Mining Software Aging Patterns by Artificial Neural Networks

Hisham El-Shishiny, Sally Deraz, and Omar Bahy

IBM Cairo Technology Development Center
P.O.B. 166 Ahram, Giza, Egypt.
shishiny@eg.ibm.com,sally@eg.ibm.com,obadr024@uottawa.ca

Abstract This paper investigates the use of Artificial Neural Networks (ANN) to mine and predict patterns in software aging phenomenon. We analyze resource usage data collected on a typical long-running software system: a web server. A Multi-Layer Perceptron feed forward Artificial Neural Network was trained on an Apache web server dataset to predict future server swap space and physical free memory resource exhaustion through ANN univariate time series forecasting and ANN nonlinear multivariate time series empirical modeling. The results were benchmarked against those obtained from non-parametric statistical techniques, parametric time series models and other empirical modeling techniques reported in the literature.

Data Mining, Artificial Neural Network, Pattern Recognition, Software Aging

1 Introduction

It has been observed that software applications executing continuously over a long period of time, such as Web Servers, show a degraded performance and increasing rate of failures [5]. This phenomenon has been called software aging [4]. This may be due to memory leaks, unreleased file-locks and round-off errors. Currently, researchers are looking for methods to counteract this phenomenon by what is so called software rejuvenation methods such as applying a form of preventive maintenance. This could be done by, for example, occasionally stopping the software application, cleaning its internal state and then restarting [9] to prevent unexpected future system outages. This allows for scheduled downtime at the discretion of the user, which suggests an optimal timing of software rejuvenation.

In this work, we investigate the use of Artificial Neural Networks (ANN) univariate time series forecasting and ANN nonlinear multivariate time series empirical modeling to mine and predict software aging patterns in a typical long-range software system: a web server, in order to assess ANN suitability for the analysis of the software aging phenomenon. ANN are used to forecast swap space and free physical memory of an Apache web server and results are cross

benchmarked against those reported in the literature based on parametric and non-parametric statistical techniques and other empirical modeling techniques.

This research aims at providing some empirical evidence on the effectiveness of artificial neural networks on modeling, mining and predicting software aging patterns, and the ultimate goal is an optimization model that uses the prediction of resources exhaustion as well as further information for deriving the best rejuvenation schedule.

The rest of this paper is organized as follows: in section2, we review related work and in section3, the data collected is described. The adopted Neural Network approach is illustrated in section 4. Finally, conclusion and future work are presented in section 5.

2 Related Work

The software aging problem is currently approached either by building analytical models for system degradation such as probability models, linear and nonlinear statistical models, expert systems and fractal base models [7,3,1], or by empirically studying the software systems based on measurements. Few attenpts were reported on the use of Wavelet Networks in software aging [12,10].

The rate to which software ages is usually not constant, but depends on the time-varying system workload. Therefore, time series models are usually fitted to the data collected to help predicting the future resource usage. Attributes subject to software aging are monitored and related data is collected aiming at predicting the expected exhaustion of resources like real memory and swap space. Then, non-parametric statistical techniques and parametric time series models are employed to analyze the collected data and estimate time to exhaustion via extrapolation for each resource [5], usually assuming linear functions of time.

3 Software aging data

We make use of the data reported in [5] and [7] to carry on further analysis using an Artificial Neural Network approach. The collected data is from a Linux web server with an artificial load approaching its maximum optimal load level.

The setup that was used for collecting the data consisted of a server running Apache version 1.3.14 on a Linux platform, and a client connected via an Ethernet local area network. Among the system parameters of the web server monitored during a period of more than 3.5 weeks are the free physical memory and the used swap space. Data were collected during experiments in which the web server was put in a near overload condition indicating the presence of software aging.

4 The Neural Network Approach

4.1 Artificial Neural Networks

ANN is a class of flexible nonlinear models that can discover patterns adaptively from the data. Given an appropriate number of nonlinear processing units, neural networks can learn from experience and estimate any complex functional relationship with high accuracy. Numerous successful ANN applications have been reported in the literature in a variety of fields including pattern recognition and forecasting. For a comprehensive overview of ANN the reader is referred to [8].

4.2 ANN for mining patterns in software aging

In software aging, we do not have a well defined model describing the aging process that one would like to study. All that is available are measurements of the variables of interest (i.e. time series). Therefore, we propose, in this work, an artificial neural network approach for mining software aging patterns, with the objective of predicting the expected exhaustion patterns of resources like real memory and swap space used. We investigate in this work two ANN based methods for this problem; a univariate time series forecasting method and a multivariate time series empirical modeling method.

4.3 The Proposed Neural Network Structure

Although many types of neural network models have been proposed, the most popular one is the Multi-Layer Perceptron (MLP) feed forward model [13]. A multi layer feed forward network with at least one hidden layer and a sufficient number of hidden neurons is capable of approximating any measurable function [11]. A feed-forward network can map a finite time sequence into the value that the sequence will have at some point in the future [6]. Feed forward ANNs are intrinsically non-linear, non-parametric approximators, which makes them suitable for complex prediction tasks.

For this problem, we choose to use a fully connected, MLP, feed forward ANN with one hidden layer, a logistic activation function as in figure 1, and the back propagation learning algorithm [6].

4.4 Forecasting the exhaustion of the Apache server resources

We use the ANN described above and the data introduced in [5] and [7] to predict the Apache server Free Physical Memory and Swap Space Used performance variables, in order to obtain predictions about possible impending failures due to resource exhaustion. An ANN based univariate time series method is used for forecasting the Swap Space Used and an ANN based non-linear multivariate time series empirical modeling method is used to predict the Free Physical Memory.

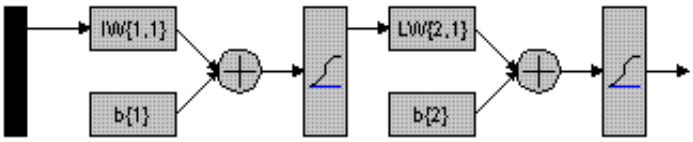

Figure1. The implemented MLP Neural Network

This dataset was split into three segments; the first segment is used to train the ANN and the second segment is used to tune the ANN parameters (i.e. number of time lags and number of neurons in the hidden layer) and validation. The third segment is used to measure the ANN generalization performance on data which has not been presented to the NN during parameter tuning.

Forecasting Swap Space Used of the Apache server. The Swap Space Used of the Apache server is forecasted using ANN based univariate time series forecasting. The usage of ANN for time series analysis relies entirely on the data that were observed and is powerful enough to represent any form of time series. ANN can learn even in the case of noisy data and can represent nonlinear time series. For example, Given a series of values of the variable x at time step t and at past time steps $x(t), x(t-1), x(t-2) \cdots x(t-m)$, we look for an unknown function F such that; $X(t+n) = F[x(t), x(t-1)$,x(t-2)$\cdots x(t-m)]$, which gives an $n-step$ predictor of order m for the quantity x.

The ANN sees the time series $X1, \cdots, Xn$ in the form of many mappings of an input vector to an output value [2]. The time-lagged values $x(t), x(t-1), x(t-2) \cdots x(t-m)$ are fed as inputs to the network which once trained on many input-output pairs, gives as output the predicted value for yet unseen x values. The ANN input nodes in this case are the previous lagged observations while the output nodes are the forecast for the future values. Hidden nodes with appropriate non-linear transfer (activation) functions are used to process the information received by the input nodes.

The number of ANN input neurons determine the number of periods the neural network looks into the past when predicting the future. Whereas it has been shown that one hidden layer is generally sufficient to approximate continuous function [8], the number of hidden units necessary is not known in general. To examine the distribution of the ANN main parameters (i.e. number of time lags and number of neurons in the hidden layer), we conducted a number of experiments, where these parameters were systematically changed to explore their effect on the forecasting capabilities. These estimations of the networks most important parameters although rough, allowed us to choose reasonable parameters for our ANN.

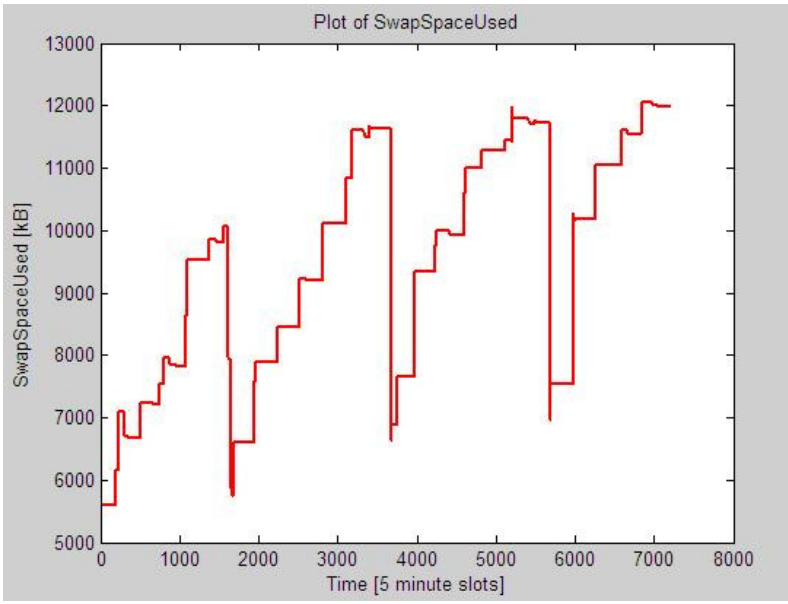

Figure2. Swap Space Used

The Swap Space Used dataset was collected on a 25-day period with connection rate of 400 per second. We divided the collected data into three segments, one to train the ANN, one for validation, and the third for testing. The testing segment is used to evaluate the forecasting performance of the ANN in predicting the performance parameters values.

The training and forecasting accuracy is measured by Root Mean Square Error (RMSE) and two other common error measures, MAPE and SMAPE.

Mean Absolute Percentage Error (MAPE). MAPE is calculated by averaging the percentage difference between the fitted (forecast) line and the original data:

$MAPE = \sum_t |e_t/y_t| * 100/n$

Where y represents the original series and e the original series minus the forecast, and n the number of observations

Symmetric Mean Absolute Percentage Error (SMAPE). SMAPE calculates the symmetric absolute error in percent between the actual X and the forecast F across all observations t of the test set of size n. The formula is

$MAPE = \frac{1}{n} \sum_{t=1}^{n} \frac{|X_t - F_t|}{(X_t + F_t)/2} * 100$

Results. Figure 2 shows Swap Space Usage for the Apache server. It is clear that it follows a seasonal pattern and that considerable increases in used swap space occur at fixed intervals.

Table 1 shows the RMSE, MAPE and SMAPE for the forecasts of Swap Space Used of the Apache server for the testing dataset using the MLP described in

Table1. Swap Space Used evaluation

Error measures for the predicted data	SMAPE (Symmetric Mean Absolute Percent Error)	MAPE (Mean Absolute Percent Error)	RMSE (Root Mean Square Error)
Non-Parametric Statistical approach	4.313%	4.47%	612.46
ANN approach	0.354%	0.357%	116.68

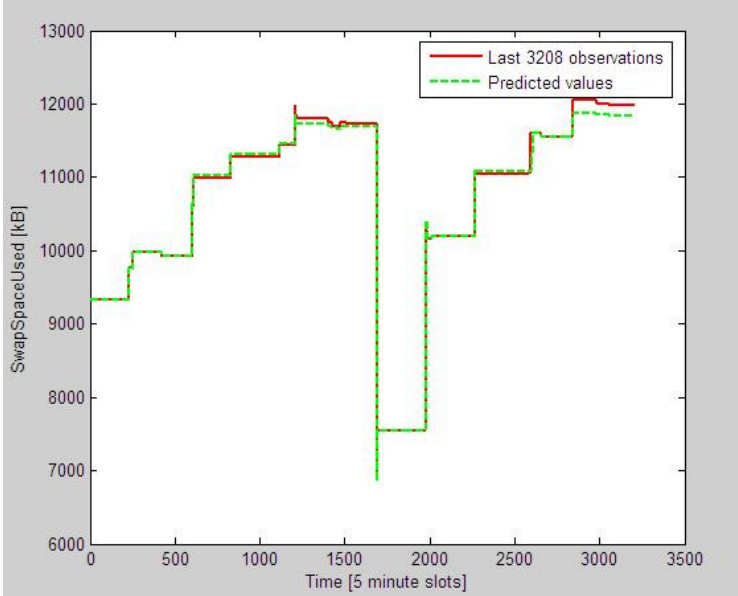

Figure3. Swap Space Used results

Figure 1 with 3 input neurons (time lags), 3 neurons in the hidden layer and a sigmoid nonlinear transfer function. As seen in Table 1, the results obtained by the ANN approach are far more accurate than the results obtained by the non-parametric statistical approach reported in [5].

In Figure 3, we show a plot of the last 3208 observations of the measured SwapSpaceUsed (the testing dataset) and the predicted values obtained by the ANN approach, which shows accurate predictions.

Forecasting Free Physical Memory of Apache Server. In order to model and predict the Apache server Physical Free Memory performance variable, we have developed an ANN based non-linear multivariate time series empirical modeling procedure that involves parameter set reduction and selection, model building and sensitivity analysis.

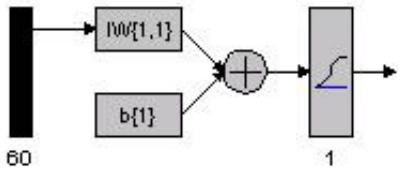

Figure4. A single neuron for logistic regression

Parameter set reduction and selection. Since there are 100 different Apache parameters that were monitored in addition to Free Physical Memory, an important question will be which of these parameters are the most important predictors. Some parameters may encode the same information and therefore are redundant and some others may have a trivial or no effect at all on future values of Free Physical Memory. Since parameter set reduction is a subset selection problem, therefore for 100 parameters we have 2 to the power of 100 possible subsets, which is not practical to evaluate.

In order to determine the smallest subset of input parameters which are necessary and sufficient for Free Physical Memory prediction, we have adopted the following approach:

a We have excluded 41 parameters because they had constant values during the monitoring period.
b We have performed non-linear logistic regression for the remaining 59 parameters in addition to the Free Physical Memory at time $(t-1)$, taking them as inputs to a simple one neuron ANN with a sigmoid activation function (Figure 4).
c We have selected seven parameters that had ANN weights values above an arbitrary small threshold (Fig. 5).
d We have repeated step (c) above but using a tan-sigmoid activation function and obtained nine parameters that had ANN weight values above the same threshold in step (c).
e We have selected the parameters in common between step (c) and (d) which were: si_tcp_tw_bucket, si_tcp_bind_bucket, si_mm_struct, si_files_cache, si_size_1024, udp_socks_high and PhysicalMemoryFree at time $t-1$.

Emperical model building. Having selected the seven parameters that look more significant in section 4.4 above, we have used them as input nodes for the MLP feed forward ANN of Figure 1. We have used a sigmoid activation function and two neurons in one hidden layer (2 neurons gave the least MAPE, SMAPE and RMSE during validation of this dataset). The output node was selected to be the Free Physical memory. We therefore have formulated the problem as a non-linear multivariate time series model.

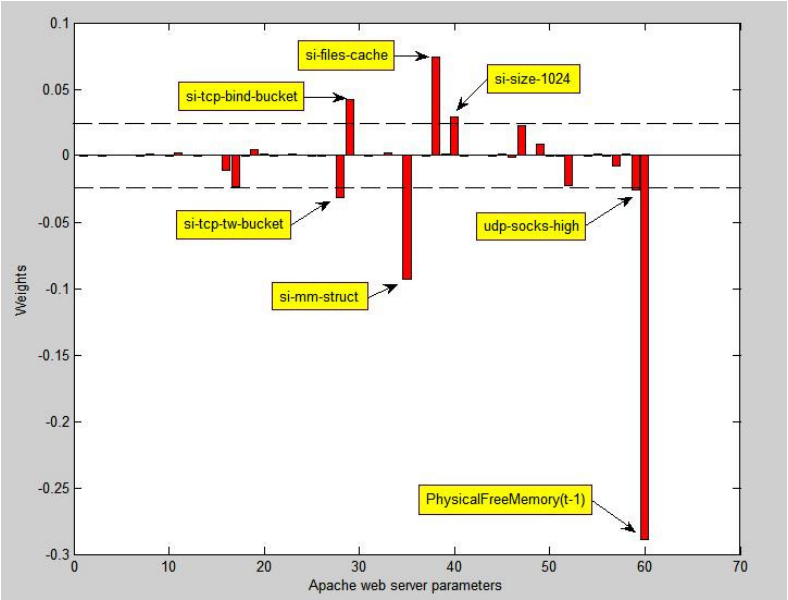

Figure5. ANN weights for the Apache web server parameters (Sigmoid activation function)

Parameter sensitivity analysis. We have conducted sensitivity analysis on the parameters of the ANN model developed in section 4.4 above in order to gain some insight into the type of interactions among the different parameters and the Free Physical Memory and to assess the contribution of each parameter on the predicted value of Free Physical Memory.

We removed one parameter at a time from the input of the developed ANN above and each time we computed the SMAPE, MAPE and RMSE on the testing dataset and recorded the change. We noted that the developed ANN model was particularly sensitive to the Free Physical Memory at time $(t-1)$ and the si_files_cache which is in accordance with the results reported in [7].

Results. Figure 6 shows a plot over time of the Free Physical Memory of the Apache server that was collected in a 7-day period with a connection rate of 350 per second. The shown irregular utilization pattern can be explained by the fact that the Free Physical Memory cannot be lower than a preset threshold value. If physical memory approaches the lower limit, the system frees up memory by paging [7].

Table 2 shows the RMSE, MAPE and SMAPE for the forecasted Free Physical Memory of the Apache server for the testing dataset using the ANN empirical model.

In Figure 7, we show a plot of the last 2483 observations of the measured physical free memory (the testing dataset) and the predicted values obtained by

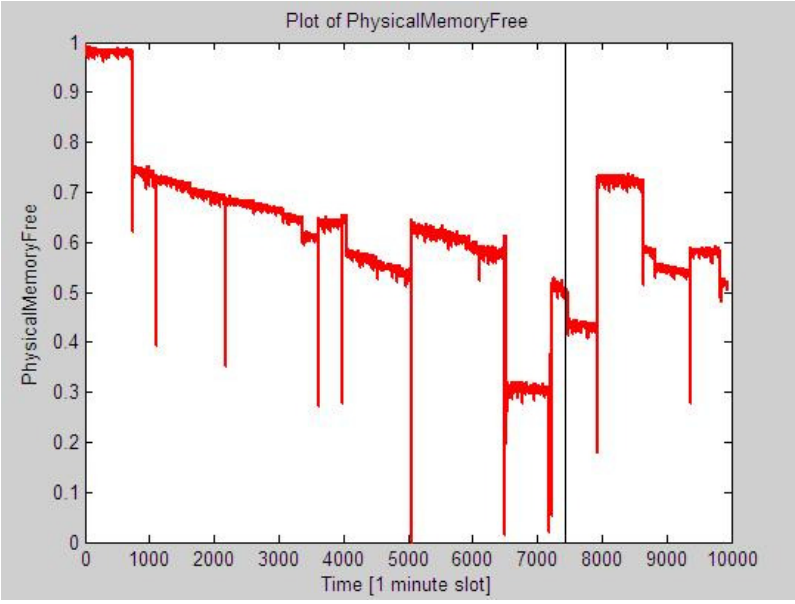

Figure6. Physical Free Memory

Table2. Physical Free Memory evaluation

Error measures for the predicted data	SMAPE (Symmetric Mean Absolute Percent Error)	MAPE (Mean Absolute Percent Error)	RMSE (Root Mean Square Error)
ANN approach	1.295%	1.275%	0.01354

the ANN empirical model, which shows accurate forecasts. Based on RMSE the obtained results are more accurate than the results reported in [7] obtained from universal basis functions, multivariate linear regression, support vector machines and radial basis functions empirical modeling techniques.

5 Conclusion

In this work we have investigated the use of ANN for mining the software aging patterns in a typical long-running software system: an Apache web server. ANN based univariate time series forecasting method and ANN nonlinear multivariate time series empirical model were developed to predict swap space used and memory usage that are related to software aging, of an Apache web server subjected to a synthetic load for 25 days. We showed that a Multi-Layer Perceptron (MLP) feed forward ANN is able to accurately predict the future behavior of these performance variables. The results obtained were benchmarked against those reported in the literature that are based on parametric and non-parametric

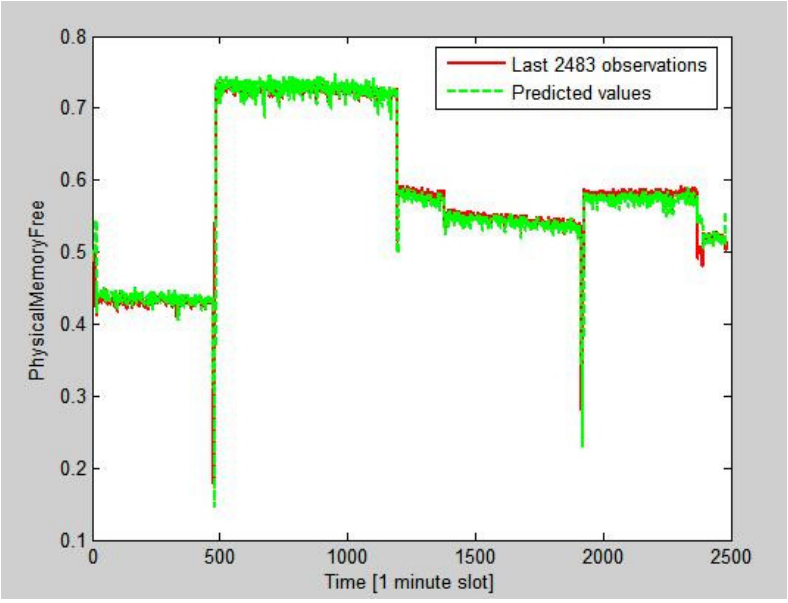

Figure7. Physical Free Memory results

statistical techniques and other empirical modeling techniques and were more accurate.

Future work involves extending the proposed Artificial Neural Network approach to attempt to define an optimal software rejuvenation policy.

Acknowledgements

The authors would like to thank Professor Kishor S. Trivedi, the Hudson Chair in the Department of Electrical and Computer Engineering at Duke University, for valuable discussions during the development of this work and for his review of the manuscript and Michael Grottke for providing the Apache performance dataset

This work is part of a research project conducted at IBM Center for Advanced Studies in Cairo.

References

1. G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications.* Wiley-Interscience, New York, NY, USA, 1998.
2. K. Chakraborty, K. Mehrota, K. M. Chilukuri, and S. Ranka. Forecasting the behaviour of multivariate time series using neural networks. *Neural Networks 5*, pages 961–970, 1992.

3. M. Y. Chen, E. Kiciman, E. Fratkin, A. Fox, and E. Brewer. Pinpoint: Problem determination in large, dynamic internet services. In *DSN '02: Proceedings of the 2002 International Conference on Dependable Systems and Networks*, pages 595–604, Washington, DC, USA, 2002. IEEE Computer Society.
4. T. Dohi, K. Goseva-Popstojanova, and K. S. Trivedi. Analysis of software cost models with rejuvenation. *hase*, 00:25, 2000.
5. M. Grottke, L. Li, K. Vaidyanathan, and K. S. Trivedi. Analysis of software aging in a web server. *IEEE Transactions on Reliability*, 55(3):411–420, 2006.
6. M. H. Hassoun. *Fundamentals of Artificial Neural Networks*. MIT Press, Cambridge, MA, USA, 1995.
7. G. A. Hoffmann, K. S. Trivedi, and M. Malek. A best practice guide to resource forecasting for computing systems. *IEEE Transactions on Reliability*, pages 615–628, 2007.
8. K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Netw.*, 2(5):359–366, 1989.
9. N. Kolettis and N. D. Fulton. Software rejuvenation: Analysis, module and applications. In *FTCS '95: Proceedings of the Twenty-Fifth International Symposium on Fault-Tolerant Computing*, page 381, Washington, DC, USA, 1995. IEEE Computer Society.
10. M. H. Ning, Y. Q., H. Di, C. Ying, and Z. J. Zhong. Software aging prediction model based on fuzzy wavelet network with adaptive genetic algorithm. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, pages 659–666, 2006.
11. H. Siegelmann and D. S. Eduardo. Neural nets are universal computing devices. Technical Report SYSCON-91-08, Rugters Center for Systems and Control, 1991.
12. J. Xu, J. You, and K. Zhang. A neural-wavelet based methodology for software aging forecasting. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 59–63, 2005.
13. G. P. Zhang and M. Qi. Neural network forecasting for seasonal and trend time series. *European Journal of Operational Research*, pages 501–514, 2005.