Journal of Mathematical Sciences: Advances and Applications
Volume 4, Number 1, 2010, Pages 73-94

PREDICTING CHAOTIC TIME SERIES USING NEURAL NETWORKS WITH DELAY COORDINATES AS INPUTS

JIIN-PO YEH

Department of Civil and Ecological Engineering
I-Shou University
Dashu Township
Kaohsiung County 84001
Taiwan
e-mail: jpyeh@isu.edu.tw

Abstract

With delay coordinates as inputs of neural networks, this paper presents a forecasting technique for chaotic time series. By using delay coordinates, the chaotic time series is first embedded in a reconstructed state space. Then the delay coordinates of each state in the reconstructed state space serve as the input vector of the neural network and the first delay coordinate of the next state as the target of the neural network. The neural networks used in this paper are two-layer feedforward networks with one hidden layer of tan-sigmoid neurons followed by an output layer of one linear neuron. Meanwhile, Bayesian regularization and early stopping of training are applied to improve the network generalization. Traffic flows of three different time scales are used as examples to show the effectiveness of the technique. Numerical results show that with the number of neurons in the hidden layer not more than the number of elements in the input vector and for only a few iterations, the neural network will have acceptable performance. Although, more neurons or iterations can enhance the network performance for the training set, it does not have the tendency to benefit the validation and the prediction sets. In addition, the prediction accuracy becomes higher, when the traffic volume time scale increases.

2010 Mathematics Subject Classification: 92B20, 37D45.

Keywords and phrases: neural networks, chaotic time series, delay coordinates, reconstructed state space.

Received July 2, 2009

1. Introduction

The neural network was originated by McCulloch and Pitts [20], who claimed that neurons with binary inputs and a step-threshold activation function were analogous to first order systems. Hebb [14] revolutionized the perception of artificial neurons. Rosenblatt [24], using the McCulloch-Pitts neuron and the findings of Hebb, developed the first perception model of the neuron, which is still widely accepted today. Hopfield [15] and Hopfield et al. [16] demonstrated from work on the neuronal structure of the common garden slug that ANNs (artificial neural networks) can solve non-separable problems by placing a hidden layer between the input and output layers. Rumelhart and McClelland [25] developed the most famous learning algorithm in ANN-backpropagation, which uses a gradient descent technique to propagate error through a network to adjust the weights in an attempt to find the global error minimum, marking a milestone in the current artificial neural networks. Since then, a huge proliferation in the ANN methodologies has occurred.

The fact that chaotic behaviors exist in the traffic flow system has been known for decades. Gazis et al. [11] developed a generalized car-following model, known as the GHR (Gazis-Herman-Rothery) model, whose discontinuous behavior and nonlinearity suggested chaotic solutions for a certain range of input parameters. Traffic systems without signals, bottlenecks, intersections, etc. or with a coordinate signal network, modelled with the traditional GHR traffic-flow equation, were tested for presence of chaotic behaviors [8, 10]. Chaos was also observed in a platoon of vehicles described by the traditional GHR model with a nonlinear inter-car separation dependent term added [1].

Up to now, a variety of methodologies has been applied to short-term chaotic time series prediction, including local linear models [9], polynomial models [4], and neural network-based black-box models [2, 5, 6, 23]. This paper adopts neural networks as well, applying the MATLAB software [7] to build feedforward-backpropagation neural networks, but uses the delay coordinates [3, 22] of the state in the reconstructed state space of the time series as the input vector of the neural network and the first delay coordinate of next state as the target of the neural network, which is distinguished from other authors' previous works.

2. Embedding Dimension

When mathematical models of dynamical systems are difficult to derive and there is only one measured variable, to obtain more information about their dynamical behaviors, one usual method is to reconstruct its state space by delay coordinates: $x(t)$, $x(t - \tau)$, $x(t - 2\tau)$, $\ldots$, $x(t - (q - 1)\tau)$, where q is the dimension of the reconstructed state space, τ is the time delay, and $x(t)$ is the time series of the measurement. To get the right dimension of the reconstructed state space to embed the attractor (chaotic or periodic), the dimension of the attractor must first be found. There are a number of ways to measure the attractor dimension [12, 22]. Among them, correlation dimension is adopted in this paper.

Given an orbit discretized to a set of N points in the state space, as shown in Figure 1, a sphere of radius r is constructed at each point of the orbit, and the number of points in each sphere is counted. A correlation function is then defined as

$$C(r) = \lim_{N \to \infty} \frac{1}{N(N-1)} \sum_{i}^{N} \sum_{j}^{N} H(r - |\mathbf{x}_i - \mathbf{x}_j|), \tag{1}$$

$$(i \neq j)$$

where $|\mathbf{x}_i - \mathbf{x}_j|$ is the Euclidean distance between points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ of the orbit and H is the unit step function. For many attractors, this function exhibits a power law dependence on r, as $r \to 0$; that is,

$$\lim_{r \to 0} C(r) = ar^{d}. \tag{2}$$

Hence, the correlation dimension is defined by the expression

$$d_c = \lim_{r \to 0} \frac{\ln C(r)}{\ln r}. \tag{3}$$

The dimension of the attractor will approach an asymptote d with the dimension q of the reconstructed state space gradually increasing. To represent the attractor one to one without causing self-intersection, the embedding dimension of the attractor must be at least $2d + 1$ [26]. Therefore, the appropriate dimension for the reconstructed state space will be the smallest integer $\nu \geq 2d + 1$. For a chaotic attractor, the dimension d is always fractal, not an integer. The delay coordinates of the ν-dimensional state space will serve as the input vector of the neural networks.

3. Neural Networks

The forecasting model used in this paper is built, based on a two-layer feedforward neural network with the backpropagation training algorithm, as shown in Figure 2. The transfer function used in the single hidden layer is the tan-sigmoid function for mapping the input to the interval (–1, 1) of the following form

$$a_i = f(n_i) = \frac{1 - e^{-n_i}}{1 + e^{-n_i}}, \; i = 1, 2, 3, \dots, s, \tag{4}$$

where $n_i = w_{i1}x_1 + w_{i2}x_2 + \dots + w_{iR}x_R + b_i$, $x_1, x_2, \dots, x_R$ are the inputs, s is the number of neurons, $w_{i1}, w_{i2}, \dots, w_{iR}$ are the weights connecting the input vector and the i-th neuron, and b_i is the bias of the i-th neuron. The output layer with a single neuron uses the linear transfer function

$$a = f(n) = n, \tag{5}$$

where $n = W_{11a_1} + W_{12a_2} + \dots\dots + W_{is}a_s + b$, $W_{11}, W_{12}, \dots, W_{is}$ are the weights connecting the neurons of the hidden layer and the neuron of the output layer, and b is the bias of the output neuron.

There are many variations of the backpropagation algorithm, which is aimed at minimizing the network performance function, i.e., the mean square error between the network outputs and the target outputs, which is

$$Minimize \quad MSE = \frac{1}{m}\sum_{j=1}^{m}(t_j - a_j)^2, \tag{6}$$

where t_j and a_j are the j-th target output and network output, respectively. This paper selects the Levenberg-Marquardt algorithm [13, 17, 19] as the training function to minimize the network performance function. This algorithm interpolates between the Newton's algorithm and the gradient descent method. If a tentative step increases the performance function, this algorithm will act like the gradient descent method, while it shifts toward Newton's method, if the reduction of the performance function is successful. In this way, the performance function will always be reduced at each iteration of the algorithm. To make the neural networks more efficient, it is often useful to scale inputs and targets so that they will always fall within a specific range. For example, the following formula

$$k' = 2(\frac{k - \min}{\max - \min}) - 1, \tag{7}$$

is used in this paper to scale both inputs and targets, where k is the original value, k' is the scaled value, and max and min are the maximum and minimum of the inputs or targets, respectively. Equation (7) produces inputs and targets in the range [–1, 1]; the scaled outputs of the networks will be usually converted back to the original units.

There are two methods to improve the network generalization: Bayesian regularization [18] and early stopping. The Bayesian regularization provides a measure of how many network parameters (weights and biases) are being effectively used by the network. From this effective number of parameters, the number of neurons required in the hidden layer of the two-layer neural network can be derived by the following equation

$$(Rs + s) + (s + 1) = P, \tag{8}$$

where R is the number of elements in the input vector, s is the number of neurons in the hidden layer, and P is the effective number of parameters found by the Bayesian regularization. With the strategy of early stopping incorporated into the neural network, the error on the validation set is monitored during the training process. When the network begins to overfit the training data, the error on the validation set typically begins to rise. Once the validation error increases for a specified number of iterations, the training is stopped and the weights and biases at the minimum of validation error are returned. To evaluate the performance of the trained network, this paper makes use of a regression analysis between the network outputs and the corresponding targets, and expresses it by both scatter plots and correlation coefficients [21].

4. Numerical Results

Examples given in this paper are the westbound passing traffic volume at the intersection of Xingai Road and Guanfu S. Road, Taipei City, Taiwan. The data were collected by the vehicle traffic counter from August 22, 2005 to September 2, 2005, totaling 10 weekdays with the weekend excluded. There are three time scales involved: 5-min, 10-min, and 15-min. The data is divided into three sets: the training set (the first 7 days), the validation set (the 8th and 9th days), and the prediction set (the 10th day).

4.1. Reconstruction dimension

Time series of the three different time intervals show no repeat of themselves and have the tendency to be aperiodic. For example, Figure 3 shows the time series of the 5-min traffic volume for the training set (first 7 days totaling 2016 observations). The state-space dimension q for the delay-coordinate reconstruction is increased gradually from 3 to 22. For each reconstruction, the correlation dimension of the chaotic attractor is found. Figures 4(a), (b), and (c) show the limiting behavior of the correlation dimension with time delay τ fixed at 10, for the 5-min, 10-min and 15-min interval traffic volumes, respectively. Asymptotes of the correlation dimension are obtained for a variety of time delays, as shown in Table 1. These dimensions range from 6.408 to 6.462; accordingly, the embedding dimension is found to be 14 for different time intervals and time delays. The results also indicate the choice of time delay in fact is not decisive.

4.2. Prediction of traffic volume

This paper employed the neural network toolbox of MATLAB software to set up neural networks and perform the training. The elements of the input vector consist of 14-dimensional delay coordinates: $x(i), x(i-\tau), x(i-2\tau), \ldots, x(i-13\tau)$, where $x(i)$ is the i-th observation of the time series of traffic volume, and τ is the time delay, which is chosen to be 20, 10, and 5 for 5-min, 10-min, and 15-min traffic volumes, respectively. The network target corresponding to this input is $x(i+T)$, where T is the time interval of the traffic volume. All forecasts are only one time interval ahead of occurrence, i.e., 5-min, 10-min or 15-min ahead of time. When using the strategy "early stopping" to monitor the training process, the allowed number of iterations for the validation error to increase is set to be 5.

4.2.1. 5-min traffic volume

First of all, generate a feedforward, backpropagation neural network with the Bayesian regularization to get the effective number of network parameters. This network inputs and targets are imported from the 14-dimensional delay coordinates of the training set: $x(i), x(i-\tau), x(i-2\tau), \ldots, x(i-13\tau)$, and $x(i+T)$, respectively. The results are shown in Figure 5, which shows the network uses approximately 216 parameters; therefore, the appropriate number of neurons in the hidden layer is found to be 14 (equal to the number of elements in the input vector). Then, replace the number of neurons in the hidden layer with 14 and train the network again by the Levenberg-Marquardt algorithm coupled with the strategy "early stopping." The training process stops at 10 epochs because the validation error already has increased for 5 iterations. Figure 6 shows the scatter plot for the training set with correlation coefficient $\rho = 0.90249$. Lastly, simulate the trained network with the prediction set. Figure 7 shows the scatter plot for the prediction set with the correlation coefficient $\rho = 0.83086$. Time series of the observed value (network targets) and the predicted value (network outputs) are shown in Figure 8. If the strategy “early stopping” is disregarded and 100 epochs is chosen for the training process, the trained network performance indeed improves for the training set, but gets worse for the validation and prediction sets. If the number of neurons in the hidden layer is increased to 28 and 42, the performance of the network for the training set tends to improve, but does not have the tendency to significantly improve for the validation and prediction sets, as listed in Table 2.

4.2.2. 10-min traffic volume

Similarly, the effective number of parameters is first found to be 108, as shown in Figure 9; therefore, the appropriate number of neurons in the hidden layer is found to be 7 (one half of the number of elements in the input vector). Replace the number of neurons in the hidden layer with 7 and train the network again. The training process stops at 11 epochs because the validation error has increased for 5 iterations. Figure 10 shows the scatter plot for the training set with correlation coefficient $\rho = 0.93874$. Simulate the trained network with the prediction set. Figure 11 shows the scatter plot for the prediction set with the correlation coefficient $\rho = 0.91976$. Time series of the observed value (network targets) and the predicted value (network outputs) are shown in Figure 12. If the strategy “early stopping” is disregarded and 100 epochs is chosen for the training process, the performance of the network improves for the training set, but gets worse for the validation and prediction sets. If the number of neurons in the hidden layer is increased to 14 and 28, the performance of the network for the training set tends to improve, but does not have the tendency to improve for the validation and prediction sets, as listed in Table 3.

4.2.3. 15-min traffic volume

Likewise, the effective number of parameters is found to be 88, as shown in Figure 13. Instead of using 6 neurons, 7 neurons (one half of the number of elements in the input vector), are used in the hidden layer for consistence. Replace the number of neurons in the hidden layer with 7 and train the network again. The training process stops at 11 epochs because the validation error has increased for 5 iterations. Figure 14 shows the scatter plot for the training set with correlation coefficient $\rho = 0.95113$. Simulate the trained network with the prediction set. Figure 15 shows the scatter plot for the prediction set with the correlation coefficient $\rho = 0.93333$. Time series of the observed value (network targets) and the predicted value (network outputs) are shown in Figure 16. If the strategy "early stopping" is disregarded and 100 epochs is chosen for the training process, the performance of the network gets better for the training set, but gets worse for the validation and prediction sets. If the number of neurons in the hidden layer is increased to 14 and 28, the performance of the network for the training set tends to improve, but does not have the tendency to significantly improve for the validation and prediction sets, as listed in Table 4.

5. Conclusions

Numerical experiments have shown the effectiveness of the technique introduced in this paper to predict short-term chaotic time series. When reconstructing the state space of the chaotic time series, the choice of time delay is not decisive, when building the neural networks, the effective number of neurons in the hidden layer can be derived with the aid of the Bayesian regularization instead of using the trial and error. Using neurons in the hidden layer more than the number decided by the Bayesian regularization can indeed improve the performance of neural networks for the training set, but does not significantly contribute to the performance for the validation and prediction sets. Although, disregarding the strategy "early stopping" can improve the network performance for the training set, it has worse performance for the validation and prediction sets. For chaotic traffic flow, the longer the traffic volume scales, the better the prediction of the traffic flow.

References

[1] P. S. Addison and D. J. Low, order and chaos in the dynamics of vehicle platoons, Traffic Engineering + Control, July/August, (1996), 456-459.

[2] A. M. Albano, A. Passamante, T. Hediger and M. E. Farrell, Using neural nets to look for chaos, Physica D 58 (1992), 1-9.

[3] K. T. Alligood, T. D. Sauer and J. A. Yorke, Chaos: An Introduction to Dynamical Systems, Springer-Verlag, Inc., New York, (1997).

[4] L. A. Aquirre and S. A. Billings, Validating identified nonlinear models with chaotic dynamics, International Journal of Bifurcation and Chaos in Applied Sciences and Engineering 4(1) (1994), 109-125.

[5] R. Bakker, J. C. Schouten, F. Takens and C. M. van den Bleek, Neural network model to control an experimental chaotic pendulum, Physical Review E 54A (1996), 3545-3552.

[6] G. Deco and B. Schurmann, Neural learning of chaotic system behavior, IEICE Transactions, Fundamentals E77-A(11) (1994), 1840-1845.

[7] H. Demuth and M. Beale, Neural Network Toolbox for Use with MATLAB, User's Guide, Version 4, The Math Works, Inc., Natick, MA, (2004).

[8] J. E. Disbro and M. Frame, Traffic flow theory and chaotic behavior, Transportation Research Record 1225 (1989), 109-115.

[9] J. D. Farmer and J. J. Sidorowich, Predicting chaotic time series, Phys. Rev. Letters 59 (1987), 62-65.

[10] H. Fu, J. Xu and L. Xu, Traffic chaos and its prediction based on a nonlinear car-following model, J. Control Theory Appl. 3(3) (2005), 302-307.

[11] D. C. Gazis, R. Herman and R. W. Rothery, Nonlinear follow-the-leader models of traffic flow, Operational Research 9(4) (1961), 545-567.

[12] P. Grassberger and I. Proccacia, Characterization of strange attractors, Phys. Rev. Letters 50 (1983), 346-349.

[13] M. T. Hagan and M. Menhaj, Training feedforward networks with the Marquardt algorithm, IEEE Transactions on Neural Networks 5(6) (1994), 989-993.

[14] D. D. Hebb, The Organization Behavior, John Wiley, New York, (1949).

[15] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proc. Nat. Acad. Sci., USA 79(8) (1982), 2554-2558.

[16] J. J. Hopfield, D. I. Feinstein and R. G. Palmers, Unlearning has a stabilizing effect in collective memories, Nature 304 (1983), 158-159.

[17] K. Levenberg, A method for the solution of certain problems in least squares, Quart. Appl. Math. 2 (1944), 164-168.

[18] D. J. C. MacKay, Bayesian interpolation, Neural Computation 4(3) (1992), 415-447.

[19] D. Marquardt, An algorithm for least squares estimation of nonlinear parameters, SIAM J. Appl. Math. 11 (1963), 431-441.

[20] W. S. McCulloch and W. Pitts, A logical calculus of ideas immanent in nervous activity, Bull. Math. Biophys. 5 (1943), 115-133.

[21] W. Mendenhall, R. L. Scheaffer and D. D. Wackerly, Mathematical Statistics with Applications, Third Edition, Duxbury Press, Boston, (1986).

[22] F. C. Moon, Chaotic and Fractal Dynamics: An Introduction for Applied Scientists and Engineer, John-Wiley and Sons, New York, (1992).

[23] J. C. Principe, A. Rathie and J. M. Kuo, Prediction of chaotic time series with neural networks and the issue of dynamic modelling, International Journal of Bifurcation and Chaos in Applied Sciences and Engineering 2 (1992), 989-996.

[24] F. Rosenblatt, The perception: a probabilistic model for information storage and organization in the brain, Psychological Review 65(6) (1958), 386-408.

[25] D. E. Rumelhart and J. L. McClelland, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1 (Foundations), The MIT Press, Cambridge, MA, (1986).

[26] F. Takens, Detecting strange attractors in turbulence, Lecture Notes in Mathematics 898 (1981), 366-381.

Table 1. Asymptotes of correlation dimension for different time intervals and delays

Time Delay Time Interval	τ = 10	τ = 20	τ = 30	τ = 40	τ = 50	τ = 60	τ = 70
5-min	6.408	6.427	6.447	6.433	6.432	6.462	6.440
10-min	6.423	6.415	6.416	6.419			
15-min	6.444	6.443	6.430				

Table 2. Correlation coefficients for training, validation and prediction data sets with the number of neurons in the hidden layer increasing (5-min traffic volume)

No. of Neurons Data	14	28	42
Training Set	0.90249	0.90593	0.94371
Validation Set	0.86535	0.86614	0.86757
Prediction Set	0.83086	0.85049	0.82901

Table 3. Correlation coefficients for training, validation and prediction data sets with the number of neurons in the hidden layer increasing (10-min traffic volume)

No. of Neurons Data	7	14	28
Training Set	0.93874	0.95814	0.96486
Validation Set	0.92477	0.87930	0.88337
Prediction Set	0.91976	0.90587	0.91352

Table 4. Correlation coefficients for training, validation and prediction data sets with the number of neurons in the hidden layer increasing (15-min traffic volume)

No. of Neurons Data	7	14	28
Training Set	0.95113	0.96970	0.97013
Validation Set	0.88594	0.93893	0.92177
Prediction Set	0.93333	0.94151	0.94915

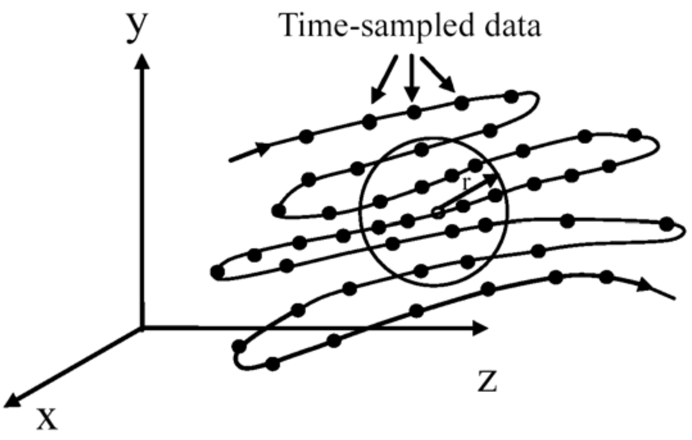


Figure 1. A trajectory in the state space.

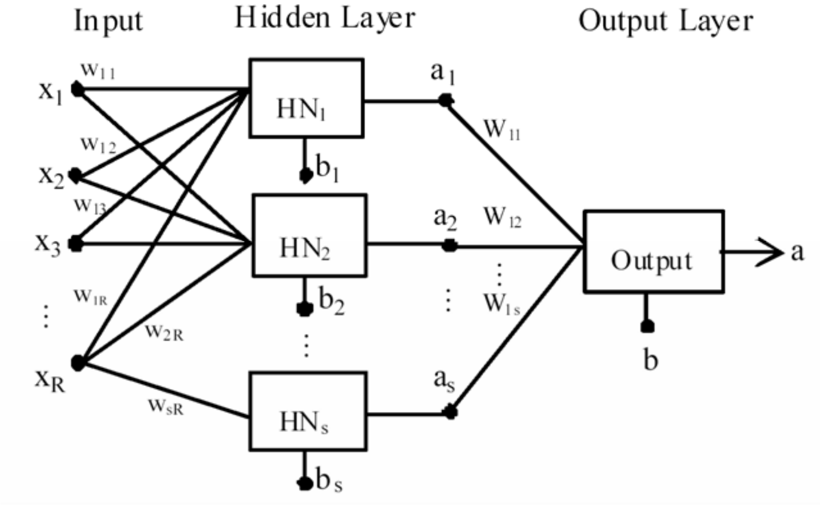


Figure 2. The feedforward neural network with two layers.

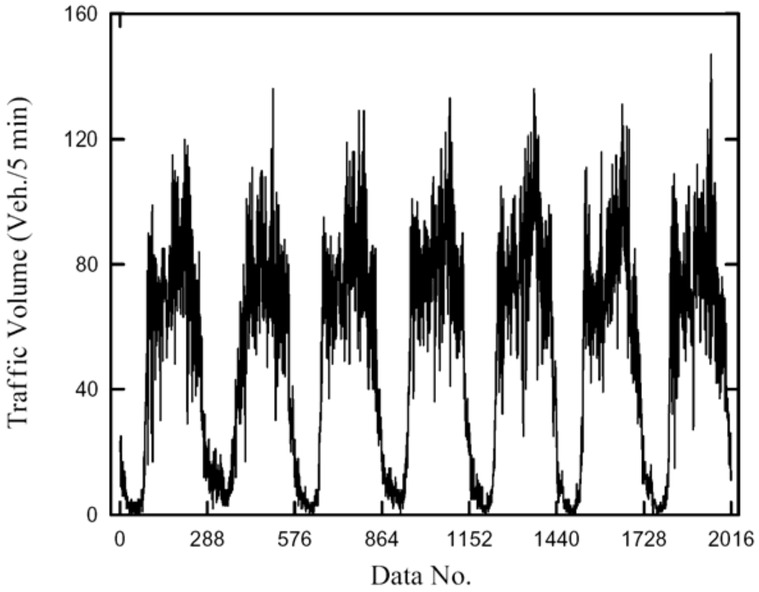
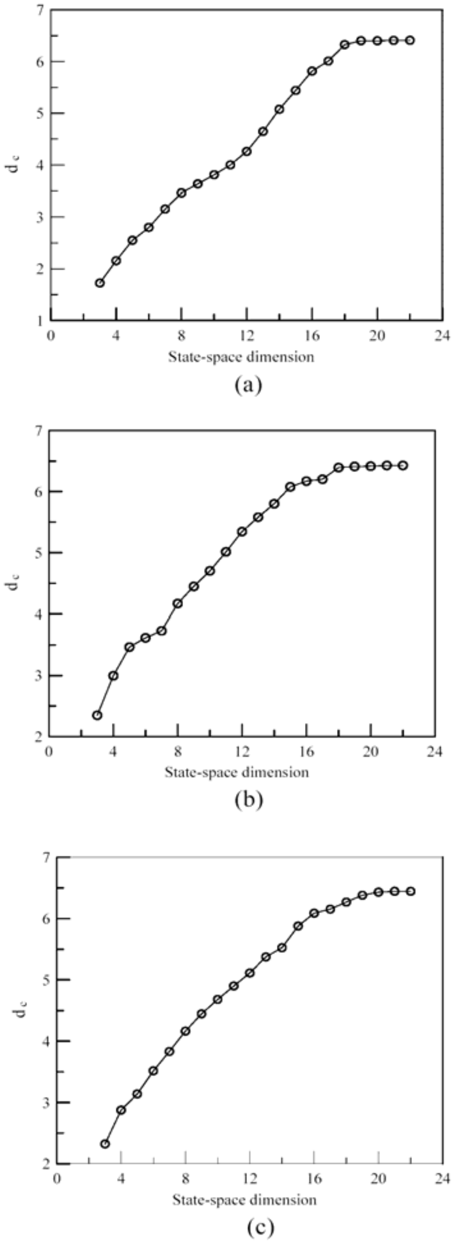


Figure 3. Time series of the 5-min traffic volume.



(a)

(b)

(c)

Figure 4. The asymptotes of correlation dimension with the state-space dimension n increasing from 3 to 22 and time delay τ fixed at 10, (a) the 5-min traffic volume, (b) the 10-min traffic volume, and (c) the 15-min traffic volume.

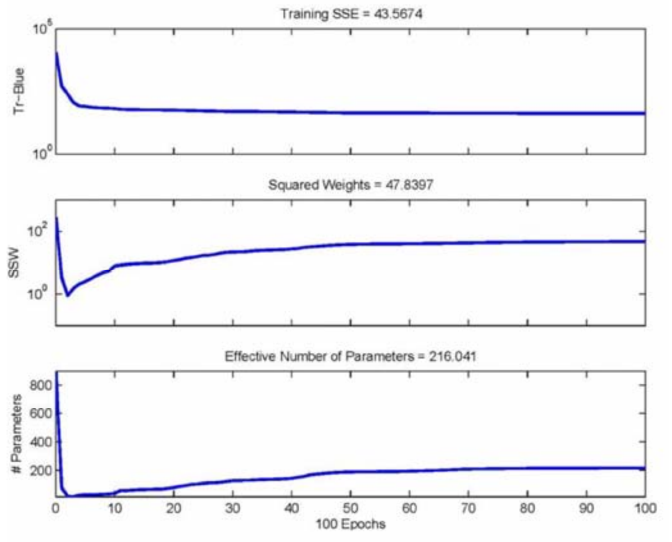


Figure 5. The convergence process to find effective number of parameters used by the network for the 5-min traffic volume.

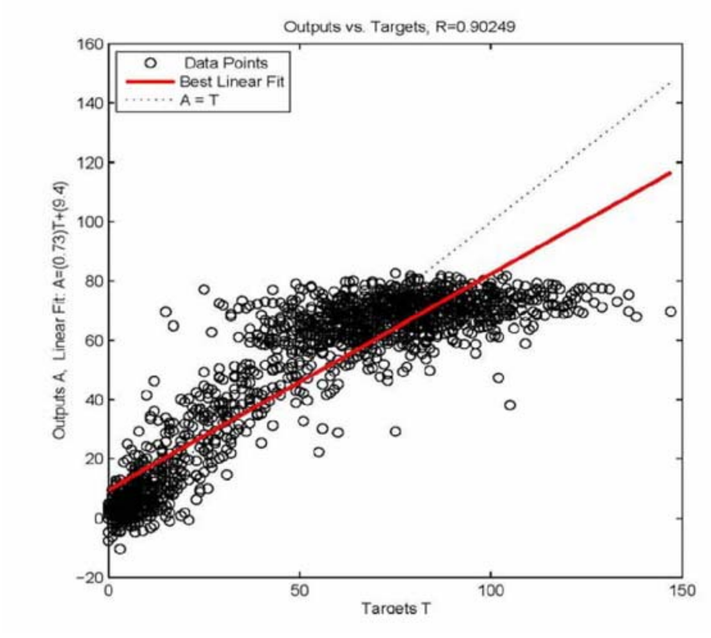


Figure 6. The scatter plot of the network outputs and targets for the training set of the 5-min traffic volume.

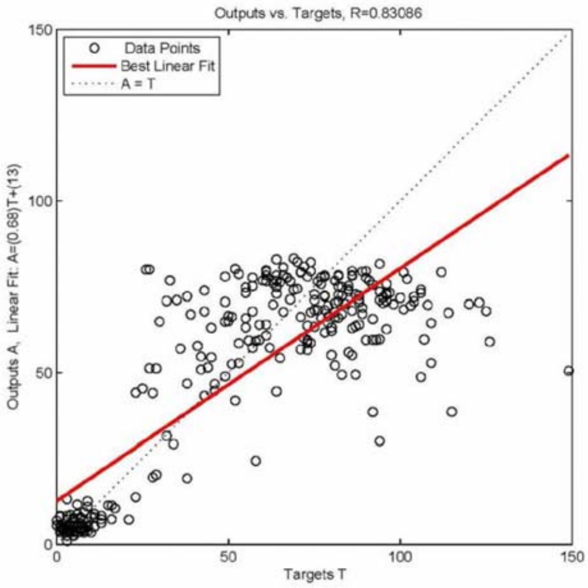


Figure 7. The scatter plot of the network outputs and targets for the prediction set of the 5-min traffic volume.

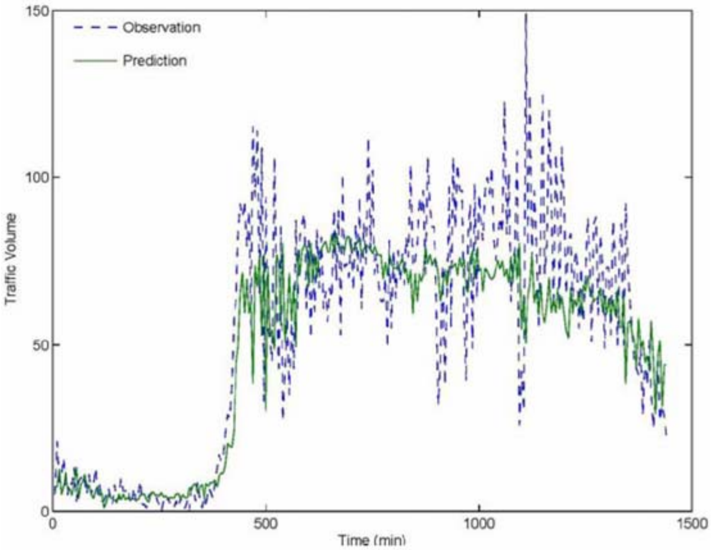


Figure 8. Time series of the observed value (network targets) and the predicted value (network outputs) for the 5-min traffic volume.

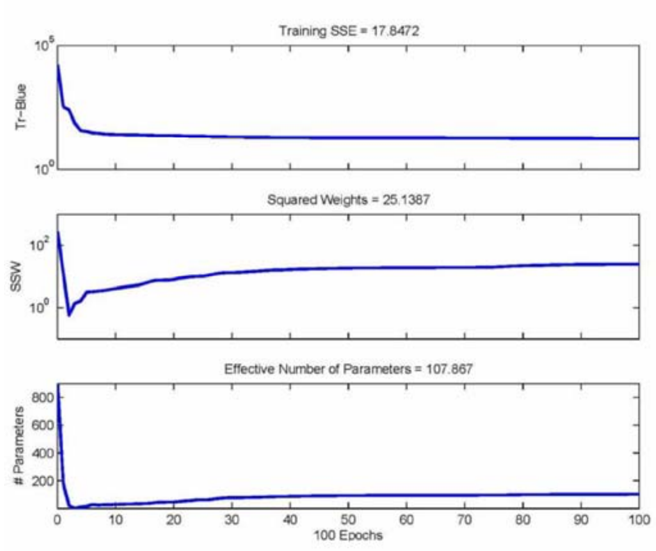


Figure 9. The convergence process to find effective number of parameters used by the network for the 10-min traffic volume.

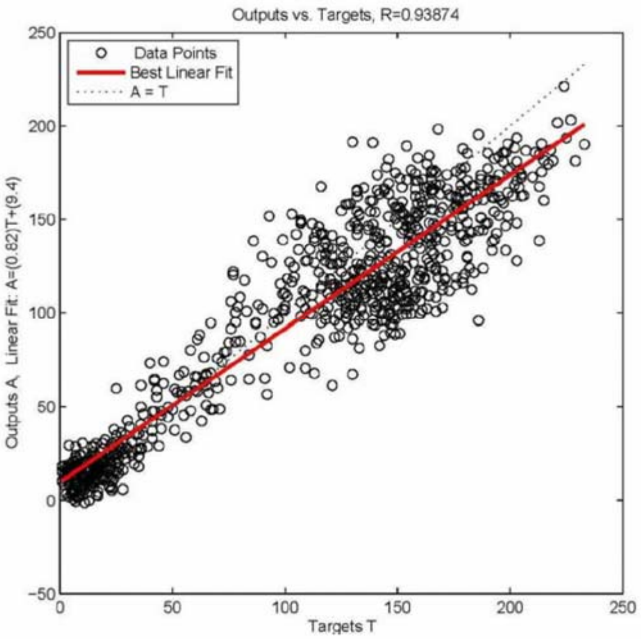


Figure 10. The scatter plot of the network outputs and targets for the training set of the 10-min traffic volume.

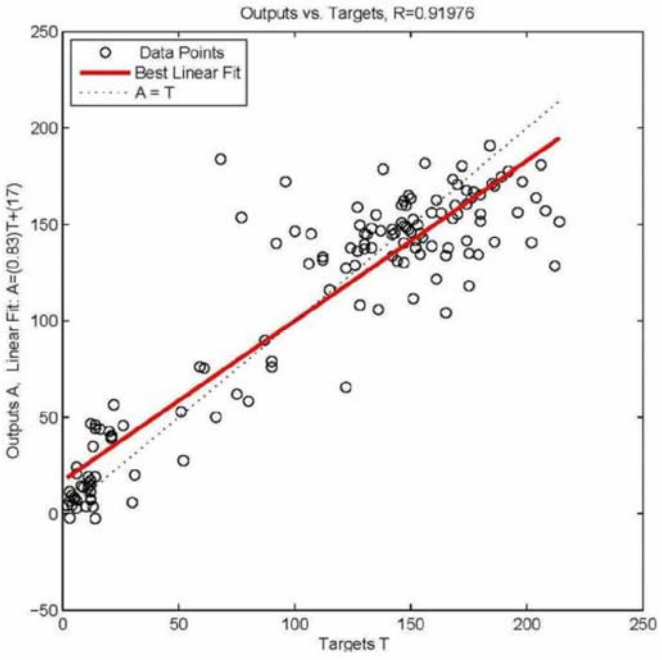


Figure 11. The scatter plot of the network outputs and targets for the prediction set of the 10-min traffic volume.

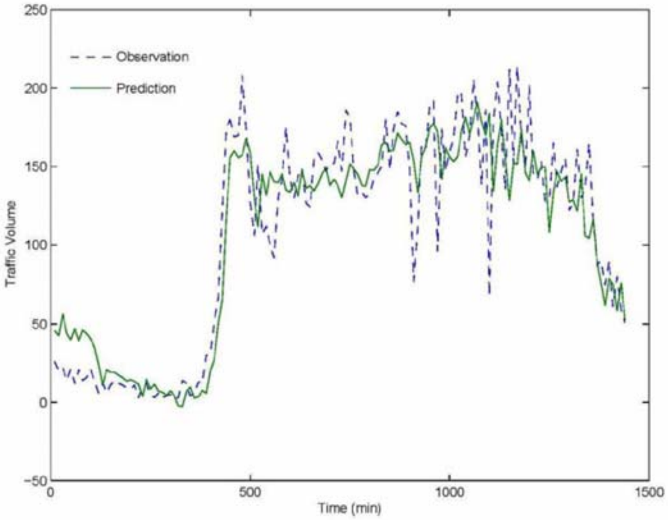


Figure 12. Time series of the observed value (network targets) and the predicted value (network outputs) for the 10-min traffic volume.

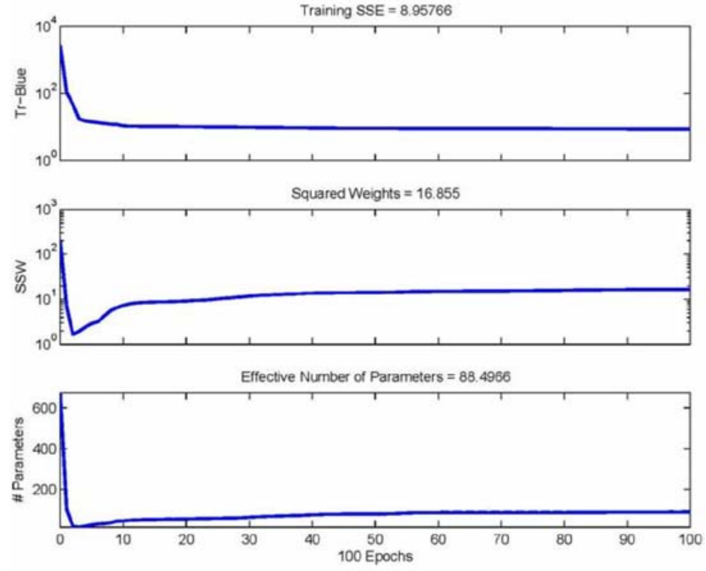


Figure 13. The convergence process to find effective number of parameters used by the network for the 15-min traffic volume.

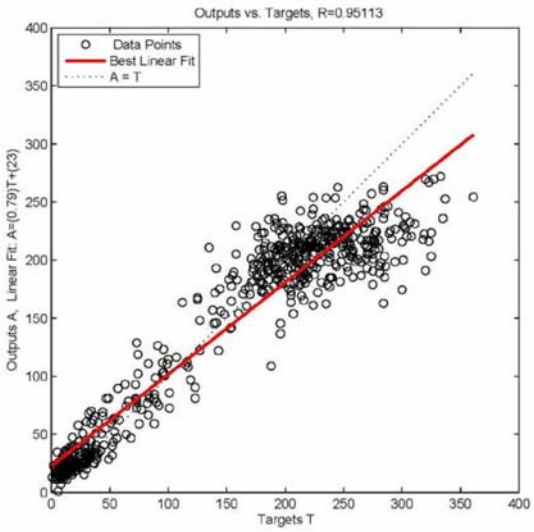


Figure 14. The scatter plot of the network outputs and targets for the training set of the 15-min traffic volume.

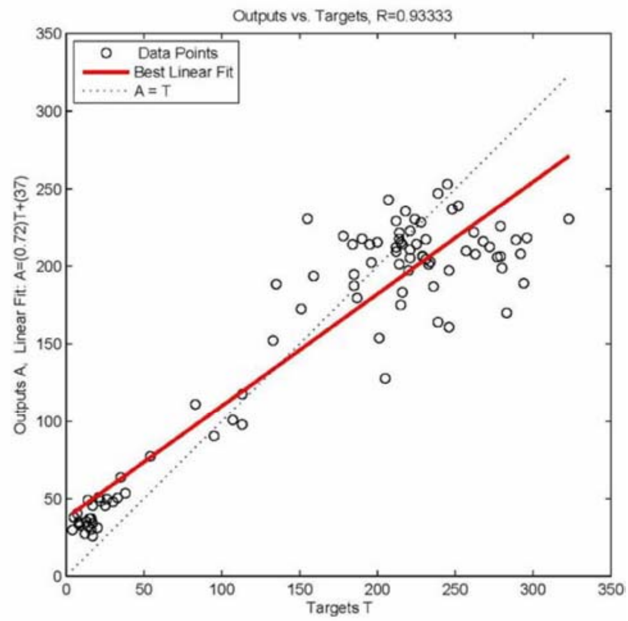


Figure 15. The scatter plot of the network outputs and targets for the prediction set of the 15-min traffic volume.

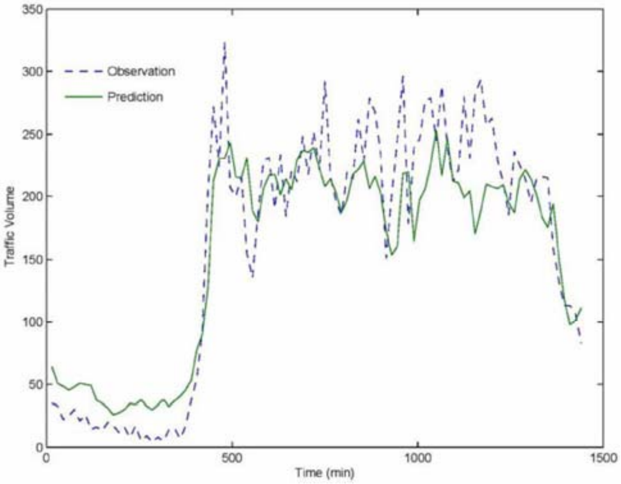


Figure 16. Time series of the observed value (network targets) and the predicted value (network outputs) for the 15-min traffic volume.

■