Proc. of the Intl. Conf. on Advances In Computer and Electronics Technology- ACET 2014.

ISBN: 978-1-63248-024-8 doi: 10.15224/ 978-1-63248-024-8-14

A Nonlinear ARIMA Technique for Debian Bug Number Prediction

Jayadeep Pati and K.K. Shukla

***Abstract*—A bug in a software application may be a requirement bug, development bug, testing bug or security bug, etc. To predict the bug numbers accurately is a challenging task. Both end users and software developers get benefit by predicting the number of bugs in a new version of software application in advance. The choice of predicting models becomes an important factor for improving the prediction accuracy. This paper provides a combination methodology that combines ARIMA and ANN models for predicting the bug numbers in advance. This method is examined using bug number data for Debian which is publicly available. This paper also gives a comparative analysis of forecasting performance of hybrid Nonlinear ARIMA, ARIMA and ANN models. Empirical results indicate that an Nonlinear ARIMA model can improve the prediction accuracy.**

***Keywords*—Debian, Bug, Bug Pattern, Artificial Neural Network, ARIMA, Hybrid Model.**

I. Introduction

The bug information about different software applications is maintained in software repositories [4]. The bug may be a system application bug or a user application bug. The monthly, weekly and daily bug number information can be extracted from the repositories. These extracted bug number data can be used for time series analysis.

Previous studies on bug number predictions are based on traditional time series modelling like ARIMA [1] that is used to predict a stationary time series data. In another paper by Hongyu et al, [2] Polynomial Regression is used to predict the bug growth patterns in Eclipse. In another paper [3] we have used Nonlinear Autoregressive Neural Net to predict the bug numbers in Debian.

One of the popularly used methods for time series analysis is ARIMA modelling. The advantages of ARIMA model are due to its statistical properties and Box–Jenkins methodology [2] in the model building process. The limitation of the model is the linear form of the model.

Neural networks are generally used for classification and prediction task in different domains. Neural networks are also frequently used in time series modelling [16]. They are good approximators for nonlinear patterns in data. As real time bug number data shows nonlinearity, neural network is a good approach for modelling.

In this paper we have used a hybrid ARIMA + neural network model to predict the Debian bug number series. There are many hybrid approach used in time series modelling [12, 13, 14, 15]. We have also performed a comparative analysis of performance hybrid model, ARIMA model and neural network model in predicting the bug numbers of Debian.

Advance knowledge about bug numbers will help the software managers to take decision on resource allocation and effort investments. The developers will be aware of the number of bugs in advance and can take effective steps to reduce the number of bugs. The end user can take decision on adopting a particular software application by knowing the bug growth patterns of the particular software application.

The rest of the paper is organized as follows: Section 2 presents a description of neural network modeling. Section 3 describes about ARIMA in shot. Section 4 describes about the hybrid approach and its flow diagram. Section 5 describes about model evaluation. Section 6 describes about data preparation. Section 7 describes about Implementation, Results and Discussion. Section 8 gives comparison among the models to find the best model. Section 9 concludes the paper and shows direction for the future work.

II. Artificial Neural Network

A. *Neural Network Model*

A feed-forward neural network (multilayer) is nonlinear learning model and it uses supervised training approach. It performs adjustment of the network weights on its inputs and the internal nodes in an iterative manner to minimize the errors between actual and predicted values. It commonly uses simple gradient descent to find the local minima and optimize the network accordingly. The Levenberg– Marquardt algorithm [5, 6] with Bayesian Regularization [11] is the most updated algorithm and commonly used for neural network training.

The data needs to be normalized before being trained by a neural network training algorithm. The normalization is crucial to obtain good results as well as to speed up the calculations significantly. Sometimes we use special transfer functions which normalize the data before training [9].

Jayadeep Pati
Indian Institute of Technology (BHU)
India

K.K. Shukla
Indian Institute of Technology (BHU)
India

B. *Transfer Function*

There are different transfer functions to catch the linear and nonlinear patterns in the data. As the real time bug number data shows nonlinear behaviour, we have used only nonlinear transfer functions for the hidden layer.

III. ARIMA

ARIMA (P, D, Q) consists of AR (P), MA(q) and ARMA (P, Q) classes. AR (P) is a autoregressive model of order P which represents the value of a series as a linear regression of previous P values. The moving average MA (Q) against the white noise terms. ARMA (P, Q) is the combination of AR (P) and MA (Q) model. The time series needs to be differentiated before applying ARMA (P, Q) model. ARIMA includes the differentiating operator D. The details of the ARIMA model can be found in the paper by [8].

IV. HYBRID MODEL (NONLINEAR ARIMA)

Although there are numerous time series models available, but none of them fit the real time data perfectly as real time bug number data contains mixture of linear and nonlinear terms. Hybrid models reduce the risk by combining different models. The concepts of hybrid model come from the fact that it is a very complex task to detect the data generation process or a single model is not competent enough to identify the characteristic of time series.

In the paper we have used an advanced hybridization model approach to bug number prediction to yield more accurate result. Here, we have used a nonlinear integration of autoregressive and moving average terms i.e. a time series is taken as a nonlinear function of some past observations and some error terms.

In ARIMA we have:

$$Y_t = F_l\left(\left(Y_{t-1,} Y_{t-2,} \ldots \ldots Y_{t-m}\right), \left(e_{t-1,} e_{t-2,} \ldots \ldots e_{t-n}\right) \right) \ldots .(1)$$

Where F_l is a linear function.

In the hybrid model we use a neural network to have a nonlinear integration of some past observations and some error terms.

Here we have:

$$Y_t = F_n\left(\left(Y_{t-1,} Y_{t-2,} \ldots \ldots Y_{t-m}\right), \left(e_{t-1,} e_{t-2,} \ldots \ldots e_{t-n}\right) \right) \ldots .(2)$$

Where F_n is a nonlinear function used in neural network.

In the first step we use ARIMA to get the residual terms and in the second step we use neural network to model the time series as a nonlinear combination of some past observations and some error terms. Figure 1 shows the representation of the hybrid system.

V. EVALUATION ACCURACY

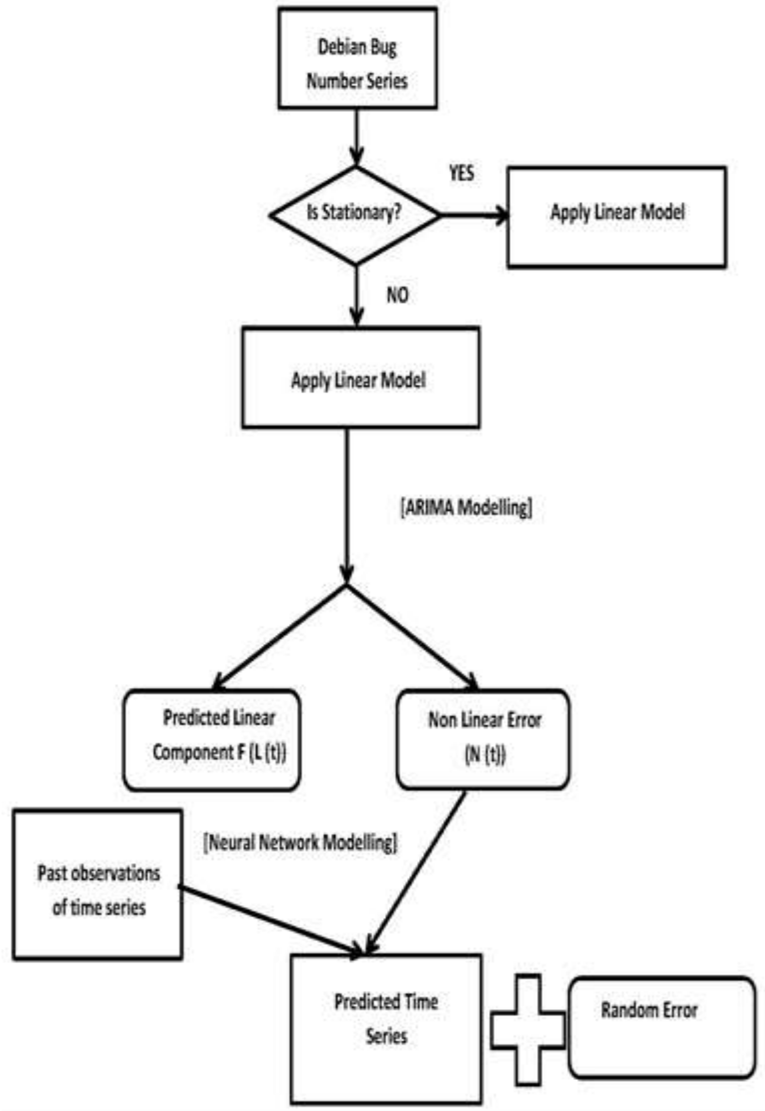

Figure 1: Hybrid System

To evaluate the accuracy of the models different formulas are used. We use following methods for this purpose.

1. Mean Absolute Error (MAE):

$$MAE = \frac{1}{N} \sum_{t=1}^{N} |(A(t) - F(t)) /A(t)| \ldots .(3)$$

2. Root Mean Square Error (RMSE):

$$MSE = \sqrt{\frac{1}{N}} \sum_{t=1}^{N} |\left(A(t) - F(t)\right)^2 \ldots \ldots \ldots (4)$$

3. Average Error per Mean (Em):

$$E_m = Mean\frac{(Output - Input)}{Mean\ (Data)} \ldots\ldots\ldots\ldots (5)$$

VI. DATA PREPARATION

A. *Data Collection*

The detailed bug reports for Debian are available in the Ultimate Debian Database (UDD) [4]. The original data in the repository looks like this: “id, “ Last Modified” : 718624; 2013-08 -19. We have considered total bug counts in our analysis. The monthly bug number data from Jan – 2000 to Dec – 2013 is taken for analysis. For Debian, we have 168 monthly bug counts. The bug number series from Jan – 2000 to Dec -2013 shows an increasing trend. The time series of monthly bug counts from Jan – 2000 to Dec – 2013 is given figure 2.

B. *Data Partition*

The entire bug number series is partitioned in to a training set and a testing set. Partitioning helps in finding how our model which fits the data pattern (both trained and unseen data) properly. The model which gives more accuracy on test data is preferably selected. For all the models the size the training dataset is 140 and size of test dataset is 10.

VII. IMPLEMENTATION AND RESULT

In this paper we have experimentally verified the predictive performance of three models: ARIMA, ANN and the Hybrid Model (ARIMA + ANN). In this section we will describe in details about the application of these models to Debian bug number series.

A. *ARIMA*

As discussed in section 2, ARIMA model consists of AR (p), MA (q) and a Differencing (d) operator. To have a good predictive performance the series must be stationary. The stationary behavior can be checked from the Auto Correlation Function (ACF) and Partial Auto Correlation Function (PACF) plots. If the ACF decays slowly, then the series is said to be non-stationary. If the ACF cuts instantly, then the series is said to be stationary.

Results and Discussion:

For the Debian bug number series the ACF plot is given in Figure 1. From the figure 1, it is observed that the series is non-stationary. To make it stationary, we have used differencing operator.

After differencing the ACF cuts at lag2 instantly and PACF also cuts approximately at lag 2. There are some abnormalities in higher lags which can be ignored. So, the most appropriate model for the Debian bug number series becomes ARIMA (2, 1, 2). We have also checked 15 combination of p, d, q parameters for ARIMA to find the best fit.

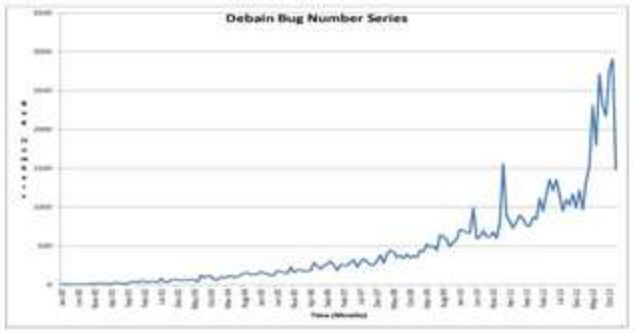

Figure 2: Debian Bug Number Series

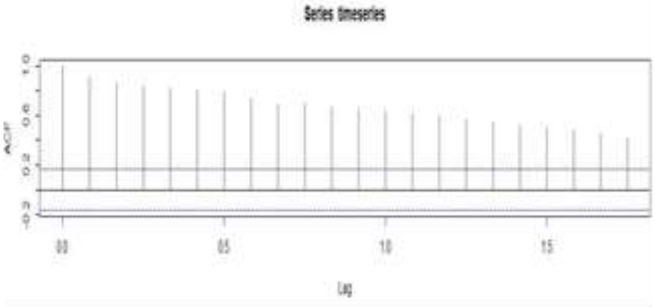

Figure 3: ACF Plot for Debian Bug Number Series

It is observed that ARIMA (2, 1, 2) is the best among all the other models as it has least RMSE and MAE values. After applying the ARIMA (2, 1, 2) we got a predicted series and some error terms.

The RMSE, MAE and the Error function (E) for ARIMA model for Debian bug number series is shown in table 2.

Table2: Error Values (ARIMA)

ARIMA Model Prediction	RMSE	MAE	Mean Error Value (E)
Training Set	83.510860	38.82696125	0.142604885
Test Set	250.8259	197.2487	0.190137575

B. *Artificial Neural Network (ANN)*

Here we have taken a feed forward neural network for predicting the Debian bug number series. The multiple layers of neurons with nonlinear transfer functions allow the network to match the nonlinear patterns in the data. The output layer uses linear transfer function for function fitting problem. The neural network is trained by Levenberg– Marquardt algorithm with Bayesian Regularization (trainbr) [11], an advanced neural network training algorithm. We have taken different nonlinear transfer functions to find the best fitted function for the Debian bug number series. All the experiments were conducted in Matlab environment.

Proc. of the Intl. Conf. on Advances In Computer and Electronics Technology- ACET 2014.
Copyright © Institute of Research Engineers and Doctors. All rights reserved.
ISBN: 978-1-63248-024-8 doi: 10.15224/ 978-1-63248-024-8-14

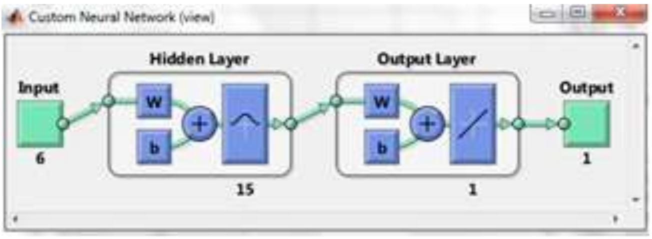

Figure 7: Neural Network Design

Table3: Error Values (Neural Network)

ANN Model Prediction	RMSE	MAE	Mean Error Value (E)
Training Set	58.6773	20.4584	0.0751
Test Set	230.847	193.7327	0.1867

Results and Discussion:

We have used feed forward neural network having 6 input neurons, 15 hidden layer neurons and single output neuron. The 11 inputs represent previous bug number data of previous 5 months and serial number of corresponding input vectors. The hidden layer is having 15 neurons with nonlinear transfer function that calculates layer's output from its net input. For this model the training data size is 140 and test data size is 10. As the bug number series is nonlinear we only apply nonlinear activation functions to match the nonlinear patterns in the data. . Figure 7 shows the architecture of our ANN model. The RMSE, MAE and the Error function (E) for neural network model for Debian bug number series for training set and test set is given in table 3.

C. *Hybrid Model (Nonlinear ARIMA):*

The next step is to apply the linear and non-linear combination of models to predict the Debian bug number series. As discussed in section 2, the Debian bug number series is first modeled with ARIMA. The result is predicted linear component and some non-linear error term . Now the error contains some non-linear patterns which need non-linear technique to model the error series.

Some past observations and some error series are modeled as non-linear neural network model of order 5, i.e. five previous values of observations and error terms are required for predicting the current value. We have used non-linear activation function (radbasn) to match the non-linear patterns in the error series. We have also used a factor called Goodness factor (G) series as an additional input vector for improving the accuracy of neural network. The G value is "+1" when error is positive and "-1" when error is negative. After applying neural network model we get a predicted bug number series and some random error.

The RMSE, MAE and the Error function (E) for neural network model for Debian bug number series for training set and test set is given in table 4.

Table 4: Error Values (Nonlinear ARIMA)

ARIMA + ANN Model Prediction	RMSE	MAE
Training Set	23.22046679	8.420530711
Test Set	129.5394011	49.64705963

VIII. COMPARISON BETWEEN MODELS

In this section we give a comparative analysis of predictive performance of the ARIMA, Neural network and the hybrid (Nonlinear ARIMA) model. For training dataset, figure 4 shows the graphical representation of the actual data series with predicted series by the three models respectively. Figure 5 shows same representation for test set.

Form the graph it is observed that thy hybrid model match the bug number series more accurately than the other three models. For the test sample the hybrid model also outperforms the ARIMA and Neural Network model with a large margin. We also found that ARIMA is a poor predictor of Debian bug number series which contain both linear and non-linear terms. Also neural network won't fit the linear patterns perfectly. The results confirms the hybrid model as a good predictor for Debian bug number series.

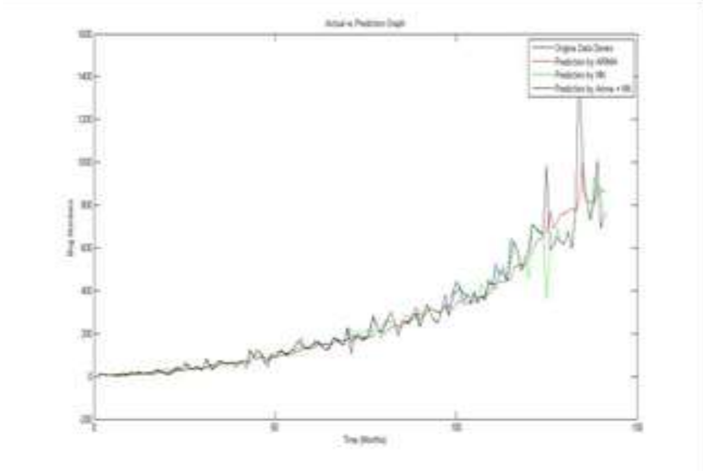

Figure 4: Comparison between Actual series and Predicted series by three models. (Training Set)

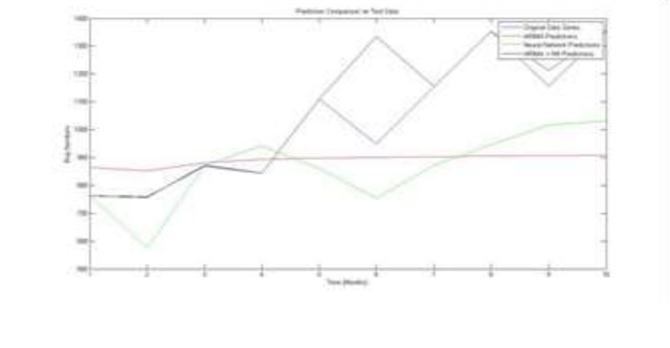

Figure 5: Comparison between Actual series and Predicted series by three models. (Test Set)

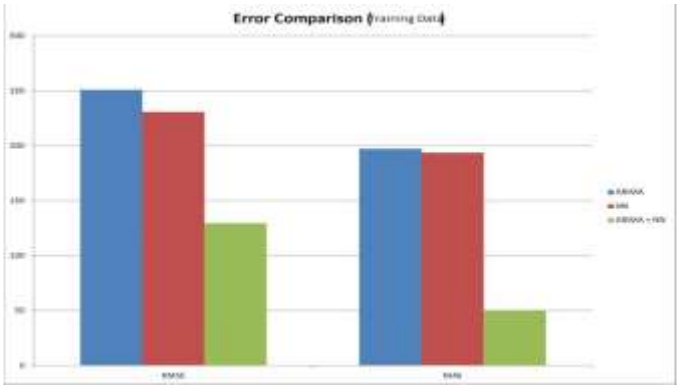

Figure 6: Comparison of RMSE and MSE value between the three Models (Training Set)

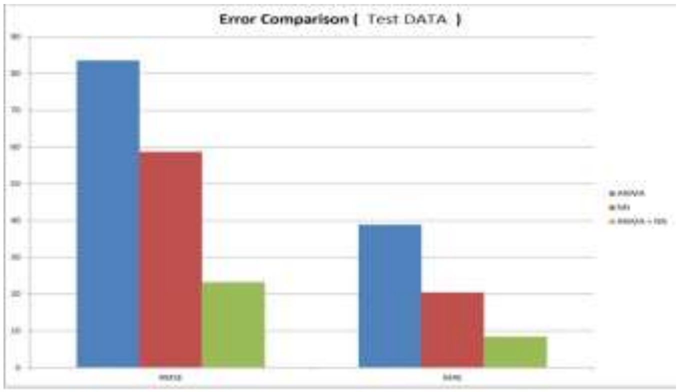

Figure 7: Comparison of RMSE and MSE value between the three Models (Test Set)

Figure 6 and Figure 7 show the bar chart representation of comparison between RMSE and MAE for both training and test set for the three models respectively.

IX. CONCLUSION AND FUTURE WORK

This paper uses a hybrid approach to predict the trend of the bug numbers in Debian. This paper also compares the predictive performance between ARIMA, Neural Network and the Hybrid model (Nonlinear ARIMA) to find the best fit for Debian bug number series. The result confirms the hybrid model (Nonlinear ARIMA) as a most appropriate model for Debain bug number series. This paper also confirms the poor performance of ARIMA model in predicting the non-linear patterns in data and low predictive performance of neural network when data contain mixture of linear and non-linear patterns.

The contribution of the paper consists of two parts: first, it adopts hybrid approach to predict the trend of the bug numbers in Debain. Second, it uses efficient way of interpreting the results obtained by the models.

In the future we will apply advanced ensemble techniques like Neural Net + PSO, Neural Net+ GA, etc. to predict bug number series in software applications. We will apply time series analysis to other large open source software applications.

Acknowledgment

The first author gratefully acknowledges the assistantship provided by IIT(BHU).

REFERENCE

[1] Wenjin Wu; Wen Zhang; Ye Yang; Qing Wang, "Time series analysis for bug number prediction," Software Engineering and Data Mining (SEDM), 2010 2nd International Conference on , vol., no., pp.589,596, 23-25 June 2010

[2] Hongyu Zhang,An initial study of the growth of eclipse defects, MSR ‟08 Proceedings of the 2008 international working conference on Mining software repositories Pages 141-144 ,ACM New York, NY, USA 2008

[3] Pati, J.; Shukla, K.K., "Time series prediction of debian bug data using autoregressive neural network," Computer and Communication Technology (ICCCT), 2013 4th International Conference on , vol., no., pp.110,115, 20-22 Sept. 2013

[4] http://udd.debian.org/bugs

[5] K. Levenberg, A method for the solution of certain problems in least squares, Quarterly of Applied Mathematics, 5, 164168, 1944.

[6] D. Marquardt, An algorithm for least-squares estimation of nonlinear parameters, SIAM Journal on Applied Mathematics, 11(2), 431441, June 1963.

[7] Montgomery, Douglas C., and Douglas C. Montgomery. Design and analysis of experiments. Vol. 7. New York: Wiley, 1997.

[8] Box, G. E. P., and Jenkins, G. (1994), Time Series Analysis: Forecasting and Control, Holden-Day. 3rd Edition, Prentice-Hall: New York, NY

[9] http://www.mathworks.in/help/nnet/function-approximation-and-nonlinear-regression.html

[10] http://mathworld.wolfram.com/NonparametricEstimation.html

[11] Payal, A.; Rai, C.S.; Reddy, B.V.R., "Comparative analysis of Bayesian regularization and Levenberg-Marquardt training algorithm for localization in wireless sensor network," Advanced Communication Technology (ICACT), 2013 15th International Conference on , vol., no., pp.191,194, 27-30 Jan. 2013

[12] Mehdi Khashei, Mehdi Bijari, A novel hybridization of artificial neural networks and ARIMA models for time series forecasting, Applied Soft Computing, Volume 11, Issue 2, March 2011, Pages 2664-2675, ISSN 1568-4946, http://dx.doi.org/10.1016/j.asoc.2010.10.015.

[13] Mehdi Khashei, Mehdi Bijari, An artificial neural network (p,&#xa0;d,&#xa0;q) model for timeseries forecasting, Expert Systems with Applications, Volume 37, Issue 1, January 2010, Pages 479-489, ISSN 0957-4174, http://dx.doi.org/10.1016/j.eswa.2009.05.044.

[14] Eswaran, C., and R. Logeswaran. "An enhanced hybrid method for time series prediction using linear and neural network models." Applied Intelligence 37.4 (2012): 511-519.

[15] Areekul, Phatchakorn, et al. "Combination of artificial neural network and ARIMA time series models for short term price forecasting in deregulated market." Transmission & Distribution Conference & Exposition: Asia and Pacific, 2009. IEEE, 2009.

[16] Zhang, G. Peter, and Min Qi. "Neural network forecasting for seasonal and trend time series." European journal of operational research 160.2 (2005): 501-514.