ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 39-44

Self-Organizing Maps and Functional Networks for Local Dynamic Modeling

Noelia Sánchez-Maroño[*], Oscar Fontela-Romero,
Amparo Alonso-Betanzos, Bertha Guijarro-Berdiñas

Laboratory for Research and Development in Artificial Intelligence,
Department of Computer Science, University of A Coruña,
Campus de Elviña s/n, 15071 A Coruña, Spain

Abstract. The paper presents a method for times series prediction using a local dynamic modeling based on a three step process. In the first step the input data is embedded in a reconstruction space using a memory structure. The second step, implemented by a self-organizing map (SOM), derives a set of local models from data. The third step is accomplished by a set of functional networks. The goal of the last network is to fit a local model from the winning neuron and a set of neighbors of the SOM map. Finally, the performance of the proposed method was validated using two chaotic time series.

1 Introduction

The problem of time series prediction can be viewed as a function approximation problem where the aim is to estimate a complex function $f(\boldsymbol{x})$ by means of another function $\hat{f}(\boldsymbol{x})$. There are two basic alternatives to proceed with this approximation: global and local. In the global approach, only one model is used to characterize the process, so $\hat{f}(\boldsymbol{x})$ is estimated as:

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \phi_i(\boldsymbol{x}) \tag{1}$$

where α_i are coefficients and $\phi_i(\boldsymbol{x}); i = 1, 2, \cdots, n$ are basic functions. However, if $f(\boldsymbol{x})$ is complicated, there is no guarantee that $\hat{f}(\boldsymbol{x})$ will approximate it adequately. This problem can be reduced by using the local approach where the domain of $f(\boldsymbol{x})$ is broken into local neighborhoods and a separate model is

This work is partially supported by the Xunta de Galicia (project PGIDT-01PXI10503PR) and the first author is supported by a grant from Diputación de A Coruña.

[*]Corresponding author. Email: noelia@mail2.udc.es

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 39-44

used for each one [3], so the overall predictive function is the union of several local models as it is expressed in equation (2). The division of the data set is usually carried out with some clustering or quantization algorithm such as Self-Organizing Map (SOM) or a k-means algorithm.

$$\hat{f}(\boldsymbol{x}) = \bigcup_{i=1}^{m} \hat{f}_i(\boldsymbol{x}) \tag{2}$$

In this paper, a system for local dynamic modeling based on a previous work designed by the same authors [2] is presented. The system is composed by a SOM, that divides the data set into smaller sets, to get several local models and a set of functional networks [1] that approximate each local model.

2 Dynamic Modeling

Takens' Embedding Theorem [4] proved that a sample of a dynamic system $x(n)$ and its delayed versions $\boldsymbol{x}(n) = [x(n), x(n-\tau), \ldots, x(n-(N-1)\tau)]$, where τ is a specific time delay, can be created to get a trajectory of the system in an euclidean space of size N without modifying the dynamic invariants of the original system. N must be greater than 2D, where D is the dimension of the original system. According to this theorem, exists a map $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$ that transforms the current reconstructed state $\boldsymbol{x}(n)$ to the next state $\boldsymbol{x}(n+\tau)$. For simplicity, we consider $\tau = 1$, which means:

$$\boldsymbol{x}(n+1) = \boldsymbol{f}(\boldsymbol{x}(n)) \tag{3}$$

Note that (3) specifies a multiple input, multiple output system. For a system with one output, the predictive mapping: $f : \mathbb{R}^N \to \mathbb{R}$ can be expressed as:

$$x(n+1) = f(\boldsymbol{x}(n)) \tag{4}$$

Dynamic modeling implies a two-step process. The first step is to transform the observed time series into a trajectory in the reconstruction space by using an embedding technique. The most common one is a time delay embedding which can practically be implemented with a delay line with the size specified by Takens' Embedding Theorem. The second step is to build the predictive model (4) from the trajectory in the reconstruction space. As it was mentioned, both local and global modeling can be used for this purpose.

3 Functional Networks

Functional networks are a generalization of neural networks that combine both knowledge about the structure of the problem, to determine the architecture of the network, and data, to estimate the unknown functional neurons [1]. As it can be seen in Fig.1, a functional network consists of: a) several layers of storing units, one layer for containing the input data ($x_i; i = 1...4$), another

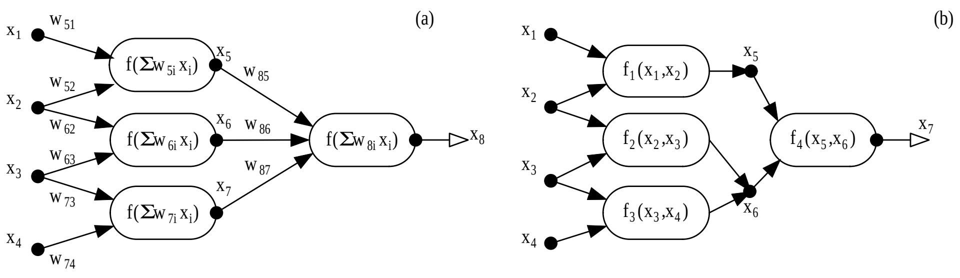


Figure 1: (a)A neural network. (b)A functional network

for containing the output data (x_7) and none, one or several layers to store intermediate information (x_5 and x_6); b) one or several layers of processing units that evaluate a set of input values and delivers a set of output values (f_i) and c) a set of directed links. Functional networks extend neural networks by allowing neural functions f_i to be not only true multiargument and multivariate functions, but to be different and learnable, instead of fixed functions. In functional networks, the activation functions are unknown functions from a given family, i.e. polynomial, to be estimated during the learning process. In addition, functional networks allow connecting neuron outputs, forcing them to be coincident. Some differences between a neural network and a functional network are shown in Fig. 1.

4 System Design

As the system in [2], the proposed method is composed of three blocks:

1. *An embedding layer*, implemented by a time delay line, to transform the original space of the time series into a reconstruction space. The output of this layer is a sequence of N-dimensional state vectors, $\boldsymbol{x}(n) = [x(n), x(n-\tau), \ldots, x(n-(N-1)\tau)]^T$, created from the input signal.

2. *A self-organizing map* trained using Kohonen's learning. The input of this network is the pair $(d(n), \boldsymbol{x}(n))$, where $d(n)$ is the desired response. If a time series prediction scenario is employed then $d(n) = x(n + \gamma\tau)$, where γ is a predetermined prediction step. The SOM, formed by $P \times Q$ neurons, will represent the system dynamics in the discrete output lattice, but enhanced with neighborhood relationships. These neighborhood relationships of the SOM assure the desirable property of continuity among the local models. The weights of the k^{th} neuron of the SOM is represented by the vector $\boldsymbol{w}_k = [w_{0k}, w_{1k}, \ldots, w_{Nk}]^T; k = 1, \ldots, P \times Q$, where w_{0k} is associated with the input $d(n)$ and the other N correspond to the $\boldsymbol{x}(n)$ vector. It is important to say that once the SOM has been trained, $d(n)$ will not be used as input in a real environment.

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 39-44

3. *A set of $P \times Q$ functional networks.* Each functional network k is associated with the k^{th} neuron of the SOM and it was used to fit each local model from a subset of weights of the SOM. For the training process, the following cost function was used:

$$J^k = \sum_{t \in T_k} (\epsilon_t^k)^2 = \sum_{t \in T_k} (d_t^k - y_t^k)^2 \tag{5}$$

where T_k is the set formed by the index of the k^{th} neuron of the SOM and the indexes of its N_L neighbors, and the desired output d_t^k is w_{0t}. To determine y_t^k, the model shown in (6) was employed, where the polynomial family is used to learn the neural functions f_i^k and the coefficients of these polynomials functions, a_{ij}^k and a_0, are the parameters to be learned.

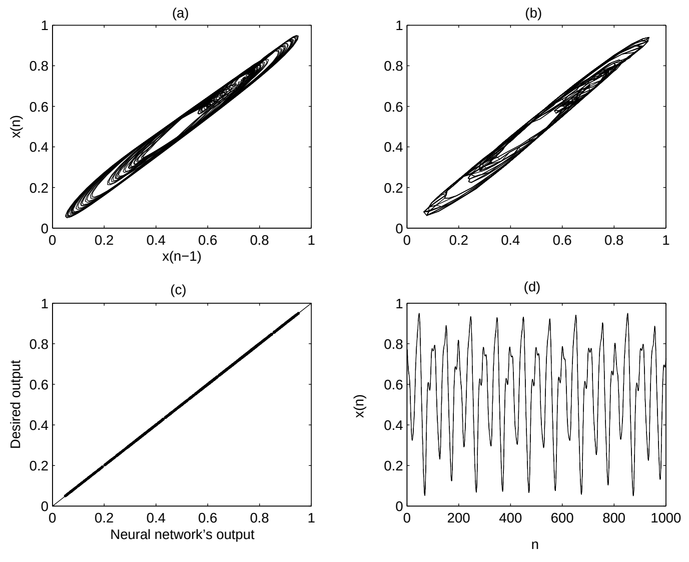
$$\begin{gathered} y_t^k = a_0^k + \sum_{i=1}^{N} f_i^k(w_{it}) = a_0^k + \sum_{i=1}^{N} \sum_{j=1}^{m} a_{ij}^k (w_{it})^j \\ t = 1, \ldots, card(T_k); \quad k = 1, \ldots, P \times Q \end{gathered} \tag{6}$$

Once the system was trained, it will work as follows:

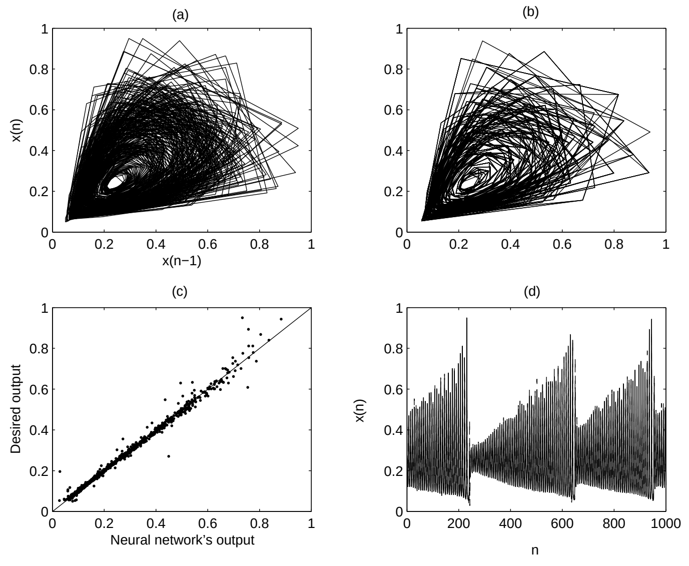
- The winning neuron of the SOM is determined from an input vector $\boldsymbol{x}(n)$ obtained from the embedding layer.
- The functional network associated with this winning neuron will be activated and it will give the output related to $\boldsymbol{x}(n)$.

5 Simulations

The system proposed was validated using synthetic and real chaotic time series. The synthetic data set employed was the Mackey-Glass chaotic series and the real-world data set used was the laser time series from the Santa Fe Time Series Competition [5]. The number of samples selected for training and testing in both series was 2000 and 1000, respectively. A comparative between the method proposed and our previous work[2] based on a SOM and a set of single layer neural networks is shown in Table 1. It can be noticed that the normalized mean square error (NMSE) is reduced considerably in our new approach. Moreover, the SOM employed is simpler than the one needed in our previous work. Fig. 2 shows the results obtained for the Mackey-Glass data (subfigure 2.1) and the laser data (subfigure 2.2). In both subfigures (a) contains the two-dimensional attractor of the chaotic time series, (b) depicts, for the train data, the sequence of winning neurons connected with lines showing that the SOM is able to learn the trajectory of the time series, (c) shows the real versus the desired output and (d) shows the series used for the test data (solid line) and the output of the system proposed (dashed line). As it can be seen in Fig. 2.1(c) and 2.1(d), there is no difference between the real Mackey-Glass time series and the predicted series.



2. 1: Results for Mackey-Glass Data



2. 2: Results for Laser Data

Figure 2: Results for Mackey-Glass and Laser data: (a) two-dimensional attractor, (b) trajectory of winning neurons in the SOM, (c) real versus desired output and (d) observed (solid line) and predicted (dashed line) time series.

		N	$P \times Q$	N_L	m	$NMSE$
$Mackey - GlassData$	$SOM + FN$	7	20×20	41	2	$5.74e - 7$
	$SOM + NN$	7	25×25	35	–	$7.62e - 4$
$LaserData$	$SOM + FN$	5	20×20	48	2	$1.10e - 2$
	$SOM + NN$	5	30×30	45	–	$1.36e - 2$

Table 1: Results for the proposed method using a SOM and Functional Networks(FN) and our previous method using a SOM and Neural Networks(NN).

6 Conclusion

In this work, a new method for local dynamic modeling is presented. The proposed method uses a SOM for getting the local models and later a functional network to approximate each one. Simulations over two benchmark data demonstrated that this method is able to capture accurately the underlying dynamics of chaotic signal. The method shows a better performance than a previous work based on a SOM and a set of single layer neural networks.

References

[1] E. Castillo, A. Cobo, J.M. Gutierrez, and R.E. Pruneda. *Functional Networks with Applications. A Neural-Based Paradigm.* Kluwer Academic Publishers, Dordrecht, 1998.

[2] O. Fontenla-Romero, A. Alonso-Betanzos, E. Castillo, J.C. Principe, and B. Guijarro-Berdiñas. Local modeling using self-organizing maps and single layer neural networks. In J.R. Dorronsoro, editor, *Lectures Notes in Computer Science*, volume 2415, pages 945–950, Heidelberg, Germany, 2002. Springer-Verlag.

[3] S. Lawrence, A.C. Tsoi, and A.D. Back. Function approximation with neural networks and local methods: Bias, variance and smoothness. In P. Bartlett, A. Burkitt, and R. Williamson, editors, *Australian Conference on Neural Networks*, pages 16–21. Australian National University, 1996.

[4] F. Takens. Detecting strange attractors in turbulence. In D.A. Rand and L.S. Young, editors, *Dynamical systems and Turbulence (Lecture Notes in Mathematics)*, volume 898, pages 365–381, New York, 1980. Springer-Verlag.

[5] A.S. Weigend and N.A. Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past.* Addison-Wesley, Reading, MA, 1994.