SOFT COMPUTING APPROACH FOR TIME SERIES PREDICTION

Yang Gao [*] **Meng Joo Er** [**]

[*] *School of Electronics & Physical Sciences*
University of Surrey, UK
yang.gao@surrey.ac.uk
[**] *School of Electrical & Electronic Engineering*
Nanyang Technological University, Singapore
emjer@ntu.edu.sg

Abstract: This paper focuses on the modeling and prediction of nonlinear time series using a soft computing approach, namely fuzzy neural network (FNN). An efficient algorithm for model structure determination and parameter identification with the aim of producing improved predictive performance for noninear time series is developed. Experiments and comparative studies demonstrate that the proposed approaches can effectively learn complex temporal sequences in an adaptive way and they outperform some well-known fuzzy neural methods.

Keywords: Time series analysis, Prediction problems, Fuzzy logic, Neural networks, Autoregressive model

1. INTRODUCTION

Time series prediction is an important practical problem with a variety of applications in business and economic planning, inventory and production control, meteorology forecasting, sociology, and many other fields. In the last decade, neural networks (NNs) have been extensively applied for complex time series processing tasks (Chen *et al.*, 1991; Platt, 1991; Salmeron *et al.*, 2001). This is due to their capability to learn and capture the nonlinear functional dependencies between past and current values from prior observations. More recently, fuzzy logic has been incorporated with the neural models for time series prediction (Cho and Wang, 1996; Chao *et al.*, 1966; Jang, 1993; Kim and Kasabov, 1999; Kasabov and Song, 2002). These approaches are generally known as fuzzy neural networks (FNNs) or NN-based fuzzy inference systems (FISs) approaches. FNNs possess both the advantages of FISs, such as human-like reasoning and ease of incorporating expert knowledge, and NNs, such as learning abilities, optimization abilities and connectionist structures. By virtue of this, low-level learning and computational power of NNs can be incorporated into the FISs on one hand and high-level human-like reasoning of FISs can be incorporated into NNs on the other hand.

The FNNs used in this paper are equivalent to Takagi-Sugeno-Kang-Type FISs. A sequential and hybrid (supervised/unsupervised) learning algorithm, namely generalized fuzzy neural network (G-FNN) learning algorithm, is employed to form the FNN prediction model. The proposed FNN predictors have the following salient features: 1) They are capable of modeling complex time series with on-line adjustment; 2) Fast learning speed; 3) Self-organizing FNN topology; 4) Good generalization and computational efficiency. Various comparative studies show that the proposed approaches are superior to many existing fuzzy/neural methods.

The rest of the paper is organized as follows. Section 2 reviews the NARX models and the concept of optimal predictors. Section 3 provides a mathematical description of the FNN predictors including the prediction models as well as the G-FNN learning algorithm for establishing those models. Two experiments are presented in Section 4 to demonstrate the predictive ability of the proposed methods. The results are compared with some popular fuzzy/neural methods. Finally, conclusions and directions for future research are given in Section 5.

2. NARX MODEL AND OPTIMAL PREDICTORS

2.1 General NARX(n_y,n_x) Model

The statistical approach for forecasting begins with selection of mathematical models to predict the value of an observation y_t using previous observations. A very general class of such models is the nonlinear autoregressive model with exogenous inputs (NARX) given by

$$y_t = \mathbf{F}[y_{t-1}, \ldots, y_{t-n_y}, x_{t-1}, \ldots, x_{t-n_x}] + e_t \quad (1)$$

where y and x are output and external input of the system model respectively, n_y and n_x are the maximum lags in the output and input respectively, and $\mathbf{F}$ is an unknown smooth nonlinear function. It is assumed that e_t is zero mean, independent and identically distributed, independent of past y and x, and has a finite variance σ^2.

2.2 Optimal Predictors

Optimum prediction theory revolves around minimizing mean squared error (MSE). Given the infinite past and provided the conditional mean exists, the optimal predictor $\hat{y}_t$ is the conditional mean $\mathbf{E}[y_t|y_{t-1}, y_{t-2}, \ldots]$ (Jazwinski, 1970).

Assuming that e_t is zero mean, independent and identically distributed, independent of past y and x, and has a finite variance σ^2 in (1), the optimal predictor for NARX or NARX model can be approximated as

$$\begin{aligned} \hat{y}_t &= \mathbf{E}[y_t|y_{t-1}, \ldots, y_{t-n_y}] \\ &= \mathbf{F}[y_{t-1}, \ldots, y_{t-n_y}, x_{t-1}, \ldots, x_{t-n_x}] \quad (2) \end{aligned}$$

where the optimal predictor (2) has MSE σ^2.

3. FNN PREDICTORS

3.1 FNN Prediction Models

In this paper, FNN predictors are proposed to emulate optimal predictor in (2). In other words, the FNN is used to approximate the function F. Functionality of the FNN is given by

$$\begin{aligned} \hat{y}_t &= \mathbf{F}_{FNN}[\mathbf{z}_t] = \sum_{j=1}^{n_r} \phi_j(\mathbf{z}) w_j \\ &= \sum_{j=1}^{n_r} \exp[-(\mathbf{z} - \mathbf{c}_j)^T \Sigma_j (\mathbf{z} - \mathbf{c}_j)] w_j \quad (3) \end{aligned}$$

where $\mathbf{z} = [z_1 \ldots z_{n_i}]^T$ is the input vector, $\mathbf{c}_j = [c_{1j} \ldots c_{n_i j}]^T$ and $\mathbf{\Sigma}_j = diag(\frac{1}{\sigma_{1j}^2} \cdots \frac{1}{\sigma_{n_i j}^2})$ are the center vector and the width matrix of the Gaussian membership function ϕ_j respectively, $w_j = k_{0j} + k_{1j} z_1 + \ldots + k_{n_i j} z_{n_i}$ is the TSK-type weight, k_{ij}s are real-valued parameters, n_i and n_r are the number of inputs and rules in the FNN respectively.

Eq. (3) can be represented in matrix form as follows:

$$\hat{y}_t = \mathbf{\Phi}_t^T \mathbf{w}_t \quad (4)$$

where the regression vector $\mathbf{\Phi} = [\phi_1 \;\; \phi_1 z_1 \ldots \phi_1 z_{n_i} \ldots \ldots \phi_{n_r} \;\; \phi_{n_r} z_1 \ldots \phi_{n_r} z_{n_i}]^T$ and the weight vector $\mathbf{w} = [k_{01} \;\; k_{11} \ldots k_{n_i 1} \ldots\ldots k_{0n_r} \;\; k_{1n_r} \ldots k_{n_i n_r}]^T$.

With proper choice of the input vector z, FNNs can be used to emulate NARX models for time series prediction. An FNN is a nonlinear approximation to function $\mathbf{F}$ that is equivalent to optimal predictor in (2) as follows:

$$\hat{y}_t = \mathbf{F}_{FNN}\{[y_{t-1} \ldots y_{t-n_y} \;\; x_{t-1} \ldots x_{t-n_x}]^T\} \quad (5)$$

3.2 G-FNN Learning Algorithm

In this paper, G-FNN learning algorithm is used to establish the FNN prediction model from prior time series values. It provides an efficient way of constructing the prediction model online and combining structure and parameter learning simultaneously. Structure learning includes determining the proper number of rules n_r. The parameters learning corresponds to premise and consequent parameters learning of the FNN. Premise parameters include membership function parameters $\mathbf{c}_j$ and $\mathbf{\Sigma}_j$, and consequent parameter refers to the weight w_j of the FNN.

Given the supervised training data, the proposed learning algorithm first decides whether to generate a rule based on two criteria. If structure learning is necessary, premise parameters of a new rule will be determined. The learning will also decide whether there are redundant rules to be deleted, and it will change the consequents of all the rules properly. If no structure learning is necessary, parameter learning will be performed to adjust the current premise and consequent parameters. This structure/parameter learning will be repeated for each training input-output data pair.

Two Criteria of Rule Generation: For each training data pair $[\mathbf{z}_t, y_t] : t = 1 \dots n_d$, where y_t is the desired output or the supervised teaching signal and n_d is the total number of training data, the system error is defined as $e_t = ||y_t - \hat{y}_t||$. If e_t is bigger than a designed threshold K_e, a new fuzzy rule should be considered.

At sample time t, regularized Mahalanobis distance is calculated as $md_j = \sqrt{[\mathbf{z}_t - \mathbf{c}_j]^T \mathbf{\Sigma}_j [\mathbf{z}_t - \mathbf{c}_j]}$ $j = 1 \dots n_r$. The accommodation factor is defined as $d_t = \min md_j$. If d_t is bigger than $K_d = \sqrt{\ln \frac{1}{\epsilon}}$, a new rule should be considered because the existing fuzzy system does not satisfy ϵ-completeness(Wang, 1997). Otherwise, the new input data can be represented by the nearest existing rule.

Pruning of Rules: The Error Reduction Ratio (ERR) concept proposed in (Chen *et al.*, 1991) is adopted here for rule pruning. At sample time t, we have from (4) $\mathbf{y} = \mathbf{\Theta w} + \mathbf{e}$, where $\mathbf{y} = [y_1 \quad y_2 \dots y_t]^T \in \Re^t$ is the teaching signal, $\mathbf{w} \in \Re^v$ is the real-valued weight vector, $\mathbf{\Theta} = [\mathbf{\Phi}_1 \dots \mathbf{\Phi}_t]^T \in \Re^{t \times v}$ is known as the regressor, $\mathbf{e} = [e_1 \quad e_2 \dots e_t]^T \in \Re^t$ is the system error vector that is assumed to be uncorrelated with the regressor $\mathbf{\Theta}$, and $v = n_r(n_i + 1)$.

For any matrix $\mathbf{\Theta}$, if its row number is larger than the column number, i.e. $t \geq v$, it can be transformed into a set of orthogonal basis vectors by QR decomposition (Press *et al.*, 1992), i.e. $\mathbf{\Theta} = \mathbf{QR}$, where $\mathbf{Q} = [\mathbf{q}_1 \ \mathbf{q}_2 \ \dots \ \mathbf{q}_v] \in \Re^{t \times v}$ has orthogonal columns, and $\mathbf{R} \in \Re^{v \times v}$ is an upper triangular matrix.

An ERR due to $\mathbf{q}_\gamma$ can be defined as (Chen *et al.*, 1991)

$$err_\gamma = \frac{(\mathbf{q}_\gamma^T \mathbf{y})^2}{\mathbf{q}_\gamma^T \mathbf{q}_\gamma \mathbf{y}^T \mathbf{y}} \tag{6}$$

The ERR matrix of the FNN is defined as

$$\begin{aligned} \mathbf{E}RR &= \begin{bmatrix} err_1 & err_2 & \dots & err_{n_r} \\ err_{n_r+1} & err_{n_r+2} & \dots & err_{n_r+n_r} \\ \vdots & \vdots & \dots & \vdots \\ err_{n_i \times n_r+1} & err_{n_i \times n_r+2} & \dots & err_{n_i \times n_r+n_r} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{e}rr_1 & \mathbf{e}rr_2 & \dots & \mathbf{e}rr_{n_r} \end{bmatrix} \end{aligned} \tag{7}$$

Total ERR $Terr_j, j = 1 \dots n_r$ corresponding to the jth rule is defined as

$$Terr_j = \sqrt{\frac{(\mathbf{e}rr_j)^T \mathbf{e}rr_j}{n_i + 1}} \tag{8}$$

If $Terr_j$ is smaller than a designed threshold $0 < K_{err} < 1$, the jth fuzzy rule should be deleted, and vice versa.

Determination of Premise Parameters: Premise parameters or Gaussian membership functions of the FNN are allocated to satisfy the ϵ-completeness of fuzzy rules.

In case of $e_t > K_e$ and $d_t > K_d$, we compute the Euclidean distance $ed_{ij_n} = ||z_i - b_{ij_n}||$ between z_i and the boundary point $b_{ij_n} \in \{c_{i1}, c_{i2}, \dots, c_{iN_r}, z_{i,\min}, z_{i,\max}\}$. Next, we find

$$\tilde{j}_n = \arg\min ed_{ij_n} \tag{9}$$

If $ed_{i\tilde{j}_n}$ is less than a threshold or a dissimilarity ratio of neighboring membership function K_{mf}, we choose

$$c_{i(n_r+1)} = b_{i\tilde{j}_n}, \sigma_{i(n_r+1)} = \sigma_{i\tilde{j}_n} \tag{10}$$

Otherwise, we choose

$$c_{i(n_r+1)} = z_i \tag{11}$$
$$\sigma_{i(n_r+1)} = \frac{\max\left(|c_{i(n_r+1)} - c_{i(n_r+1)_a}|, |c_{i(n_r+1)} - c_{i(n_r+1)_b}|\right)}{\sqrt{\ln \frac{1}{\epsilon}}}$$

In case of $e_t > K_e$ but $d_t \leq K_d$, the ellipsoidal field needs to be decreased to obtain a better local approximation. A simple method to reduce the Gaussian width is as follows

$$\sigma_{i\tilde{j}_{new}} = K_s \times \sigma_{i\tilde{j}_{old}} \tag{12}$$

where K_s is a reduction factor which depends on the sensitivity of the input variables.

In case of the rest, the system has good generalization and nothing need to be done except adjusting weight.

Determination of Consequent Parameters: TSK-type consequent parameters are determined using the Linear Least Squared (LLS) method as $\mathbf{w} = \mathbf{\Theta}^\dagger \mathbf{y}$, where $\mathbf{\Theta}^\dagger$ is the pseudoinverse of $\mathbf{\Theta}$. LLS method provides a computationally simple but efficient procedure of determining the weight so that it can be computed very quickly and used for real-time control.

4. EXPERIMENTS AND COMPARATIVE STUDIES

To demonstrate the validity of our proposed method, the G-FNN predictor are tested on Box-Jenkins Gas Furnace Data and chaotic Mackey-Glass time series. In addition, performances of the proposed strategies are compared with some recently developed fuzzy neural methods, including FNN methods such as adaptive network-based fuzzy inference system (ANFIS) (Jang, 1993), RBF-based adaptive fuzzy system (RBF-AFS) (Cho and Wang, 1996), hybrid neural fuzzy inference system (HyFIS) (Kim and Kasabov, 1999), and dynamic evolving neural-fuzzy inference system (DENFIS) (Kasabov and Song, 2002), and NN methods such as orthogonal-least-squares-based RBF network (OLS-RBFN) (Chen *et al.*, 1991), resource-allocating network (RAN) (Platt, 1991), pseudo-Gaussian basis function network (PG-BF)(Rojas *et al.*, 2000), and resource-allocating network using orthogonal techniques (RANO) (Salmeron *et al.*, 2001) etc.

4.1 Box-Jenkins Gas Furnace Data

In this example, the proposed predictors are applied to the Box-Jenkins gas furnace data (Box and Jenkins, 1970), a benchmark problem for testing identification algorithms. This data set was recorded from a combustion process of a methane-air mixture. During the process, the portion of methane was randomly changed, keeping a constant gas flow rate. The data set consists of 296 pairs of input-output measurements. The input $u(t)$ is the methane gas flow into the furnace and the output $y(t)$ is the CO_2 concentration in the outlet gas.

To facilitate comparison, the following fitting model is chosen:

$$y(t) = \mathbf{F}[y(t-1), u(t-4)] \quad (13)$$

As a result, there are in total 292 input/output data pairs with $y(t-1)$ and $u(t-4)$ as input variables and $y(t)$ as an output variable. The data were partitioned in 200 data pairs as a training set, and the remaining 92 pairs as a test set for validation.

A total of 7 fuzzy rules are generated for the G-FNN during training as shown in Figure 1(a). The corresponding Gaussian membership functions with respect to the input variables are shown in Figure 2. It can be seen that the membership functions are evenly distributed over the training interval. This is in line with the aspiration of "local representation" in fuzzy logic. The root mean square error (RMSE) is able to converge very quickly during training as illustrated in Figure 1(b). As the G-FNN algorithm uses one-pass learning method to avoid iterative learning loops, fast learning speed can be achieved. Figure 3 shows the actual and predicted values for both training and testing data. In this experiment, performance comparison of the G-FNN predictor is carried out with three learning models, i.e. ANFIS, HyFIS and PG-BF. The results are summarized in Table 1. It is shown that the G-FNN provides better generalization and more compact rule base.

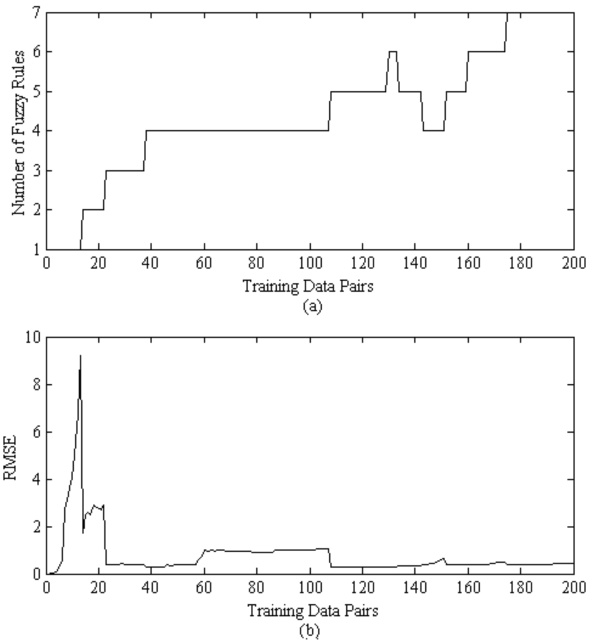

Fig. 1. Training performances: (a) Fuzzy rule generation; (b) RMSE during training process.

4.2 Chaotic Mackey-Glass Time Series

Chaotic Mackey-Glass time series is a benchmark problem that has been considered by a number of researchers (Cho and Wang, 1996; Chen *et al.*, 1991; Jang, 1993; Platt, 1991; Salmeron *et al.*, 2001). The time series is generated from the following equation

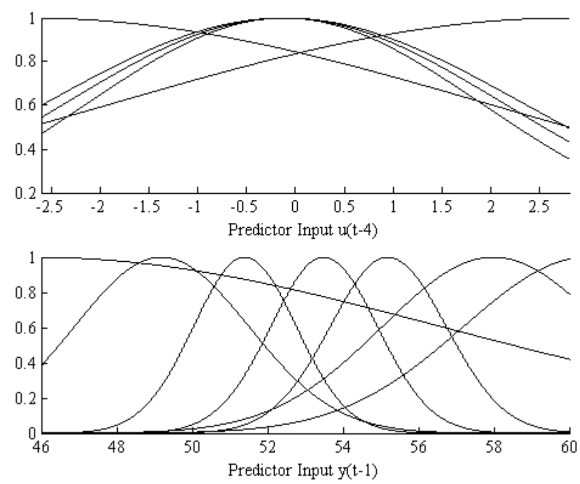

Fig. 2. Gaussian membership functions w.r.t input variables.

Table 1. Performance Comparisons with ANFIS, HyFIS and PG-BF

Model	n_r	Testing MSE	Learning Method
ANFIS	25	0.1643	parameter (iterative loops)
HyFIS	15	0.0945	structure (one pass) +[1] parameter (200 loops)
PG-BF	10	0.157	structure ×[2] parameter (one pass)
G-FNN	7	0.0744	structure × parameter (one pass)

[1] + implies structure learning and parameter learning are performed offline separately. [2] × implies structure and parameter learnings are performed simultaneously.

Table 2. Performance Comparisons with RBF-AFS and OLS-RBFN

Model	n_r	n_p	Training RMSE	Testing RMSE	Learning Method
RBF-AFS	21	210	0.0107	0.0128	structure (one pass) × parameter (iterative loops)
OLS-RBFN	35	211	0.0087	0.0089	structure × parameter (one pass)
G-FNN	10	90	0.0063	0.0056	structure × parameter (one pass)

Table 3. Performance Comparisons with RAN, RANO and DENFIS

Model	n_r	Testing NDEI[1]
RAN	40	0.1642
RANO	18	0.1492
DENFIS	883	0.042
G-FNN	8	0.0226

[1] Nondimensional error index (NDEI), also known as normalized RMSE, is defined as the RMSE divided by the standard deviation of the target time series.

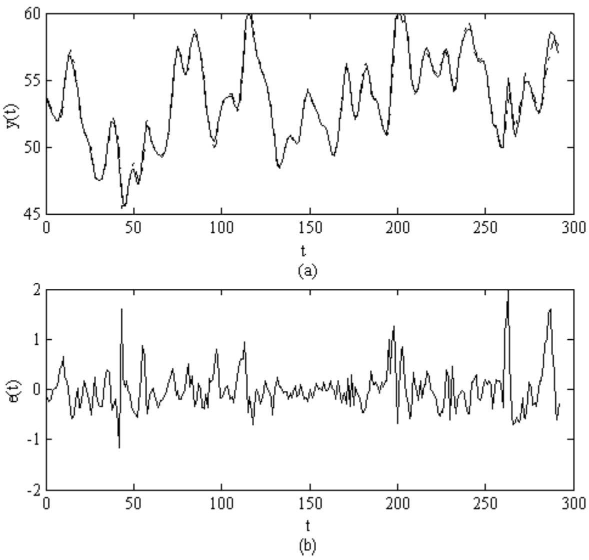

Fig. 3. Prediction results: (a) Box-Jenkins gas furnace data and one-step ahead prediction; (b) Prediction error.

$$y(t) = (1-a)y(t-1) + \frac{by(t-\tau)}{1+y^{10}(t-\tau)} \quad (14)$$

where $\tau > 17$ gives chaotic behavior. Higher value of τ yields higher dimensional chaos. For the ease of comparison, parameters are selected as: $a = 0.1$, $b = 0.2$ and $\tau = 17$.

The fitting model of (14) is chosen to be

$$y_t = \mathbf{F}[y_{t-p}, y_{t-p-\triangle t}, y_{t-p-2\triangle t}, y_{t-p-3\triangle t}] \quad (15)$$

For simulation purpose, it is assumed that $y_t = 0$, $\forall t < 0$ and $y(0) = 1.2$. The following values are chosen: $p = \triangle t = 6$ and $118 \leq t \leq 1140$. The first 500 input/output data pairs generated from (14) are used for training the G-FNN while the following 500 data pairs are used for validating the identified model.

Using the G-FNN learning algorithm, a total of 8 fuzzy rules are generated for the G-FNN predictor during training as shown in Figure 4(a). The root mean square error (RMSE) is able to converge very quickly using one-pass learning method as illustrated in Figure 4(b). Figure 5 shows that the actual and predicted values are essentially the same and their differences can only be seen on a finer scale.

Performance comparisons of the G-FNN with RBF-AFS and OLS-RBFN are shown in Table 2. The G-FNN predictor again forms competitively compact rule base. Another comparison is performed between G-FNN and some one-pass also known as on-line learning models, i.e. RAN, RANO, GNGN, and DENFIS, applied on the same task (see Table 3). The G-FNN predictor outperforms these methods in terms of predictive accuracy as well as computational efficency.

5. CONCLUSIONS

In this paper, a FNN predictor is proposed, tested and compared. The proposed predictor provide a sequential and hybrid learning method for model structure determination and parameter identification which greatly improves predictive performance. Experiments and comparative studies demonstrate superior performance of the proposed approaches over some well-known fuzzy neural methods. Further study is necessary to investigate the robustness of the proposed methods.

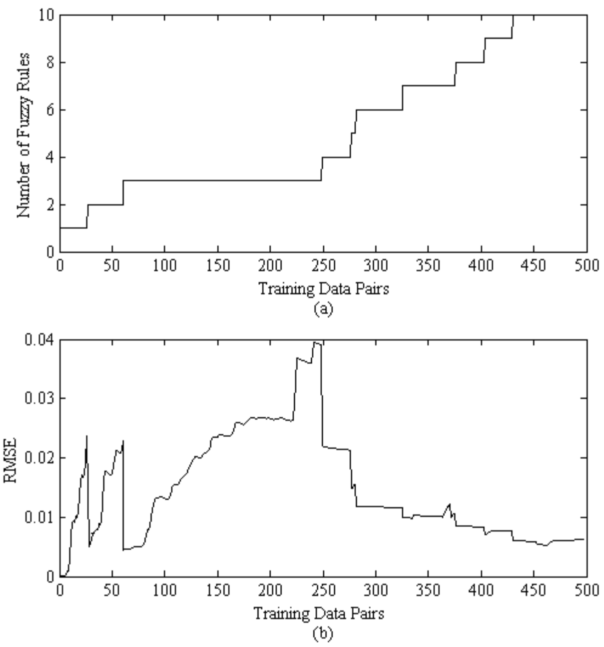

Fig. 4. Training performances: (a) Rule generation; (b) RMSE during learning process.

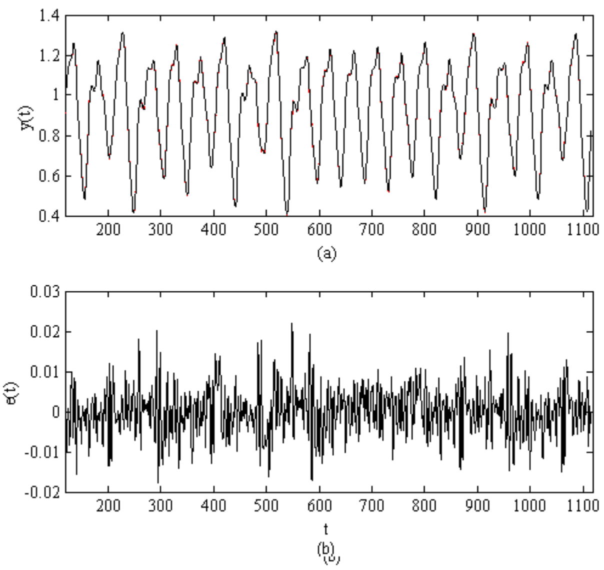

Fig. 5. Prediction results: (a) Mackey-Glass time series from $t = 118$ to 1140 and six-step ahead prediction; (b) Prediction error.

One way is to train the prediction models using different training data groups. If the prediction models are formed consistently, the proposed G-FNN method is robust in this sense.

REFERENCES

Box, G. E. P. and G. M. Jenkins (1970). *Time Series Analysis, Forecating and Control.* Holden Day.

Chao, C. T., Y. J. Chen and C. C. Teng (1966). Simplification of fuzzy-neural systems using similarity analysis. *IEEE Trans. System, Man and Cybernetic* **26**, 344–354.

Chen, S., C. F. N. Cowan and P. M. Grant (1991). Orthogonal least squares learning algorithm for radial basis function network. *IEEE Trans. Neural Networks* **2**, 302–309.

Cho, K. B. and B. H. Wang (1996). Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction. *Fuzzy Sets and Systems* pp. 325–339.

Jang, J. S. R. (1993). Anfis: Adaptive-network-based fuzzy inference system. *IEEE Trans. System, Man and Cybernetic* **23**, 665–684.

Jazwinski, A. (1970). *Stochastic Processes and Filtering Theory.* Academic Press. New York.

Kasabov, N. K. and Q. Song (2002). Denfis: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE Trans. Fuzzy Systems* pp. 144–154.

Kim, J. and N. Kasabov (1999). Hyfis: Adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems. *Neural Networks* **12**, 1301–1319.

Platt, J. (1991). A resource-allocating network for function interpolation. *Neural Computation* pp. 213–225.

Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery (1992). *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press. Cambridge.

Rojas, I., J. Gonzalez, A. Canas, A. F. Diaz, F. J. Rojas and M. Rodriguez (2000). Short-term prediction of chaotic time series by using rbf network with regression weights. *Int. J. of Neural Systems* pp. 353–364.

Salmeron, M., J. Ortega, C. G. Puntonet and A. Prieto (2001). Improved ran sequential prediction using orthogonal techniques. *Neurocomputing* pp. 153–172.

Wang, L. X. (1997). *A Course in Fuzzy Systems and Control.* Prentice Hall. New Jersey.