Statistika, Vol. 10 No. 1, 1 – 8
Mei 2010

Forecasting Malaysia Load Using a Hybrid Model

NORIZAN MOHAMED[1], MAIZAH HURA AHMAD[2]

[1]Mathematics Department, Faculty Of Science And Technology, Universiti Malaysia Terengganu (UMT), 21030 Kuala Terengganu, Terengganu, MALAYSIA.
[2]Department of Mathematics, Faculty Of Science, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, MALAYSIA.
E-mail: norizan@umt.edu.my, maizah@.utm.my

ABSTRACT

A hybrid model, which combines the seasonal time series ARIMA (SARIMA) and the multilayer feed-forward neural network to forecast time series with seasonality, is shown to outperform both two single models. Besides the selection of transfer functions, the determination of hidden nodes to use for the non linear model is believed to improve the accuracy of the hybrid model. In this paper, we focus on the selection of the appropriate number of hidden nodes on the non linear model to forecast Malaysia load. Results show that by using only one hidden node, the hybrid model of Malaysia load performs better than both single models with mean absolute percentage error (MAPE) of less than 1%.
Keywords: Load Forecasting; Seasonal Autoregressive Integrated Moving Average; Multilayer Feed-forward Neural Network; Hybrid Model; Hidden Nodes.

1. INTRODUCTION

An important area of forecasting is time series forecasting where the model of an observed time series is developed to forecast the future values. Load demand is one of the time series data and it is one of the major input factors in economic development especially in a developing country such as Malaysia. Forecasting load demand with accurate forecast is hoped to help our country, especially the Department of Operation and Planning (Tenaga Nasional Berhad, Malaysia) to generate an appropriate load of required power supply.

One of the most important and widely used time series models is the autoregressive integrated moving average (ARIMA) model. An ARIMA process combines three different processes comprising an Autoregressive (AR) functions regressed on the past values of the process, moving average (MA) functions regressed on a purely random process with mean zero and variance σ^2 [6] and an integrated (I) part to make the data series stationary by differencing [6][20]. The ARIMA model is extended in order to handle seasonal aspects of time series, and the general notation is $ARIMA(p,d,q)(P,D,Q)_s$ where (p,d,q) is the non-seasonal part of the model, (P,D,Q) is the seasonal part of the model and s is the seasonal length with abbreviated as seasonal ARIMA, SARIMA [11].

Recently, artificial neural networks (ANNs) have been extensively studied and received increasing attention in time series forecasting in which the greatest advantage of a neural network is its ability to model a nonlinear process without priori assumptions on the nature of the process [4][10]. Lately, a hybrid model which combines the linear model and the non linear model were shown to outperform both two single models, hence improve the forecasting performance. Besides the selection of transfer functions, the determination of hidden nodes to be used for the non linear model is believed to improve the accuracy of the hybrid model.

The number of hidden nodes is closely linked to the number of degrees of freedom. The degrees of freedom strongly influence the shape of the function in the network structure. An increasing number of hidden nodes will increase the number of freedom and thus increase the capability of the network to model complex function accurately. However, an increase in the number of hidden nodes will also increase the capacity of network to over fit [16]. Thus, to reduce the capacity of network to over-fit, one should increase the number of data points to several thousands or partition the data points to three parts which are training, validation and testing.

Working with few hidden nodes saves time in training the network. However, if a network is defined with too few hidden nodes, the network probably would not train at all or train with an unreliable forecasting because of the inability of the network to reproduce the system dynamics accurately [7][9]. If we defined just enough hidden nodes, the network may train but it might not be robust in the face of noisy data, or it would not recognize new patterns [7]. Too many hidden neurons, in the best case, will provide reliable forecasting, but with an excessive computing time and a high memory consumption, while, in the worse case, will only learn the presented patterns and will not be able to generalize the acquired knowledge to predict new patterns [7][9]. Therefore, the selection of the appropriate number of hidden nodes is not an easy task to get back-propagation neural network to train successfully.

The main purpose of this study is to show that a hybrid model can overcome the problem of selecting the appropriate number of hidden nodes and hence improve forecasts produced by two single methods, namely SARIMA and multilayer feed-forward neural network. The current paper presents a hybrid model with a selection between 1 to 19 hidden nodes for the non linear model.

The remainder of this paper is organized as follows. In section 2, we present the Box-Jenkins seasonal ARIMA model and the multilayer feed-forward neural network model. The hybrid model is presented in section 3. In section 4, details of the results are discussed. Finally in section 5 we give our conclusions.

2. FORECAST METHODOLOGY

A variety of different forecasting approaches is available to forecast time series data and it is important to realize that no single model is universally applicable [5]. Two approaches presented here are the Box-Jenkins seasonal ARIMA (SARIMA) and the multilayer feed-forward neural network models.

2.1 Box-Jenkins Seasonal ARIMA (SARIMA) Model

In practice, many time series contain a seasonal periodic component, which repeats every s observation. To deal with seasonality, the ARIMA model is generalized hence a general multiplicative seasonal ARIMA (SARIMA) model is defined [2][3] which follows the ARIMA general procedure represented as follows:

$$\phi_p(B)\Phi_P(B^s)(1-B)^d(1-B^s)^D\dot{Z}_t = \theta_q(B)\Theta_Q(B^s)a_t \qquad (1)$$

with

$$\phi_p(B) = 1-\phi_1 B-\phi_2 B^2-\cdots-\phi_p B^p,$$
$$\Phi_P(B^s) = 1-\Phi_s B^s-\Phi_{2s} B^{2s}-\cdots-\Phi_{Ps} B^{Ps},$$
$$\theta_q(B) = 1-\theta_1 B-\theta_2 B^2-\cdots-\theta_q B^q,$$
$$\Theta_Q(B^s) = 1-\Theta_s B^s-\Theta_{2s} B^{2s}-\cdots-\Theta_{Qs} B^{Qs}$$

where B denotes the backward shift operator and a_t denotes a purely random process.

The original series Z_t is differenced by appropriate differencing to remove non-stationary terms. $(1-B)^d$ and $(1-B^s)^D$ are the seasonal and non-seasonal differencing operators, respectively [11]. If the integer D is not zero, then seasonal differencing is involved. The above model is called a SARIMA model of order $(p,d,q)(P,D,Q)_s$. If d is non-zero, then there is a simple differencing to remove trend, while seasonal differencing $(1-B^s)^D$ may be used to remove seasonality. In practice D is rarely more than one and P and Q are typically less than three [3].

2.2 Multilayer Feedforward Neural Network

The multi-layer feed-forward neural network (MFNN) is a well known neural model, which consists of an input layer, one or several hidden layers and an output layer. The neurons in the feed-forward neural network, are generally grouped into layers. Signals flow from the input layer to the output layer via unidirectional connections, the neurons being connected from one layer to the next, but not within the same layer [15].

An essential factor of successes of the neural networks depends on the training network. Among the several learning algorithms available, back-propagation has been the most popular and most widely implemented learning algorithm of all neural networks paradigms [4][5]. Among the advantages of back-propagation (BP) is its ability to store numbers of patterns that exceed its built-in vector dimensionality [4]. Basically, the BP training algorithm with three-layer feed-forward architecture means that, the network has an input layer, one hidden layer and an output layer. More hidden layer can be used but three layers are sufficient to enable this type of network to model any deterministic process within reasonable limit [18].

Neural network function sends the vector $(x_1,\cdots,x_N)$ in R^N to the vector $(y_1,\cdots,y_M)$ in R^M. Thus, the feed-forward network can be represented as [18]:

$$y = F(x) \tag{2}$$

where $x=(x_1,\cdots,x_N)$ and $y=(y_1,\cdots,y_M)$. The values y_k for the feed-forward network with N input nodes, H hidden layer nodes and M output nodes are given by [8]:

$$y_k = g_2\left(\sum_{j=1}^{H} w_{kj}h_j + w_{k0}\right), k = 1,\cdots,M \tag{3}$$

Here w_{kj} is an output "weight" from hidden node j to output node k, w_{ko} is the bias for output node, and g_2 is an activation function. The values of the hidden layer nodes h_j, $j=1,\cdots,H$ are given by:

$$h_j = g_1\left(\sum_{i=1}^{N} v_{ji}x_i + v_{j0}\right), j = 1,\cdots,H \tag{4}$$

Here, v_{ji} is the input "weight" from input node i to hidden node j, v_{jo} is the bias for hidden node j, x_i is the value at input node i and g_1 is an activation function .The activation function, g_1 may be the same as activation function, g_2 or may be a different function [18].

Several different BP training algorithms were employed and all of them use the gradient of the performance function in the determination of the weights adjustment. The basic BP algorithm adjusts the weights in the steepest descent direction (negative of gradient) in which, this direction decreases the performance function most rapidly [6][8]. The weights, in the network are adjusted by comparing the actual response with the target response for the minimization of an error function [10][18]. The sum squared error is used as error function and for the input pattern p, the sum squared error is defined as [1][8][18] :

$$E = E^P = \frac{1}{2}\sum_{k=1}^{m}\left(t_k^p - y_k^p\right)^2 \tag{5}$$

where y_k^p is the output of the kth output neuron for the pth pattern, t_k^p is desired or target output for the kth output neuron for the pth pattern and $k=1,2,\ldots,m$.

To obtain the weight adjustment, we have to perform the partial derivative of the E^p (the superscript p is dropped, then $E^p = E$) with respect to weights v_{ji} and w_{kj}. The weight adjustment for output units can be obtained by [14]:

$$w_{kj}(s+1) = w_{kj}(s) + \Delta w_{kj} \tag{6}$$

where $\Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}}$

Thus, to evaluate $\frac{\partial E}{\partial w_{kj}}$ terms, we use the relations in Eq. 4 and Eq. 6. Assuming $l_k = \sum_{j=1}^{H} w_{kj} h_j + w_{k0}$ therefore, $y_k = g_2(l_k)$. Using the chain rule will give:

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial l_k} \cdot \frac{\partial l_k}{\partial w_{kj}} \tag{7}$$

Solving partial derivative terms in Eq. 8 gives:

$$\frac{\partial}{\partial w_{kj}} \left(\sum_{j=1}^{H} w_{kj} h_j + w_{k0} \right) = h_j ,$$

$$\frac{\partial E}{\partial l_k} = \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial l_k} = -(t_k - y_k) g_2{}'(l_k)$$

Then, the partial derivative of the E with respect to weights w_{kj} $\frac{\partial E}{\partial w_{kj}}$, is given by:

$$\frac{\partial E}{\partial w_{kj}} = -(t_k - y_k) g_2{}'(l_k) h_j \tag{8}$$

Thus, the update rule for the weights to the output units is given by [8][14]:

$$\Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}} = \eta (t_k - y_k) g_2{}'(l_k) h_j = \eta \delta_k h_j \tag{9}$$

where $\delta_k = (t_k - y_k) g_2{}'(l_k)$.

For the hidden layer weight v_{ji}, we do not have target values to compute the errors. Thus, we must use the errors from the output units Eq. 6 to adjust the input to hidden layer weights. The chain rule is repeatedly used to relate the output errors to these weights. The weight adjustment for hidden layer units can be obtained by [14]:

$$v_{ji}(s+1) = v_{ji}(s) + \Delta v_{ji} \tag{10}$$

where $\Delta v_{ji} = -\eta \frac{\partial E}{\partial v_{ji}}$.

To evaluate $\frac{\partial E}{\partial v_{ji}}$ terms, we use the relations in Eq. 4, Eq. 5 and Eq. 6. Assuming $a_j = \sum_{i=1}^{N} v_{ji} x_i + v_{j0}$, then $h_j = g_1(a_j)$. By using the chain rule, we will get:

$$\frac{\partial E}{\partial v_{ji}} = \frac{\partial E}{\partial h_j} \cdot \frac{\partial h_j}{\partial a_j} \cdot \frac{\partial a_j}{\partial v_{ji}} \tag{11}$$

Solving the partial derivative terms in Eq. 12, then the partial derivative of the E with respect to weights v_{ji} $\frac{\partial E}{\partial v_{ji}}$, is given by:

$$\frac{\partial E}{\partial v_{ji}} = -x_i g_1'(a_j) \sum_k (t_k - y_k) g_2'(l_k) w_{kj} \tag{12}$$

Thus, the update rule for the weights to the hidden units is given by [8][14]:

$$\Delta v_{ji} = \eta \delta_j x_i \tag{13}$$

where $\delta_j = g_1'(a_j) \sum_k \delta_k w_{kj}$.

3. HYBRID MODEL

In a real data problem, the difficulty in forecasting often arises due to the data characteristics. To overcome this difficulty, a hybrid model which combines both linear and nonlinear capabilities that employ seasonal ARIMA model and artificial neural network model can be a good strategy to practice [17][19]. A hybrid model is considered to be composed of a linear and a nonlinear component and can be represented as [19]:

$$y_t = L_t + N_t \tag{14}$$

where L_t denotes the linear component and N_t denotes the nonlinear component. Both of these two components have to be estimated from the data. Firstly, to forecast the linear component, seasonal ARIMA model is fitted to the data series. Hence, the residuals from the linear model, seasonal ARIMA model is assumed to contain only the nonlinear associations. Let r_t be the residual at time t of the linear model, then,

$$r_t = y_t - \hat{L}_t \tag{15}$$

where $\hat{L}_t$ is the forecast value at time t from the linear model. Nonlinear relationships can be discovered by modeling residuals using artificial neural network model as follows:

$$r_t = F(x) \tag{16}$$

where $x = (x_1, \cdots, x_N)$. To determine the values r_t for the feed-forward network with N input nodes, H hidden layer at time t, we now let

$$r_t = g_2\left(\sum_{j=1}^{H} w_{1j} h_j + w_{10}\right) \tag{17}$$

Here w_{1j} is an output "weight", w_{10} is the bias for output node, and g_2 is an activation function. The values of the hidden layer nodes h_j, is obtained from Eq. 5. Let $\hat{N}_t$ be the forecast value at time t from artificial neural network model, then the combined forecast will be

$$\hat{y}_t = \hat{L}_t + \hat{N}_t \tag{18}$$

Hence, the hybrid method involves a two-stage process:

a) Forecast the linear part using seasonal ARIMA;

b) Forecast the nonlinear part (the residuals from seasonal ARIMA) using the neural network model.

4. RESULTS

Using Malaysia load data, Mohamed et al. [12][13] showed that the MAPE values using SARIMA and multilayer feed-forward neural network are 0.9774104 and 3.7928802 respectively. The data used are four-month half hourly load demand measured in Megawatts (MW) from September 01, 2005 to December 31, 2005 gathered from Tenaga Nasional Berhad (TNB), Malaysia as illustrated in Figure 1. Using the same data the performances of the hybrid model with selection of the number of hidden nodes for the non linear model are compared. Section 4.1 reports the results obtained from the number of hidden nodes which varies from 1 to 19.

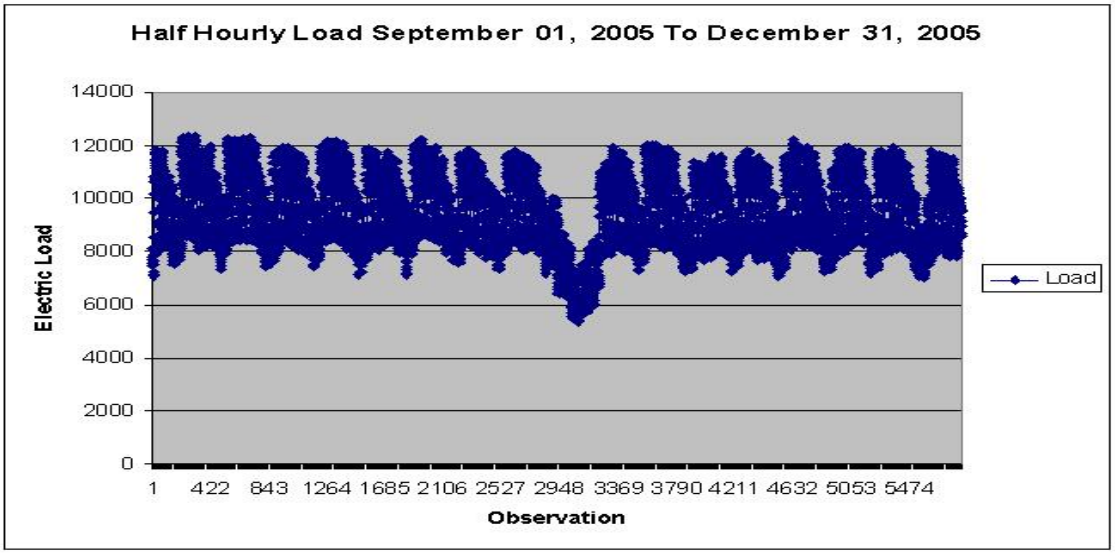

Figure 1. A half Hourly Load from September 01, 2005 to December 31, 2005

4.1 Hidden Nodes of Hybrid Model

In this section we report the effects of the number of hidden nodes which were chosen in the hidden layer. Using the log sigmoid function for both the hidden layer and the output layer as transfer function, Leverberg Marquardt as training algorithm and three input nodes, we present the selection number of hidden nodes between 1 and 19 for forecasting the nonlinear model. The performances of the hybrid model with selected number of hidden nodes are shown in Table 1.

Table 1. The RMSE, MAE and MAPE
of the Different Numbers of Hidden Nodes

Hidden Nodes Number	Performance (MAPE)
1	0.9744785
4	0.9678751
7	0.9584093
10	0.9532558
13	0.9466177
16	0.9374856
19	0.9351380

5. CONCLUSIONS

In this paper, we focus on the selection of the appropriate number of hidden nodes in a hybrid model to forecast Malaysia load. Using the mean absolute percentage error (MAPE) as performance measurement, we found that, by using only one hidden node, the hybrid model of Malaysia load performed better than both single models with mean absolute percentage error (MAPE) of less than 1%. Thus, it is concluded that a hybrid model as proposed by this paper will not only improve forecasting performance but will also reduce training time.

REFERENCES

[1]. Bengio, Y., Neural Networks for Speech and sequence recognition, International Thomson Computer Press, 1995.

[2]. Box, G. E. P., Jenkins, G. M. & Reinsel, G. C., Time Series Analysis: Forecasting and Control. Prentice Hall, New Jersey, 1994.

[3]. Brockwell, P. J., & Davis, R. A. Introduction to Time Series and Forecasting, Springer, New York, 1996.

[4]. BuHamra, S., Smaoui, N., & Gabr, M., The Box-Jenkins analysis and neural networks: prediction and time series modeling, Applied Mathematical Modeling, 27, 2003, 805-815.

[5]. Chatfield, C., The Analysis of Time Series: An Introduction, Sixth Edition, Chapman & Hall, New York, 2004.

[6]. Chen, W. H., Shih, J. Y. and Wu, S., Comparison of support vector machines and back propagation neural networks in forecasting the six major Asian stock markets, Int. J. Electronic Finance, 1, 1, 2006, 49-67.

[7]. Eberhart, R. C., Dobbins, R. W., Neural Network PC Tools, A Practice Guide, Academic Press, Inc. London, 1990.

[8]. Fausett, L., Fundamentals of neural networks architecture, algorithms and applications, Prentice Hall, New Jersey, 1994.

[9]. Gonzalez-Romera, E, Jaramillo-Moran, M.A, Carmona-Fernandez, D., Monthly electric energy demand forecasting with neural networks and Fourier series. Energy Conversion and Management, 2008.

[10]. Khoa, N. L. D., Sakakibara, K. and Nishikawa, I., Stock Price Forecasting using Back Propagation Neural Networks with Time and Profit Based Adjusted Weight Factors. SICE-ICASE International Joint Conference, 2006, 5484-5488.

[11]. Makridakis, S., Wheelwright, S. C and Hyndman, R. J., Forecasting: Methods and application. John Willey and Sons, New York, 1998.

[12]. Mohamed, N., Ahmad, M. H., Ismail, Z., and Arshad, K. A., Multilayer Feedforward Neural Network Model and Box-Jenkins Model for seasonal load Forecasting International Journal of Physical Sciences, Ultra Science,20, 2008,767-772.

[13]. Mohamed, N., Ahmad, M. H., Ismail, Z., and Arshad, K. A., A hybrid of artificial neural networks and SARIMA models for load forecasting, International Journal of IT and Knowledge Management, 1, 2008, 179-190.

[14]. Patterson, D. W., Artificial neural networks, theory and applications, Prentice Hall, Singapore, 1995.

[15]. Pham, T. D., and Liu, X., Neural Networks for Identification, Prediction and Control, Great Britain, 1995.

[16]. Smith, M., Neural Networks for Statistical Modelling, Van Nostrand Reindhold, New York, 1993.
[17]. Tseng, F. -M., Yu, H. -C., and Tzeng, G. -H.., Combining neural network model with seasonal time series ARIMA model, Technological Forecasting & Social Change, 69, 2002, 71-87.
[18]. Welstead, S. T., Neural Network and Fuzzy logic applications in C/C++, John Willey and Sons, Canada, 1994.
[19]. Zhang, G. P., Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing, 50, 2003, 159-175.
[20]. Zhang, G. P., and Qi, M., Neural network forecasting for seasonal and trend time series. European Journal of operation Research 160, 2005, 501-514.