em.Int-06-2018-Warsito_2018_J._Phys.-_Conf._Ser._1025_012097.pdf

by

Submission date: 18-Mar-2019 03:36PM (UTC+0700)
Submission ID: 1095233138
File name: em.Int-06-2018-Warsito_2018_J._Phys.-_Conf._Ser._1025_012097.pdf (635.46K)
Word count: 4149
Character count: 21173

Cascade Forward Neural Network for Time Series Prediction

IOP Conf. Series: Journal of Physics: Conf. Series **1025** (2018) 012097 doi:10.1088/1742-6596/1025/1/012097

Cascade Forward Neural Network for Time Series Prediction

Budi Warsito[1], Rukun Santoso[1], Suparti[1], Hasbi Yasin[1]

[1]Department of Statistics, Faculty of Science and Mathematics, Diponegoro University Jl. Prof. Soedharto, SH, Tembalang, Semarang 50275, Indonesia

E-mail: budiwrst2@gmail.com

Abstract. Cascade-forward neural network is a class of neural network which is similar to feed-forward networks, but include a connection from the input and every previous layer to following layers. In a network which has three layers, the output layer is also connected directly with the input layer beside with hidden layer. As with feed-forward networks, a two-or more layer cascade-network can learn any finite input-output relationship arbitrarily well given enough hidden neurons. Cascade-forward neural network can be used for any kind of input to output mapping. The advantage of this method is that it accommodates the nonlinear relationship between input and output by not eliminating the linear relationship between the two. In this study, we apply the network in time series field. The optimal architecture was determined computationally by using incremental search method in both input and hidden units. The simple one was built first, and then the more complex is constructed by adding the units one by one. The optimal one is chosen then by using the mean square error criteria.

1. Introduction

Forecasting is usually based on identifying, modeling, and extrapolating the patterns found in historical data [9]. The changing pattern certainly gives much influence to the precision of a prediction model. The idea of using mathematical models to explain the behavior of phenomena of pattern change has been widely developed. But not all phenomena are entirely deterministic because unknown factors can occur and affect the physical phenomenon. In this case, a time-dependent phenomenon is required in the stochastic models. However, the widely used model is more parametric with many limitations. Neural Network (NN) is a nonparametric model that can be utilized for time series data modeling which does not require assumptions on its residual. Several studies have found that this model yields better prediction accuracy than parametric models. The NN application on seasonal time series prediction models is expected to provide more accurate and robust results against data fluctuations. Because it does not require an assumption test, the main thing to consider is how to get the error as small as possible. Some studies have shown that nonparametric models such as NN are capable of resulting good predictions. One of the classes of the NN model -beside Feed Forward Neural Network (FFNN) model as the main class- is the Cascade Forward Neural Network (CFNN). The main thing that characterizes this class is the direct link between the input layer and the output, in addition to the indirect link through the hidden layer. The only difference of CFNN and FFNN is that each neuron in the input layer is attached to each neuron in the hidden layer and each neuron in the output layer [7]. The advantage of this method is that it accommodates the nonlinear relationship between input and output by not eliminating the linear relationship between the two.

Based on the background, it can be formulated problem that is how to build prediction time series data based on the stochastic (probabilistic) process with a high level of accuracy. ARIMA method that is often used for forecasting have limitations, such as ignoring the possibility of nonlinear relationships between data. The stationarity assumption is also less likely to analyze data with period patterns that vary over time. This often results in less accurate predictions being obtained. The development of CFNN model that is nonlinear-nonparametric and more flexible can be utilized to make predictions of time series data so that more precise results can be obtained. The problem in CFNN modeling is in terms of input selection procedure so that the optimal architecture is obtained.

NN modeling has progressed rapidly along with developments in the computing field. Several studies have been conducted in forecasting of time series data. Some studies that use the NN model for time series prediction are [14] and [8]. Several studies have also been conducted related to NN modeling for seasonal time series [4]. Some of the other studies are [5] and [11]. The special class of the FFNN model is the Cascade Forward Neural Network (CFNN). In this class, there is a direct connection between the input and output besides the indirect relationship through the hidden layer. Some studies related to the CFNN model for time series are [1], [2], [3], [6], [10], [12], and [13]. The results show that the resolution of the NN model is strongly influenced by input determination. Often the selected input is not appropriate or the input data contains noise so the weighting is less precise. In this research, the development of CFNN model focused on modeling procedure. The obtained model is used to predict the time series data from both the generated data and the real data. The developed procedure is tested first on time series data generated from the seasonal model. Repetition is done to determine the stability of the proposed method.

2. Neural Network

Artificial Neural Network (ANN, for simplicity often abbreviated as NN only), is a series consisting of interconnected simple elements called neurons. Each neuron represents a mapping, especially with multiple inputs and a single output. The output of the neuron is a function of the sum of its inputs. The function used in the output of a neuron is called an activation function. The single output of the neuron can be applied as an input to some other neurons, and therefore the symbol for a single neuron shows the number of arrows coming from the neuron. A simple network architecture that contains only the input layer and the output layer is called perceptron. In perceptron, the signal is sent directly from the input layer to the output layer. The relationship between input and output is linear. Signals that are sent from input to output are weighted sums indicating the direct relationship between the two layers.

The addition of layers between input and output makes the neural network contains many layers called Multi-Layer Perceptron (MLP). In the neural network modeling MLP network is also called Feed Forward Neural Network (FFNN). In FFNN the network comes with an additional layer called the hidden layer. The signal is sent from the input layer to the hidden layer in the weighted form. The signals from the input are distributed to the neurons in the hidden layer. Signals that enter the neurons in the hidden layer are then processed by the activation function on the layer. The activation function in the hidden layer is a nonlinear function useful as a transfer function. The output of each neuron in the hidden layer is then sent to the output layer in the weighted sum. The incoming signal to the output layer is then processed by the activation function on this layer. Usually the activation function of the output layer is the identity mapping so that the output obtained in this layer is the same as the incoming signal. Figure 1 shows the MLP with n input x_i, where i = 1, ..., n; one output neuron y; and k neurons in the hidden layer. The output of the neuron in the hidden layer is expressed by z_j, where j=1, ..., k. The input x_1, ..., x_n is distributed to neurons in the hidden layer. The activation function of each neuron in the input layer is the identity mapping. The activation function of the neuron in the hidden layer is expressed with f^h, while the neuron activation function in the output layer is written with f^o. Each activation function is a function of R to R.

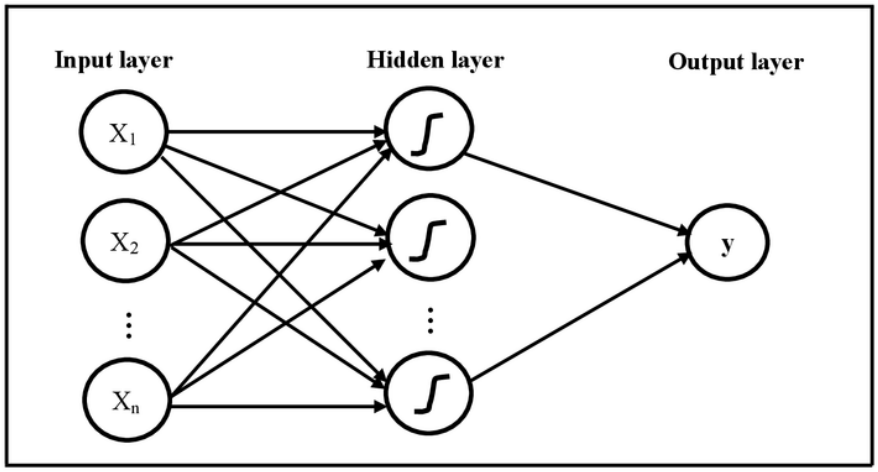


Figure 1. MLP with *n* neuron in input layer, k neuron in hidden layer and one neuron in output layer

The mathematical equation of the architecture in Figure 1 can be written as follows:

$$y = f^o\left(\sum_{j=1}^{k} \omega_j^o f_j^h\left(\sum_{i=1}^{n} \omega_{ji}^h x_i\right)\right) \quad (1)$$

Where f^o is the activation function on the output layer and f_j^h is an activation function on the hidden layer and If a bias is added to the input layer and the activation function of each neuron in the hidden layer is f^h then equation (1) becomes

$$y = f^o\left(\omega^b + \sum_{j=1}^{k} \omega_j^o f^h\left(\omega_j^b + \sum_{i=1}^{n} \omega_{ji}^h x_i\right)\right) \quad (2)$$

Where ω_b is the weight from bias to output and ω_j^b is the weight from bias to hidden layer.

3. Cascade Forward Neural Network

In perceptron connection that is formed between input and output is a form of direct relation while in FFNN connection formed between input and output is indirect relationship. The connection is nonlinear in shape through an activation function in the hidden layer. If the connection form on perceptron and multilayer network is combined, then the network with direct connection between the input layer and the output layer is formed, besides the connection indirectly. The network formed from this connection pattern is called Cascade Forward Neural Network (CFNN). The equations are formed from the CFNN model can be written as follows:

$$y = \sum_{i=1}^{n} f^i \omega_i^i x_i + f^o\left(\sum_{j=1}^{k} \omega_j^o f_j^h\left(\sum_{i=1}^{n} \omega_{ji}^h x_i\right)\right) \quad (3)$$

Where f^i is the activation function from the input layer to the output layer and ω_i^i is weight from the input layer to the output layer. If a bias is added to the input layer and the activation function of each neuron in the hidden layer is f^h then equation (2) becomes

$$y = \sum_{i=1}^{n} f^i \omega_i^i x_i + f^o\left(\omega^b + \sum_{j=1}^{k} \omega_j^o f^h\left(\omega_j^b + \sum_{i=1}^{n} \omega_{ji}^h x_i\right)\right) \quad (4)$$

In this research, the CFNN model is applied in time series data. Thereby, the neurons in the input layer are the lags of time series data X_{t-1}, X_{t-2}, ..., X_{t-p}, whereas the output is the current data X_t. The architecture of CFNN model in predicting time series is shown at Fig. 2.

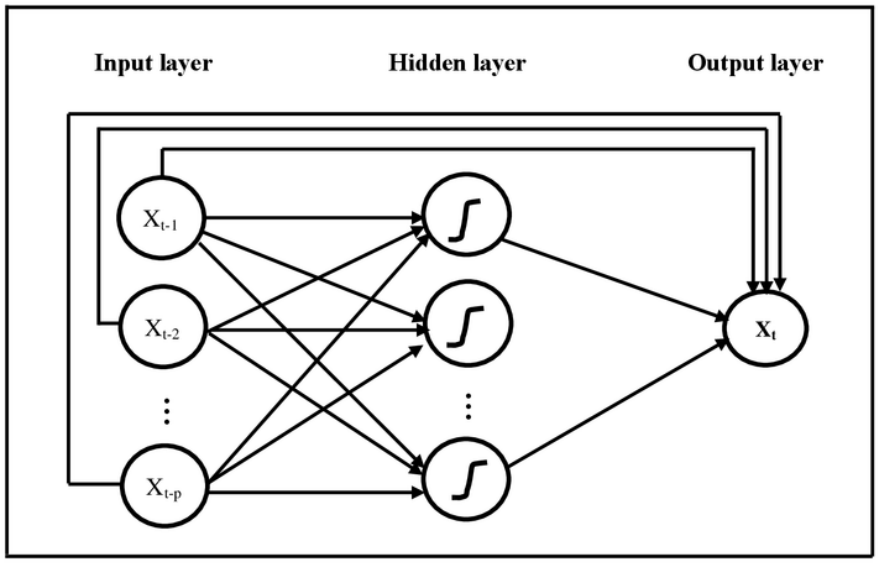


Figure 2. Architecture of CFNN for time series prediction

Figure 2 shows that there is a direct relationship between input and output. The consequence of this form of relationship is that the network weight to be estimated increases as much as the neurons in the input layer. As with FFNN, backpropagation algorithm on CFNN also consists of three stages: feedforward of the input pattern, error counting and weight adjustment. As explained before, after the feedforward stage the process continues with the error calculation (the difference from the output to the target). The next step is to update the weights and do the recalculation. This step is done until no error occurs or until iteration stops according to the specified stop criteria. In this section we briefly discuss the conjugate gradient optimization method for weighting adjustments of the CFNN model.
Suppose that belonging to a weight vector ω of length s is the set of all network weights and the objective function is $e = \frac{1}{2}\left(X_t - \hat{X}_t\right)^2$. Defined Q is the positive definite matrix of size $s \times s$ where $Q^T = Q$. Stages of the algorithm on Conjugate Gradient optimization are described as follows:

- Set k = 0, select the initial point $\Omega^{(0)}$
- Calculate the gradient of the initial weight

  $$g^{(0)} = \frac{\partial e}{\partial \omega^{(0)}} = \frac{\partial e}{\partial \omega}\Big|_{\omega=\omega^{(0)}} = \left[\frac{\partial e}{\partial \omega_1^{(0)}} \quad \cdots \quad \frac{\partial e}{\partial \omega_s^{(0)}}\right]^T$$

  If $g^{(0)} = 0$ then stop, and it obtained the optimal weight = $\Omega^{(0)}$. Else, set $d^{(0)} = g^{(0)}$.
- Calculate $\alpha_k = \arg\min_{\alpha \geq 0} e\left(\omega^{(k)} + \alpha d^{(k)}\right) = -\frac{g^{(k)T} d^{(k)}}{d^{(k)T} Q d^{(k)}}$
- Calculate $\Omega^{(k+1)} = \Omega^{(k)} + \alpha_k d^{(k)}$
- Calculate $g^{(k+1)} = \frac{\partial e}{\partial \omega^{(k+1)}}$, if $g^{(k+1)} = 0$ stop and the optimal weight is $\omega^{(k+1)}$
- Calculate

  $$\beta_k = \frac{g^{(k+1)T} Q d^{(k)}}{d^{(k)T} Q d^{(k)}}$$

- Calculate $d^{(k+1)} = -g^{(k+1)} + \alpha_k d^{(k)}$
- k = k+1; go to step 3

As in FFNN, the iteration process for weight searching on CFNN is usually called the epoch. Suppose that the maximum number of permitted epochs is K, if the iteration termination criterion has not been met until epoch k = K then the iteration process will be stopped. On the optimization problem of the general nonlinear model, this algorithm will not always reach convergent in n steps so that the direction vector is re-initialized after a certain iteration and continued until the termination criterion. In the nonlinear model, matrix Q is a non-constant Hessian matrix and evaluated on each iteration. To avoid complicated calculations, elimination of Q is done so that the algorithm depends only on the function and the gradient value of each iteration. There are several formulas for substituting the form of Qd (k) with other forms, such as the Hestenes-Stiefel formula, i.e the form $Qd^{(k)}$ is replaced by $(g^{k+1} - g^k)/\alpha_k$. By this formula the β_k becomes:

$$\beta_k = \frac{g^{(k+1)T}\left[g^{(k+1)} - g^{(k)}\right]}{d^{(k)T}\left[g^{(k+1)} - g^{(k)}\right]}$$

The obtaining of optimal weights is done in each epoch.

4. Experimental Results in Time Series Data

In this section the CFNN model is applied to time series data. The first data is simulated data generated from the AR(2) model while the second is the real data in the financial field. The simulated data generated is obtained from the following models:

$X_t = 0.65X_{t-1} + 0.25X_{t-2} + \varepsilon_t$ where $\varepsilon_t \sim N(0,1)$

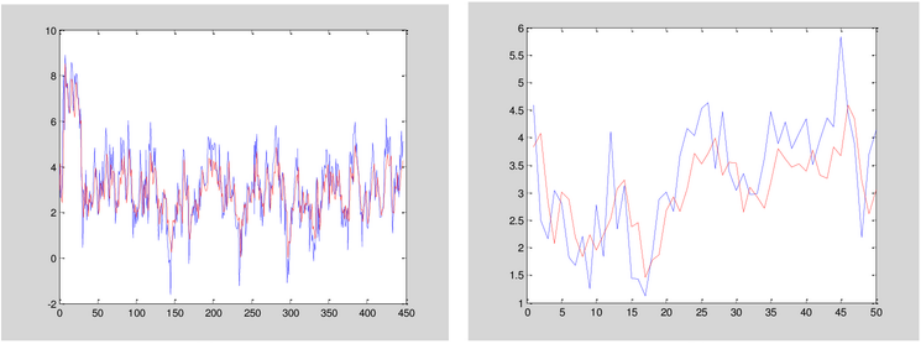
The real data used is monthly palm oil price index data in the Europe market. In each experiment, optimal architecture determination is done as follows. In the first stage, the first lag is entered as the network input. Furthermore, the network architecture with a hidden unit is built. The experiment was repeated 30 times to observe its stability. One hidden unit is added and the experiment is repeated with the same number of loops. This step is done up to ten hidden units and the smallest MSE of the training data is selected. Furthermore, the second lag is added as input so that the two input units is obtained, i.e the first lag and the second lag. The same step for determining the number of hidden unit is done on this architecture. In the first data the input selection is made to the fourth lag with the consideration that the input of the data generated only up to the second lag. The addition of some inputs is intended to be overfitting. Similarly, for the second data the selection of inputs also to the fourth lag with the consideration that the data comes from the financial field. The tendency of this data type is that past data that affect the current is not a long lag. Stability analysis is done by looking at the mean and variance of each architecture. Selection of the best architect is based on the smallest possible MSE and possible stability. The analysis results are shown in the following sections. The data generated from AR(2) model with length 500 is divided into two parts, 450 as training and 50 as testing.

Table 1. The in-sample and out-of-sample predictions of CFNN model of the data generated from AR(2) model

Input Lags	**Optimal hidden unit**	**In-sample prediction**		**Out-of-sample prediction**	
		Mean	Variance	Mean	Variance
1	8	1.117345	0.000967	0.718038	0.000686
1,2	9	1.023401	0.001673	0.695112	0.001043
1,2,3	9	1.000781	0.002449	0.707099	0.002188
1-4	1	1.020410	0.001008	0.693835	0.001129

Table 1 shows the results of each simulation with 30 repetitions. The CFNN model with 1-2 lags, 1-3 lags and 1-4 lags as input provide as good as results in both in-sample prediction and out-of-sample prediction. It can be seen from the MSE value of the three architectures that have similar value. While models with inputs only one lag produce slightly higher prediction errors than others. Similarly, the

value of variance. The three input types produce similar predictive stability. The variance of the model with just first lag as input does produce smaller values, which means that this model provides a more stable prediction, but with a larger MSE in average. Then this model is not recommended to be selected. While three other models have similar results, one of these models should be selected. Taking into account the principle of parsimony, the selected CFNN model is the one with 1-2 lags as input. This also corresponds to the data which is generated from the AR(2) model. Thus, it can be said that the CFNN model succeeds in approaching the empirical model of the data. As illustrated visually in Figures 3(a) and 3(b), plot of actual and the predictions on one repetition is presented for in-sample and out-of-sample, respectively.



(a) (b)

Figure 3. Plot of actual and prediction of (a) in-sample and (b) out-of-sample on data generated from AR(2) model

The plot of Figure 3 shows that the CFNN model provides good results in both in-sample and out-of-sample predictions. This is apparent from predicted results that coincide with the actual. Variations of data pattern can be followed well by the prediction.

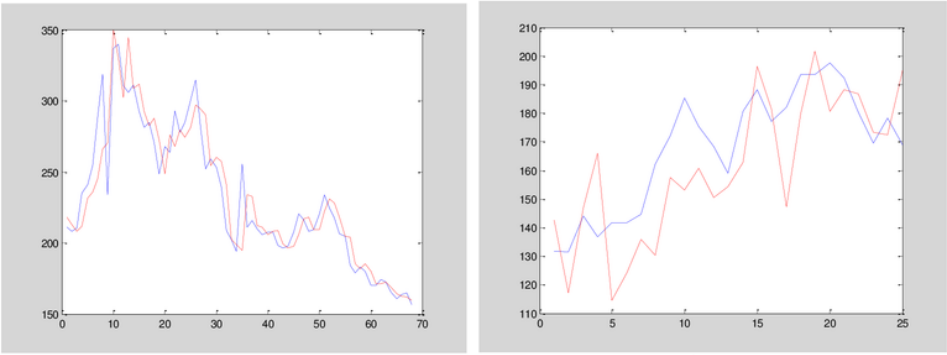
The next analysis is done on the real data, which is the monthly palm oil price index data in the Europe market, from January 2010 up to June 2017. The length of the data is 97. The first 72 data is used as training and the remaining as testing. As in the first experiment, in this second data the experiment was also carried out through 30 repetitions for each architecture. The best result of each input type is compared by the mean and variance of the resulting MSE through 30 repetitions. The in-sample and out-of-sample prediction results for each architecture are shown in Table 2.

Table 2. The in-sample and out-of-sample predictions of CFNN model of the monthly palm oil price index data

Input Lags	Optimal hidden unit	MSE of in-sample prediction		MSE of out-of-sample prediction	
		Mean	Variance (x 10^6)	Mean	Variance (x 10^6)
1	2	501.40010	0.00807	251.66719	0.21998
1,2	1	747.43785	2.14337	233.65334	0.01664
1,2,3	1	578.55679	0.11531	372.60853	0.33587
1-4	1	540.57820	0.02230	238.52723	0.02085

Table 2 shows that CFNN models with 1-2 lags as inputs produce poor in-sample predictions, whereas models with 1-3 lags as input produce poorly out-of-sample predictions. Therefore, both models are not recommended to be selected. At the in-sample data, architecture with one lag as input succeeded in generating the smallest average of MSE of the other models, as well as the variance. These results

indicate that the in-sample prediction from this model is the most precise and the most stable. The mean and variance of MSE of in-sample predictions of models with 1-4 lags input is smaller, but not as large as the difference in the in-sample prediction results. Thus, also with consideration of parsimony the model with one lag as input is preferred to choose. Figures 4 (a) and 2 (b) illustrate the plots of original data and predictions on one repetition, each for in-sample and out-of-sample data.



(a) (b)

Figure 4. Plot of actual data and prediction of (a) in-sample and (b) out-of-sample for the monthly palm oil price index data

As in the previous experiment, Figure 2 shows that the CFNN model successfully predicted well. In the plot of in-sample predictions, actual data and predictions produce a coinciding value. In-sample predictive values differ significantly from real data to only a few observation points. Similarly, in the out-of-sample predictions, the pattern of the actual data can be followed well by the predictions.

5. Conclusion

The Cascade Forward Neural Network model has been applied in time series data, both simulated data generated from AR(2) model and real data of the monthly palm oil price index data. CFNN modeling procedures have been developed for the selection of input units and hidden units to obtain the optimal network architecture based on the accuracy of the resulting predictions. Repetition is done on each experiment to get a model that produces the most accurate and most stable predictions. Experimental results show that the built CFNN model has successfully predicted both types of data. The proposed modeling procedure can also be developed for various other time series data types. Developing model can be done through the extension of the seasonal lag as the input candidate. A certain subset of the candidate or a partial lag can be selected as the chosen input from all of the candidates. Development of modeling procedures can also be done through the use of k-fold time series as the evaluation criteria for selecting the best architecture.

References

[1] Allaf ONA 2012 Cascade-Forward vs. Function Fitting Neural Network for Improving Image Quality and Learning Time in Image Compression System, *Proceedings of the World Congress on Engineering* Vol II WCE 2012, July 4 - 6, 2012, London, U.K.

[2] Badde DS, Gupta AK and Patki VK 2012 Cascade and Feed Forward Back propagation Artificial Neural Network Models for Prediction of Compressive Strength of Ready Mix Concrete, *IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE)* ISSN: 2278-1684, 01-06

[3] Bolanca T, Stefanovic SC, Ukic S, Rogosic M 2009 Development of Temperature Dependent Retention Models in Ion Chromatography by the Cascade Forward and Back Propagation

Artificial Neural Networks, *Journal of Chromatography and Related Technologies*, 32: 2765–2778

[4] Devi SR, Arulmozhivarman P, Venkatesh C and Agarwal P 2016 Performance Comparison of Artificial Neural Network Models for Daily Rainfall Prediction, *International Journal of Automation and Computing* 13(**5**), October 2016, 417-427

[5] Ghiassi M, Zimbra DK and Saidane H 2006 Medium term system load forecasting with a dynamic artificial neural network model. *Electric Power Systems Research*, Volume 76, Issue 5, March 2006, 302-316. a. ISSN 0378-7796, DOI: 10.1016/j.epsr.2005.06.010.

[6] Hedayat A, Davilu H, Barfrosh AA, Sepanloo S 2009 Estimation of research reactor core parameters using cascade feed forward artificial neural networks, *Progress in Nuclear Energy* 51, 709–718

[7] Karaca Y 2016 Case Study on Artificial Neural Networks and Applications, Applied Mathematical Sciences, 10, 45, 2225 - 2237

[8] Kumar N, Middey A, Rao PS 2017 Prediction and examination of seasonal variation of ozone with meteorological parameter through artificial neural network at NEERI, Nagpur, India, *Urban Climate*, 20, 148-167

[9] Montgomery DC, Jennings CL, Kulahci M 2008 *Introduction to Time Series Analysis and Forecasting*, John Wiley & Sons, Inc.

[10] Narad S. And Chavan P 2016 Cascade Forward Back-propagation Neural Network Based Group Authentication Using (n, n) Secret Sharing Scheme, *Procedia Computer Science*, 78, 185 – 191

[11] Nigrini LB 2012 Developing a Neural Network Model to Predict the Electrical Load Demand in the Mangaung Municipal Area, *thesis in the School of Electrical and Computer Systems Engineering*, Faculty of Engineering and Information Technology, Central University of Technology, Free State

[12] Pwasong A and Sathasivam S 2016 A new hybrid quadratic regression and cascade forward backpropagation neural network, *Neurocomputing* 182, 197-209

[13] Tengeleng S and Armand N 2014 Performance of Using Cascade Forward Back Propagation Neural Networks for Estimating Rain Parameters with Rain Drop Size Distribution, *Atmosphere* ***5***, 454-472

[14] Zhang JP, Qi M 2005 Neural network forecasting for seasonal and trend time series, *European Journal of Operational Research*, 160, 501–514

em.Int-06-2018-Warsito_2018_J._Phys.-_Conf._Ser._1025_012097.pdf

ORIGINALITY REPORT

3%	3%	5%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Chong, . "Unconstrained Optimization and Neural Networks", An Introduction to Optimization Chong/An Introduction, 2011. Publication	3%

Exclude quotes	On	Exclude matches	< 3%
Exclude bibliography	On		