PERGAMON

Computers & Operations Research 28 (2001) 381–396

computers & operations research

www.elsevier.com/locate/dsw

A simulation study of artificial neural networks for nonlinear time-series forecasting

G. Peter Zhang[a,*], B. Eddy Patuwo[b], Michael Y. Hu[b,c]

[a]*Department of Decision Sciences, J. Mack Robinson College of Business, Georgia State University,*
Atlanta, GA 30303-3083, USA
[b]*Graduate School of Management, Kent State University, Kent, OH 44242-0001, USA*
[c]*Chinese University of Hong Kong, People's Republic of China*

Received 1 February 1999; received in revised form 1 July 1999

Abstract

This study presents an experimental evaluation of neural networks for nonlinear time-series forecasting. The effects of three main factors — input nodes, hidden nodes and sample size, are examined through a simulated computer experiment. Results show that neural networks are valuable tools for modeling and forecasting nonlinear time series while traditional linear methods are not as competent for this task. The number of input nodes is much more important than the number of hidden nodes in neural network model building for forecasting. Moreover, large sample is helpful to ease the overfitting problem.

Scope and purpose

Interest in using artificial neural networks for forecasting has led to a tremendous surge in research activities in the past decade. Yet, mixed results are often reported in the literature and the effect of key modeling factors on performance has not been thoroughly examined. The lack of systematic approaches to neural network model building is probably the primary cause of inconsistencies in reported findings. In this paper, we present a systematic investigation of the application of neural networks for nonlinear time-series analysis and forecasting. The purpose is to have a detailed examination of the effects of certain important neural network modeling factors on nonlinear time-series modeling and forecasting. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Artificial neural networks; Nonlinear time series; Forecasting; Simulation

* Corresponding author Tel.: + 404-651-4065; fax: + 404-651-3498.
E-mail address: gpzhang@sur.edu (G. Peter Zhang).

0305-0548/00/$ - see front matter © 2000 Elsevier Science Ltd. All rights reserved.
PII: S0305-0548(99)00123-9

1. Introduction

Since the invention of backpropagation algorithm [1] to train feedforward multi-layered neural networks a decade ago, artificial neural networks (ANNs) have been widely used for many types of problems in business, industry and science [2]. One major use of ANNs is for time-series forecasting. Many successful applications suggest that ANNs can be a promising alternative tool for both forecasting researchers and practitioners. Recently, Zhang et al. [3] presented a review of the current status in applications of neural networks for forecasting.

The popularity of ANNs is derived from the fact that they are generalized nonlinear forecasting models. Forecasting has been dominated by linear statistical methods for several decades. Although linear models possess many advantages in implementation and interpretation, they have serious limitations in that they cannot capture nonlinear relationships in the data which are common in many complex real world problems [4]. Approximation of linear models to complicated nonlinear forecasting problems is often not satisfactory. In the early 1980s, Makridakis [5] organized a large-scale forecasting competition (M-competition) in which the majority of commonly used *linear* forecasting methods were tested using 1001 real-time-series data. The results showed that no single forecasting method is globally the best. In our view, one of the major reasons for this conclusion is that there is a varying degree of nonlinearity in the data which cannot be handled properly by linear statistical methods.

During the past two decades, a number of nonlinear time-series models such as the bilinear model [6], the threshold autoregressive (TAR) model [7], the smoothing transition autoregressive (STAR) model [8], and the autoregressive conditional heteroscedastic (ARCH) model [9] have been developed (see De Grooijer and Kumar [10] and Tjostheim [11] for reviews of this field). While these models can be useful for a particular problem and data, they do not have a general appeal for other applications. The pre-specification of the model form restricts the usefulness of these parametric nonlinear models since there are too many possible nonlinear patterns. In fact, the formulation of an appropriate nonlinear model to a particular data set is a very difficult task compared to linear model building because "there are more possibilities, many more parameters and thus more mistakes can be made" ([12, p. 233]. Furthermore, one particular nonlinear specification may not be general enough to capture all nonlinearities in the data. As Diebold and Nason [13, p. 318] pointed out, "the overwhelming variety of plausible candidate nonlinear models makes determination of a good approximation to the true data-generating process (DGP) a difficult task and the seemingly large variety of parametric nonlinear models is in fact a very small subset of the class of plausible nonlinear DGPs".

As opposed to the model-based nonlinear methods, ANNs are nonparametric data driven approaches which can capture nonlinear data structures without prior assumption about the underlying relationship in a particular problem. ANNs are more general and flexible modeling and analysis tools for forecasting applications in that not only can they find nonlinear structures, they also can model linear processes. In fact, linear autoregressive (AR) models are special cases of ANNs without hidden nodes. In a recent forecasting competition [14], most participants have considered using neural networks and almost all of the best predictions for each data set are made with ANN models. This is not surprising given that all of the six time series used in that competition are quite nonlinear in nature [15].

Interest in using ANNs for forecasting has led to a tremendous surge in research activities in the past decade. Yet, researchers to date are still not certain about the effects of key modeling factors on ANNs' forecasting performance. The questions of whether, why, and under what conditions they are better than the established conventional forecasting methods still remain [3]. Most results or conclusions of neural network forecasting are obtained from limited empirical studies. The shot-gun methodology for specific problems is typically used by most researchers. Hence, the experimental designs in these studies are limited and results from these studies often cannot be extended to general applications. The lack of systematic investigations in ANN model building and forecasting is probably the primary cause of inconsistencies in reported findings. As Zhang et al. [3] pointed out that "given too many factors could affect the performance of the ANN method, limited empirical study alone may not be sufficient to address all the key issues".

The overall objective of this paper is to have a systematic investigation of ANNs for time-series analysis and forecasting. Specifically, the effects of three important factors on the neural network forecasting ability are examined by conducting an experimental study. The factors investigated include the number of input nodes, the number of hidden nodes, and the training sample size. Eight simulated nonlinear time series each with 30 replications are used and undergo detailed examinations. Using simulated time series for model selection [16], model evaluation [17], model comparison [18–20], and others [8,20,21] is quite common in the forecasting literature. Results from the simulation study are useful in general applications of ANNs and in providing guidelines for nonlinear time-series forecasting.

The paper is organized as follows. The next section reviews key issues in time-series forecasting with ANNs. Then we present the research design which is followed by the discussions of results. The final section provides concluding remarks.

2. Issues in ANNS for time-series forecasting

In this study, we focus on the feedforward multilayer networks, also known as the multilayer perceptrons (MLPs). This is the most popular type of ANNs used for forecasting purposes [3,22]. An MLP is typically composed of several layers of input, hidden and output nodes. For a univariate time-series forecasting problem, the inputs of the network are the past, lagged observations and the output is the predicted value. Each input pattern is composed of a moving window of fixed length along the series. A single output MLP actually performs the following mapping from the inputs to the output:

$$y_t = f(y_{t-1}, y_{t-2}, \dots, y_{t-p}), \tag{1}$$

where y_t is the observation at time t, p is the dimension of the input vector or the number of past observations used to predict the future, and f in general is a nonlinear function determined by the MLP structure and the data. From Eq. (1), the feedforward network can be viewed as a general nonlinear autoregressive model.

In neural network forecasting applications, total available data are usually divided into a training set (in-sample data) and a test set (out-of-sample or hold-out sample). The training set is used for the construction of the neural network while the test set is used for measuring the predictive ability of the model. The construction process of the network is called network training. Training is the process of determining the function f in Eq. (1) which is uniquely determined by the linking arc weights of the network. Suppose we have N time-lagged observations $y_1, y_2, \dots, y_N$ in the training set and we need the one-step-ahead forecasts, then using a network with p input nodes and one output node, we have $N-p$ training patterns. The first training pattern is composed of $y_1, y_2, \dots, y_p$ as the inputs and y_{p+1} as the target output. The second training pattern contains $y_2, y_3, \dots, y_{p+1}$ for the inputs and y_{p+2} for the desired output. Finally, the last training pattern is $y_{N-p}, y_{N-p+1}, \dots, y_{N-1}$ for the inputs and y_N for the target.

It is often not an easy task to build an MLP for time-series analysis and forecasting because of the large number of factors related to the model selection process. Although there are many rules of thumb proposed, none of them can be universally applied. Guidelines are either heuristic or obtained from limited empirical studies. This often causes inconsistent reports in the literature.

The most difficult problem is how to develop a network of appropriate size for capturing the underlying patterns in the training data. More importantly for a network model to be useful, it must have generalization or forecasting capability. Although several different methods such as the pruning algorithm [23,24], the polynomial time algorithm [25], the canonical decomposition technique [26], and the network information criterion [27] have been proposed for building the optimal architecture of an ANN, none of these methods can guarantee the best solution for all forecasting situations.

The size of an MLP largely depends on the number of input nodes and the number of hidden nodes. Theoretical results [28,29] prescribe that an MLP with one hidden layer is capable of approximating any continuous function. The number of input nodes is perhaps the most important parameter since it corresponds to the number of lagged observations used to discover the underlying patterns and/or autocorrelation structures in a time series. There are no systematic reports on the effect of input nodes. Lachtermacher and Fuller [22] observed both undesirable effects of more input nodes for one-step-ahead forecasting and good effects for multi-step prediction. They also found that correct identification of the number of input nodes is more important than the selection of the number of hidden nodes. Obviously, too few or too many input nodes can have significant impact on the learning and prediction ability of the network since the former will result in under-learning and the latter over-specification.

Hidden nodes are used to capture the nonlinear structures in a time-series. Determination of how many hidden nodes to use is another difficult issue in ANN model construction process. Since no theoretical basis exists to guide the selection, in practice the number of hidden nodes is often chosen through experimentation or by trial-and-error. Although ANN theory suggests that more hidden nodes typically lead to improved accuracy in approximating a functional relationship, they also cause the problem of overfitting, that is, the network fits the training data very well (learning everything including spurious features and noises) but generalizes or forecasts very poorly in out-of-samples. The overfitting problem is more likely to occur in neural network models than in other statistical models due to the typical large parameter set to be estimated. Universally accepted effective and systematic approaches to dealing with overfitting do not exist although several ways have been proposed. For example, different weight elimination and node pruning methods have been proposed by Weigend et al. [30–32], Cottrell et al. [33], and Schitienkopf et al. [34]. Regardless of the method used to overcome overfitting, the central idea is to find a parsimonious model that fits the data well. Generally, a parsimonious model not only gives adequate representation of the data, but also has the more important generalization capability. The principle of parsimony should be emphasized [35] and can be a guiding rule in selection of the number of hidden nodes. Another way to tackle the overfitting problem is to divide the time series into three sets of training, testing and validation parts [36]. The first two parts are used for model building with the last part used for validation or evaluation of the model. The best neural network model is the one which gives the best results in the test set. This approach, however, requires a large amount of data and may not be applicable in situations where data is limited. Tang and Fishwick [37] have studied the effect of hidden nodes on forecasting but no clear patterns were found.

The determination of other parameters of an MLP is relatively straightforward and less controversial. For example, the number of output nodes is often one for both one-step-ahead and multi-step-ahead forecasting. Of course, more output nodes can be included for direct multi-step forecasts [37,38].

Another closely related issue in ANN model building is how large the training and/or test sample sizes should use. Furthermore, given a data set, what is the best way to split up the data? In the ANN literature, large sample size for training is often suggested for sufficient learning and to ease the overfitting effect in training a neural network. However, Kang [39] found that neural network models do not necessarily require large data sets to perform well. ANN models forecast pretty well even with sample size less than 50 while the Box–Jenkins models typically require at least 50 observations to forecast successfully. The literature offers little guidance in selecting the training and test samples as well as their sizes. Granger [12], however, suggested that at least "20 percent of any sample should be held back for a post-sample, forecasting evaluation." It should be noted that the selection of data for training and test may affect both in-sample fitting and out-of-sample forecasting performance due to model uncertainty [40,41].

3. Experimental design

In order to clear some cloud in ANN forecasting applications outlined in the previous section, we have performed a simulation experiment. Our major purpose is to systematically study the effects of several key factors on the neural network forecasting performance. The number of input nodes, the number of hidden nodes, and the training sample size are selected as the experimental factors. The one-step-ahead forecasts are the focus of this study.

3.1. Data

Eight nonlinear univariate time series are generated from a variety of nonlinear models commonly used in the forecasting literature. Each type of time series is replicated 30 times using different initial random seeds for the error term. These nonlinear time series are listed below. In each case, ε_t is assumed to be i.i.d. $N(0, 1)$.

Series 1. Sign autoregressive (SAR) model

$$y_t = sign(y_{t-1}) + \varepsilon_t,$$

where

$$
\begin{aligned}
sign(x) &= 1 && \text{if } x > 0,\\
&= 0 && \text{if } x = 0,\\
&= -1 && \text{if } x < 0.
\end{aligned}
$$

Series 2. Bilinear model 1 (BL1)

$$y_t = 0.7y_{t-1}\varepsilon_{t-2} + \varepsilon_t.$$

Series 3. Bilinear model 2 (BL2)

$$y_t = 0.4y_{t-1} - 0.3y_{t-2} + 0.5y_{t-1}\varepsilon_{t-1} + \varepsilon_t.$$

Series 4. Threshold autoregressive (TAR) model

$$
\begin{aligned}
y_t &= 0.9y_{t-1} + \varepsilon_t && \text{for } |y_{t-1}| \leqslant 1,\\
&= -0.3y_{t-1} - \varepsilon_t && \text{for } |y_{t-1}| > 1.
\end{aligned}
$$

Series 5. Nonlinear autoregressive (NAR) model

$$y_t = \frac{0.7|y_{t-1}|}{|y_{t-1}| + 2} + \varepsilon_t.$$

Series 6. Nonlinear moving average (NMA) model

$$y_t = \varepsilon_t - 0.3\varepsilon_{t-1} + 0.2\varepsilon_{t-2} + 0.4\varepsilon_{t-1}\varepsilon_{t-2} - 0.25\varepsilon_{t-2}^2.$$

Series 7. Smooth transition autoregressive (STAR1) model

$$y_t = 0.8y_{t-1} - 0.8y_{t-1}[1 + \exp(-10y_{t-1})]^{-1} + \varepsilon_t.$$

Series 8. Smooth transition autoregressive (STAR2) model

$$y_t = 0.3y_{t-1} + 0.6y_{t-2} + (0.1 - 0.9y_{t-1} + 0.8y_{t-2})[1 + \exp(-10y_{t-1})]^{-1} + \varepsilon_t.$$

These eight time-series models are chosen to represent a variety of problems which have different characteristics in time-series. For example, some of the series have pure autoregressive (AR) or pure moving average (MA) correlation structures while others have mixed AR and MA components. Theoretical background and applications of some of these series can be found in [4,6,7,58,59].

3.2. Design of experiments

In this research, the number of input nodes, the number of hidden nodes, and the training sample size are the experimental factors. Five levels of input nodes from 1 to 5 are selected for experimentation. The selection of these levels of the input nodes is based on (1) the time-series models we selected all have lags of no more than 2; and (2) a majority of the real-time-series forecasting problems have AR terms in the order of 1 or 2 and very few are of order 3 or higher for nonseasonal time series [35,42]. Since feedforward neural networks per se are generalized nonlinear AR models, the upper limit of 5 for the number of input nodes seems reasonable. There is no upper limit on the possible number of hidden nodes in theory. However, from the literature, it is rare that the number of hidden nodes is more than double the number of input nodes. Hence, the number of hidden nodes in this study varies from 1 to 10.

Three levels of training sample size will be used for simulated series. They are 100, 200, and 400 respectively. For simplicity in illustration, training sample size will refer to the number of observations in a time series, not the number of training patterns which is often defined as the training sample size in the literature. However, these two are related through the number of input nodes. Let L be the training sample size (the length of a time-series data in a training set), N be the number of input nodes, then the number of training patterns will be $L - N$. The test sample for each case is composed of the last 80 data points. Three forecast horizons of 20, 40 and 80 are used to study the effect of forecasting horizon.

A factorial design is used to investigate the effects of these three factors on the performance of ANNs. Each of the eight time-series types is replicated 30 times using different initial random seeds for the error terms, yielding a total of 240 different time-series. Four hundred and eighty data points are generated for each individual series with the last 80 points used for testing purposes. Depending on the sample size, the most recent 100, 200, and 400 points are used as training samples. A 5×10 factorial layout with the number of input nodes (from 1 to 5) and the number of hidden nodes (from 1 to 10) is then applied to each of the total 720 simulated series with different sample sizes.

Since there is no consensus on the appropriate error measures for a forecasting situation [5,43–46], we elect to use the mean squared error (MSE) and the median absolute percentage error (MdAPE) as performance measures. The former is an absolute measure of forecasting accuracy and is appropriate for comparing different methods on the same data. The latter is a relative accuracy measure and is useful for comparison of methods on data with different scales. Both measures were recommended by Gardner [47] who discussed the reasons why these two measures are the most appropriate ones for forecasting comparisons.

For comparison purposes, the Box–Jenkins ARIMA [35] models are also used in each time-series. ARIMA is one of the most popular models in traditional time-series forecasting and is often used as a benchmark model for comparison with neural networks [3,37,39]. ARIMA modeling and forecasting is implemented by SCA statistical software [48]. In particular, we use the SCA-EXPERT function to automate the ARIMA model building. Results from neural networks are compared to those of Box–Jenkins models through paired t-tests.

3.3. *Neural networks*

The fully connected feedforward neural networks are used in this study. Only one hidden-layer MLPs are considered. The number of input nodes and the number of hidden nodes are used as major experimental factors. Since the one-step-ahead forecasting is exclusively examined, only one output node is employed. That is, we use actual rather than predicted values to forecast the future values.

The logistic function is used for all hidden nodes as the activation function. The linear activation function is employed for the output node. There is a bias term associated with the output node and each hidden node. The initial values for all arc weights and biases are uniformly distributed in the range of -5–5.

A GRG2-based system is used for training neural networks [49,50]. GRG2 [51] is a widely used optimization software which solves nonlinear optimization problems, such as those in neural network training, using the generalized reduced gradient method. As shown in a number of previous studies [50,52,53], the GRG2 training algorithm has many advantages over the popular backpropagation-based training systems. A number of parameters need to be specified for using GRG2, and most of them can be set at their default values. The stopping criteria is to terminate the program if the error function is not reduced by at least 10^{-5} for four consecutive iterations. The bounds on weights are set at 100.

Data normalization is not used since neural networks are believed to be able to adjust the weights automatically and adaptively [3]. Furthermore, because of the adoption of the linear output activation function, it is not necessary to normalize the output values. Previous studies [54] indicate that data normalization is not critical for the performance of neural networks.

4. Results

We perform SAS ANOVA with the Duncan option [57] for each of the eight time series to examine the effects of the number of input and hidden nodes along with three different sample sizes. Since three test sets of 20, 40 and 80 data points are used to represent different forecasting horizons, we have three sets of performance measures for each time horizon. The MSE used for these three horizons are denoted as MSE1, MSE2, and MSE3. The MdAPE is indexed as MdAPE1, MdAPE2, and MdAPE3, respectively. Because of the similarity in the results in all the time-series studied, we only report the detailed results for one series of the smooth transition autoregressive (STAR2) model. The overall results will also be discussed.

Table 1 gives the overall ANOVA result for the effects of input and hidden nodes as well as sample sizes on both training and test performance of neural networks. Notice that there are no interaction effects for all cases. The significant impact of input nodes on MSE and MdAPE is clearly seen for both training and test sets across different sample sizes. While the number of hidden nodes is significant on training MSE, it is not significant judging from training and test MdAPE. Hidden nodes have significant effects on test MSE only when the sample size is 100. As the sample size increases, the number of hidden nodes has lesser impact on both in-sample and out-of-sample performance of ANNs. When the sample size is 200 and 400, there is no significant hidden node effect on the test results. This finding seems to agree with our expectation that the number of input nodes is a more important factor for ANNs to identify the patterns in time-series.

Duncan's multiple range test [57] is employed to examine the specific effect of the input as well as hidden nodes on the training and testing performance. Because of the similarity in results across different sample sizes, only the results with sample size of 100 are used for illustrations. Figs. 1 and 2 display the results of input node effect on MSE and MdAPE, respectively. From Fig. 1, it is quite clear that as the number of input nodes increases, the training MSE decreases consistently. The test sample MSEs, however, exhibit a different pattern. Across the three test periods, MSE is the highest with one input node, achieves its minimum at two input nodes, and then monotonically increases after two input nodes. This is a common overfitting phenomenon in many ANN applications. As the model becomes more complex, the in-sample fit generally improves while out-of-sample performance gets worse. This observation also suggests that the in-sample MSE is not a good

Table 1
ANOVA results by sample size (F and P-values)

Sample size	Factor	Training		Test					
		MSE	MdAPE	MSE1	MdAPE1	MSE2	MdAPE2	MSE3	MdAPE3
100	Input	878.47 (0.0001)	87.81 (0.0001)	60.25 (0.0001)	8.93 (0.0001)	116.77 (0.0001)	11.37 (0.0001)	230.13 (0.0001)	15.17 (0.0001)
	Hidden	9.75 (0.0001)	1.29 (0.2373)	2.40 (0.0106)	0.38 (0.9468)	3.87 (0.001)	0.47 (0.8945)	80.13 (0.0001)	0.85 0.5674)
	Input × hidden	0.13 (1.0000)	0.23 (1.0000)	0.32 (1.0000)	0.20 (1.0000)	0.35 (0.9999)	0.11 (1.0000)	0.43 (0.9987)	0.13 (1.0000)
200	Input	1121.11 (0.0001)	154.65 (0.0001)	76.53 (0.0001)	16.88 (0.0001)	193.36 (0.0001)	11.43 (0.0001)	414.74 (0.0001)	16.05 (0.0001)
	Hidden	2.64 (0.0049)	1.35 (0.2070)	0.32 (0.9693)	0.13 (0.9988)	0.55 (0.8353)	0.09 (0.9997)	0.98 (0.4538)	0.13 (0.9990)
	Input × hidden	0.05 (1.0000)	0.24 (1.0000)	0.07 (1.0000)	0.05 (1.0000)	0.06 (1.0000)	0.04 (1.0000)	0.11 (1.0000)	0.07 (1.0000)
400	Input	1915.92 (0.0001)	220.87 (0.0001)	142.18 (0.0001)	29.05 (0.0001)	281.05 (0.0001)	13.75 (0.0001)	604.82 (0.0001)	23.43 (0.0001)
	Hidden	1.84 (0.0573)	1.26 (0.2526)	0.05 (1.0000)	0.07 (0.9999)	0.09 (0.9998)	0.05 (1.0000)	0.13 (0.9989)	0.06 (0.9999)
	Input × hidden	0.04 (1.0000)	0.10 (1.0000)	0.04 (1.0000)	0.03 (1.0000)	0.02 (1.0000)	0.02 (1.0000)	0.03 (1.0000)	0.03 (1.0000)

criterion for model selection. Judging from the test sample results, the best number of input nodes for this series is two, although there is no significant difference between using two and three input nodes. The result is consistent for different sample sizes across different forecasting horizons. This indicates the ability of neural networks to correctly identify the number of lagged observations used to predict future values since in this STAR series the future value should be related to the past two observations. Note that using only one input node yields the worst performance for both training and test samples, indicating the network is not able to learn sufficiently from one past observation alone — an under-learning situation.

Using MdAPE as the performance measure, Fig. 2 shows very similar pattern as in Fig. 1. Again, we do not find any significant differences in using 2–5 input nodes for almost all situations. If the principle of parsimony is applied, then two input nodes should be the best choice to model this series.

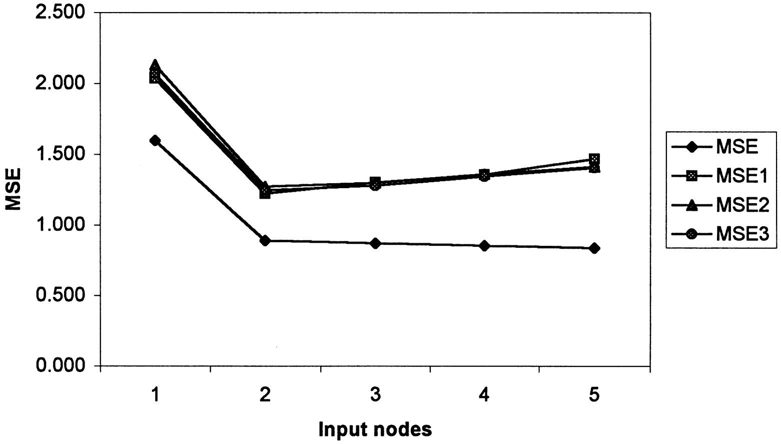

Fig. 1. The effect of input nodes on MSE.

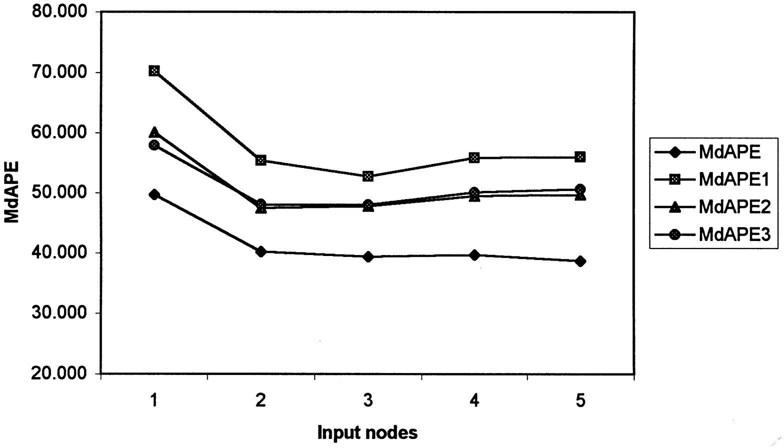

Fig. 2. The effect of input nodes on MdAPE.

The hidden node effects on MSE and MdAPE are illustrated in Figs. 3 and 4. The overall patterns of MSE and MdAPE are similar to those observed in Figs. 1 and 2. As more hidden nodes are used, in-sample MSEs and MdAPEs consistently decrease while test MSEs and MdAPEs decrease from one hidden node to two hidden nodes and then gradually increase. This overfitting effect, however, is found relatively small when larger sample sizes of 200 and 400 are used. That is, larger sample sizes may reduce the effects of overfitting. In general, there is no significant difference among different levels of hidden nodes although one or two hidden nodes are found to have the lowest test sample MSE and MdAPE.

Another general observation from Figs. 1–4 is that test set results are worse than those in the training sample no matter what performance measure is used. This supports the findings of other empirical studies, for example, Chatfield [55] and Fildes and Makridakis [56].

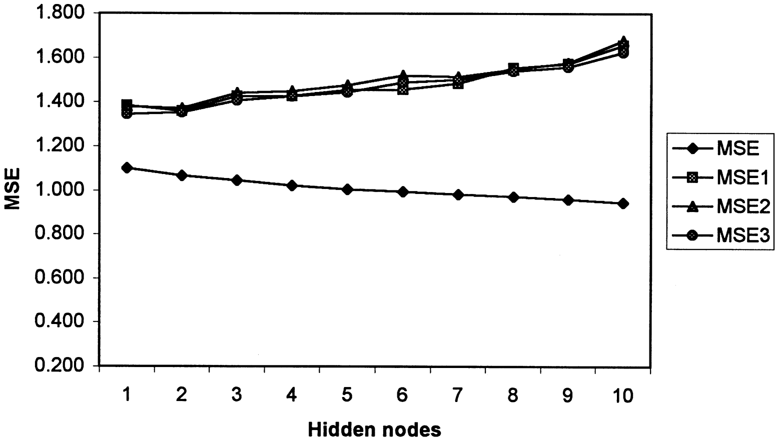

Fig. 3. The effect of hidden nodes on MSE.

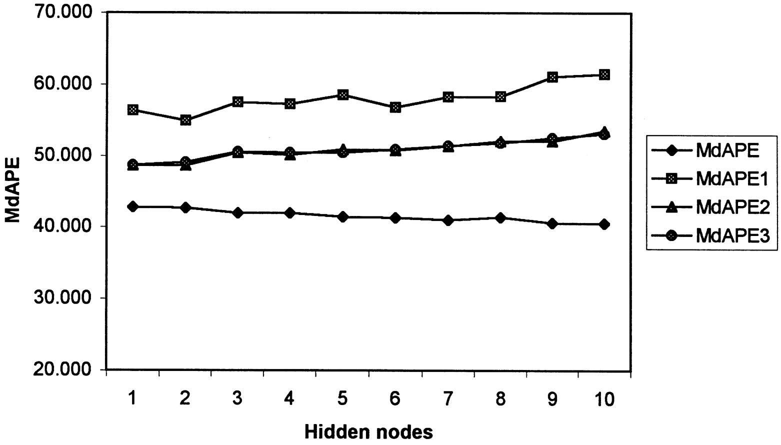

Fig. 4. The effect of hidden nodes on MdAPE.

In Table 2, paired-t tests are performed comparing the forecasting performance of neural networks with that of the Box–Jenkins model. The test statistic used is $t = \sqrt{n}\bar{d}/s_d$, where n is the sample size, $\bar{d}$ is the average of individual performance differences (d) between ANNs and ARIMA models, and s_d is the standard deviation of variable d. As mentioned earlier, the Box–Jenkins model is implemented with the SCA statistical system, using SCA-EXPERT capabilities. The SCA-EXPERT function employs the expert system technology to facilitate automatic ARIMA modeling. Through the iterative process of model identification, parameter estimation, and diagnostic checking, Box–Jenkins method is assumed to produce the best linear model fitted to the data. It is an accepted statistical paradigm that the correctly specified ARIMA model for the historical data would also be the optimum model for the forecasting purpose [56].

Table 2
Paired comparisons of forecasting performance: ANNs vs. BJ models

Sample size	Statistics	MSE1	MdAPEP1	MSE2	MdAPE2	MSE3	MdAPE3
100	Difference[a]	– 0.3245	– 13.6439	– 0.2874	– 11.3222	– 0.1905	– 11.2879
	P-value	0.0001	0.0001	0.0005	0.0003	0.0002	0.0001
200	Difference	– 0.2967	– 11.8821	– 0.2361	– 9.6224	– 0.1541	– 9.6507
	P-value	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
400	Difference	– 0.2432	– 11.4219	– 0.2049	– 8.8767	– 0.1497	– 8.9048
	P-value	0.0001	0.0001	0.0001	0.0005	0.0001	0.0001

[a] Difference = ANNs — BJ Models.
Negative values indicate preference to ANNs.

Table 3
The best ANN structure for nonlinear time series

Time series model	AR terms	MA terms	Mixed terms	Discontinuity	Number of input nodes	Number of hidden nodes
1. SAR	AR(1)	—	—	Yes	1	1 or 2
2. BL1	—	—	Yes	—	1	1
3. BL2	AR(2)	—	Yes	—	2	1 or 2
4. TAR	AR(1)	—	—	Yes	2 or 3	1 or 2
5. NAR	AR(1)	—	—	Yes	1	1
6. NMA	—	MA(2)	Yes	—	1 or 2	1
7. STAR1	AR(1)	—	Yes	—	1	1 or 2
8. STAR2	AR(2)	—	Yes	—	2	1 or 2

Table 2 clearly shows the superiority of neural network models over Box–Jenkins models. The difference in the table represents the performance measure of ANNs minus that of Box–Jenkins. Hence, negative values indicate preference for ANNs. All the differences are significant at the 0.0005 level, indicating that ANNs are able to produce significantly better forecasting for different time horizons and sample sizes. This may not be surprising since Box–Jenkins models are linear. They cannot capture nonlinear structures and hence predict poorly in nonlinear time-series.

Results from other series are very similar to those obtained above. The similarity in the patterns of input and hidden node effect as well as sample size effect is obvious. The major difference among different series is in the selected number of input nodes and/or hidden nodes required for model building and forecasting. Table 3 gives the overall result in terms of the selected network structure for each time-series. It also lists the major characteristic components of the eight nonlinear series. We classify nonlinear time-series based on their components of autoregressive (AR) terms, moving average (MA) terms, mixed AR and MA terms, and if the series present any discontinuities. It is seen from the table that for most series ANNs can identify the optimal number of input nodes, which is the number of AR lags. Discontinuity and mixed AR/MA terms seem to have no significant impact on the selection of an ANN model. For the nonlinear moving average (NMA) series, we find either 1 or 2 input nodes can give the best forecasting performance. One special case is in the threshold autoregressive (TAR) model with 1 AR lag where the optimal number of input nodes is 2 or 3. Because of the limit cycle property of a TAR model [7], a simple TAR model can have a very complicated structure. On the other hand, one or at most two hidden nodes usually give the best forecasts, suggesting simple ANN structure is often desirable for forecasting purposes.

5. Conclusions

We presented an experimental study on the application of neural networks for nonlinear time-series forecasting. Specifically, the effects of input nodes, hidden nodes and sample size on ANN modeling and forecasting behavior have been investigated. We considered eight time-series models with different characteristics. The conclusions from this study are summarized below:

(1) Both the number of input nodes and the number of hidden nodes have significant effects on ANN model building and predictive ability. Generally, the number of input nodes has much stronger effects than the number of hidden nodes in both in-sample fit and out-of-sample forecasting. This suggests that users should pay more attention to selecting the number of input nodes.
(2) Neural networks are able to identify the appropriate number of lagged observations for forecasting future values. Hence, neural networks can be a useful tool in analyzing the characteristics of time series such as the autocorrelation structure.
(3) Simple or parsimonious neural networks are effective in forecasting. For most time-series investigated, neural networks with one or two hidden nodes typically give the best forecasting performance in terms of MSE and MdAPE.
(4) ANNs are shown to be more competent than Box–Jenkins models in forecasting nonlinear time series. This seems to be quite obvious given the limitations of linear models.
(5) The number of observations (training sample size) used for training a neural network has limited effects on performance. In most cases, there is no significant difference in forecasting capability for the three sample sizes investigated. However, more data is found helpful for overcoming overfitting problems.

Neural networks in this study show much promise for nonlinear time-series forecasting. The ANN model is demonstrated to have nonlinear pattern recognition capability which is valuable for modeling and forecasting complex nonlinear problems in practice. It should be noted, however, that the findings reported in this study are subject to the constraint imposed by the design of experiment. Future research should focus on the development of ANN model selection criteria or model building methodology with a special emphasis on the theoretical aspect of the nonlinear structure of time-series.

References

[1] Rumelhart DE, Hinton GE, Williams RJ. Learning internal representation by back-propagating errors. In: Rumelhart DE, McCleland JL, and the PDP Research Group, editors. Parallel distributed processing: explorations in the microstructure of cognition. MA: MIT Press, 1986.

[2] Widrow B, Rumelhart DE, Lehr MA. Neural networks: applications in industry, business and science. Communications of the ACM 1994;37:93–105.

[3] Zhang G, Patuwo BE, Hu MY. Forecasting with artificial neural networks: the state of the art. International Journal of Forecasting 1998;14:35–62.

[4] Granger CWJ, Terasvirta T. Modelling nonlinear economic relationships. Oxford: Oxford University Press, 1993.

[5] Makridakis S, Anderson A, Carbone R, Fildes R, Hibdon M, Lewandowski R, Newton J, Parzen E, Winkler R. The accuracy of extrapolation (time series) methods: results of a forecasting competition. Journal of Forecasting 1982;1:111–53.

[6] Granger CWJ, Anderson AP. An introduction to bilinear time series models. Göttingen: Vandenhoeck and Ruprecht, 1978.

[7] Tong H, Lim KS. Threshold autoregression, limit cycles and cyclical data. Journal of Royal Statistical Society B 1980;42:245–92.

[8] Chan WS, Tong H. On tests for non-linearity in time series analysis. Journal of Forecasting 1986;5:217–28.

[9] Engle RF. Autoregressive conditional heteroscedasticity with estimates of the variance of UK inflation. Econometrica 1982;50:987–1008.

[10] De Gooijer JG, Kumar K. Some recent developments in non-linear time series modelling, testing, and forecasting. International Journal of Forecasting 1992;8:135–56.

[11] Tjostheim D. Non-linear time series: a selective review. Scandinavian Journal of Statistics 1994;21:97–130.

[12] Granger CWJ. Strategies for modelling nonlinear time-series relationships. The Economic Record 1993; 69:233–8.

[13] Diebold FX, Nason JA. Nonparametric exchange rate prediction? Journal of International Economics 1990;28:15–332.

[14] Weigend AS, Gershenfeld NA. Time series prediction: forecasting the future and understanding the past. Reading, MA: Addison-Wesley, 1993.

[15] Gershenfeld NA, Weigend AS. The future of time series: learning and understanding. In: Weigend AS, Gershenfeld NA, editors. Time series prediction: forecasting the future and understanding the past. Reading, MA: Addison-Wesley, 1993. p. 1–70.

[16] Gardner ES, Dannenbring DG. Forecasting with exponential smoothing: some guidelines for model selection. Decision Sciences 1980;11:370–83.

[17] Adam EE. Individual item forecasting model evaluation. Decision Sciences 1973;4:458–70.

[18] Dielman TE. A comparison of forecasts form least absolute value and least squares regression. Journal of Forecasting 1986;5:189–95.

[19] Lin JL, Granger CWJ. Forecasting from non-linear models in practice. Journal of Forecasting 1994;13:1–9.

[20] Davies N, Petruccelli JD. Detecting non-linearity in time series. The Statistician 1986;35:271–80.

[21] Lee TW, White H, Granger CWJ. Testing for neglected nonlinearity in time series models. Journal of Econometrics 1993;56:269–90.

[22] Lachtermacher G, Fuller JD. Backpropagation in time-series forecasting. Journal of Forecasting 1995;14:381–93.

[23] Sietsma J, Dow R. Neural net pruning-why and how? In: Proceedings of the IEEE International Conference on Neural Networks 1988;1:325–33.

[24] Reed R. Pruning algorithms — a survey. IEEE Transactions on Neural Networks 1993;4:740–7.

[25] Roy A, Kim LS, Mukhopadhyay S. A polynomial time algorithm for the construction and training of a class of multilayer perceptrons. Neural Networks 1993;6:535–45.

[26] Wang Z, Massimo CD, Tham MT, Morris AJ. A procedure for determining the topology of multilayer feedforward neural networks. Neural Networks 1994;7:291–300.

[27] Murata N, Yoshizawa S, Amari S. Network information criterion-determining the number of hidden units for an artificial neural network model. IEEE Transactions on Neural Networks 1994;5:865–72.

[28] Cybenko G. Approximation by superpositions of a sigmoidal function. Mathematical Control Signals Systems 1989;2:303–14.

[29] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. Neural Networks 1989;2:359–66.

[30] Weigend AS, Huberman BA, Rumelhart DE. Predicting the future: a connectionist approach. International Journal of Neural Systems 1990;1:193–209.

[31] Weigend AS, Huberman BA, Rumelhart DE. Predicting sunspots and exchange rates with connectionist networks. In: Casdagli M, Eubank S, editors. Nonlinear modeling and forecasting. Redwood City: Addison-Wesley, 1992.

[32] Weigend AS, Rumelhart DE, Huberman BA. Generalization by weight-elimination with application to forecasting. Advances in Neural Information Processing Systems 1991;3:875–82.

[33] Cottrell M, Girard B, Girard Y, Mangeas M, Muller C. Neural modeling for time series: a statistical stepwise method for weight elimination. IEEE Transactions on Neural Networks 1995;6:1355–64.

[34] Schittenkopf C, Deco G, Brauer W. Two strategies to avoid overfitting in feedforward networks. Neural Networks 1997;10:505–16.

[35] Box GEP, Jenkins GM. Time series analysis: forecasting and control. San Francisco: Holden-Day, 1976.

[36] Kaastra I, Boyd M. Designing a neural network for forecasting financial and economic time series. Neurocomputing 1996;10:215–36.

[37] Tang Z, Fishwick PA. Feedforward neural nets as models for time series forecasting. ORSA Journal on Computing 1993;5:374–85.

[38] Zhang X. Time series analysis and prediction by neural networks. Optimization Methods and Software 1994;4:151–70.

[39] Kang S. An investigation of the use of feedforward neural networks for forecasting. PhD Thesis, Kent State University, 1991.

[40] Chatfield C. Model uncertainty, data mining and statistical inference. Journal of Royal Statistical Society, A 1995;158:419–66.

[41] Chatfield C. Model uncertainty and forecast accuracy. Journal of Forecasting 1996;15:495–508.

[42] Pankratz A. Forecasting with univariate Box–Jenkins models: concepts and cases. New York: Wiley, 1983.

[43] Makridakis S. Wheelwright SC, McGee VE. Forecasting: methods and applications. 2nd ed. New York: Wiley, 1983.

[44] Armstrong JS, Fildes R. Correspondence: on the selection of error measures for comparisons among forecasting methods. Journal of Forecasting 1995;14:67–71.

[45] Armstrong JS, Collopy F. Error measures for generalizing about forecasting methods: Empirical comparisons International Journal of Forecasting 1992; 8: 69–80, 99–111.

[46] Yokum JT, Armstrong JS. Beyond accuracy: comparison of criteria used to select forecasting methods. International Journal of Forecasting 1995;11:591–7.

[47] Gardner ES. The trade-offs in choosing a time series method. Journal of Forecasting 1983;2:263–7.

[48] SCA (Scientific Computing Associates). The SCA statistical system reference manual (Version 4.3). Oak Brook: SCA, 1994.

[49] Hung MS, Denton JW. Training neural networks with the GRG2 nonlinear optimizer. European Journal of Operational Research 1993;69:83–91.

[50] Subramanian V, Hung MS. A GRG2-based system for training neural networks: design and computational experience. ORSA Journal on Computing 1993;5:386–94.

[51] Lasdon LS, Waren AD. GRG2 user's guide. School of Business Administration, University of Texas at Austin, 1986.

[52] Patuwo E, Hu MY, Hung MS. Two-group classification using neural networks. Decision Science 1993;24:825–45.

[53] Lenard MJ, Alam P, Madey GR. The application of neural networks and a qualitative response model to the auditor's going concern uncertainty decision. Decision Science 1995;26:209–26.

[54] Shanker M, Hu MY, Hung MS. Effect of data standardization on neural network training. Omega 1996;24:385–97.

[55] Chatfield C. Calculating interval forecasts (with discussion). Journal of Business and Economic Statistics 1993;11:121–44.

[56] Fildes R, Makridakis S. The impact of empirical accuracy studies on time series analysis and forecasting. International Statistical Review 1995;63:289–308.

[57] Miller R. Simultaneous statistical inferences. 2nd ed. New York: Springer, 1981.

[58] Terasvirta T, Anderson HM. Characterizing nonlinearities in business cycles using smooth transition autoregressive models. Journal of Applied Econometrics 1992;7:S119–S36.

[59] Tong H. A personal overview of non-linear time series analysis from a chaos perspective 1995;22:399–445.

G. Peter Zhang is Assistant Professor of Decision Sciences at Georgia State University. He received his Ph.D. in Operations Management/Operations Research from Kent State University. His research interests include neural networks and time-series forecasting. His research has appeared in *Decision Sciences*, *European Journal of Operational Research*, *International Journal of Forecasting*, *OMEGA*, and others.

B. Eddy Patuwo is Associate Professor in the Administrative Sciences Department at Kent State University. He earned his Ph.D. in IEOR from Virginia Polytechnic Institute and State University. His research interests are in the study of stochastic inventory systems and neural networks. His work has been published in *Decision Sciences*, *IIE Transactions*, *Computers and Operations Research*, *Journal of Operational Research Society*, among others.

Michael Y. Hu is Professor of Marketing at Kent State University and a visiting professor at the Chinese University of Hong Kong. He earned his Ph.D. in Management Science from the University of Minnesota in 1977. He has published more than 80 papers in the areas of neural networks, marketing research, international business, and statistical process cotrol. His articles have been published in numerous journals including *Decision Sciences*, *Computers and Operations Research*, *OMEGA*, *Journal of International Business Studies*, *Financial Management*, and many others.