Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

ELSEVIER

Applied Soft Computing 2 (2003) 255–268

Applied Soft Computing

www.elsevier.com/locate/asoc

Application of neural networks in forecasting engine systems reliability

K. Xu[*], M. Xie, L.C. Tang, S.L. Ho

Department of Industrial and System Engineering, National University of Singapore, Engineering Driver 2, 117576 Singapore, Singapore

Received 20 January 2001; received in revised form 19 December 2001; accepted 22 October 2002

Abstract

This paper presents a comparative study of the predictive performances of neural network time series models for forecasting failures and reliability in engine systems. Traditionally, failure data analysis requires specifications of parametric failure distributions and justifications of certain assumptions, which are at times difficult to validate. On the other hand, the time series modeling technique using neural networks provides a promising alternative. Neural network modeling via feed-forward multilayer perceptron (MLP) suffers from local minima problems and long computation time. The radial basis function (RBF) neural network architecture is found to be a viable alternative due to its shorter training time. Illustrative examples using reliability testing and field data showed that the proposed model results in comparable or better predictive performance than traditional MLP model and the linear benchmark based on Box–Jenkins autoregressive-integrated-moving average (ARIMA) models. The effects of input window size and hidden layer nodes are further investigated. Appropriate design topologies can be determined via sensitivity analysis.

Keywords: Time series forecasting; Neural networks; RBF model; Reliability analysis; MLP model; ARIMA models; Predictive performance

1. Introduction

Reliability modeling and prediction plays a very important role for assessing the performance of engineering systems. Techniques based on lifetime distribution models, parts count and parts stress, fault tree analysis and Markov models have been developed to forecast reliability. Although these methods are widely used, they impose certain restrictions on the failure classes such as infant mortality, random or wear out failure patterns. A priori assumptions on the failure distributions need to be specified in the reliability modeling process, which at times are difficult to validate. In addition, only reliability measures in some fixed time interval are predicted and it is not easy to forecast the variability of reliability indices with time. Accurate estimates of reliability indices are of increasing importance in reliability-related decision making in industry. Advanced knowledge of reliability information would allow a more accurate forecast of spares requirements, support costs, warranty costs and hence, appropriate preventive maintenance and corrective maintenance plans can be initiated. Especially for automotive industries, the main concern is to satisfy the increasing demands from customers and conform to stricter acts and regulations by governments.

In the existing literatures, the use of neural networks is not widespread in reliability engineering, especially

[*] Corresponding author. Tel.: +65-687-446-08; fax.: +65-677-714-34.
E-mail address: kaixu@nus.edu.sg (K. Xu).

in engine systems analysis. Liu et al. [1] demonstrated how feed-forward multilayer perceptron (MLP) networks can successfully identify underlying failures distribution and estimate the parameters. Amjady and Ehsan [2] presented an expert system based on neural networks in evaluating the reliability of power systems. However, instead of evaluating the reliability of the complex systems, this paper attempts to forecast reliability by analyzing the past historical failure data information, using neural network techniques. We proposed using radial basis function (RBF) neural network model in predicting failures and reliability due to its inherent advantages outlined in Section 3. In recent years, there has been an increasing interest in using RBF modeling for time series analysis [3], for fault diagnosis and identification in power systems [4], and also in contingency analysis of power systems [5]. Furthermore, positive results in using RBF for time series prediction are shown by Whitehead and Choate [6], where lower prediction errors are achieved for predicting the Mackey–Glass time series.

This paper is organized as follows: Section 2 presents some approaches for time series modeling and describes the framework of a time series reliability forecasting model. Section 3 describes the architecture of the RBF neural network. In Section 4, case studies on analyzing the failure data from turbochargers of diesel engines and car engines are illustrated to demonstrate the proposed approach in predicting failures and reliability. A comparative study of the forecast errors with both MLP and Box–Jenkins autoregressive-integrated-moving average (ARIMA) models is summarized. The effects of model parameters on the predictive performance and sensitivity analysis are discussed in Section 5. The concluding remarks are presented in Section 6.

2. Time series modeling approaches for systems reliability

2.1. Box–Jenkins models

In assessing the reliability of engine systems, time series modeling approach offers a viable alternative to fitting parametric failure distributions. The failure data or some equivalent reliability indices can be construed as a time series $x(t-p), \dots, x(t-2), x(t-1), x(t)$. Traditionally, time series forecasting problem is tackled using linear techniques such as the autoregressive, moving average and autoregressive-integrated-moving average models popularized by Box and Jenkins [7]. The general form of the ARIMA model is given by:

$$y_t = a_0 + \sum a_i y_{t-1} + \sum b_j e_{t-j},\quad i = 1, \dots, p \quad \text{and} \quad j = 0, 1, \dots, q$$

where y_t is a stationary stochastic process with non-zero mean, a_0 is the constant coefficient, e_i the white noise disturbance term, a_i represents the autoregressive coefficients, and b_j denotes the moving average coefficients. The time series linear models are widely used due to its simplicity, flexibility and more importantly, the systematic model building approach that allows even the non-specialists to get the essence of the methodology. ARIMA models are preferred over methods based on Bayesian approach [8,9] which is constrained by the necessary conditions pertaining to the failure process, which itself might be arbitrary. In reliability analysis, no a priori specification of linear models for the failure process is necessary. However, Box–Jenkins models are sometimes inadequate for situations where the underlying failure behavior varies dynamically with time. Hence, other approaches based on nonlinear techniques such as neural networks can be a promising alternative.

2.2. Neural network models

The artificial neural networks present an important class of nonlinear prediction model family that has generated considerable interest in the forecasting community in the past decade. While parameters of the aforementioned nonlinear models need to be determined, neural networks are appealing because no a priori postulation of the models is necessary for the system or process under consideration. The model parameters are iteratively adjusted and optimized through network learning of historical patterns. As time series prediction is performed entirely by inference of future behavior from examples of past behavior, neural networks are therefore viable alternatives that could lead to improved predictive performance. Neural nets have found to be the domain for numerous successful applications of prediction tasks [10];

particularly in electric load forecasting [11], economic forecasting [12], river flow forecasting [13], etc.

Traditionally, the standard feed-forward MLP neural network architecture, trained using the backpropagation algorithm, is commonly used as benchmark in time series forecasting. However, the long training time and local minima problems are often impediment to their applications. Alternative architectures like the radial basis function [14,15] neural network presents a promising model for forecasting the future reliability indices of engine systems. This network typically trains faster than conventional MLP networks [16] because it can choose suitable parameters of hidden layer units without having to perform a full nonlinear optimization of networks and the relationship is linear between hidden layer and output layer. At the same time, it possess the universal function approximation capabilities just as in MLP [17]. As RBF's hidden layer nodes influence the outputs of the network only for inputs near its center, an exponential number of neighborhoods to cover the entire domain are required. This suggests that RBF are well suited for time series problems with a small number of inputs [18], which is the key to successful parsimonous model building.

2.3. *The proposed time series forecasting of reliability framework*

Our experiments are conducted in accordance to the general time series forecasting model represented in this form:

$$X_t = f(X', A) \tag{1}$$

where X' is a vector of lagged variables $\{x_{t-1}, x_{t-2}, \ldots, x_{t-p}\}$, $A = \{a_t\}$ denotes a vector of the external explanatory variables, i.e. variables on which the time series is thought to have a dependence. In reliability analysis, X can be represented in different forms such as "time between failures", "time-to-failure" or "total failure numbers per unit time interval." The neural network approach attempts to discover the appropriate internal representation of the time series reliability data. The key to the solution of forecasting problems is how to approximate the function f. By iteratively adjusting the weights in the modeling process, the autocorrelation between the data can be explored and better estimates can be obtained.

In time series reliability analysis, neural networks can first be trained to learn the relationship between past historical reliability indices and the corresponding targets, and then predict future failures. Below is an illustration of how training patterns can be designed in the neural network modeling process.

$$\begin{array}{cccccc}
 & & X & & & Y \\
x_1 & x_2 & \cdots & x_p & a_p & x_{p+1} \\
x_2 & x_3 & \cdots & x_{p+1} & a_{p+1} & x_{p+2} \\
x_3 & x_4 & \cdots & x_{p+2} & a_{p+2} & x_{p+3} \\
\vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\
x_{t-p} & x_{t-p+1} & \cdots & x_{t-1} & a_{t-1} & x_t
\end{array}$$

where p denotes the number of lagged variables, $(t - p)$ is the total number of training samples. X represents the input nodes and Y is the predicted output node. After successful training, the neural network will be able to forecast future outcomes x_{t+k} at different time steps k. If $k = 1$, the prediction is a single-step-ahead (short-term) forecast and when $k > 1$, this leads to multi-step or long-term forecasts. Although multi-step forecasting may capture some system dynamics, the performance will be quite poor due to the accumulation of errors. In practice, short-term forecasting results are more useful as they provide timely information for preventive maintenance and corrective maintenance plans. Thus, we will only consider single-step-ahead predictions in our analysis. To evaluate the prediction errors between competing models, we use the normalized root mean square error measure (NRMSE), defined as follows:

$$\text{NRMSE} = \sqrt{\frac{\sum[x(t) - \hat{x}(t)]^2}{\sum x^2(t)}} \tag{2}$$

where $\hat{x}(t)$ is the forecast of $x(t)$.

Another index used is the improvement rate, which measures the relative improvement between the NN model under evaluation and the benchmark AR model:

$$\begin{aligned}&\text{improvement rate}\\ &\quad = \text{NRMSE}_{\text{AR}} - \text{NMRSE}_{\text{NN}}/\text{NRMSE}_{\text{AR}}\end{aligned}$$

where the subscripts of NRMSE refer to the specific model under study.

3. The RBF neural network for time series prediction

The RBF neural network model has been proven to be a universal function approximator, see for example [16,19]. As an extension of the MLP networks, it can perform similar function mappings, but its architecture and functionality are quite different. First, RBF are local networks as compared to the feed-forward networks that perform global mapping. This means that RBF uses a single set of processing units, each of which is most receptive to a local region of the input space. Second, the hidden layer in RBF performs nonlinear local mapping and its neurons, known as kernel, each has a centroid c_i and smoothing radius factor σ_i. Furthermore, the output layer performs linear transformation. Similar input vectors are clustered and input to the various hidden nodes. If an input vector lies near the centroid of a particular cluster, the hidden node will be activated. In other words, the output of the neuron decreases as the input is moved away from the centroid, at a rate determined by the radius.The ability of RBF to recognize whether an input is near the training set or if it is in an untrained region of the input space gives the RBF a significant advantage over the MLP structure. Furthermore, RBF networks can be trained more rapidly. The use of Gaussian activation functions can result in networks that learn more accurately and form a compact representation using small number of neurons. The basic architecture of the RBF comprising three layers is shown in Fig. 1. The input layer is made up of source nodes whose number is equal to the dimension of the input vector X. The second layer is the hidden layer composed of nonlinear units that are connected directly to all of the nodes in the input layer. The activation function of the individual hidden nodes are defined by the Gaussian function expressed as follows:

$$u_j = \exp\left[-\frac{||X - C_j||^2}{2\sigma_j^2}\right], \quad j = 1, 2, \dots, N \qquad (3)$$

where u_j denotes the output of the jth node in hidden layer, $X = [x_1, x_2, \dots, x_n]^{\mathrm{T}}$ is the input vector, C_j the position vector of the centers of the jth Gaussian function, σ_j^2 the width of the Gaussian function of the jth node and N denotes the number of hidden layer nodes. The output layer supplies the response of the network to the activation patterns applied to the input layer. The transformation from the input space to the hidden-unit space is nonlinear, whereas the transformation from the hidden-unit space to the output space is linear and the solution of the outputs has the form:

$$y_k = \sum_{j=1}^{N} w_{kj} u_j = W_k^{\mathrm{T}} U, \quad k = 1, 2, \dots, m \qquad (4)$$

and the linear weights vector associated with the output layer denoted as follows:

$$W_k = [w_{k1}, w_{k2}, \dots, w_{kN}]^{\mathrm{T}}$$

There are several techniques for determining the weights. The C_j and σ_j^2 are typically found using an unsupervised k-means clustering technique and

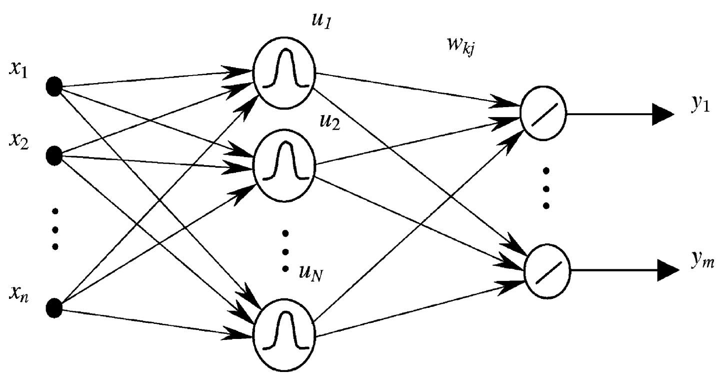

Fig. 1. Radial basis function network.

W_k is obtained by a supervised learning algorithm using steepest descent method. For effective learning, the input normalization and output encoding are very important. In this present study, the inputs are scaled from 0.1 and 0.9 as the network is less accurate in discriminating inputs close to the boundary values and the normalization process also span similar ranges. Note that the performance of the RBF is very sensitive to the design parameters. However, it is observed that k-means method can find suitable position and width of the basic functions from the reliability data. In order to prevent over-training, we adjust the number of hidden layer neurons step by step. A suitable number can be found by comprising the training and forecasting errors, i.e. when both of the training and forecasting errors approach to the minimum.

Table 1
Turbochargers failure data

Failure order number (i)	Time-to-failure (T/1000 h)
1	1.6
2	2.0
3	2.6
4	3.0
5	3.5
6	3.9
7	4.5
8	4.6
9	4.8
10	5.0
11	5.1
12	5.3
13	5.4
14	5.6
15	5.8
16	6.0
17	6.0
18	6.1
19	6.3
20	6.5
21	6.5
22	6.7
23	7.0
24	7.1
25	7.3
26	7.3
27	7.3
28	7.7
29	7.7
30	7.8
31	7.9
32	8.0
33	8.1
34	8.3
35	8.4
36	8.4
37	8.5
38	8.7
39	8.8
40	9.0

Table 2
Reliability of turbochargers

i	T_i/1000 h	$R(T_i)$
1	1.6	0.9930
2	2.0	0.9831
3	2.6	0.9731
4	3.0	0.9631
5	3.5	0.9532
6	3.9	0.9432
7	4.5	0.9333
8	4.6	0.9233
9	4.8	0.9133
10	5.0	0.9034
11	5.1	0.8934
12	5.3	0.8835
13	5.4	0.8735
14	5.6	0.8635
15	5.8	0.8536
16	6.0	0.8436
17	6.0	0.8337
18	6.1	0.8237
19	6.3	0.8137
20	6.5	0.8038
21	6.5	0.7938
22	6.7	0.7839
23	7.0	0.7739
24	7.1	0.7639
25	7.3	0.7540
26	7.3	0.7440
27	7.3	0.7341
28	7.7	0.7241
29	7.7	0.7141
30	7.8	0.7042
31	7.9	0.6942
32	8.0	0.6843
33	8.1	0.6743
34	8.3	0.6643
35	8.4	0.6544
36	8.4	0.6444
37	8.5	0.6345
38	8.7	0.6245
39	8.8	0.6145
40	9.0	0.6046

Table 3
Forecasting results of turbochargers reliability using different neural network architectures (without explanatory variable, $p = 3$, $N = 25$) and AR model

Number	Reliability (actual)	MLP (logistic activation)	MLP (Gaussian activation)	RBF (Gaussian activation)	AR
36	0.6444	0.6589	0.6515	0.6466	0.646000
37	0.6345	0.6532	0.6446	0.6369	0.639234
38	0.6245	0.6479	0.6383	0.6270	0.633807
39	0.6145	0.6430	0.6328	0.6170	0.629354
40	0.6046	0.6384	0.6278	0.6072	0.625562
NRMSE		0.0397	0.0250	0.0039	0.0199
Improvement rate		−99.497%	−25.628%	80.402%	

Table 4
A summary of reliability forecasts using different neural network models (with explanatory variables $p = 3$, $N = 10$)

Number	Reliability (actual)	MLP (logistic activation)	MLP (Gaussian activation)	RBF (Gaussian activation)
36	0.6444	0.6601	0.6539	0.6440
37	0.6345	0.6542	0.6476	0.6331
38	0.6245	0.6471	0.6441	0.6214
39	0.6145	0.6419	0.6389	0.6110
40	0.6046	0.6357	0.6360	0.6004
NRMSE		0.0384	0.0338	0.0046

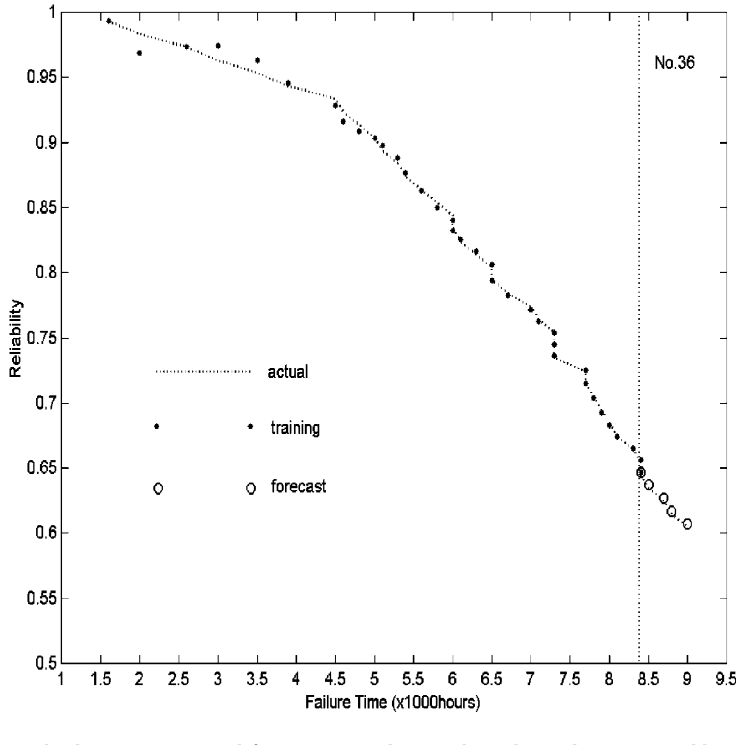

Fig. 2. Reliability turbocharger training and forecasting resulting without the explanatory variables ($p = 3$, $N = 25$).

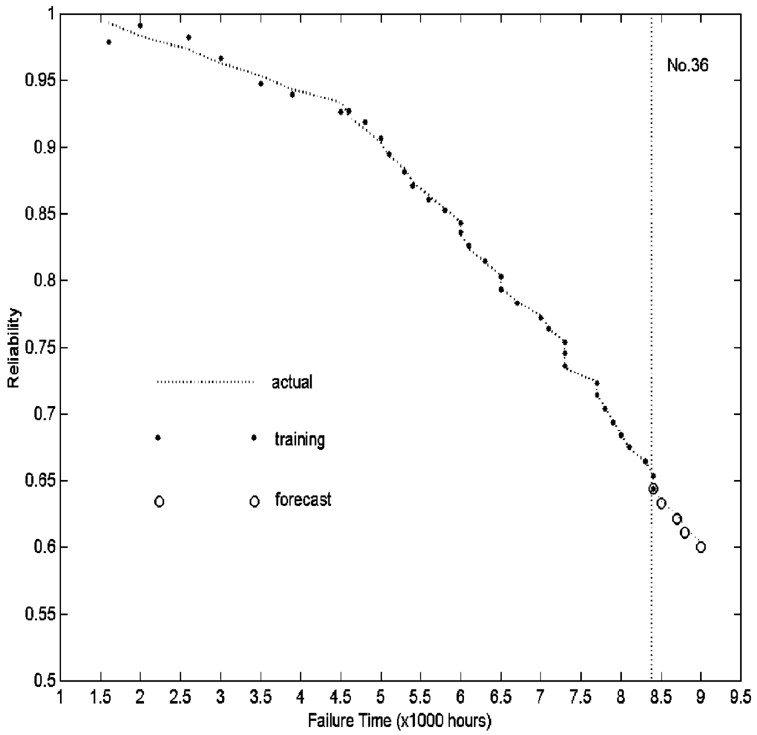

Fig. 3. Reliability turbocharger training and forecasting result with the explanatory variables ($p = 3$, $N = 10$).

4. Application examples and numerical comparisons

4.1. Reliability forecast of turbochargers in diesel engines

The turbocharger is a critical component in the turbo-charged diesel engine. As reliability is one of the most important considerations for diesel engine systems design, an accurate forecast of its reliability will provide a good assessment of its performance. Table 1 tabulates the original test records of the time-to-failure data for 40 suits of turbochargers. When analyzing ungrouped failure data, the cumulative failure distribution can be estimated from generating the median plotting positions for the ith ordered failures. This approach is preferred because the cumulative failure distribution is skewed for values

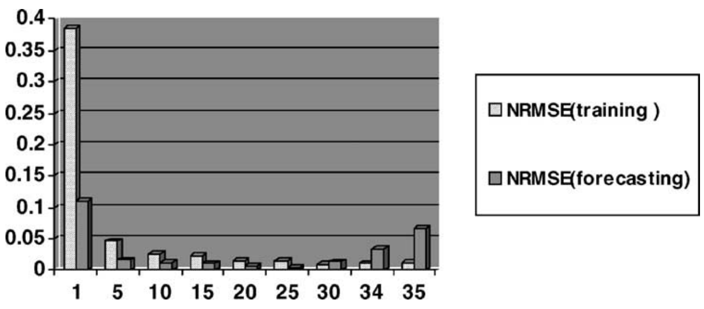

Fig. 4. The effect of hidden layer units N on training and testing forecast performance ($p = 3$, without explanatory variable).

of i close to zero and close to the sample size n. For skewed distributions, median ranking provides a better correction than mean ranking. The reliability estimates can thus be computed according to the following formula [20] due to Benard approximation.

$$R(T_i) = 1 - \frac{i - 0.3}{n + 0.4}$$

The estimated reliability information of turbochargers are summarized in Table 2.

In this example, two designs of training patterns are investigated. The first design includes the explanatory variable 'failure time' as an input node for

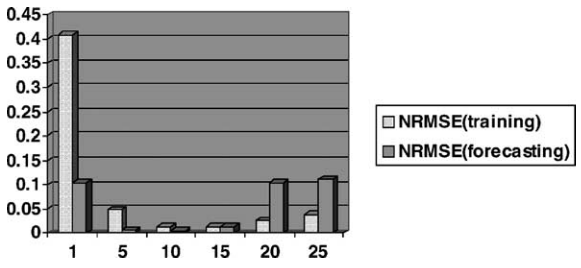

Fig. 5. The effect of hidden layer units N on training and testing forecast performance ($p = 3$, without explanatory variable).

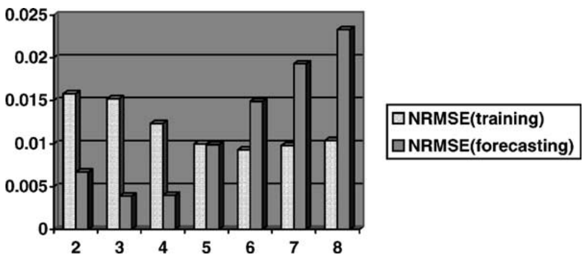

Fig. 6. The effect of lagged variables number p on training and testing forecast performance.

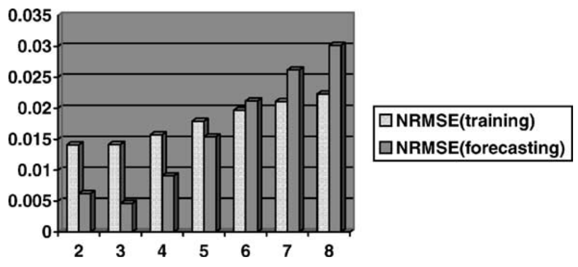

Fig. 7. The effect of lagged variables number p on training and testing forecast performance.

Table 5
Car engines' miles-to-failure (× 1000 miles)

Number	MTF
1	37.1429
2	37.4286
3	37.6190
4	38.5714
5	40.0000
6	35.8095
7	36.2857
8	36.2857
9	36.4762
10	38.1905
11	36.1905
12	36.8571
13	37.6190
14	37.8095
15	38.7619
16	35.9048
17	36.4762
18	36.8571
19	37.1429
20	37.4286
21	37.4286
22	37.6190
23	38.3810
24	38.5714
25	39.4286
26	35.8095
27	36.9524
28	37.6190
29	37.8095
30	38.0952
31	36.8571
32	38.0952
33	38.0952
34	38.3810
35	39.0476
36	37.2381
37	37.3333
38	37.5238
39	37.8095
40	38.5714
41	37.1429
42	37.2381
43	37.6190
44	38.1905
45	38.5714
46	36.0952
47	37.2381
48	37.4286
49	37.5238
50	39.0476
51	37.1429
52	37.8095
53	38.0952

Table 5 (*Continued*)

Number	MTF
54	38.6667
55	40.0619
56	36.1905
57	36.3810
58	37.0476
59	37.2381
60	38.0000
61	35.7143
62	36.4762
63	37.3333
64	37.6190
65	38.4762
66	36.8571
67	37.1429
68	37.9048
69	38.0952
70	38.8571
71	37.1429
72	37.6190
73	37.6190
74	37.8095
75	38.3810
76	36.3810
77	38.0000
78	38.1905
79	38.6667
80	38.6667
81	37.1429
82	37.6190
83	37.6190
84	38.0952
85	39.0476
86	36.2857
87	37.1429
88	37.5238
89	37.8095
90	38.0000
91	36.8571
92	37.0476
93	37.9048
94	38.1905
95	39.5238
96	35.4286
97	36.0000
98	37.7143
99	38.0952
100	38.5714

the input layer, whereas the other design excludes this variable in the training process. We have trained the proposed RBF network for 6000 iterations and the forecasting results for both designs, i.e. with and without the explanatory variables, are analyzed. At the same time, the predictive performances of both MLP and Box–Jenkins models are compared. In the ARIMA modeling process, it was found that the AR model is appropriate as it provides a good fit to the failure data. The comparative results are shown in Tables 3 and 4. The variables p and N denotes the design parameters for input window size and number of hidden nodes respectively.

Optimal results can be attained from simulation experiments. For the case with explanatory variable, the setting is $p = 3$ and $N = 10$. As for the design without the explanatory variable, $p = 3$ and $N = 25$. It is noted that the former (with explanatory variable) gives rise to higher forecast errors. In addition, the errors are also higher when identical network parameters ($p = 3$, $N = 25$) are simulated. The results here suggest that inputs with explanatory variables might not definitely enhance the predictive performance. The graphical comparisons between the actual and predicted reliability are shown in Figs. 2 and 3. It is observed that the proposed RBF neural model fits this particular data set very well. Figs. 4–7 investigate the effect of hidden layer nodes N and the number of lagged variables p on the training and testing forecast performance of RBF networks for the two designs. It is observed that network without the explanatory variable seems to have more satisfactory predictive performance. A design with 5 or 10 hidden nodes has a larger forecasting error than that with 25 hidden nodes. We also found that the training error is significantly larger, which means that the generalization is not good when the number of hidden nodes is 5 or 10.

4.2. *Miles-to-failure forecast of car engines*

The miles-to-failure data for 100 units of a specific brand of car engine is collected in Table 5. The objective is to forecast future miles-to-failure of car engines based on past failure observations. Hence, using the reliability forecasting framework explained earlier, the training and test patterns can be generated. We employ patterns 1–90 as the training samples and the last 10 from 91 to 100 as the testing samples.

After training the network for 20000 iterations, the predictive performances of the MLP, RBF and ARIMA models are summarized in Table 6. It is found that the RBF predictive performance is still satisfactory and

Table 6
Forecasting results of car engines' MTF using different neural network architectures ($p = 5$, $N = 16$)

Number	Miles-to-failure (actual) (× 1000 miles)	MLP (logistic activation) (× 1000 miles)	MLP (Gaussian activation) (× 1000 miles)	RBF (Gaussian activation) (× 1000 miles)	AR
91	36.85710	37.0259	36.9917	37.1096	37.3546
92	37.04760	37.8178	37.4988	37.7532	36.3213
93	37.90480	37.8254	37.7549	38.0020	37.0603
94	38.19050	38.5449	38.4238	37.9306	38.3013
95	39.52380	38.8205	39.4327	38.2064	38.7083
96	35.42860	36.5156	36.1145	36.9375	39.3056
97	36.00000	36.4749	36.3036	36.3272	34.177
98	37.71430	36.8491	36.9533	36.8791	36.3164
99	38.09520	37.9428	37.2913	37.2870	38.3696
100	38.57140	38.7185	38.3821	38.1781	40.1801
NRMSE		0.0156	0.0122	0.0211	0.0422
Improvement rate		63.03%	71.09%	50%	

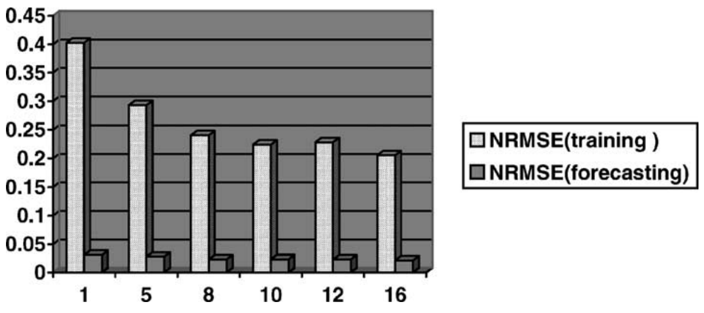

Fig. 8. The effect of hidden layer neurons N on training and testing forecast performance ($p = 5$).

comparable with both MLP and AR models. Higher training and testing errors are expected compared to the previous example due to the stochastic behavior and cyclic patterns of the time series. This is evident in Figs. 8 and 9 that evaluates the effects of hidden layer units N and lagged variables number p respectively, on the forecast performance of RBF network. The graphical plot of the actual and predicted miles-to-failure is presented in Fig. 10.

5. Discussions

Some observations can be drawn by analyzing Figs. 4–9. First, there appears to be a strong correlation between training error and forecasting error for most of the comparisons. Secondly, the parameters of RBF architecture (i.e. the number of hidden layer neurons N and the number of lagged variables P) affect the forecasting performance greatly. The sensitivity analysis curves of these two parameters are shown in Figs. 11 and 12, respectively. Note that for a more meaningful comparison, the y-axis of both the graphs has been normalized using $\mathrm{NRMSE}(i) = \log_{10}\{\mathrm{NRMSE}(i)/\min(\mathrm{NRMSE})\}$. The results showed that varying the number of hidden layer neurons affects the forecasting performance. Generally, the NRMSE decreases with increasing number of hidden layer neurons.

It can be seen that for the car engine failure data, the prediction error is the smallest when the maximum number of hidden layer neurons is specified, which could be obtained by the k-means clustering techniques. This is reasonable because with more hidden neurons in the RBF one can thus produce finer grid for the inputs by using a suitable clustering method.

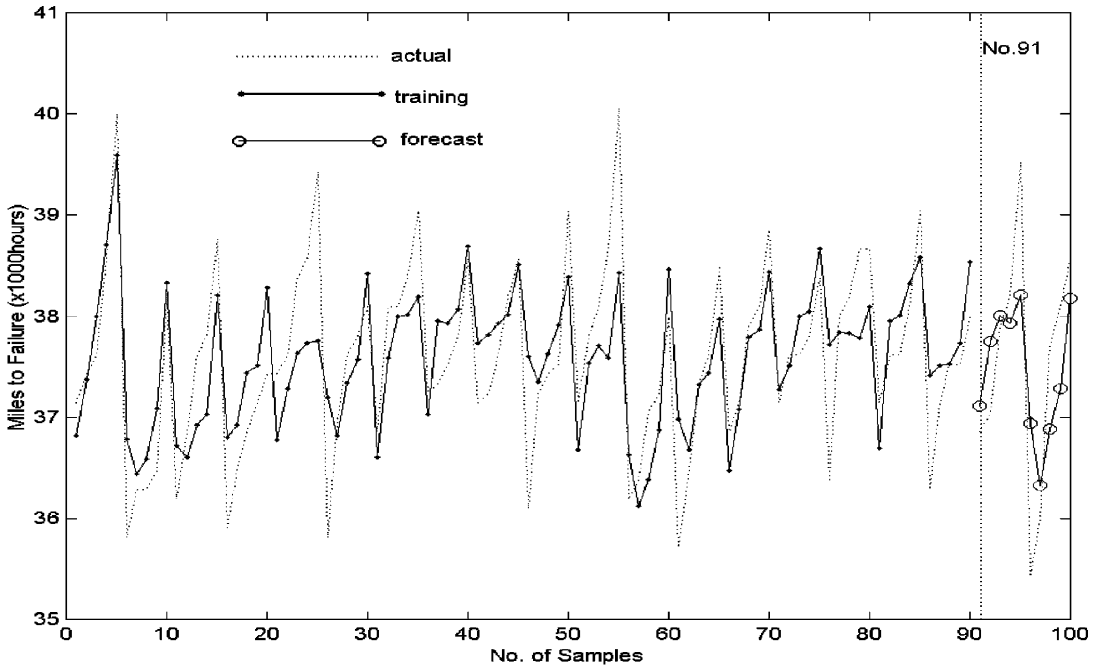

Fig. 9. Predictive performance of MTF for car engines ($p = 5$, $N = 16$).

On the other hand, a rough partition could make the training impossible. However, it is cautioned that a very fine partition could compromise its generalization ability. Too large a number of basis functions in RBF can cause over-fitting of the data and will result in poor generalization. Too small a number of these functions may lead to poor approximation accuracies [21]. This can be seen clearly in Fig. 11.

Furthermore, the results showed that there exists some optimal number of hidden neurons with least prediction errors. In our analysis, the range and number of hidden neurons evaluated is determined a priori and the optimal number is obtained via trial-and-error. When the predefined number is larger than the maximum, some clustering groups will be empty. It leads to the question of how the optimum number of basis function and optimum choice of RBF centers can be determined. Techniques such as orthogonal least squares learning [22] and genetic algorithm [5] has been proposed to circumvent this problem.

In addition, for different failure data sets, the optimal number of lagged variables p in the forecasting model can be found. In the case of car engine failures as highlighted in Fig. 12, using a number less than five will make NRMSE increase quickly, while any number larger than this optimal does not seem to have an impact enhancing the forecasting performance or cause it to worsen quickly. Similar interpretations can be drawn from the turbocharger failures.

Chaos theory implies that a time series, which seems to be random, may be generated by a deterministic function. As a consequence of Takens's embedding theorem [23], where the unknown dynamics in the true

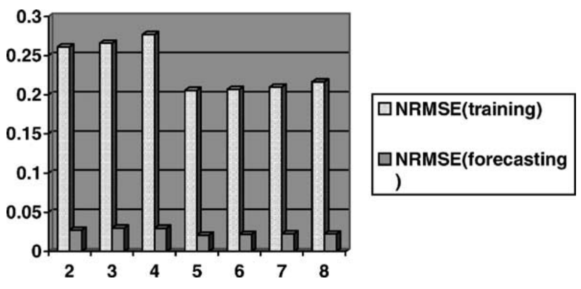

Fig. 10. The effect of lagged variable number p on training and testing forecast performance (N are different values).

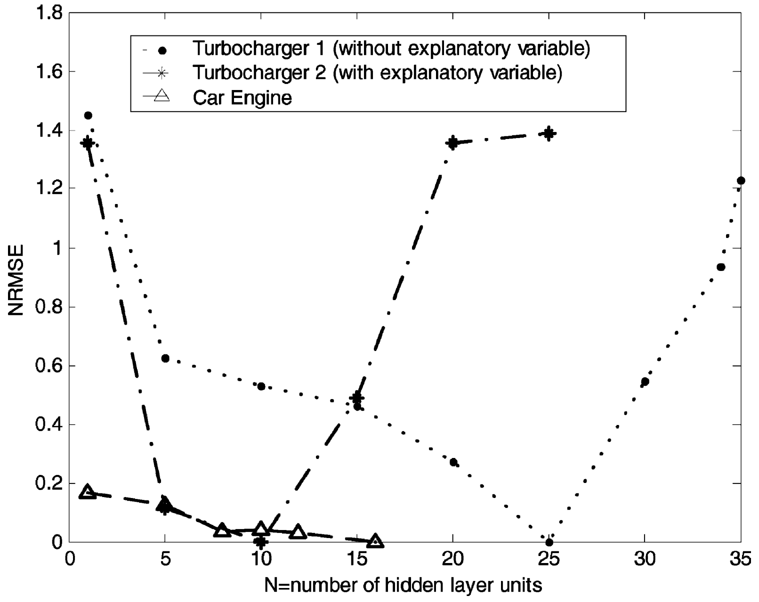

Fig. 11. The sensitivity analysis curve of *N*.

space can be estimated from the dynamics in the reconstructed space of lag vectors, prediction of future observation $x(t+1)$ is made possible from past historical lagged variables, using appropriate time series models. Takens proved that there exits a smooth unknown function that will correctly predict future value of a time series using at most $2n+1$ past measurement, n is the information dimension. Results from our

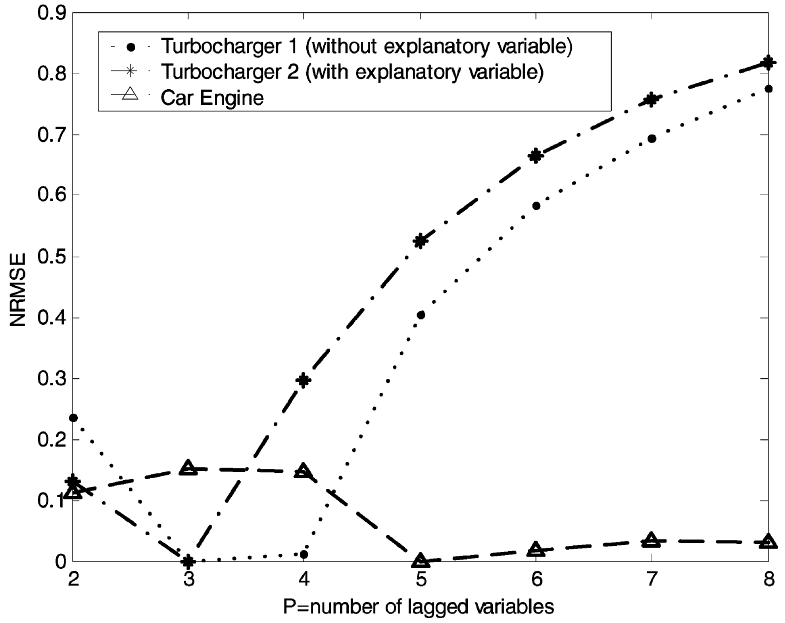

Fig. 12. The sensitivity analysis curve of *P*.

examples showed that the optimal number of lagged variable is three or five. This implies that the information dimension of these engine failures time series is close to one or two. Note that, as the dimensionality of the input space increases, the data may occupy an ever smaller sub-volume so that the number of hidden nodes of RBF does not have to expand so as to cover all of the possible domain of functions, the increasing of the predicting error is due to the breaking down of the concept of local neighborhood as the input space dimensionality grows. Thus, in some cases, it may become a limitation in view of the minimum number of the embedded space that is required by Takens's theorem. Thus a reasonable estimation of the minimum dimension in the choice of time lags is of interest. From our experiments, we recommend that initially 3–5 is a suitable value for the choice of time lags.

6. Conclusion

In this paper, the neural network approach for forecasting reliability and failures of engine systems is investigated. A comparative study of the predictive performance of various time series models are evaluated. Results from two case studies showed that the use of neural network models in forecasting the failures and reliability of engine systems is appropriate. The significance of this research is that no a priori specifications of parametric failure distributions need to be assumed and tested. A time series modeling technique using neural networks provides a promising alternative and leads to better predictive performance than the linear benchmark ARIMA models. Furthermore, it was demonstrated that the proposed RBF architecture is capable of achieving comparable or lower prediction errors, compared to traditional feed-forward MLP network and Box–Jenkins models.

The application of neural networks for reliability data analysis is relatively new. The positive results shown in this research have clearly demonstrated the potential of this approach in predicting failures and reliability. Future works can be centered on designing optimal neural network forecasting models and exploring design parameters for improved predictive performance.

References

[1] M.C. Liu, T. Sastri, W. Kuo, An exploratory study of a neural network approach for reliability data analysis, Qual. Reliabil. Eng. Int. 11 (1995) 107–112.
[2] N. Amjady, M. Ehsan, Evaluation of power systems reliability by artificial neural network, IEEE Trans. Power Syst. 14 (1) (1999) 287–292.
[3] D.K. Wedding, K.J. Cios, Time series forecasting by combining RBF networks, certainty factors and Box–Jenkins model, Neurocomputing 10 (1996) 149–168.
[4] K.G. Narendra, V.K. Sood, K. Khorasani, R. Patel, Application of a RBF neural network for fault diagnosis in a HVDC system, IEEE Trans. Power Syst. 13 (1) (1998) 177–183.
[5] J.A. Refaee, M. Mohandes, H. Maghrabi, Radial basis function networks for contingency analysis of bulk power systems, IEEE Trans. Power Syst. 14 (2) (1999) 772–778.
[6] B.A. Whitehead, T.D. Choate, Cooperative–competitive genetic evolution of radial basis function centers and widths for time series prediction, IEEE Trans. Neural Networks 7 (1996) 869–880.
[7] G.E.P. Box, G.M. Jenkins, Time Series Analysis: Forecasting and Control, Holden-Day Inc., San Francisco, CA, 1976.
[8] J.A. Beiser, S.E. Rigdon, Bayes prediction for the number of failures of a repairable system, IEEE Trans. Reliabil. 46 (2) (1997) 291–295.
[9] O.T. Ogunyemi, P.I. Nelson, Prediction of gamma failure times, IEEE Trans. Reliabil. 46 (3) (1997) 400–405.
[10] A. Weigend, N. Gerschenfeld (Eds.), Time Series Forecastion: Forecasting the Future and Understanding the Past, Addison-Weisley, Reading, MA, 1994.
[11] A. Khotanzad, R. Afkhami-Rohani, T.-L. Lu, et al., ANNSTLF—a neural-network-based electric load forecasting system, IEEE Trans. Neural Networks 8 (4) (1997) 835–846.
[12] J.V. Hansen, R.D. Nelson, Neural networks and traditional time series methods: a synergistic combination in state economic forecasts, IEEE Trans. Neural Networks 8 (4) (1997) 863–873.
[13] A.F. Atiya, S.I. Shaheen, A comparison between neural-network forecasting techniques—case study: river flow forecasting, IEEE Trans. Neural Networks 10 (1999) 402–409.
[14] M.J.D. Powell, Radial basis functions for multivariate interpolation: a review, in: J.C. Mason, M.G. Cox (Eds.), Proceedings of the IMA Conference on Algorithms for the Approximation of Functions and Data, Oxford University Press, Oxford, UK, 1987, pp. 143–167.
[15] D.S. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, Comp. Syst. 2 (1988) 321–335.
[16] J.E. Moody, C.J. Darken, Fast learning in networks of locally-tuned processing units, Neural Comput. 1 (1989) 282–294.
[17] J. Park, I.W. Sandberg, Universal approximation using radial-basis-functions networks, Neural Comput. 3 (2) (1991) 246–257.
[18] F. Scarselli, A.C. Tsoi, Universal approximation using feed-forward neural networks: a survey of some existing methods, and some new results, Neural Networks 11 (1) (1998) 15–37.

[19] E.J. Hartman, J.D. Keeler, J.M. Kowalski, Layered neural networks with Gaussian hidden units as universal approximations, Neural Comput. 2 (1990) 210–215.
[20] W. Nelson, Applied Life Data Analysis, Wiley, New York, 1988.
[21] W. Kaminski, P. Strumillo, Kernel orthonormalization in radial basis function neural networks, IEEE Trans. Neural Networks 8 (5) (1997) 1177–1183.
[22] S. Chen, C.F.N. Cowan, P.M. Grant, Orthogonal least squares learning algorithm for radial function networks, IEEE Trans. Neural Networks 2 (1991) 302–309.
[23] F. Takens, Detecting strange attractors in fluid turbulence, in: D. Rand, S. Yong (Eds.), Dynamical Systems and Turbulence, Springer, Berlin, 1981.