Predicting Chaotic Time Series Using Recurrent Neural Network

To cite this article: Jia-Shu Zhang and Xian-Ci Xiao 2000 *Chinese Phys. Lett.* **17** 88

Predicting Chaotic Time Series Using Recurrent Neural Network *

ZHANG Jia-Shu (张家树), XIAO Xian-Ci (肖先赐)
Department of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu 610054

(Received 10 July 1999)

A new proposed method, i.e. the recurrent neural network (RNN), is introduced to predict chaotic time series. The effectiveness of using RNN for making one-step and multi-step predictions is tested based on remarkable few datum points by computer-generated chaotic time series. Numerical results show that the RNN proposed here is a very powerful tool for making prediction of chaotic time series.

PACS: 05. 45. +b, 05. 45. Tp, 07. 05. Mh

The problem of the chaotic time series prediction has become an active field due to its potential applications.[1-6] While chaos places a fundamental limit on long-term prediction, it suggests possibilities for short-term predictions: Random-looking chaotic data contain simple deterministic nonlinear relationships, involving only a few irreducible degree of freedom.[5] In fact, a real time series may be comprised of a low-dimensional attractor (deterministic part of the time series) and high-dimensional noise (stochastic part of the time series). By building an appropriate model, it is possible to make the multi-step prediction of the deterministic part of chaotic time series.

Suppose a scalar time series $\{x(t), t = 1, 2, \cdots, N\}$ is a measurement on the chaotic dynamical system in the state space. If the embedding dimension m and the time delay τ have been chosen for the phase reconstruction, a reconstructed vector can be written as follows

$$\boldsymbol{x}(t) = [x(t), x(t+\tau), \cdots, x(t+(m-1)\tau)]. \quad (1)$$

By Taken's embedding theorem or its extensions, there exists a function F such that

$$x(t+T) = F[\boldsymbol{x}(t)], \quad t = 1, 2, \cdots, N, \quad (2)$$

where integer T is the look-ahead time. Let the reconstructed attractor be denoted by S, then F: $S \to S$. Thus it is necessary to fit F only in the domain S by using prediction model. A model for time series is obtained by finding a functional form $\hat{F}$ for the training set pairs $[\boldsymbol{x}(t), d(t)]$, $t = 1, \cdots, N$, where $d(t) = x(t+T)$ is the output of the model, and $\boldsymbol{x}(t)$ is the input of the model. $\hat{F}$ relates $\boldsymbol{x}(t)$ to $d(t)$:

$$x(t+T) = \hat{F}[\boldsymbol{x}(t)] + \varepsilon(t), \quad (3)$$

where $\varepsilon(t)$ represents the noise or fitting error.

In this case, a good predictive model for a time series should have good training and predictive accuracy in the presence of noise or fitting error, and it should only require a reasonable amount of computing time and data. As is well-known, artificial neural networks have gained a lot of interest as empirical models for their powerful representational capacity, fault tolerance ability and multi-input-and-output mapping characteristics. More recently, neural networks, including feed-forward neural network (FFNN),[6] radial basis function network (RBFN),[7,8] and wavelet network (WN),[9] are used to make prediction of chaotic time series. The effectiveness of these networks has been confirmed by testing a large number of examples.[8,10] The FFNN, RBFN, and WN are essentially static systems. However, time series prediction involves processing of patterns that evolve over time — the appropriate response at a particular point in time depends not only on the current value of the observable but also on the past. Even if these networks are very well trained to make one-step predictions, they may fail to make n-step predictions.

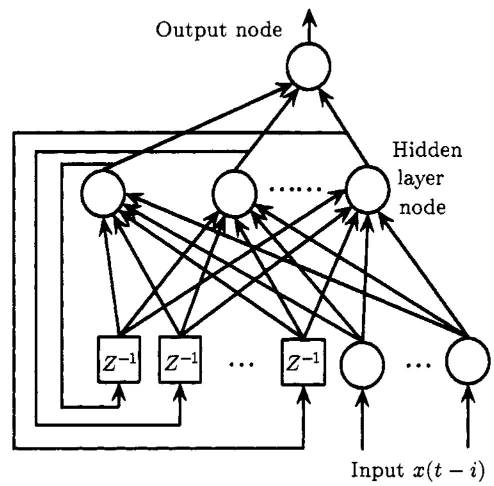

Fig. 1. RNN architecture.

The recurrent neural network (RNN) considered in this letter(Fig. 1) is a variant type of traditional discrete-time recurrent network. It is capable of representing and encoding deeply hidden states with tapped-delay lines, in which a network's output depends on an arbitrary number of its previous inputs.

*Supported by National Defense Foundation of China Under Grant Nos. 98JS05. 4.1.DZ0205.

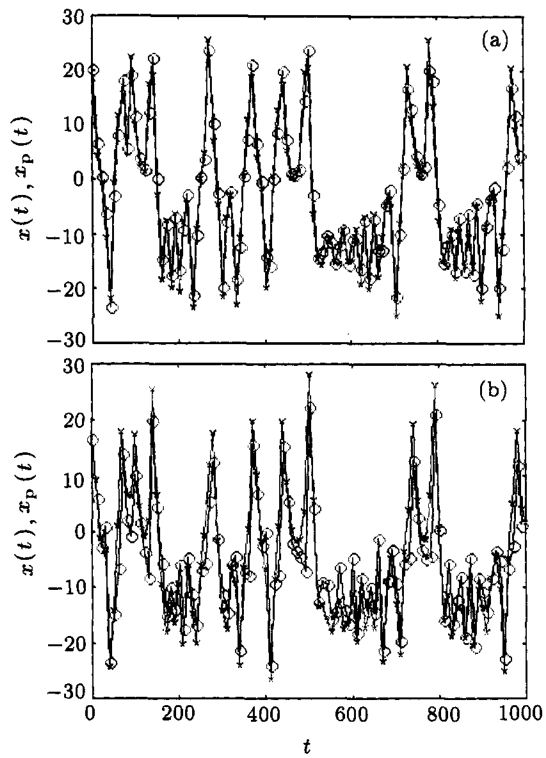

Fig. 2. Results of Lorenz system with (a) $T = 1$ and $E = 0.0051$ (b) $T = 20$ and $E = 0.2058$ (○: predicted values $x_p(t)$, ×: true values $x(t)$).

This RNN is a specific network with both forward path and feedback path, in which the output of the hidden neurons at instant t fully feed themselves as partial inputs at $t+1$. Unlike other NN, it is an essential dynamic system. So we can use various training multi-layer feed-forward network's method to train this RNN. Based on the above discussion, we think that the RNN might be a useful tool for making predictions of chaotic time series, especially in making multi-step predictions. Let M is the input cell number (i.e., embedding dimension), L the number of the hidden neurons, then there exist relationships between the input and output of neurons on every layer of this RNN as follows:

$$x(t+T) = \sum_{j=1}^{L} w_j^2 h_j + \theta_2, \tag{4}$$

$$h_j = f\left(\sum_{i=1}^{M} w_{ij}^1 x(t-i) + \sum_{i=1}^{L} w_{ij}^h h_j + \theta_j^1\right), \tag{5}$$

$$f(x) = \frac{1-\mathrm{e}^{-x}}{1+\mathrm{e}^{-x}}, \tag{6}$$

where h_j is the output of j-th neuron in the hidden layer, w_{ij}^1 the weight of i-th neuron of the hidden layer connecting to j-th input node, $x(t-i)$ the i-th input signal, w_{ij}^2 the weight of the hidden neuron connecting to output node, the θ_j^1 and θ_2 are the thresholds of the hidden neurons and the output node, respectively.

Among many methods proposed for training multi-layer FFNN, the recursive least squares (RLS) method[11] stands out. The RLS training method is a parameter identification technique for a nonlinear dynamical system (RNN). This method adapts weights of the network pattern-by-pattern accumulating training information in approximate error covariance matrices and providing individually adjusted updates for the network's weights. It is required by the RLS method that we compute deviates of the RNN's outputs, rather than output errors, with respect to the weights. These deviates are obtained by back-propagation (BP) algorithm through time. The following equations form the basis of the RLS training procedure:

$$\varepsilon_i^{(k)} = \begin{cases} d(t) - x(t+T), & \text{for } k=2, \\ \sum_{j=1}^{L+M} f'(s_i^{(k)}(n)) w_{ji}^{(k+1)}(n) \varepsilon_j^{(k+1)}(n), & \text{for } k=1, \end{cases} \tag{7}$$

$$g(n) = \frac{f'[s_i^{(k)}(n)] P_i^{(k)}(n-1) X_i^{(k)}(n)}{\lambda + f'^2[s_i^{(k)}(n)] X^{(k)T}(n) P_i^{(k)}(n) X_i^{(k)}(n)}, \tag{8}$$

$$P_i^{(k)}(n) = \lambda^{-1}[I - f'(s_i^{(k)}(n)) g_i^{(k)}(n) X^{(k)T}(n)] \cdot P_i^{(k)}(n-1), \tag{9}$$

$$w_i^{(k)}(n) = w_i^{(k)}(n-1) + g_i^{(k)}(n) \varepsilon_i^{(k)}(n), \tag{10}$$

where λ is a positive constant (called the forgetting factor) close to and less than 1, $\varepsilon_i^{(k)}(n)$ the error of the i-th neuron in the k-th layer, $X_i^{(k)}(n)$ the vector of weight of the i-th neuron in the k-th layer, $w_{ji}^{(k)}(n)$ the weights of the i-th neuron of in the k-th layer connecting with the j-th input. The weights vector can be updated according to the formulae (7)−(10) with

$$w_{ij}^{(k)}(0) = 0, \tag{11}$$

$$P(0) = \delta I, \quad \delta \gg 1, \tag{12}$$

where I is the identity matrix and δ is a large constant. Alternatively, the initial values of the weight vector can be chosen as random numbers.

To illustrate the effectiveness of the used RNN for making predictions, the Lorenz equation and the Mackey-Glass delay differential equation are chosen for examples of chaotic time series. Each example uses corresponding 2100 values, respectively. The generated time series is divided into two parts: the first 100 values are the training dataset used to build the model, and the last 1000 points are the test dataset to the prediction ability of the resulting. To evaluate the accuracy of the RNN predictions, we compute the root-mean-square (rms) error, $\sigma_T = \langle [x(t+T) - \hat{x}(t+T)] \rangle^{1/2}$. For convenience we normalize it by rms deviation of the data $E = \sigma_T/\langle x(t+T)^2\rangle^{1/2}$, forming the normalized error E. If $E = 0$, the predictions are perfect. $E = 1$ indicates that the prediction performance is not better than a constant predictor $\hat{x}(t+T) = \langle x(t)\rangle$.

Our final network's architecture is the one-hidden layer RNN with 4 inputs of the normalized data, 10 fully recurrent hidden neurons and 1 output neuron. Recurrent and output nodes have the common bipolar sigmodal nonlinear function defined by Eq. (6). To show clearly the predictive results of the used RNN for making 1 step and 20 steps future predictions of chaotic series, only one is plotted for each 4 points compared with actual values. Numerical experiments are given as follows (Fig. 2 and 3):

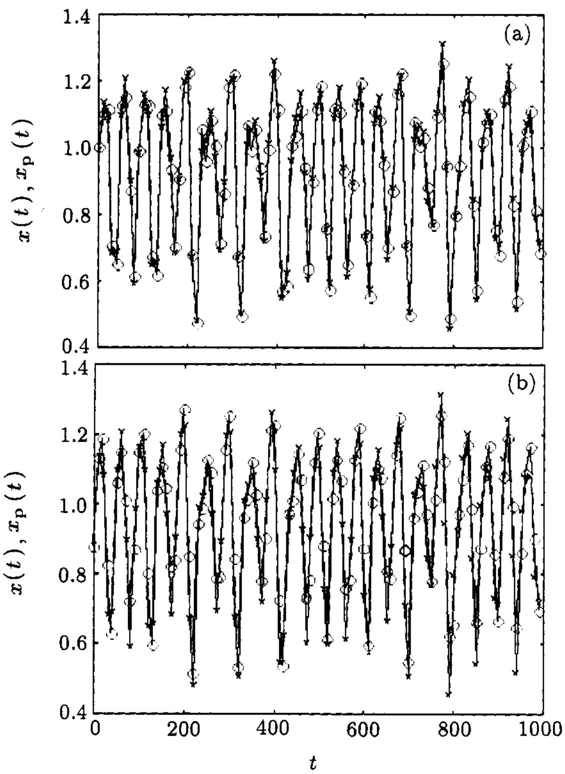

Fig. 3. Results of Mackey-Glass equation with (a) $T = 1$ and $E = 9.9554 \times 10^{-4}$ (b) $T = 20$ and $E = 0.0064$ (prediction values $x_p(t)$ by ○, true values $x(t)$ by ×).

Example 1: Lorenz system with an integal step of 0.005. After all transients have been diminished, 2100 $x-$componen values from these points are taken as our experimental data:

$$\begin{cases} \dot{x} = 10(-x+Y), \\ \dot{y} = 49.5x - y - xz, \\ \dot{z} = xy - 4.0z. \end{cases}$$

Example 2: Mackey-Glass differential delay equation with $\Delta = 17$, and initial conditions $x(t) = 0.9$, $0 \leq t \leq 17$, $D_2 = 2.1$, for $\lambda = 0.0086$, 0.001, -0.0395, and -0.0504, $\mathrm{d}x(t)/\mathrm{d}t = -0.1x(t) + 0.2x(t-\Delta)/[1 + x(t-\Delta)^{10}]$.

From these experimental results listed above, we can see clearly that the predicted values for the case of one step agree well with true values, the predicted results for 20 steps are not better than one step. Obviously, the RNN proposed here is capable of capturing the underlying chaotic dynamics of the system based on a few data point. As we expect, the multi-step predictions by the RNN are very successful. This is due to the RNN's internal recurrence.

In summary, we have described a technique of using RNN for making one step and multi-step accurate value predictions of chaotic time series. This method has following advantages. First, the RNN proposed in this letter has the capacity to dynamically incorporate past experience due to internal recurrence. Second, unlike other neural network, it can make accurate predictions based on a few data. So it does not need large memory storage. Third, this RNN with the recursive least squares training method can be a faster convergence than that with the conventional back-propagation.

REFERENCES

1 H. D. I. Abarbanel, R. Brown, J. B. Kadtke, Phys. Lett. A 138 (1989) 401.
2 J. D. Farmer, J. J. Sidorowich, Phys. Rev. Lett. 59 (1987) 845.
3 D. B. Murray, Physica, D 59 (1993) 318.
4 L. A. Aguirre, S. A. Billings, Physica, D 85 (1995) 239.
5 K. Judd, A. Mees, Physica, D 82 (1995) 426.
6 S. Haykin, X. B. Li, Proc. IEEE, 83 (1995) 94.
7 X. D. He and A. Lapedes, Physica, D 70 (1993) 151.
8 M. Casdali, Physica, D 35 (1989) 335.
9 L. Y. Cao, Y. G. Hong, H. Fang and G. W. He, Physica, D 85 (1995) 225.
10 J. C. Principle, A. Ratheie and J. M. Kuo, Int. J. Bifur. and Chaos, 2 (1990) 989.
11 J. Bilski, L. Rutkowsi, IEEE trans. Circuit and System II, 45 (1998) 749.