Forecasting the Behavior of Gas Furnace Multivariate Time Series Using Ridge Polynomial Based Neural Network Models

Waddah Waheeb*, Rozaida Ghazali

Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor (Malaysia)

Received 13 October 2018 | Accepted 22 April 2019 | Published 26 April 2019

unir LA UNIVERSIDAD EN INTERNET

Abstract

In this paper, a new application of ridge polynomial based neural network models in multivariate time series forecasting is presented. The existing ridge polynomial based neural network models can be grouped into two groups. Group A consists of models that use only autoregressive inputs, whereas Group B consists of models that use autoregressive and moving-average (i.e., error feedback) inputs. The well-known Box-Jenkins gas furnace multivariate time series was used in the forecasting comparison between the two groups. Simulation results show that the models in Group B achieve significant forecasting performance as compared to the models in Group A. Therefore, the Box-Jenkins gas furnace data can be modeled better using neural networks when error feedback is used.

Keywords

Error Feedback, Nonlinear Autoregressive Moving-Average Model, Multivariate Time Series, Recurrent Network.

DOI: 10.9781/ijimai.2019.04.004

I. Introduction

TIME series data are everywhere. Examples of time series data are the daily weather temperature, monthly sales, and annual milk production.

Time series can be categorized into univariate and multivariate time series. Univariate time series can be obtained by recording a single phenomenon over time, for example, recording air pollution concentration. Multivariate time series, on the other hand, can be obtained by recording more than one phenomenon over time, for example, recording air pollution concentration and some weather information.

Multivariate time series can be found in many real problems such as in business and engineering domains. The importance of having multivariate time series is to take advantage of the available additional information from related time series in order to enhance the forecasting accuracy for each series individually.

Various models have been used for multivariate time series forecasting such as vector autoregression models, adaptive network-based fuzzy inference system, belief rule based system, neural networks and other hybrid models [1]-[4].

Neural networks (NNs) have been applied extensively in time series forecasting due to some distinguishing features as reported in [5]. These reported features are the universal function approximator property in NNs that allow them to approximate any continuous function with an arbitrary degree of accuracy. Furthermore, NNs are a nonlinear data-driven method with few a priori assumptions about underlying models. In addition, their generalization ability is good.

* Corresponding author.

E-mail address: waddah.waheeb@gmail.com

Higher order neural network (HONN) is a type of NNs that utilizes high order terms (e.g., multiplicative units) besides the commonly used summing units. HONNs are simple in their architecture and have fewer trainable parameters to deliver the input-output mappings as compared to multilayered NNs.

An example of a HONN is the feedforward ridge polynomial neural network (RPNN) [6]. RPNN has good mapping capabilities and it utilizes univariate polynomials that are easy to handle, unlike other HONNs which suffer an explosion of free parameters due to the use of multivariate polynomials [6]. Furthermore, the RPNN provides a more efficient and regular architecture compared to the ordinary HONNs while maintaining their fast learning property [6].

Based on the structure of the RPNN model, there are three recurrent neural networks, namely the dynamic ridge polynomial neural network (DRPNN) [7]-[8], the ridge polynomial neural network with error feedback (RPNN-EF) [9], and the ridge polynomial neural network with error and output feedbacks (RPNN-EOF) [10]. The DRPNN has a feedback based on its network output as an additional input to the network, whereas the feedback in the RPNN-EF is its network error. The RPNN-EOF has both feedbacks.

The feedforward RPNN and the three recurrent networks can be grouped into two groups. The first group, which is referred in this paper as Group A, contains the RPNN and DRPNN that use only autoregressive inputs (i.e., lagged variables of one or more time series), whereas Group B contains the RPNN-EF and RPNN-EOF that use autoregressive and moving-average (i.e., error feedback) inputs.

Based on studies about ridge polynomial based neural network models retrieved in this paper, the four existing ridge polynomial based neural network models have not yet been applied in multivariate time series forecasting.

The well-known multivariate time series called Box-Jenkins gas furnace time series was used in this paper [3]. This time series has been frequently used for the assessment of new identification and modeling techniques. A combustion process of a methane-air mixture was used to record this series. The gas flow rate was kept constant, but the methane rate was randomly changed. The resulting carbon dioxide concentration (CO_2) in the output gases was measured.

The main objective of this paper is to investigate and compare the forecasting efficiency of neural network models in the two groups in forecasting this well-known multivariate time series. To the best to our knowledge, this is the first study that has attempted to achieve this objective. This is important to investigate the difference in the forecasting ability between the two groups. Also, a comparison with some models reported in the literature with this time series was reported.

We organized our paper as follows. The neural network models used in this paper are described in Section II. Section III discusses the methodology used to conduct the experiments. The findings are presented and discussed in Section IV, and in Section V the conclusion and future works are given.

II. Related Works

Neural networks provide a promising alternative tool for forecasters [11]. Numerous applications have been reported on using different types of neural networks to model time series. A basic example to show how a NN is used to learn the nonstationary time series is shown in Fig. 1. Simply, during training, the past time series (i.e., lagged variables) are fed to the neural network. Network output (i.e., forecast) based on the given inputs is produced then compared with the desired output to calculate the error. This error is used to update the network parameters until a specified goal is attained.

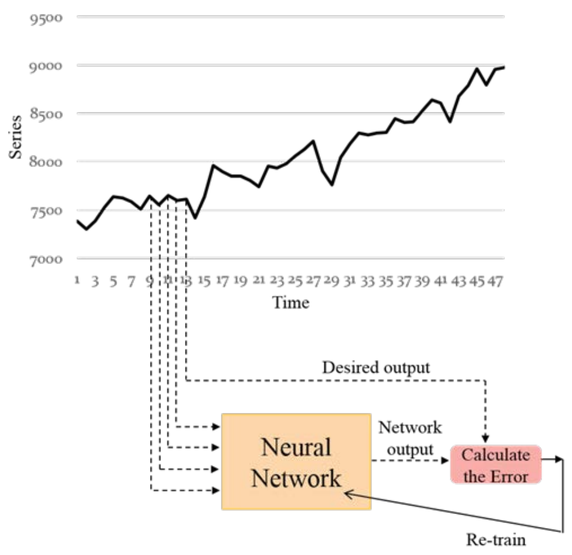

Fig. 1. Learning time series with a neural network.

The RPNN is a type of neural network that has only one layer of adjustable weights, and it uses the pi-sigma neural network (PSNN) as basic building blocks [6]. According to [6], the RPNN utilizes a constructive learning algorithm which allows it to automatically determine the necessary network order. For a given problem, the RPNN starts with a small order of the PSNN block(s), and then an extra block of higher orders is added during the learning process. The learning process stops when the desired level of approximation error is attained, or when the maximum number of epochs or the appropriate order of the network has been reached [6]-[9].

According to [6], the RPNN has the ability to uniformly approximate any continuous function on a compact set in a multidimensional input space with arbitrary degree of accuracy. Moreover, the RPNN utilizes univariate polynomials which are easy to handle, thus it avoids the explosion of the number of trainable parameters as the number of inputs increases, which happens with some neural networks [6]. Fig. 2 shows an example of the RPNN. It can be seen that the weights linking the hidden and the output units are fixed to unity. The inputs to the RPNN are *only* the lagged variables from the time series.

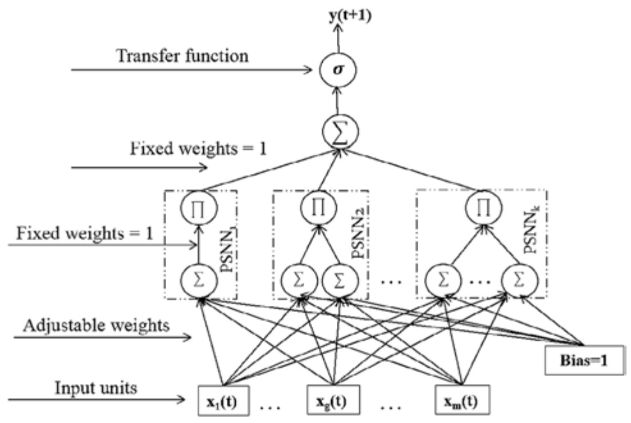

Fig. 2. Architecture of the ridge polynomial neural network [6].

Like any feedforward network, the output of the RPNN is only a function of the current input; in other words, the node equations are memoryless. Therefore, the dynamic RPNN which is a recurrent version of the RPNN was proposed [7]. The DRPNN uses its network output value as an additional input to the input layer. Therefore, it is provided with memory that helps to retain information to be used later to affect the processing of future inputs [7]. Fig. 3 shows an example of the DRPNN.

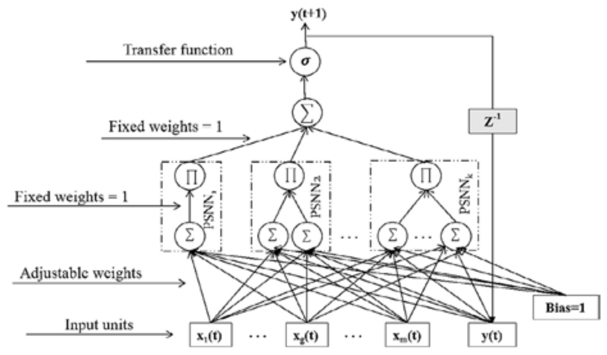

Fig. 3. Architecture of the dynamic ridge polynomial neural network [7].

Due to the existence of the recurrent feedback, learning instability problem could occur in the DRPNN. Therefore, the learning rate was controlled by a Lyapunov function [8]. This solution improves the forecasting accuracy of the DRPNN and reduces its training time. In this paper, we used the DRPNN that uses Lyapunov function and referred to it as DRPNN.

The inputs of the DRPNN is given as follows:

$$z_g(t) = \begin{cases} x_g(t) & 1 \le g \le m \\ y(t) & g = m+1 \end{cases} \qquad (1)$$

where $x_g(t)$ is the lagged variables from the time series, m is the number of past lagged values from the time series, and $y(t)$ is network output at time t.

Both the RPNN and DRPNN use only autoregressive inputs (i.e., lagged variables of one or more time series). However, an alternative to autoregressive modeling has been suggested by feeding back network error to the input layer of a neural network [9]-[10], [12]-[13] in order to model the nonlinear moving-average processes more directly and parsimoniously [13].

The RPNN-EF [9] is a network based on the RPNN and is incorporated with network error as shown in Fig. 4. As shown in Fig. 4, the error term is calculated by subtracting the target from network output (i.e., forecast).

The RPNN-EF was used to forecast univariate time series [9]. It was found that the RPNN-EF showed better forecasting performance with respect to the RPNN and DRPNN using four univariate time series. Furthermore, the forecasting performance of the RPNN-EF was found better than some existing techniques with some benchmark time series.

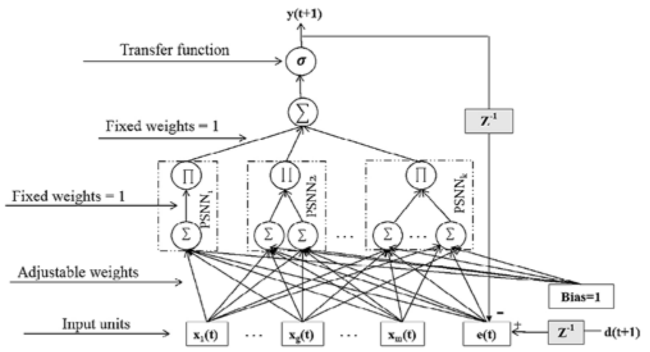

Fig. 4. Architecture of the ridge polynomial neural network with error feedback [9].

The inputs to the RPNN-EF are given as follows:

$$z_g(t) = \begin{cases} x_g(t) & 1 \le g \le m \\ e(t) = d(t) - y(t) & g = m+1 \end{cases} \tag{2}$$

where $x_g(t)$ is the lagged variables from the time series, m is the number of past lagged values from the time series, $e(t)$ is the error, $y(t)$ is the network output, and $d(t)$ is the desired output.

The RPNN-EOF [10] is a network based on the DRPNN and is incorporated with network error as shown in Fig. 5. The inputs for the RPNN-EOF is given by the following equation:

$$z_g(t) = \begin{cases} x_g(t) & 1 \le g \le m \\ e(t) = d(t) - y(t) & g = m+1 \\ y(t) & g = m+2 \end{cases} \tag{3}$$

where $x_g(t)$ is the lagged variables from the time series, m is the number of past lagged values from the time series, $e(t)$ is the error, $y(t)$ is network output, and $d(t)$ is the desired output.

The RPNN-EOF was evaluated using the Mackey-Glass differential delay equation time series [10]. Based on the findings in [10], the RPNN-EOF produced smaller error as compared to some state-of-the-art models.

The ridge polynomial based neural network models have been utilized in different applications as tabulated in Table I. The feedforward RPNN was used in various applications, whereas the three recurrent networks were used only for time series forecasting. With regard to time series forecasting, all the studies in Table I were applied to forecast only univariate time series [7]-[10], [14], [19]. Based on [7], [9], and [14], the recurrent RPNN models are better than the feedforward RPNN model for univariate time series forecasting. Moreover, the RPNN-EF is better than those without error feedback (i.e., RPNN and DRPNN) on average [9].

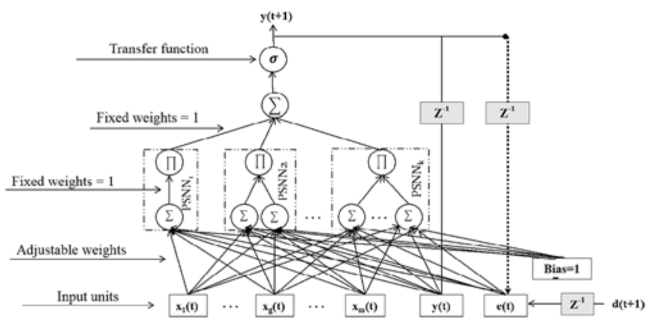

Fig. 5. Architecture of the ridge polynomial neural network with error-output feedbacks [10].

Based on the aforementioned, there is a scope to do further research for applying ridge polynomial based neural network models for multivariate time series forecasting. Therefore, the main objective of this paper is to apply and compare the forecasting ability of two groups of neural networks in multivariate time series forecasting: neural network models that use only autoregressive inputs (i.e., RPNN and DRPNN), simply referred to as Group A; and neural network models that use autoregressive and moving-average (i.e., RPNN-EF and RPNN-EOF) inputs, simply referred to as Group B.

III. Methodology

A. Box-Jenkins Gas Furnace Data

It is one of the well-known and frequently used benchmark problems [3] that has been frequently used for the assessment of new identification and modeling techniques. This time series consists of 296 records and was downloaded from the following website http://www.stat.purdue.edu/~chong/stat520/bjr-data/gas-furnace. A combustion process of a methane-air mixture was used to record this series. The gas flow rate was kept constant, but the methane rate was randomly changed. The resulting carbon dioxide concentration (CO_2) in the output gases was measured. The sampling interval is 9 seconds.

There are different fitting models used by researchers. In this paper, the following fitting model was chosen to compare our findings with [1] and [2]:

$$y(t) = F(v(t-4), y(t-1)) \tag{4}$$

where $v(t)$ is the methane gas flow into the furnace, and $y(t)$ is the CO_2 concentration in the outlet gas.

Based on (4), there are 292 input/output data pairs. By following [1] and [2], these pairs were partitioned in 200 and 92 pairs for training and out-of-sample sets, respectively. Fig. 6 and 7 show the training and out-of-sample values of the CO2 concentration $y(t)$.

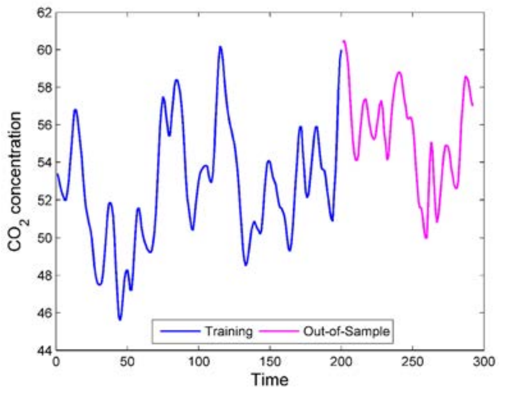

Fig. 6. CO_2 concentration time series for the Box-Jenkins gas furnace data.

TABLE I. Studies Applied The Ridge Polynomial Based Neural Network Models

Study	Model	Inputs	Application
[6]	RPNN	Autoregressive	Multivariate function approximation, realization of multivariate polynomial, and data classification
[15]			Image compression
[16]			OFDM (Orthogonal Frequency Division Multiplexing) systems signals as a nonlinear compensator
[17]			Character recognition
[18]			Modeling nonlinear chemical kinetics
[20]			Microwave characterization of dielectric materials
[21]			Data classification
[19]			Univariate time series forecasting
[7]-[8], [14]	Dynamic RPNN	Autoregressive	
[9]	RPNN with error feedback	Autoregressive & Moving-average	
[10]	RPNN with error-output feedbacks		

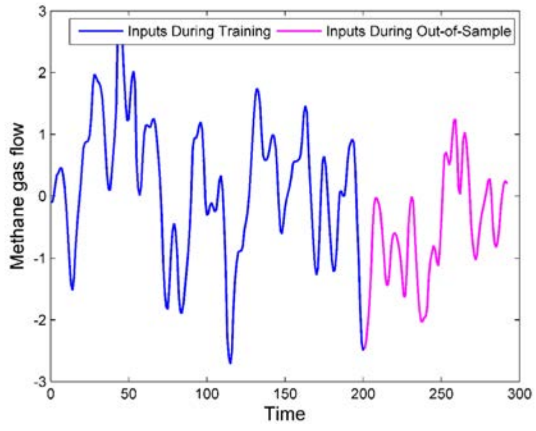

Fig. 7. Methane gas flow time series for the Box-Jenkins gas furnace data.

The time series were scaled to the range [0.2 - 0.8] by using the minimum and maximum normalization method which is given by:

$$\hat{y} = (\max_2 - \min_2) * \left(\frac{y - \min_1}{\max_1 - \min_1} \right) + \min_2 \tag{5}$$

where min_1 and max_1 are the respective minimum and maximum values of all observations, min_2 and max_2 refer to the desired minimum and maximum of the new scaled series, y refers to the original value, and $\hat{y}$ is the normalized version of y. The min_1 and max_1 for the methane gas flow into the furnace are equal to -2.716 and 2.834, respectively. Similarly, the min_1 and max_1 for the CO2 concentration in the outlet gas are equal to 45.6 and 60.5, respectively.

B. Network Topology and Parameters' Settings

Network structure and the values used to train the four models are shown in Table II. These settings are based on previous works using these models for time series forecasting [8]-[10].

C. Performance Metric

Root mean squared error (RMSE) metric was used in this paper. The equation for the RMSE metric is given by:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{6}$$

where N, y and $\hat{y}$ represent the number of test pairs, actual output and network output, respectively.

TABLE II. Network Topology and Training for the Ridge Polynomial Based Neural Network Models

Setting	Value
Initial weights	[-0.5,0.5]
Network order	Incrementally grown from 1 to 5.
Stopping criteria	Minimum Squared Error = 0.000001 or after accomplishing the 5th order network learning or reaching the maximum number of epochs = 3000. Extra criterion for the DRPNN, RPNN-EF and RPNN-EOF: network becomes unstable.
Learning rate (n)	[0.01-1]
Decreasing factors for n	0.8
Momentum	[0.4-0.8]
Threshold of successive PSNN addition (r)	[0.00001-0.1]
Decreasing factors for r	[0.05, 0.2]

D. Model Selection

As mentioned before, the data were partitioned into two sets only; training and out-of-sample sets as used in [1] and [2]. In order to evaluate the forecasting performance of the four models, the adjustable parameters after finishing the training of each network's order were selected for generalization purpose (i.e., out-of-sample forecasting) [7]. Then, the training is continued with increasing network's order, and this continues until the stopping criteria are met [7]. Network structure with the lowest RMSE on the out-of-sample set is considered the best model.

E. Wilcoxon Sign-rank Test

In order to know if there is any significance in the forecasting performance, the Wilcoxon sign-rank test [22] was used. This test is a nonparametric and distribution free test, thus it is statistically safer and more robust than parametric tests [23]. The Wilcoxon signed-rank test was conducted using IBM SPSS software.

F. Comparison with Other Forecasting Models

In this paper, we compared the forecasting performance of the four models from the two groups with other forecasting models reported in [1] and [2]. In addition, we used the well-known multilayer feed-forward network model (MLP) and extreme learning machines (ELM) [24] models in the comparison.

The function *nnetar* in the R package 'forecast' [25] was used to build the MLP model. The number of hidden neurons are selected between 1 and 25. For that, 40 simulations for each hidden neuron size were conducted, yielding 1000 total simulations. The data is scaled by subtracting the column means and dividing by their respective standard deviations because it gives better forecasting performance based on our experiments.

To build the ELM model, we used the function *elm* in the R package ‘nnfor’ [26]. We used a multiplication of 10 to 100 for the number of hidden nodes as well as the available option to be determined automatically by the algorithm. Ridge regression with cross validation was used as an estimation type for output layer weights because it gives better forecasting accuracy as compared to the other three types based on our experiments. Direct input-output connections to model strictly linear effects was used. We ran 1000 simulations using the ELM model.

IV. Results and Discussion

This section presents one-step forecasts comparison between two neural network models that use only autoregressive inputs (i.e., Group A: RPNN and DRPNN) and two neural network models that use autoregressive and moving-average inputs (i.e., Group B: RPNN-EF and RPNN-EOF) in forecasting the Box-Jenkins gas furnace time series.

After finishing the training of each network’s order with 225 different settings, the final values for the trainable parameters were selected for generalization purpose.

Among 1,125 simulations, the average RMSE results of best 20, 50 and 100 simulations for each model are shown in Table III. In addition, the results of the Wilcoxon sign-rank test are shown in Table IV.

TABLE III. Average of Best Top Simulation Results Comparison Between Neural Network Models Using the Rmse (Based on the Normalized Data)

No. Simulations	Group A		Group B	
	RPNN	DRPNN	RPNN-EF	RPNN-EOF
20	0.0214	0.0207	**0.0159**	0.0165
50	0.0231	0.0215	**0.0173**	0.0176
100	0.0245	0.0225	**0.0182**	**0.0182**

Best results are in **boldface**.

TABLE IV. Results of the Wilcoxon Sign Rank Test

No. Simulations	20	50	100
RPNN-EF vs. RPNN	-3.920 ($p < 0.000089$)	-6.154 ($p < 0.000000$)	-8.682 ($p < 0.000000$)
RPNN-EOF vs. DRPNN	-3.920 ($p < 0.000089$)	-6.154 ($p < 0.000000$)	-8.682 ($p < 0.000000$)
DRPNN vs. RPNN	-3.920 ($p < 0.000089$)	-6.154 ($p < 0.000000$)	-8.682 ($p < 0.000000$)
RPNN-EOF vs. RPNN	-3.920 ($p < 0.000089$)	-6.154 ($p < 0.000000$)	-8.682 ($p < 0.000000$)

As shown in Table III and Table IV, the models in Group B produce significantly better forecasting performance as compared to the models in Group A in all cases. In other words, the RPNN-EF is significantly better than the RPNN, and the RPNN-EOF is significantly better than the DRPNN. Based on that, it can be concluded that incorporating error feedback to neural network models makes the models more suitable to deal with this time series. This is because the models in Group B have autoregressive and moving-average inputs, unlike the models in Group A that have only autoregressive inputs. Since an approach to identify the component of a non-linear model similar to the Box-Jenkins identification approach, which is used for linear model identification, is still not available, it is difficult to prove that the time series possess a non-linear moving-average component. However, it was found that neural networks with error feedback are more suitable than neural networks without error feedback in modelling univariate time series that possess a moving-average component [12]- [13].

We observe from Table III and Table IV that all recurrent models achieve significant results as compared to the feedforward RPNN that produced the highest forecasting error. Moreover, the RPNN-EF has the best average forecasting performance.

The best simulation results for the four models in the two groups are shown in Table V. In addition, the best results obtained by the MLP and ELM models are shown in Table V. For fair comparison with the models reported in [1] and [2], all the results in Table V are after the de-normalization (i.e., returned to the original data scale). Even though the models in Table V have different structures as compared to the four models, it is widely accepted to compare the final results with those reported since all the models use the same model mentioned in (4), the training and out-of-sample sets are identical, and the reported results are in the original scale. It is good to note that the best model using the MLP is with 14 hidden neurons while the best model for the ELM used 100 hidden neurons.

TABLE V. Comparison of the Performance of Various Existing Models (Based on the De-Normalized Data)

Model	RMSE
MLP	0.9098
ELM	0.6736
RPNN	0.4920
DRPNN	0.4804
Belief rule based (BRB) system with the number of 5 * 5 referential values [1]	0.4616[a]
Adaptive network-based fuzzy inference system (ANFIS) [2]	0.4053[a]
Pseudo-Gaussian basis function network (PG-BF) [2]	0.3962[a]
RPNN-EF	0.3652
RPNN-EOF	0.3617
Hybrid neural fuzzy inference system (HyFIS) [2]	0.3074[a]
Generalized fuzzy neural network (G-FNN) [2]	0.2728[a]

[a] The original work reported the results in MSE

It can be seen from Table V that among the neural network models used in this paper, the RPNN-EOF produces the best simulation. Similar to the results shown in Table III, the models in Group B produce the better forecasting performance as compared to the models in Group A. The recurrent models also produce better results as compared to the feedforward model.

Moreover, as tabulated in Table V, the RPNN-EF and RPNN-EOF outperform hybrid models such as the MLP, ELM, BRB, ANFIS and PG-BF models. However, the hybrid HyFIS and G-FNN models outperform the RPNN-EF and RPNN-EOF. Therefore, using more than one error feedback in the RPNN-EF and RPNN-EOF could help to improve their forecasting performance.

Regarding the number of final parameters (i.e., weights and biases), the minimum number of parameters is with the RPNN where it equals 18 parameters. The maximum number of parameters is 60 with the DRPNN. The RPNN-EF and RPNN-EOF have 24 and 30 parameters, respectively.

The learning curve for these best models are shown in Fig. 8 - Fig. 11. It can be seen from these figures that the MSE per epoch decreases monotonically. The fastest training model based on these figures is the RPNN-EF with less than 70 epochs.

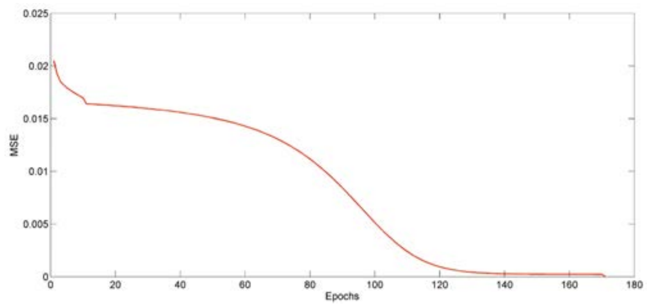

Fig. 8. Learning curve for the RPNN.

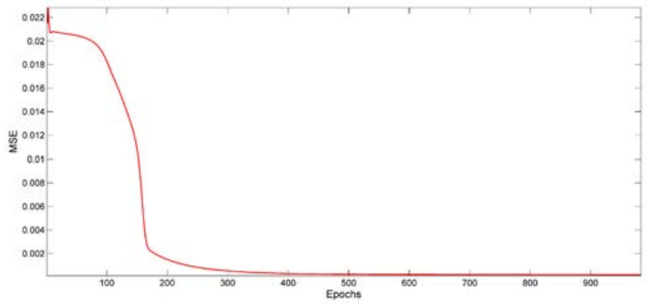

Fig. 9. Learning curve for the DRPNN.

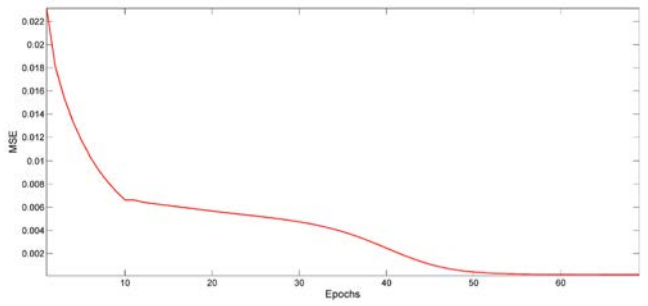

Fig. 10. Learning curve for the RPNN-EF.

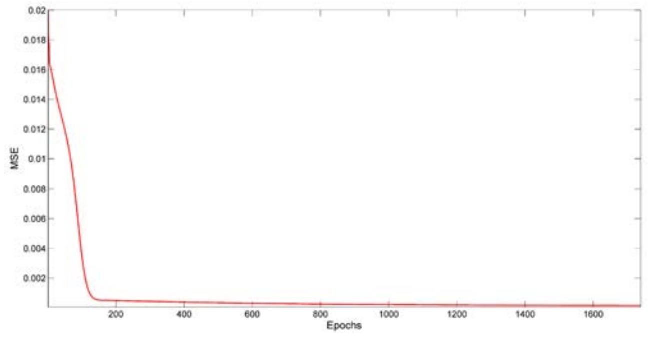

Fig. 11. Learning curve for the RPNN-EOF.

Fig. 12 and Fig. 13 show the best out-of-sample forecasting result using the four models. The idea behind dividing the results into two figures is to show the difference in the forecasting ability between the model with and without error feedback. Therefore, the RPNN is drawn with the RPNN-EF in Fig 12, whereas the DRPNN is drawn with the RPNN-EOF in Fig. 13. It is clear that all models are able to produce forecasts close to the targets in most cases. However, the RPNN-EF and RPNN-EOF have better ability to follow the peaks and troughs as compared to the RPNN and DRPNN, respectively.

The best simulation performance among the four models is achieved using the RPNN-EOF model as shown in Table V. This model is as follows:

$$Eq1 = 0.004471 * x1 + 0.201585 * x2 - 0.33254 * e(t) - 0.325995 * y(t) - 0.088927 \tag{7}$$

$$Eq2 = (-0.410376 * x1 - 0.100439 * x2 - 0.223905 * e(t) + 0.127541 * y(t) + 0.411272) * (-0.279539 * x1 + 0.167540 * x2 - 0.299313 * e(t) - 0.399306 * y(t) + 0.072087) \tag{8}$$

$$Eq3 = (-0.641134 * x1 - 0.585069 * x2 - 0.078077 * e(t) - 0.458598 * y(t) - 0.941347) * (-0.788508 * x1 - 0.510193 * x2 - 0.308327 * e(t) - 0.171573 * y(t) - 0.891403) * (-0.720051 * x1 + 0.952238 * x2 + 0.868305 * e(t) - 0.064836 * y(t) - 0.021427) \tag{9}$$

$$y(t+1) = \mathrm{Sigmoid}(Eq1 + Eq2 + Eq3) \tag{10}$$

where x_1 is the methane gas flow into the furnace, x_2 is the CO_2 concentration in the outlet gas, *e(t)* is the past network error, and *y(t)* is the past network output. The inputs should be normalized, as explained in Eq. (5). The *e(t)* and *y(t)* to be used with the first out-of-sample sample are -0.001954 and 0.781819, respectively. It is worth noting that the numbers (i.e., weights) shown in Eq. (6) – Eq. (9) are rounded to 6 decimal places.

The main contribution of this paper is that it shows how good the forecasting performance is when error feedback is incorporated into the structure of the neural networks as compared to their counterparts without error feedback when forecasting the Box-Jenkins gas furnace data. This better forecasting performance can be attributed to the direct modelling of the non-linear moving-average component via error feedback.

The limitation of the study is that we conducted only one-step forecasts comparison. However, to the best of our knowledge, neural network models available in the literature that use error feedback have not yet tested for recursive multi-step forecasting. That is because the error cannot be observed until the real value for the current time is available. However, the unobserved errors can be replaced with zeros as in the autoregressive and moving-average (ARMA) model, or other solutions can be further investigated for recursive multi-step forecasting. Another limitation is the comparison with the MLP and ELM models. There are no options to select different ranges for the learning parameters such as learning rate with both functions used. Therefore, 1000 simulations were conducted to find best performance for both models.

V. Conclusion

In this paper, one feedforward and three recurrent neural networks based on the ridge polynomial neural network were used to forecast the well-known Box-Jenkins gas furnace multivariate time series. These four models are grouped into two groups. Group A consists of models that use only autoregressive inputs, whereas Group B consists of models that use autoregressive and moving-average (i.e., error feedback) inputs. Overall, the following points can be concluded from the obtained results:

- The models in Group B achieve better and more significant forecasting performance as compared to the models in Group A in forecasting this time series.
- The recurrent models are better and more significant than the feedforward RPNN model to forecast this time series.
- The significance in the forecasting performance for Group B models reveals that the time series could possess moving-average component. Therefore, the used error feedback with the models helps to better modeling the series.

In future, more multivariate time series could be explored with the models in the two groups. Incorporating more loops into the recurrent

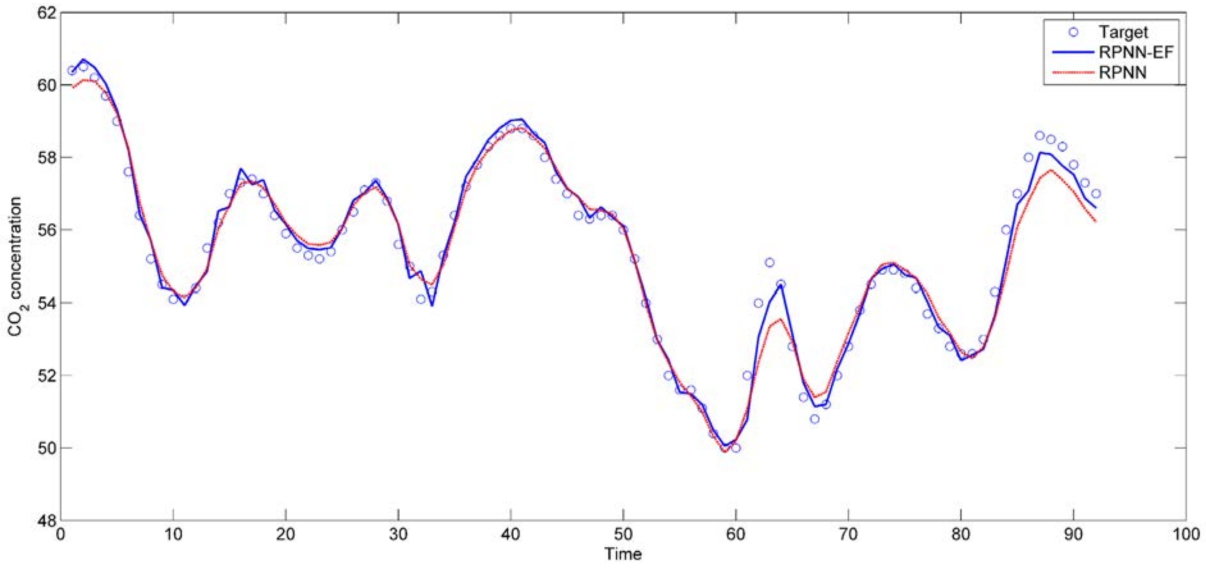

Fig. 12. The best out-of-sample forecasting using the RPNN and RPNN-EF.

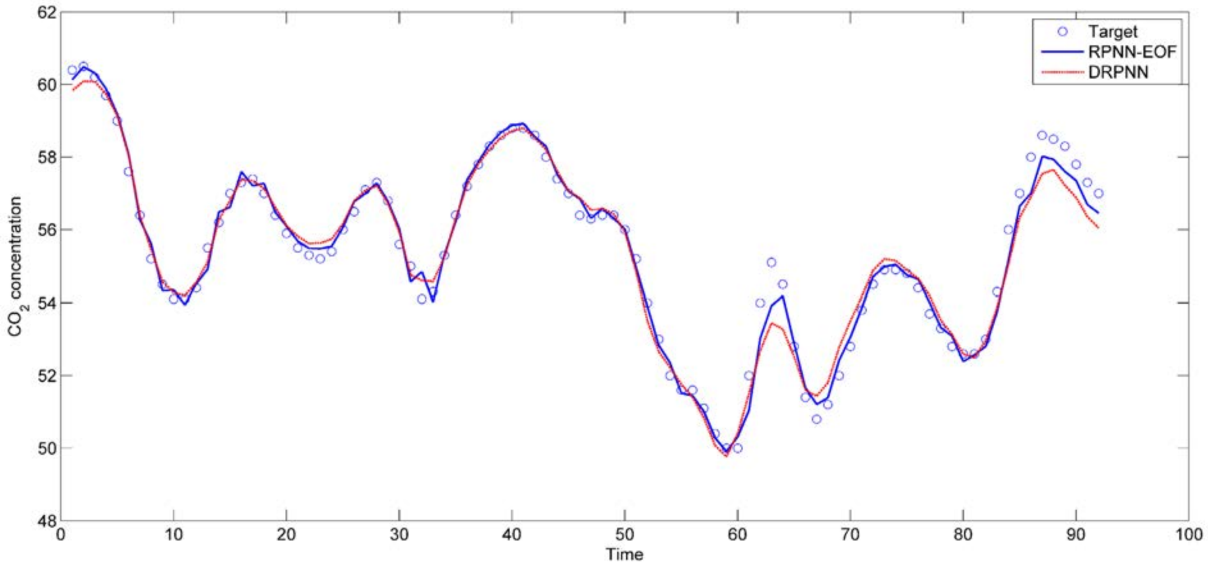

Fig. 13. The best out-of-sample forecasting using the DRPNN and RPNN-EOF.

models are also a subject of the future work. Moreover, recursive multi-step forecasting will be further explored with univariate and multivariate time series.

Acknowledgment

The authors would like to thank Universiti Tun Hussein Onn Malaysia and the Office for Research, Innovation, Commercialization and Consultancy Management (ORICC) for funding this research under the Postgraduate Research Grant (GPPS), VOT# U612.

References

[1] Y.W. Chen, J.B. Yang, D.L. Xu and S.L. Yang, "On the inference and approximation properties of belief rule based systems," *Information Sciences*, vol. 234, pp. 121-135, 2013.

[2] Y. Gao and M. J. Er, "NARMAX time series model prediction: feedforward and recurrent fuzzy neural network approaches," *Fuzzy sets and systems*, vol. 150, no. 2, pp. 331-350, 2005.

[3] G. E. Box, G. M. Jenkins, G. C. Reinsel and G. M. Ljung, *Time series analysis: forecasting and control*. New Jersey: John Wiley & Sons, 2015.

[4] K. Chakraborty, K. Mehrotra, C. K. Mohan and S. Ranka, "Forecasting the behavior of multivariate time series using neural networks," *Neural networks*, vol. 5, no. 6, pp. 961-970, 1992.

[5] G. Zhang, B. E. Patuwo and M. Y. Hu., "Forecasting with artificial neural networks: The state of the art," *International journal of forecasting*, vol. 14, no. 1, pp. 35-62, 1998.

[6] Y. Shin and J. Ghosh, "Ridge polynomial networks," *IEEE Transactions on neural networks*, vol. 6, no. 3, pp. 610-622, 1995.

[7] R. Ghazali, A. J. Hussain, N. M. Nawi, and B. Mohamad, "Non-stationary and stationary prediction of financial time series using dynamic ridge polynomial neural network," *Neurocomputing*, vol. 72, no. 10-12, pp.2359-2367, 2009.

[8] W. Waheeb, R. Ghazali, and A. J. Hussain, "Dynamic ridge polynomial neural network with Lyapunov function for time series forecasting," *Applied Intelligence*, vol. 48, no. 7, pp.1721-1738, 2018.

[9] W. Waheeb, R. Ghazali, and T. Herawan, "Ridge polynomial neural network with error feedback for time series forecasting," *PloS one*, 11(12), p.e0167248, 2016.

[10] W. Waheeb and R. Ghazali, "Multi-step Time Series Forecasting Using Ridge Polynomial Neural Network with Error-Output Feedbacks," in *International Conference on Soft Computing in Data Science*, Kuala

Lumpur, 2016, pp. 48-58.

[11] G. P. Zhang, "Neural Networks for Time-Series Forecasting," in *Handbook of Natural Computing*, Berlin, Heidelberg, Springer, 2012, pp. 461-477.

[12] J. T. Connor, R. D. Martin and L. E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 240-254, 1994.

[13] A. N. Burgess and A. N. Refenes, "Modelling non-linear moving average processes using neural networks with error feedback: An application to implied volatility forecasting," *Signal Processing*, vol. 74, no. 1, pp. 89-99, 1999.

[14] R. Ghazali, A. J. Hussain, and P. Liatsis, "Dynamic Ridge Polynomial Neural Network: Forecasting the univariate non-stationary and stationary trading signals," *Expert Systems with Applications*, vol. 38, no. 4, pp.3765-3776, 2011.

[15] P. Liatsis and A. J. Hussain, "Nonlinear 1D DPCM image prediction using polynomial neural networks," in *Applications of Artificial Neural Networks in Image Processing IV*, San Jose, 1999, pp. 58-69.

[16] S. Tertois, A. L. Glaunec and G. Vaucher, "Compensating the non-linear distortions of an OFDM signal with neural networks," in *the 9th International Workshop on Systems, Signals and Image Processing*, Manchester, 2002, pp. 484-488.

[17] C. Voutriaridis, Y. S. Boutalis and B. G. Mertzios, "Ridge polynomial networks in pattern recognition," in *Video/Image Processing and Multimedia Communications*, Zagreb, 2003.

[18] N. Shenvi, J. M. Geremia and H. Rabitz, "Efficient chemical kinetic modeling through neural network maps," *The Journal of chemical physics*, vol. 120, no. 21, pp. 9942-9951, 2004.

[19] R. Ghazali, A. J. Hussain, P. Liatsis, and H. Tawfik, "The application of ridge polynomial neural network to multi-step ahead financial time series prediction," *Neural Computing and Applications*, vol. 17, no. 3, pp.311-323, 2008.

[20] T. Hacib, Y. L. Bihan, M.-K. Smaïl, M. R. Mekideche, O. Meyer and L. Pichon, "Microwave characterization using ridge polynomial neural networks and least-square support vector machines," *IEEE Transactions on Magnetics*, vol. 47, no. 5, pp. 990-993, 2011.

[21] N. K. S. Behera and H. S. Behera, "Firefly based ridge polynomial neural network for classification," in *Advanced Communication Control and Computing Technologies (ICACCCT)*, Ramanathapuram, 2014, pp. 1110-1113.

[22] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics bulletin*, vol. 1, no. 6, pp. 80-83, 1945.

[23] J. Demšar. "Statistical comparisons of classifiers over multiple data sets." *Journal of Machine learning research,* 2006, pp. 1-30.

[24] Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2006). Extreme learning machine: theory and applications. Neurocomputing, 70(1-3), 489-501.

[25] R. Hyndman, G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay, M. O'Hara-Wild, F. Petropoulos, S. Razbash, E. Wang, F. Yasmeen, forecast: Forecasting functions for time series and linear models. R package version 8.5, 2019, http://pkg.robjhyndman.com/forecast.

[26] N. Kourentzes, nnfor: Time Series Forecasting with Neural Networks. R package version 0.9.6, 2019, https://cran.r-project.org/web/packages/nnfor/index.html.

Waddah Waheeb

Waddah Waheeb received his MSc (Soft Computing) from Universiti Tun Hussein Onn Malaysia in 2015. He is currently pursuing his PhD degree at the same university. His scientific work has been published in peer-reviewed journals and conferences. He has won numerous awards for his research such as the Best Paper Award at the 2nd International Conference on Soft Computing in Data Science and at the 3rd International Conference of Reliable Information and Communication Technology. Moreover, his ensemble method used in the worldwide M4 forecasting competition is listed among the 17 most accurate methods as compared to benchmark methods, ranked 15th among 50 submissions around the world on average, and ranked 6th among 50 submissions around the world with the weekly time series.

Rozaida Ghazali

Rozaida Ghazali is currently a Professor at the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia (UTHM). She graduated with a Ph.D. degree in Higher Order Neural Networks from the School of Computing and Mathematical Sciences at Liverpool John Moores University, United Kingdom in 2007. Earlier, in 2003 she completed her M.Sc. degree in Computer Science from Universiti Teknologi Malaysia (UTM). She received her B.Sc. (Hons) degree in Computer Science from Universiti Sains Malaysia (USM) in 1997. In 2001, Rozaida joined the academic staff in UTHM. Her research area includes neural networks, swarm intelligence, optimization, data mining, and time series prediction. She has successfully supervised a number of PhD and master students and published more than 100 articles in various international journals and conference proceedings. She acts as a reviewer for various journals and conferences, and as an editor in a few Springer conference proceedings. She has also served as a conference chair, and as a technical committee for numerous international conferences.