop
730	10	TANSIG	4.02	Validation Stop
1460	10	TANSIG	1.44	Validation Stop
365	10	PURELIN	17.9	Min Gradient
730	10	PURELIN	17.8	Min Gradient
1460	10	PURELIN	17.6	Min Gradient
365	16	TANSIG	10.4	Validation Stop
730	16	TANSIG	2.75	Validation Stop
1460	16	TANSIG	0.201	Validation Stop
365	16	PURELIN	16.8	Max MU Reached
730	16	PURELIN	16.4	Max MU Reached
1460	16	PURELIN	16	Max MU Reached

Table 3- Ten Hidden Layer

Samples	Neurons /Layer	Transfer Function of Hidden Layer	MSE	Stopping Criteria
365	2	TANSIG	16.4	Validation Stop
730	2	TANSIG	16.2	Validation Stop
1460	2	TANSIG	15.9	Validation Stop
365	2	PURELIN	18.9	Min Gradient
730	2	PURELIN	18.3	Min Gradient
1460	2	PURELIN	18	Min Gradient
365	5	TANSIG	16	Validation Stop
730	5	TANSIG	8.48	Validation Stop
1460	5	TANSIG	5.3	Validation Stop
365	5	PURELIN	17.7	Min Gradient
730	5	PURELIN	17.6	Min Gradient
1460	5	PURELIN	17.5	Min Gradient
365	8	TANSIG	13.1	Validation Stop
730	8	TANSIG	5.73	Validation Stop
1460	8	TANSIG	1.52	Validation Stop
365	8	PURELIN	16.9	Max MU Reached
730	8	PURELIN	16.6	Max MU Reached
1460	8	PURELIN	16.1	Max MU Reached

4. Results and Discussion

From the set of observations presented, we study the following factors which have affected the performance of our Artificial Neural Network Model:-

4.1 No. of Neurons/Layer:-

Increasing the number of neurons/layer decreases the MSE i.e. increases the performance.

4.2 No. of Samples:-

Increasing the number of samples (here 750 and 1460 compared to 365) decreases the MSE i.e. increases the performance.

4.3 Transfer Function for Hidden Layers:-

Pure-linear function is kept fixed for output layer in all cases. Tan-Sigmoid function is our best choice for hidden layers over pure-linear function because of its very fast learning rate and sensitivity towards change in number of samples and neurons/layer. However, increasing the number of hidden layers above an optimum can adversely increase its performance (for e.g. increasing it from 5 to 10).

4.4 No. of Hidden Layers:-

For a single hidden layer ANN and 730 samples, increasing the number of neurons to 50 and 80 from 20 gave sharp decrease in our MSE to the tune of 3.65 and 2.71 from 10.6 respectively. 3.65 is an acceptable accuracy but 2.71 is identified as *overfitting* according to the graph analysis dealt under '*overfitting*'. To overcome this, we increased the number of hidden layers to 5 and distributed the 80 neurons among each layer such that 16 neurons/layer. We got a resulting MSE of 2.75 which was accurate as well as free from *overfitting*. However when we further increased the number of layers to 10 such that 8 neurons/layer (total 80 neurons), the MSE again declined to 5.73. We concluded that the reason this distribution failed our expectation was that our number of inputs is 10 and 8<10 but 15>10. Considering only the first hidden layer and based on our notion that more number of neurons increases the performance, a the first hidden layer of a 5-hidden layer network presents 15 neurons to the 10 inputs compared to the 8 neurons of the 10-hidden layer network. Hence, the latter declined in performance while handling the same information.

4.5 Overfitting:-

Sometimes a very low MSE can be mistaken as good accuracy when in fact it points to a serious problem called 'overfitting*'*. The final MSE generated is due to the isolated test data which is random 20% of the samples. A low MSE may be possible even if the generalization is very poor, i.e. MSE of validation data may be much higher. Such a case of a single hidden-layer model having 750, 10-input samples and 80 neurons generating MSE of 2.71 is shown below.

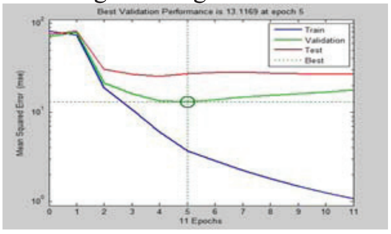

Fig- 4 Large variation between the test and validation data pointing to overfitting

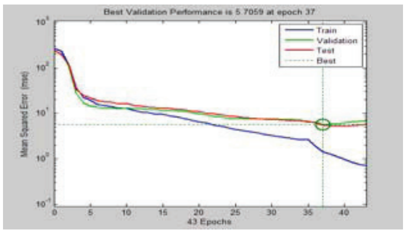

Fig.5- Closeness between the test and validation data- good generalization achieved

Now, we can contrast the previous graph with the one presented in Fig 5 with a fairly acceptable MSE of 2.75 for the same number of neurons but distributed in a 5 hidden-layer network with 16 neurons/layer. The validation MSE curve seems to closely follow the test MSE curve confirming improved generalization.

5. Conclusion

Most Artificial Neural Network approaches preprocess the input and target data into a range -1 to +1 or 0 to 1 and then post-process it. However, we investigated on finding a model that can reduce this processing cost by working on raw data. Since we have 10 inputs, a 5 hidden-layer network with 10 or 16 neurons/ layer and a tan-sigmoid transfer function for hidden layers seemed to do generalize much better over 750 and 1460 samples as compared to a single hidden-layer network with the same number of neurons. We have already discussed the method to analyze and handle *overfittin*g while aiming for accuracy in prediction. However the most important conclusion that our study resulted to was on the behavior of increased hidden layers on performance and generalization. It can be summarized as under:
Finally, the prediction that we made for the maximum temperature can be extended to other weather factors like humidity, wind speed etc. using the same model and precautions discussed. Further measures to optimize the performance of such a weather forecasting model can be based on various macro and micro-environmental factors. This study can be best used to develop supportive statistical plots and concentrate on the trend of weather over a long period of time in a particular area.

References

1. Imran Maqsood, Muhammad Riaz Khan, and Ajith Abraham. An ensemble of neural networks for weather forecasting, Neural Comput & Applic (2004) 13: 112–122.
2. Ball , R. , Tissot , P. (2006) Demonstration of Artificial Neural Network in Matlab
3. G. R. Gainieva, L. D. Nikitin, M. M. Naimark, N. N. Nazarov, and G. P. Tkachenko, Influence of Batch Composition and Clinkering Properties on the Hot Strength of Coke and Blast-Furnace Operation, ISSN 1068-364X, Coke and Chemistry, 2008, Vol. 51, No. 10, pp. 390–393. © Allerton Press, Inc., 2008
4. *http://climate.weatheroffice.gc.ca/climateData/canada_e.html* .The website of the weather data of Canada
5. Haider, Adnan and Hanif, Nadeem, M., *Inflation Forecasting In Pakistan Using Artificial Neural Networks*
6. Matlab 7.6.0 (R2008a), The Math Works Inc., *Product Help- Neural Network Toolbox*
7. Dr. S. Santhosh Baboo and I.Kadar Shereef. An e_cient weather forecasting system using Artificial neural network. International Journal of Environmental Science and Development, 1(4):321-326, 2010.
8. Amanpreet Kaur, J K Sharma, and Sunil Agrawal. Artificial neural networks in forecasting maximum and minimum relative humidity. International Journal of Computer Science and Network Security, 11(5):197-199, May 2011.
9. Sanjay Mathur, Avinash Kumar, and Mahesh Chandra. A feature based neural network model for weather forecasting, World Academy of Science, Engineering and Technology 34 2007.
10. Mohsen Hayati and Zahra Mohebi. Application of artificial neural networks for temperatureforecasting. World Academy of Science, Engineering and Technology, 28:275{279, 2007.
11. Rosmina Bustami, Nabil Bessaih, Charles Bong, and Suhaila Suhaili. Artificial neural network for precipitation and water level predictions of bedup river. IAENG International Journal of Computer Science, 34(2):228-233, 2007.]
12. Sergio Caltagirone. Air temperature prediction using evolutionary arti_cial neural networks. Master's thesis, University of Portland College of Engineering, 5000 N. Willamette Blvd. Portland, OR 97207, 12 2001.
13. Surajit Chattopadhyay. Multilayered feed forward artificial neural network model to predict the average summer-monsoon rainfall in india. Master's thesis, Department of Mathematics, Techno Model SchoolTechno India Group, EM-4/1, Sector V, Salt Lake, Kolkata 700 091, India, 2006.
14. Tony Hall, Harold E. Brooks, and Charles A. Doswell III. Precipitation forecasting using a neural network, AMS Journal online, Volume 14, Issue 3 (June 1999).
15. N. Q. Hung, M. S. Babel, S. Weesakul, and N. K. Tripathi. An arti_cial neural network model for rainfall forecasting in bangkok, Thailand. Hydrology and Earth System Sciences, 13(8):1413-1425, 2009.
16. Adrian Visoiu. Neural network based model refinement. Informatica Economica Journal,2008.