Neural Process Lett
DOI 10.1007/s11063-011-9174-0

A Neural Network Scheme for Long-Term Forecasting of Chaotic Time Series

Pilar Gómez-Gil · Juan Manuel Ramírez-Cortes · Saúl E. Pomares Hernández · Vicente Alarcón-Aquino

Abstract The accuracy of a model to forecast a time series diminishes as the prediction horizon increases, in particular when the prediction is carried out recursively. Such decay is faster when the model is built using data generated by highly dynamic or chaotic systems. This paper presents a topology and training scheme for a novel artificial neural network, named "Hybrid-connected Complex Neural Network" (HCNN), which is able to capture the dynamics embedded in chaotic time series and to predict long horizons of such series. HCNN is composed of small recurrent neural networks, inserted in a structure made of feed-forward and recurrent connections and trained in several stages using the algorithm back-propagation through time (BPTT). In experiments using a Mackey-Glass time series and an electrocardiogram (ECG) as training signals, HCNN was able to output stable chaotic signals, oscillating for periods as long as four times the size of the training signals. The largest local Lyapunov Exponent (LE) of predicted signals was positive (an evidence of chaos), and similar to the LE calculated over the training signals. The magnitudes of peaks in the ECG signal were not accurately predicted, but the predicted signal was similar to the ECG in the rest of its structure.

Keywords Long-term prediction · Hybrid-connected Complex Neural Network · Recurrent neural networks · Chaotic time series · ECG modeling · Mackey-Glass equation

P. Gómez-Gil (✉) · S. E. Pomares Hernández
Department of Computational Science, National Institute of Astrophysics, Optics and Electronics, Puebla, Mexico
e-mail: pgomez@acm.org

J. M. Ramírez-Cortes
Department of Electronics, National Institute of Astrophysics, Optics and Electronics, Puebla, Mexico

V. Alarcón-Aquino
Department of Electronic Engineering, Universidad de las Américas, Puebla, Mexico

Published online: 08 March 2011

1 Introduction

Modeling of nonlinear and chaotic systems constructed from observed data has become an important issue in the last years [1,2]. Frequently, such data comes from uniform sampling of a continuous signal $s(t)$, and the model is used to predict one or several future values of the time series. A time series may be periodic, aperiodic, stationary (whose statistics do not change with time), cycle-stationary (whose statistics change in time according to a known pattern), non-stationary (whose statistics change over time) or chaotic [3]. Chaotic signals result from deterministic systems showing an aperiodic, long-term behavior that exhibits sensitive dependence on initial conditions [4]. Examples of chaotic signals are electroencephalograms (EEG) [5], electrocardiograms (ECGs) [3], body temperature samplings [6], sunspot numbers [7], video traffic [8], etc.

It is well known that accurate long-term prediction for chaotic systems is limited by the value of the largest Lyapunov Exponent (LE) of the system [9]. However, if a good reconstructed map of the system is built, a good short-term predictor may be built and used for long-term prediction through recursive prediction, provided that the model includes ways to compensate for systematic errors [10]. *Recursive prediction* consists of calculating future values of a series based on knowledge of past values and values calculated by the predictor itself. Long-term predictors of chaotic series have applications in several fields, for example medicine, economy, meteorology, geology and finances; techniques to build them include catastrophe theory, chaos modeling and artificial neural networks [11]. Among them, Recurrent Neural Networks are considered one of the most powerful strategies to tackle problems where non-stationary and complex signals play a major role [12].

Advances in the design of machine learning algorithms and increments of computational power have allowed that a huge number of prediction problems have been tackled using neural networks or ensembles that use them, and everyday new models appear. Next we briefly describe some recent research related to long-term prediction using artificial neural networks: Beliaev and Kozma [13] introduced a chaotic neural network, called K_{III}. It is composed of a multi-layer architecture with excitatory and inhibitory neurons and massive lateral, feed-forward and delayed feedback connections between layers. They use K_{III} for multi-step time series prediction applied to the IJCNN CATS benchmark data [14]. Júnior and Barreto [8] showed that a Nonlinear Autoregressive Model with Exogenous inputs (NARX) network, originally proposed by Leontaritis and Billings [15], outperforms a Time Delay Neural Network (TDNN) [16] and that a Simple Recurrent Network architecture, originally proposed by Elman [17]. The authors applied NARX to long-term prediction of a chaotic laser time series and to a variable bit rate (VBR) video traffic time series. Park and collaborators [18] introduced a wavelet-based neural network architecture, called "Multiscale BiLinear Recurrent Neural Network" (M-BLRNN) using it for the long-term prediction of network traffic. Cai et al. [19] built an architecture that automates the design of a recurrent neural network through an evolutionary learning algorithm based on a particle swarm optimization; they used their model to predict 100 missing values of a time series provided by the IJCNN 2004 time series prediction competition [14]. Alarcón-Aquino and Barria [20] used the maximal overlap discrete wavelet transform (MODWT) to train a multi-resolution FIR Neural Network and applied it to network traffic prediction. Sapankevich and Sankar [21] reported an excellent survey on the use of support vector machines for both short and long-term prediction.

In this paper we present a novel topology and training scheme of a neural network, able to forecast chaotic signals with some degree of accuracy using long-term prediction. The scheme includes pre-processing the training signals and a particular training scheme. The net

topology, named "Hybrid-connected Complex Neural Network[1]" (HCNN) which was first presented at [22], is built using recurrent and feed-forward connections. The most important advantage of this topology is that it is able to oscillate in a chaotic way under bounded values. This allows HCNN to forecast a chaotic signal longer than other predictors using recursive prediction. The neural components of HCNN are trained in several phases using the algorithm "back-propagation through time" (BPTT), originally proposed by [23], and following an implementation proposed in [24].

This paper is organized as follows: Sect. 2 presents main concepts associated to the construction of embedded systems from data; Sect. 3 describes the architecture of HCNN and its components; Sect. 4 describes the particular way in which HCNN is trained to capture the dynamics of the system, pointing out some issues related to the implementation of the algorithm BPTT and recursive prediction; Sect. 5 shows the results obtained by HCNN when it was trained to predict large horizons of two signals: a time series obtained by the integration of a Mackey Glass equation and a real ECG. Finally, Sect. 6 discusses some conclusions and future work.

2 Main Concepts

Let $s(n)$ be a scalar time series obtained from the uniform sampling of a continuous signal $s(t)$, using a sampling time τ_s and starting at some time t_0. The state of the dynamical system producing the signal is composed of many unknown variables that may be represented in a *state vector* $\mathbf{x}(n)$ following the unknown rule [9]:

$$\mathbf{x}\,(n+1) = \{\mathbf{F}(\mathbf{x}(n))\} \tag{1}$$

Another rule defines the relation among the observations and the state variables as follows:

$$s(n) = h(\mathbf{x}(n)) \tag{2}$$

In order to predict the future behavior of $s(n)$, building a model approximating $\mathbf{F}(\cdot)$ is required. To do so, first it is required to create a d-dimensional state space of vectors $\mathbf{z}(n)$ that is a proxy for the unknown $\mathbf{x}(n)$. According to Takens [25] such space can be represented as:

$$\mathbf{z}(n) = [s(n), s\,(n - T_L)\,, s\,(n - 2T_L)\,, \ldots, s\,(n - (d-1)\,T_L)] \tag{3}$$

where T_L is the integer *time lag* that makes the components of $\mathbf{z}(n)$ independent in a nonlinear sense [9]; d the dimension required to unambiguously represent the trajectories of the system in state space; d is also known as the *embedding dimension*.

To reconstruct this embedded state space, it is assumed that $s(n)$ and $s(n + T_L)$, for some integer T_L, are independent samplings of the state of the nonlinear system [9]. During time $T_L \tau_s$ the system evolves and the unknown variables are now reflected in $s\,(n + T_L)$. Appropriate values of T_L and d have to be determined to reconstruct $\mathbf{z}(\cdot)$. Currently, there are several methods to calculate both values from $s(n)$ [1]. Abarbanel et al. [26] describe a method to calculate T_L based on finding the first zero of the auto-correlation function of $s(n)$. They also propose to calculate d in an iterative process involving the calculation of the correlation function for increasing values of d, starting at one. They choose the value of d when the correlation does not change with an increment of d. Both methods were used in this research to calculate the values of T_L and d for the experiments shown in this paper.

[1] Formerly known as "Hybrid Complex Neural Network".

A standard procedure to determine if a measured signal comes from a chaotic system is calculating the local LEs in the embedded space. LE are a measure of the mean ratio of contractions or expansions near the limit in a non-linear dynamical system [27]. There are as many LE as dimensions in the system and, in a chaotic system, at least one LE is positive. Values of local LE of an unknown dynamical system may be approximated using numerical methods. Gencay and Dechert [28] proposed an algorithm for obtaining LEs of an unknown dynamical system, based on a multivariate feed-forward network estimation technique. This idea was used to design the HCNN, as described in Sect. 3.2. The algorithm developed by Wolf et al. [29] monitors the long-term evolution of a single pair of nearby orbits of the system in order to estimate the largest LE. This algorithm was used to calculate the LE in the training and predicted signals used in the experiments reported in this paper. The algorithm attempts to approximate the local tangent space around a fiducial orbit of the system [29]. After calculating the embedded system, the algorithm finds the nearest neighbor in the reconstructed space to the first point in the orbit. The magnitude of the difference vector is recorded. Subsequently, the point evolves along its trajectory a given number of steps. The magnitude of the final separations is determined, and a contribution to the largest LE is calculated as the logarithm of the final separation divided by the initial separation. All contributions are averaged over the length of the time series. If the distance between neighbors becomes too large, the algorithm abandons this point and searches for a new neighbor. Sano and Sawada [30] introduced a method to determine the Lyapunov spectrum from a chaotic time series, based on a least-square-error optimal estimation of a linear flow map of the tangent space from the data sets. Rosenstein et al. [31] proposed an algorithmic approach for finding the largest LE based on the reconstruction of the attractor dynamics from the time series, with an estimation of lag and mean period using the Fast Fourier Transform. LE are estimated using the mean rate of nearest-neighbor separation of each point on the trajectory. Darbyshire and Broomhead [32] presented an approach for obtaining the spectrum of LE based on least square and total least square methods applied to the estimation of tangent maps from time series data.

The HCNN implements an approximation of map $\mathbf{F}(\cdot)$ (Eq. 1), which is built with information given by the embedded state space $\mathbf{z}(n)$ (Eq. 3) obtained from the observed time series $s(n)$. After being trained, HCNN represents a system with a maximum LE similar to the one calculated over the training time series $s(n)$. Next we present details of the design and implementation of HCNN.

3 Architecture of HCNN

Hybrid-connected Complex Neural Network is a combination of components connected in recurrent and feed-forward fashions; it receives as input $\mathbf{z}(n)$ and output last variable in $\mathbf{z}\,(n+1)$, that is, $z_d\,(n+1)$. When HCNN is used to predict long horizons, it receives inputs calculated by the network itself. HCNN works as a function approximator whose architecture aims to represent two main concepts: (1) the construction of a function aided by a non-linear combination of sinusoidal functions and (2) the modeling of a dynamical system with invariant characteristics similar to the ones found in the unknown system producing the training time series. To represent the first concept, the HCNN uses small fully connected neural networks able to produce discrete samplings of sine signals autonomously, that is, with no external inputs except for their initial conditions [33]. These small networks are inspired in the work of Logar [34]. The second concept is inherent to the ability of multi-layer perceptrons, described by Gencay and Dechert [28], to create a function with a maximum LE similar to the one calculated in the training time series.

The dynamics of each neuron in a recurrent neural network can be defined as [24]:

$$\frac{dy_i}{dt} = -y_i + \sigma\,(x_i) + I_i \tag{4}$$

for all neurons in the network, where

$$x_i = \sum_j w_{ji} y_j \tag{5}$$

represents the input to neuron i coming from all neurons connecting to it; I_i is an external input to neuron i; w_{ji} is the weight connecting neuron i to neuron j; $\sigma\,(x)$ is an activation function, which can be a sigmoid function or a linear function, depending upon the layer where the neuron is located.

Hybrid-connected Complex Neural Network contains recurrent and feed-forward connections, so, in order to use Eq. 4, some connections w_{ji} are set to zero to represent connections that are not required in the structure of the network. To implement HCNN in a digital computer, we approximate Eq. 4 using a first-order difference equation, as proposed by Pearlmutter [24]:

$$\tilde{y}_i\,(n+1) = (1-\mu)\,\tilde{y}_i(n) + \mu\sigma_i\,(\tilde{x}_i(n)) + \mu I_i(n) \tag{6}$$

for all networks in HCNN and where μ is the time step; $\tilde{x}_i(n) = \sum_j w_{ji}\tilde{y}_j\,(n-1)$;

$$I_i(n) = \begin{cases} z_i(n) & \text{if } i \text{ is a neutron in the input layer, that is } i \leq d, \text{ the embedding dimension;} \\ 0 & \text{otherwise} \end{cases}$$

$z_i(n)$ represents each variable of the embedded state, at iteration n, as defined by Eq. 3;

$$\sigma_i\,(x) = \begin{cases} x & \text{if } i \text{ is the neuron in output layer,} \\ \tanh(\alpha x) & \text{otherwise} \end{cases}$$

α is a scaling factor, experimentally adjusted according to magnitudes of training data.

Hybrid-connected Complex Neural Network contains L neurons arranged in four layers. Neurons in the input layer receive external inputs $I_i(n)$, corresponding to $z_i(n)$. The output layer contains only one neuron. The outputs of neurons in the input layer feed neurons in the hidden layer. There is also a layer made of several 3-node, fully connected recurrent networks called *harmonic generators* (HG), which are also connected to the hidden layer. Figure 1 shows an example of HCNN that contains five inputs and seven HGs.

The role of each part of HCNN is: HGs give information to the network about the frequency components of the time series; the hidden layer captures information about the nonlinearity in the system, and it allows internal representation of the dynamics of the model; the last layer allows an approximation of the function generating the network's output $\widetilde{y_L}(n+1)$. The activation function of the unique neuron at the output layer is a linear function, in order to allow the HCNN to approximate any arbitrary value.

3.1 Harmonic Generators

An HG is a three-node, fully connected recurrent neural network that, once trained, is able to reproduce in each neuron, discrete values of a sinusoidal function. This is achieved autonomously, that is, without any external input to the HG, except for the initial values of its neurons $\tilde{y}_i(n=0)$. An HG is trained using one cycle of a discrete sine function with any frequency and amplitude. The output of an HG is defined as the output of one of its neurons, arbitrarily selected. Figure 2 shows the structure of an HG.

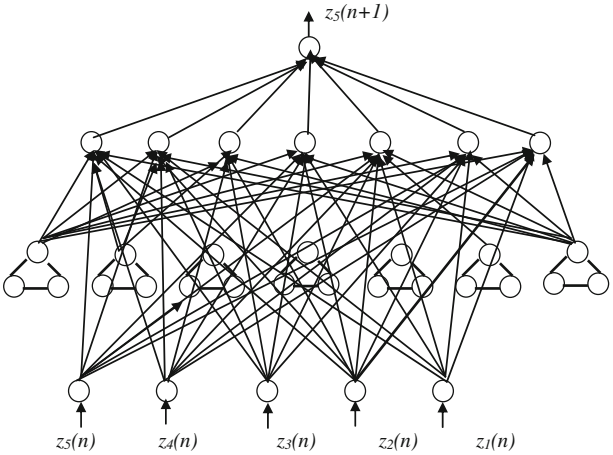

Fig. 1 HCNN with seven HGs and five inputs

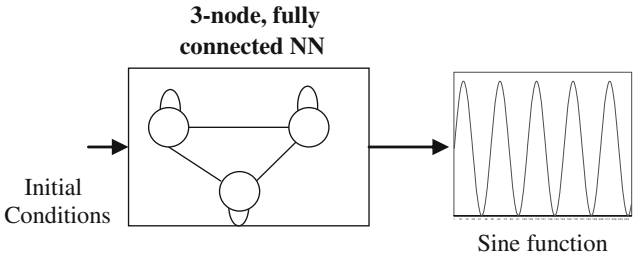

Fig. 2 A HG [33]

The HCNN includes r HGs, each trained to learn different discrete sinusoidal functions representing frequency components of the time series to be predicted. Each HG is trained with a time series $h_i(n)$ defined as:

$$h_i(n) = Asin\left(\frac{2\pi i \varphi_o}{N}\right) n \quad n = 1, \ldots, N_{h_i}/\varphi_o; \quad i = 1, \ldots, r \tag{7}$$

where φ_o is the fundamental frequency obtained from the HCNN training series $s(n)$; N_{h_i} is the number of samples of $h_i(n)$; A is the magnitude of the sine function.

An amplitude $A = 0.8$ was determined by experimentation to be used during the training stage. This value allowed the sigmoidal function to keep the output within an adequate range, which supported the network convergence.

The fundamental frequency φ_o was obtained by spectral analysis of the training series $s(n)$, using the component with largest magnitude obtained from its Discrete Fourier Transform:

$$S(k) = \sum_{n=0}^{N_s-1} s(n) e^{-j2\pi kn/N_s} \tag{8}$$

where N_s is the number of samplings in $s(n)$.

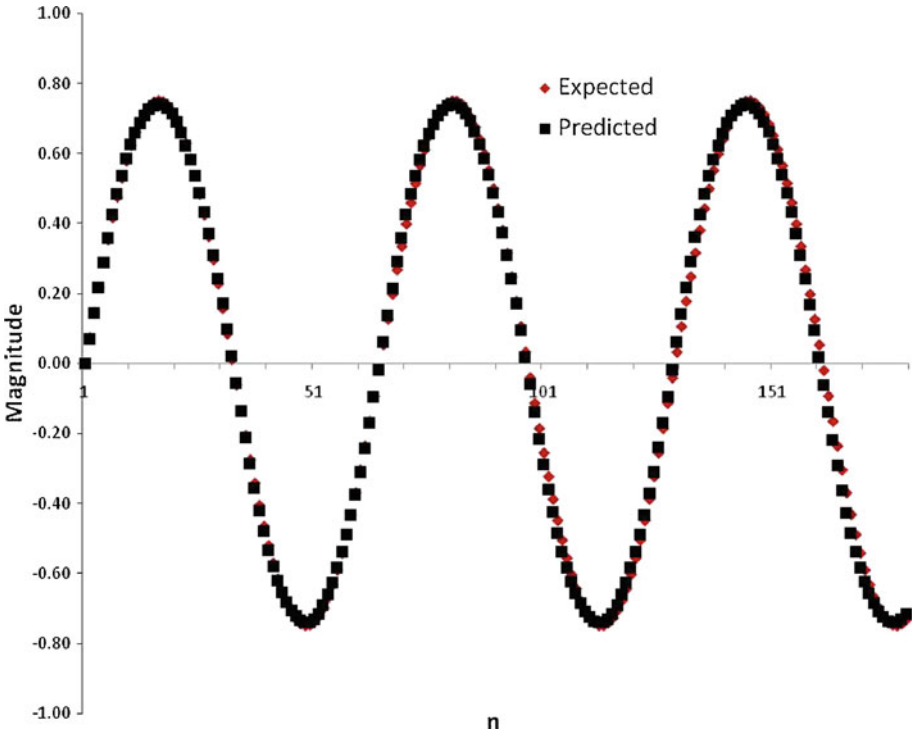

Fig. 3 Example of the output of neuron 3 of a trained HG

Figure 3 shows a plot of the first 200 discrete sine values of a training series $h_i(n)$ and the values autonomously generated by an HG in its node number 3. After being trained, the weights of this HG were:

$$w_{ij} = \{0.9064, 2.0437, -0.1731, 2.1340, 4.3370, -0.4494, 1.0664, 2.2310, 3.8437\};$$

initial values for each neuron were: $\tilde{y}_i(n=0) = \{0.3449, -0.7061, 0.0\}$ for $i, j \in \{1, 2, 3\}$; $\sigma(x) = \tanh(0.5x)$ and $\mu = 0.1$. This HG was trained using one cycle of a discrete sine function with $\varphi_o = 1.4$ and $N = 90$. Note that expected and predicted signals are almost equal, even in the segment corresponding to testing (points 91–180).

3.2 Representation of the Dynamical Invariants

After training, the HCNN is aimed to be a dynamical system with invariant characteristics similar to the ones found in the unknown system producing the training time series. A way to achieve this is to make the HCNN represent a function whose local LE are similar in magnitude to those found in the training time series. To do so, a multi-layer perceptron architecture (MLP) is built into the HCNN. It has been proved that a MLP with d inputs and one hidden layer approximates a function with d LEs, equal to the LEs of a dynamical system embedded in a time series used to train such MLP [28]; the d corresponds to the embedding dimension

of the system. Therefore, HCNN implements a function f:

$$\widetilde{z_d}(n+1) = f(\mathbf{z}(n)). \tag{9}$$

4 Training and Prediction Scheme of HCNN

There are three possible stages in a prediction system based on HCNN:

1. *Preprocessing*. Here, the embedding dimension d, time lag T_L and fundamental frequency φ_o of training series $s(n)$ are calculated using the algorithms proposed by Abarbanel et al. [26]. After that $\{\mathbf{z}(n)\}$ (Eq. 3) and $r$discrete sine functions $h_i(n)$ (Eq. 7) are built. The value of r is set by experimentation.
2. *Training*. All trainings are carried out using the algorithm back-propagation through time (BPTT) as implemented in [24] (see Sect. 4.1). HCNN is treated as a fully connected neural network where some connections are represented with $w_{ij} = 0$ to implement the feed-forward parts. Training is divided in three steps:
   - 2.1 A total of rHGs are trained to output their corresponding $h_i(n)$ discrete sine functions. Initial conditions of each neuron $y_i\ (n = 0)\,, i = 1..3$ are generated randomly. As described in Sect. 3.1, an HG requires no input signals after it has been trained, except for the initial conditions of each neuron.
   - 2.2 Next, weights of the rHGs are embedded into the HCNN as the initial weight values of this part of the architecture; the rest of the weights in HCNN are randomly generated. After that, the feed-forward part of the HCNN is trained for a number of epochs (hidden and output layer), keeping constant the weights corresponding to HGs. For neurons that are not part of an HG, their initial conditions $\tilde{y}_i\ (n = 0)$ are randomly generated at the beginning of this step. The $y_i\ (0)$ values corresponding to neurons in HGs, are provided by step 2.1.
   - 2.3 Last, the HCNN is trained as a fully connected network for a number of epochs or until a desired *Mean Square Error* (MSE) (Eq. 11) is reached, or a maximum number of sweeps in the training series is executed (see Sect. 4.1).
3. *Prediction*. Once trained, HCNN is ready to predict as many futures values as desired using recursive prediction. Each predicted value is stored in a right-shift register that feeds the inputs to be used for the prediction of the next point (see Fig. 4). Initially, the prediction stage requires as inputs the initial condition of each node $y_i\ (n = 0)$ in the HCNN and $\mathbf{z}(n = 0)$. After $n > d$ the right-shift register contains only approximations of $\mathbf{z}(n)$ and the predictor autonomously calculates as many values as desired.

4.1 Training Algorithm

Back-propagation through time extends general back-propagation algorithm so that it applies to dynamic systems [35]. BPTT looks to minimize the MSE obtained over a period of time, among real and desired values of output neurons. HCNN has only one output neuron, then MSE is defined as:

$$\mathrm{MSE_{HC}} = \frac{1}{N_z}\sum_{j=1}^{N_z}(\tilde{y}_l(j) - z_d(j+1))^2 \tag{10}$$

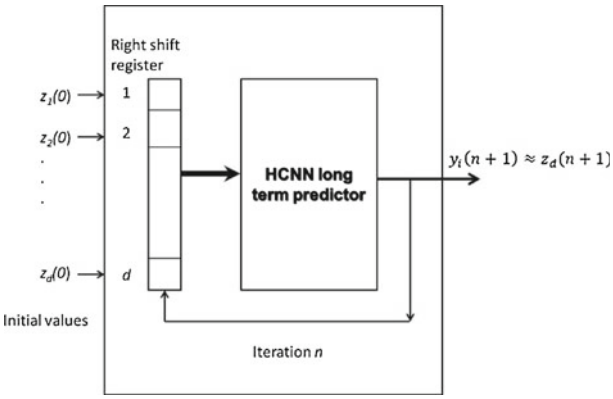

Fig. 4 Long-term prediction using HCNN

where N_z is the size of training set $\{z(\cdot)\}$; $\tilde{y}_l(j)$ is the output of last neuron in the HCNN, the only neuron in the output layer (see Fig. 1), obtained at iteration j; $z_d(j+1)$ is the last variable at embedded system $z(\cdot)$ at iteration $j+1$; it corresponds to the desired value to be predicted; d is the embedding dimension.

For the case of each HG, MSE is defined as:

$$\text{MSE}_{\text{HG}} = \frac{1}{N_{hi}} \sum_{j=1}^{N_{hi}} (\tilde{y}_3(j) - h_i(j))^2 \tag{11}$$

where N_{hi} is the number of samples in $h_i(n)$; $\tilde{y}_3(j)$ is the output of the third neuron in the HG.

Back-propagation through time modifies weights sweeping the training set as many times as needed, until a desired MSE is reached, or a maximum number of sweeps is executed. According to the derivations proposed by Pearlmutter [24], weights are updated on line, using:

$$\Delta w_{ij} = -\eta \sum_{k=1}^{N_z} \tilde{y}_i(k)\, \sigma'\left(\tilde{x}_j(k)\right) d_j(k)\, \mu \tag{12}$$

where N_z is the size of training set $\{z(\cdot)\}$; η is a learning coefficient, experimentally defined; μ is the time step as in Eq. 6; $\tilde{x}_j(k) = \sum_p w_{pj} \tilde{y}_p(k-1)$, $p \in$ {neurons connecting to neuron j};

$$d_i(k) = d_i(k+1) - \mu(d_i(k+1) - e_i(k+1) - \sum_j w_{ij} \sigma'\left(\tilde{x}_j(k+1)\right) d_j(k+1); \tag{13}$$

$d_i(k)$ is calculated backwards over training data in each sweep, before weights are modified,

$$e_i(k) = \begin{cases} \tilde{y}_i(k) - z_d(k+1) & \text{if } i \text{ is the last neuron in HCNN;} \\ \tilde{y}_3(k) - h_g(k) & \text{if } i \text{ is an output neuron in an HG;} \\ 0 & \text{if } k \text{ is not an output neuron either in HCNN or HG.} \end{cases} \tag{14}$$

4.2 Recursive Prediction Using HCNN

To predict a trajectory of arbitrary length, HCNN requires the first state vector $\mathbf{z}(0)$ and the initial conditions of each neuron in the network, that is $\tilde{y}_i(0)$, which were randomly generated at the beginning of training. Each output of the network at step n corresponds to $z_d(n+1)$. This value is stored in a right-shift register that feeds inputs to be used for next prediction (see Fig. 4). Initial values in the right-shift register are the d values of $\mathbf{z}(0)$. Each $\tilde{y}_i\ (n+1)$ is calculated according to Eq. 6; external inputs are:

$$I_i(n) = \begin{cases} \text{right-shift register}(i), & \text{if } i \text{ is an input node;} \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

5 Results

Hybrid-connected Complex Neural Network may be used with any kind of non-linear time series, but, given its architecture, it is better-suited for time series that present a pseudo-cyclic behavior. Here we present the results obtained for long-term prediction of time series generated using a Mackey-Glass function and a filtered real ECG.

The performance of HCNN was measured using:

(1) MSE, as defined in Eq. 10,
(2) A comparison of the magnitude of the maximum local LE of the output signal with the maximum local LE of the training signal. In all cases the maximum LE was calculated using the algorithm proposed by Wolf et al. [29].

Next we present the results obtained with two time series known to be chaotic: Mackey-Glass data and an ECG of a healthy patient.

5.1 Long-Term Prediction of a Mackey-Glass Time Series

The Mackey-Glass equation [6], widely used to model biological rhythms, is defined as:

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx(t) \tag{16}$$

This equation results in chaotic behavior for $a = 0.2$, $b = 0.1$ and $\tau = 17$. Figure 5 shows a normalized numerical solution of the Mackey-Glass equation using these parameters.

After pre-processing this time series as described in Sect. 4, a time lag $T_L = 1$ and an embedded dimension $d = 5$ were identified. An HCNN with seven HGs was trained to model this series using 205 points for training. This HCNN has five input neurons, 21 neurons distributed at the seven HGs, seven hidden neurons and one output neuron. Appendix A shows the connection matrix of all neurons in the network. HCNN was trained for 31,000 epochs; the first 20,000 epochs trained the feed-forward connections and the last 11,000 epochs trained all connections. The training was executed with $\mu = 0.5$, $\eta = 4.8\text{E}{-5}$, $\alpha = 0.3$. The results of HCNN for the first 205 predicted values are shown in Fig. 6, compared to training data. The MSE obtained was 0.0038. The predicted series of the same size obtained by the recursive method (black solid line) had a maximum LE of 0.0374 ± 0.006. This positive LE shows that the HCNN was able to capture the chaotic dynamics found in the training signal, which has a maximum LE of 0.0334 ± 0.003. Notice that maximum LE of training and predicted signals are similar. Figure 7 shows the results of long-term prediction for a horizon of 820 points

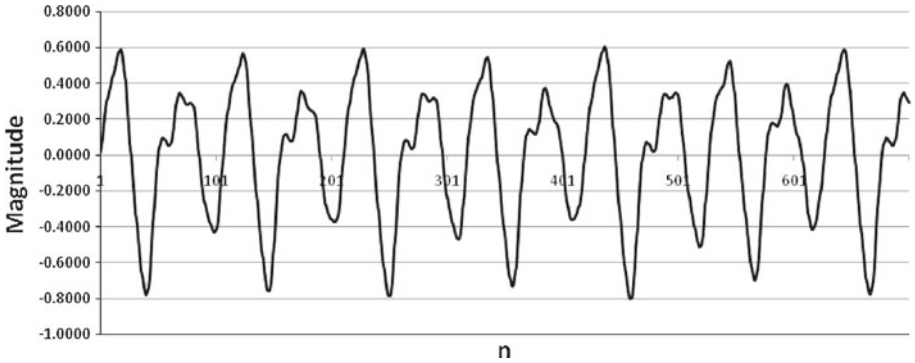

Fig. 5 A solution of Mackey-Glass equation for $a = 0.2$, $b = 0.1$ and $\tau = 17$

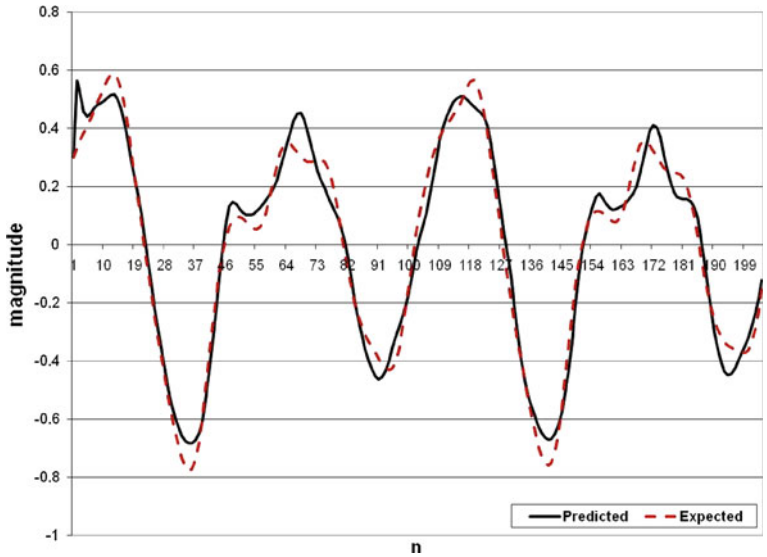

Fig. 6 Results of training an HCNN with 205 points of Mackey-Glass data

(four times the size of training series); The MSE of testing series from point 206 to point 820 was 0.2586 and the local maximum LE was 0.0845 ± 0.005. Figure 8 shows the long-term prediction compared to expected results. Notice that this prediction resembles the expected signal but it is slightly delayed in time, a condition frequently found in long-term prediction.

To compare our results with others, we looked for published research doing long-term prediction, testing it with a Mackey-Glass series and measuring performance with MSE. The closest work that we found was a long-term predictor built by Sollacher and Gao [36]. Their predictor is based on a model called "Spiral Recurrent Neural Networks," which is trained using online learning based on an extended Kalman filter and gradients, similar to Real Time Recurrent Learning. However, they evaluated the performance of their system using

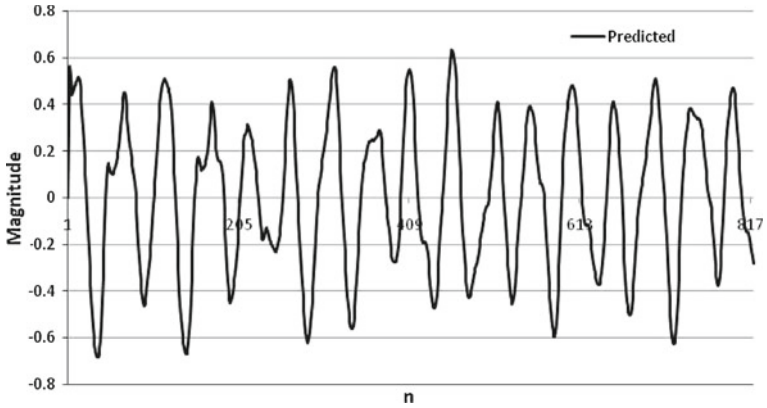

Fig. 7 Long-term prediction (820 points) obtained by an HCNN for Mackey-Glass series

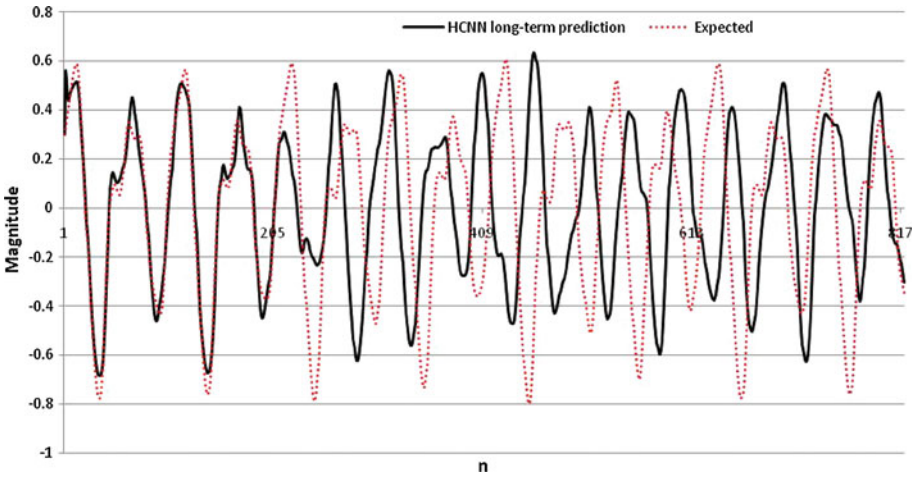

Fig. 8 HCNN 820 points of recursive long-term prediction versus original Mackey-Glass series

the "logarithmic normalized MSE" metric, and they measured the error only in the time steps when the spikes occurred, which makes difficult to compare their results with ours. On the other hand, García-Treviño and Alarcón-Aquino [37] tested their predictor using the same performance metric and same time series as we did, but they reported results only with single-step prediction. Their predictor uses a neural network with wavelets as activation functions in the hidden layer, trained with a type of back-propagation algorithm. They also compared the performance of their network with single step prediction obtained using a feed-forward network. Both networks were trained using 100 points and tested using 100 single-step predictions. Wavelet network obtained an MSE of 0.0008 and the feed-forward network obtained a MSE of 0.0359. Given the fact that this is single-step prediction, it is expected that the MSE obtained by their systems was smaller than the MSE obtained by our recursive prediction.

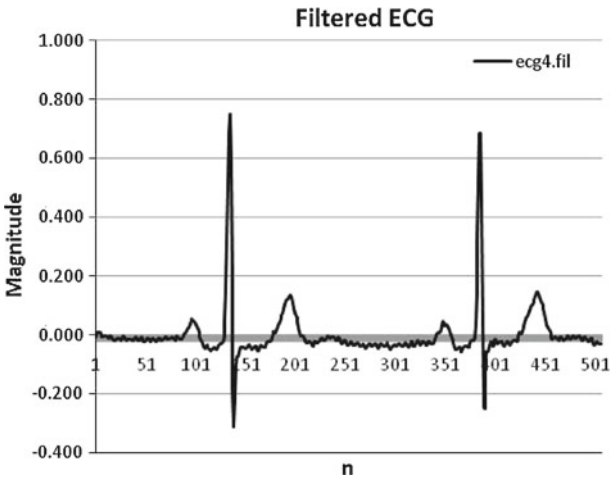

Fig. 9 A filtered ECG signal

5.2 Modeling and Predicting the Dynamics of an ECG

An ECG is a time series believed to be chaotic, because it presents a maximum positive LE, strange attractors in its phase or return maps [38] and other important characteristics [39]. Currently, several research projects are looking for modeling and predicting the behavior of an ECG, because several cardiac diseases could be promptly treated if they were predicted [40,41]. For example, ventricular fibrillation or ventricular tachycardia produce arrhythmias that may lead to death [42], but the identification of patients in risk remains a challenge [43]. Therefore, the construction of an accurate model of an ECG is still an open problem. Here, we used an HCNN to capture the dynamics embedded in an ECG, even though its exact reproduction was not achieved. The ECG signal of a healthy patient, obtained from [44] was filtered with an FIR filter of order 40, with cutoff frequencies lying at 0.5 and 105 Hz. This upper limit was chosen after observing that in the original signal the magnitude of frequencies above 100 Hz was less than 0.5. Figure 9 shows 507 points of the filtered ECG; the fundamental frequency of this signal is 1.4062 Hz and it presents a positive maximum local LE of 3.23 ± 0.27 (an evidence of chaos).

An HCNN, with the same architecture as the one described in Sect. 5.1 and in Appendix A, was used for modeling this ECG. Seven HG were trained to produce the first seven sine harmonic components of the filtered ECG. After that, the HCNN was trained for 30,000 epochs; the first 20,000 epochs trained the feed-forward connections and the last 10,000 trained all feed-forward and recurrent connections. The training was executed using $\mu = 0.3$, $\eta = 3.7\text{E}{-3}$, $\alpha = 0.1$. An MSE of 0.0028 was reached for the first 507 predictions, the training segment (see Fig. 10). This prediction has a maximum local LE $= 4.47 \pm 0.33$. Figure 11 shows 2,028 points of long-term prediction of the ECG (4 times the size of training series). This resulting series has chaotic characteristics, with a maximum local LE of 7.52 ± 1.95. Notice that, even though the predicted series does not contain the right magnitudes in the peaks of the ECG, it contains "peaks" resembling the R and T peak of a typical ECG. Figure 12 compares the original training signals with the long-term prediction.

A recursive predictor based on a fully connected, feed-forward network was built to compare these results. The feed-forward network contained five inputs, 10 hidden neurons and

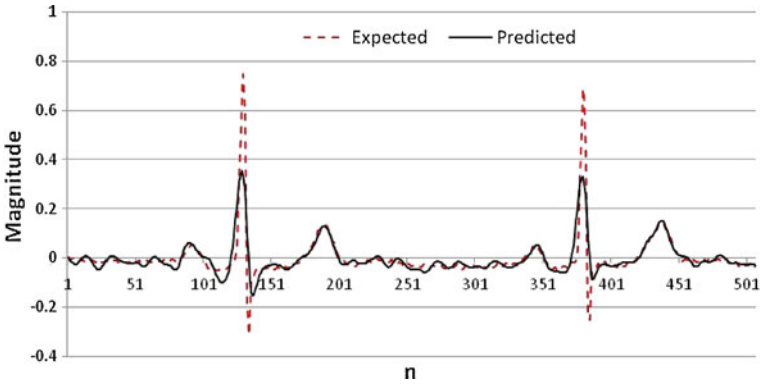

Fig. 10 Results of training an HCNN with 507 points of ECG data

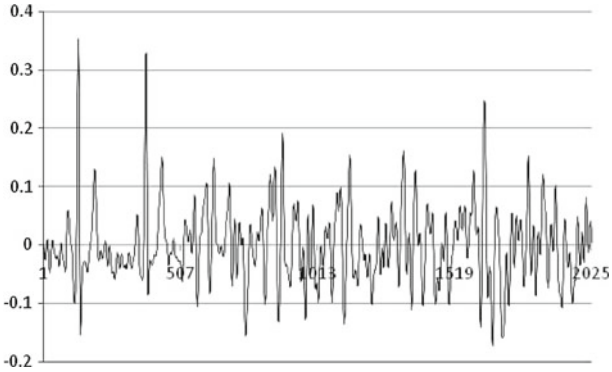

Fig. 11 Long-term prediction (2,028 points) obtained by an HCNN for ECG data

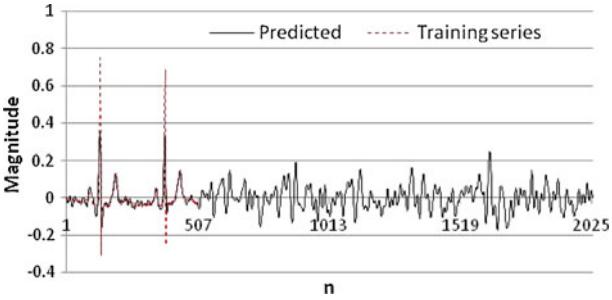

Fig. 12 Training series (507 points) and long-term prediction (2,028 points) obtained by an HCNN for ECG data

one output neuron and was trained using 470 points of a filtered ECG. This feed-forward network predicted well the training segment, as shown at Fig. 13. The MSE obtained for this segment was 0.0009. However, this network was unable to carry out the recursive prediction. Figure 14 shows the predictions obtained from points 470 to 570. For that prediction,

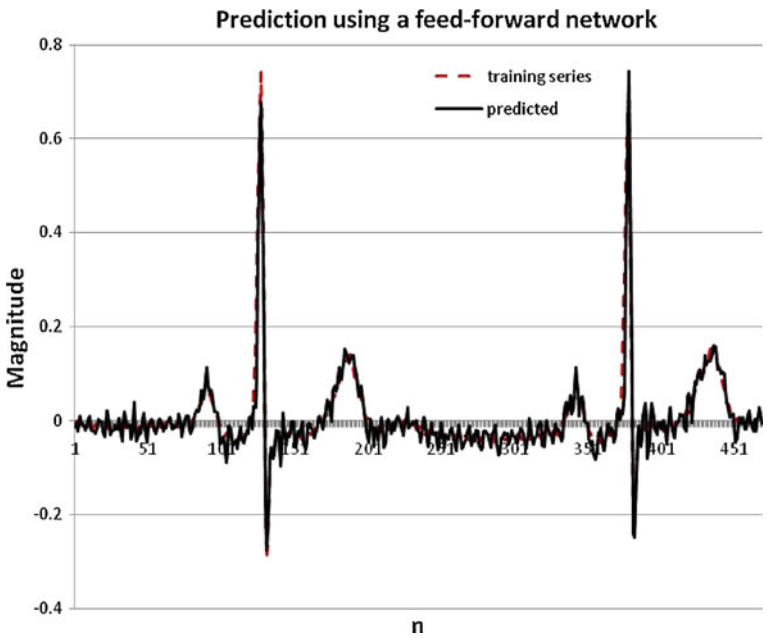

Fig. 13 Predicting the training series (point 1–470) using a feed-forward network

the MSE was 113.1381. The prediction is so distorted in magnitude that the training signal cannot be distinguished in the figure.

6 Conclusions

In this paper we presented a novel neural network architecture known as "HCNN" and its training scheme. HCNN is able to capture the dynamics embedded in highly non-linear time series and to perform some long-term prediction of chaotic time series. This architecture is composed of small fully connected neural networks, embedded in a feed-forward system with some extra recurrent connections. The small networks, called HGs, are trained to generate sine signals oscillating to frequencies that are multiples of fundamental frequency of training time series. The feed-forward part of HCNN is able to learn a function with similar maximum local LE as the one found in the training series. The HGs in HCNN contribute with some information related to frequencies; the recurrent connections in the hidden layer contribute memorizing dynamic characteristics of the past; feed-forward connections going from the input to the hidden layer allow the definition of a function approximator.

In the experiments reported here, an HCNN was used to learn a Mackey-Glass time series and an ECG to predict a horizon with a size 4 times the training signals. In both cases, chaotic signals were produced, and HCNN was able to approximate these long horizons using recursive prediction, oscillating in a stable way. However, the network was not able to generate right magnitudes in the peaks of the ECG. A reason for that could be the predictor is not

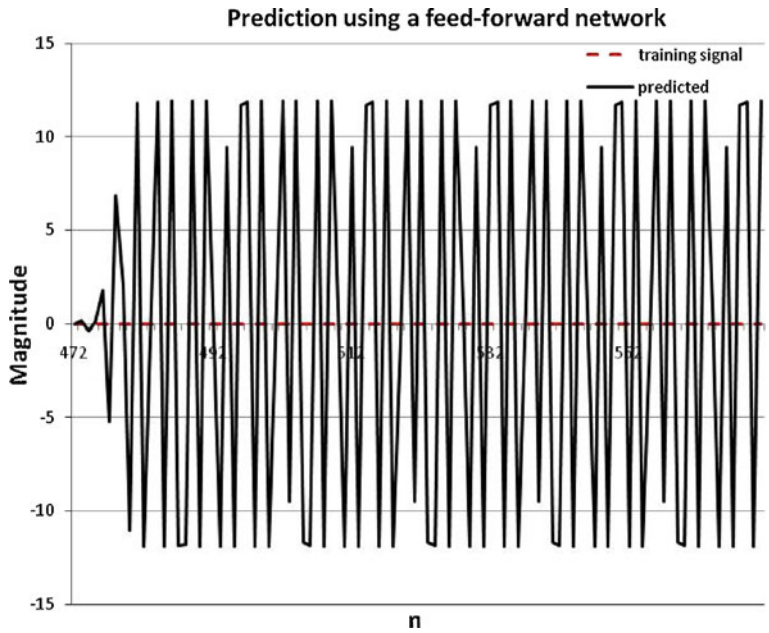

Fig. 14 Recursive Prediction from point 470 to 570 using a feed-forward network

detecting information in high frequencies. As it is well known, peaks are the most difficult part to be learnt by a predictor.

Currently, we are designing other architectures similar to the HCNN, using wavelet theory instead of Fourier transform, in order to include time-frequency information in the model [45]. Future improvements to our predictor would include faster training algorithms and the use of ensemble forecasts, which has reported good results in this kind of problems [46]. Each ensemble could be built using several HCNNs, trained with different samplings from the same dynamical system but different initial conditions.

Acknowledgments P. Gómez-Gil thanks Dr. W. Brian Oldham, from the Computer Science department at Texas Tech University, for his invaluable contributions and direction during the development of the early parts of this research. Part of this research was supported by the National Council of Science and Technology in Mexico.

Appendix A

Table A1 shows the connection matrix of the HCNN topology, used for the experiments reported in this paper and in [22]. Each row and column represents a neuron in the network. Neurons are numbered in sequence, starting at the input layer. Neurons 1–5 correspond to input neurons; 6–26 belong to HGs; neurons 27–33 belong to the hidden layer; neuron 34 corresponds to the output of the network. A value 1 in a cell in row i and column j indicates a connection going from neuron i to neuron j. Bold lines separate each layer in the network, starting with the input layer, the HG layer, the hidden layer and the output layer.

Table A1 Connection matrix of HCNN used for experiments reported in this paper [22]

	INPUT NEURONS					HARMONIC GENERATOR'S NEURONS																			HIDDEN NEURONS							OUTPUT N.
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
6	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
9	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
12	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	1	1	1	1	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

References

1. Mees AI (ed) (2000) Nonlinear dynamics and statistics. Birkhäuser, Boston
2. Galván IM, Isasi P (2001) Multi-step learning rule for recurrent neural models: an application to time series forecasting. Neural Process Lett 13:115–133
3. Kinsner W (2006) Characterizing chaos through Lyapunov metrics. IEEE Trans Syst Man Cybernet C 36(2):141–151
4. Strogatz SH (1994) Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering. Addison-Wesley, New York
5. Principe JC, Kuo JM (1995) Dynamic modeling of chaotic time series with neural networks. In: Tesauro G, Touretzky DS, Leen TK (eds) Advances in neural information processing systems 7. Morgan Koufmann, San Mateo
6. Glass L, Mackey M (1988) From clocks to chaos. Princeton University, Princeton
7. Duhau S (2003) An early prediction of maximum sunspot number in solar cycle 24. Solar Phys 213: 203–212
8. Júnior MP-J, Barreto GA (2008) Long-term time series prediction with the NARX network: an empirical evaluation. Neurocomputing 71(16-18):3335–3343
9. Abarbanel, Henry DI (2000) Challenges in modeling nonlinear systems: a worked example. In: Mees AI (ed) Nonlinear dynamics and statistics. Birkhauser, Boston
10. Judd K, Small M (2000) Towards long-term prediction. Physica D 136:31–44
11. Granger CJW (2005) Preface: somethoughts on the future of forecasting. Oxford Bull Econ Stat 67(supplement(0305-9049)):707–711
12. Mandic D, Chambers J (2001) Recurrent neural networks for prediction. Wiley, Chichester
13. Beliaev I, Kozma R (2007) Time series prediction using chaotic neural networks on the CATS benchmark. Neurocomputing 70:2246–2439

14. Lendasse A, Oja E, Simula O, Verleysen M (2004) Time series prediction competition: the CATS Benchmark. In: Proceedings of IEEE IJCNN, pp 1615–1620
15. Leontaritis J, Billings SA (1985) Input–output parametric models for nonlinear systems—part I: deterministic nonlinear systems. Int J Control 41(2):303–328
16. Principe JC, Euliano NR, Lefebvre WC (2000) Neural adaptive systems: fundamentals through simulations. Wiley, Chichester
17. Elman JL (1990) Finding structure in time. Cogn Sci 14:179–211
18. Park DC, Tran CN, Lee Y (2006) Multiscale bilinear recurrent neural networks and their application to the long-term prediction of network traffic. In: LNCS 3973, pp 196–201
19. Cai X, Zhang N, Venayagamoorthy GK, Wunsch DC II (2004) Time series prediction with recurrent neural networks using a hybrid PSO-EA algorithm. In: Proceedings of IEEE IJCNN, pp 1647–1652. doi:10.1109/IJCNN.2004.1380208
20. Alarcon-Aquino V, Barria JA (2006) Multiresolution FIR neural-network-based learning algorithm applied to network traffic prediction. IEEE Trans Syst Man Cybernet C36(2): 208–220. doi:10.1109/TSMCC.2004.843217
21. Sapankevych NI, Sankar R (2009) Time series prediction using support vector machines: a survey. IEEE Comput Intell Mag 4(2):24–38
22. Gomez-Gil P (1998) The effect of non-linear dynamic invariant in the recurrent neural networks for prediction of electrocardiograms. Dissertation in Computer Science, Texas Tech University, Lubbock, TX, USA
23. Werbos P (1990) Backpropagation through time: what it does and how to do it. Proc IEEE 74(19):1550–1560
24. Pearlmutter B (1990) Dynamic recurrent neural networks. Technical Report CMU-CS-90-196. School of Computer Science, Carnegie Mellon University, Pittsburgh, MA
25. Takens F (1981) Detecting strange attractors in turbulence. Springer, Berlin, pp 366–381
26. Abarbanel HDI, Brown R, Kadtke J (1990) Prediction in chaotic nonlinear systems: methods for time series with broad band Fourier spectra. Phys Rev A 41:1782–1807
27. Parker TS, Chua LO (1989) Practical numerical algorithms for chaotic systems. Springer, New York
28. Gencay R, Dechert WD (1992) An algorithm for the n Lyapunov exponents of an n-dimensional unknown dynamical system. Physica D 59:142–157
29. Wolf A, Swift JB, Swinney HL, Vastano JA (1985) Determining Lyapunov exponents from a time series. Physica D 16:285–317
30. Sano J, Swada Y (1985) Measurement of the Lyapunov spectrum from a chaotic time series. Phys Rev Lett 55:1082–1085
31. Rosestein M, Collins J, Deluca C (1993) A practical method for calculating largest Lyapunov exponents for small data set. Physica D 65:117–134
32. Darbyshire AG, Broomhead DS (1996) Robust estimation of tangent maps and Lyapunov spectra. Physica D 89:287–305
33. Gómez-Gil P, Oldham WB (1999) On the use of hybrid neural networks and non linear invariants for prediction of electrocardiograms. In: Proceedings of IJCNN 99, IEEE, pp 3661–3664. doi:10.1109/IJCNN.1999.836264
34. Logar A (1992) Recurrent neural networks and time series prediction. PhD Dissertation in Computer Science, Texas Tech University, Lubbock, TX
35. Werbos P (1994) The roots of backpropagation from ordered derivatives to neural networks and political forecasting. Wiley-Interscience Publication, New York
36. Sollacher R, Gao H (2008) Efficient online learning with spiral recurrent neural networks. In: IJCNN (2008), IEEE, pp 2551–2558. doi:10.1109/IJCNN.2008.4634155
37. García-Treviño ES, Alarcón-Aquino V (2006) Single-step prediction of chaotic time series using wavelet-networks. In: Electronics, robotics and automotive mechanics conference CERMA 2006 (Cuernavaca 2006), IEEE, pp 243–248
38. Glass L, Hunter P, McCulloch A (1991) Theory of heart. Springer, New York
39. Hongxuan Z, Yisheng Z, Yuhong X (2002) Complexity information based analysis of pathological ECG rhythm for ventricular tachycardia and ventricular fibrillation. Int J Bifurc Chaos 12(10):2293–2303
40. Forberg JL, Green M, Bjork J, Ohlsson M, Edenbrandt L, Ohlin H, Ekelund U (2009) Search of the best method to predict acute coronary syndrome using only the electrocardiogram from the emergency department. J Electrocardiol 42(1):58–63
41. Shanthi D, Sahoo G, Saravanan N (2009) Designing an artificial neural network model for the prediction of thrombo-embolic stroke. Int J Biom Bioinformatics 3(1):10–18

42. Lerma C, Wassel N, Schirdewan A, Kurths J, Glass L (2008) Ventricular arrhythmias and changes in heart rate preceding ventricular tachycardia in patients with an implantable cardioverter defibrillator. Med Biol Eng Comput 46: 715–727. doi:10.1007/s11517-008-0326-y
43. Al-Khatib SM, Sanders GB, Bigger JT et al (2007) Preventing tomorrow's sudden cardiac death today: part I: current data on risk stratification for sudden cardiac death. Am Heart J 153:941–950
44. Harvard-MIT Division of Health Sciences Technology (1992) The MIT_BIH Arrhythmia Database CD-ROM, 2nd edn. Biomedical Engineering Center, Cambridge, MA
45. García-Pedrero A, Gómez-Gil P (2010) Time series forecasting using recurrent neural network and wavelet reconstructed signals. In: Proceedings of 20th international conference on Electronics, communications and computers (Puebla, Mexico 2010), IEEE, pp 169–173. doi:10.1109/CONIELECOMP.2010.5440775
46. Smith LA (2000) Disentangling uncertainly and error: on the predictability of nonlinear systems. In: Mees AI (ed) Nonlinear dynamics and statics. Birkhäuser, Boston