Neural Process Lett (2017) 45:243–261
DOI 10.1007/s11063-016-9525-y

On Chaotic Neural Network Design: A New Framework

Ke Qin[1]

Published online: 27 April 2016

Abstract The theory of chaos and chaotic neural networks (CNNs) has been widely investigated in the past two decades. However, most researchers in this area have merely focused on how to make full use of CNNs to solve various problems in areas such as pattern recognition, classification, associate memory and cryptography. The philosophy of *how* to design a CNN is seldom discussed. In this paper, we present a general methodology for designing CNNs. By appropriately choosing a self-feedback mechanism, and also including coupling functions and an external stimulus, we have succeeded in proving that a dynamical system, defined by discrete time feedback equations, does, indeed, possess interesting chaotic properties. To the best of our knowledge, the results presented here are novel and pioneering.

Keywords Chaotic neural networks · Chaos · Nonlinear dynamics · Self-feedback · Recurrent neural networks

1 Introduction

At a very fundamental and philosophical level, the field of neural networks (NNs) deals with understanding the brain as an information processing machine. Conversely, from a computational perspective, it concerns utilizing the knowledge of the (human) brain to build more intelligent computational models and computer systems. Thus, in the last few decades, NNs have been widely and successfully applied to an ensemble of information processing problems, and the areas of Pattern Recognition (PR) nd forecasting are rather primary application domains. For example, the authors of [11] have applied the classical Back Propagation NN for the segmentation of Meteosat images. Similarly, the literature [9] has also reported an alternative approach to the class prediction of protein structures by means of NNs. These authors have stated that their proposed scheme maintains the same level of classification

✉ Ke Qin
qinkesci@hotmail.com

[1] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, Sichuan, China

accuracy as its competition, although it is achieved with minimum computational time. In [17], Oliver and his co-authors verified the effectiveness of an adaptation of the NN in the design and development of solutions for some urban network problems.

Since classic NNs have attracted so much attention, many researchers have proposed various improved or novel NN models, such as celluar NNs [7,8], spiking NNs [12], random NNs [27], chaotic NNs etc. In this paper, we concentrate on the philosophy and paradigm of *chaotic* neural networks (CNNs).

It is an accepted fact that a well-defined artificial neural systems could display three typical properties: convergence, oscillation and chaos. Indeed, Freeman's clinical work has clearly demonstrated that the brain, at the individual neural level and at the global level, possesses chaotic properties. He demonstrated that the quiescent state of the brain is chaos. However, during perception, when attention is focused on any sensory stimulus, the brain activity becomes more periodic [10]. Thus, as applied scientists, if we are able to develop a system which mimics the brain, it could lead to a new model of NNs—which is the motivation with designing and utilizing CNNs.

The theory of CNNs has been extensively studied in the last two decades since Aihara and Adachi et al. proposed the first CNN model [1,2] (named the AdNN in our previous papers [19,21,22]). The AdNN was, actually, based on the modeling of the giant axons of squids. CNN models with biological plausibility have been widely used in various fields such as multi-value content addressing, optimization, image segmentation, information retrieval and data encryption [16,23,29]. Motivated by the work of Aihara, Adachi et al., various types of CNNs have been proposed to solve a number of optimization problems (such as the Traveling Salesman Problem, (TSP)), or to obtain Associative Memory (AM) and/or PR properties.

An interesting step in this regard was the work reported in [28], where the authors utilized the delayed feedback and the Ikeda map to design a CNN to mimic the biological phenomena observed by Freeman [10]. Later, Hiura and Tanaka investigated several CNNs based on a piecewise sine map or the Duffing's equation to solve the TSP [13,25,26]. Another valuable work related to chaos and NNs was reported in [3]. The authors of [3] applied CNNs to the family of the original Bidirectional Associative Memory (BAM) NNs, and also designed the Chaotic BAM (C-BAM) models. Their work demonstrated that the C-BAM family can access patterns that members of the original BAM family were incapable of accessing.

The primary aim of this paper is to give a general framework for the design of CNNs. By appropriately choosing self-feedback, coupling functions and external stimulus, we are able to drive a dynamical system, defined by discrete time feedback equations, to interesting chaotic trajectories. Our general framework has the same topological structure (completely connected) as the CNNs listed above. It is characterized by the definition of a recurrent NN described in terms of a Present-State/Next-State function and a State/Output function. The sigmoid function is still used as the transfer function. But more importantly, the work presented in this article is a prelude to a novel strategy for the design of CNNs. Essentially, the chaotic feedback module of this newly proposed framework can be easily modified and then substituted for by other conditional functions. Simultaneously, the network can be controlled to possess analogous properties—as long as the respective parameters are appropriately tuned.

2 State of the Art

A classical NN can be characterized by five elements: the dynamics of the individual neuron, the network's topology, the learning rule used, the input and the output. In this paper, we mainly consider a typical NN with chaotic and recurrent characteristics.

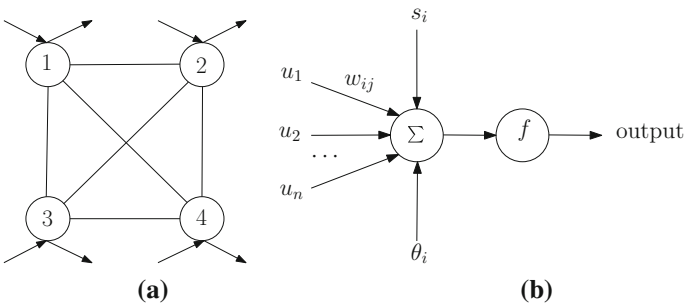

Fig. 1 **a** The structure of a recurrent NN with four neurons. **b** The typical structure of an individual neuron

A recurrent NN usually has a structure as showed in Fig. 1; it is usually fully connected. Each neuron has an external input s_i and an output u_i. Each neuron also receives outputs from other neurons. The summation of external input, the threshold value and weighted outputs from other neurons is called the "net input". The output of a neuron is given by a transform function, which is usually a sigmoid or a hard-limited function. The essential difference between a classical NN and a chaotic NN is that a CNN's neuron displays obvious chaotic properties, whereas a classical NN does not.

2.1 The Adachi Neural Network (AdNN) and its variants

The AdNN is a network of neurons with weights associated with the edges, a well-defined Present-State/Next-State function, and a well-defined State/Output function. It is composed of N neurons which are topologically arranged as a completely connected graph, as shown in Fig. 1a. A neuron, identified by the index i, is characterized by two internal states, $\eta_i(t)$ and $\xi_i(t)$ $(i = 1...N)$ at time t, whose values are defined by Eqs. (1) and (2) respectively. The output of the ith neuron, $u_i(t)$, is given by Eq. (3), which specifies the so-called sigmoid function.

$$\eta_i(t+1) = k_f \eta_i(t) + \sum_{j=1}^{N} w_{ij} u_j(t), \tag{1}$$

$$\xi_i(t+1) = k_r \xi_i(t) - \alpha u_i(t) + a_i. \tag{2}$$

$$u_i(t+1) = f(\eta_i(t+1) + \xi_i(t+1)). \tag{3}$$

As per the above definitions and illustration, the structure of a single AdNN's neuron can be described by Fig. 2. The reader must observe that it is almost the same as the one in Fig. 1b except that there is one more nonlinear unit, which leads to the chaotic phenomena. The reader should also note a fundamental difference between Figs. 2 and 1b which is that the former has a self-feedback connection which the latter does not have.

The AdNN uses the Hebbian rule to determine the weights of the connections. Under certain settings, the AdNN can behave as a dynamic AM. It can dynamically recall all the memorized patterns as a consequence of an input which serves as a "trigger". Further, if the external stimulations correspond to trained patterns, the AdNN can behave like a PR system.

By invoking an analysis based on Lyapunov Exponents (LEs), one can show that the AdNN has $2N$ *negative* LEs. In order to obtain positive LEs, Calitoiu et al. [4] proposed a model of CNNs which modifies the AdNN to enhance its PR capabilities. The most significant difference is the updating of the values of the internal state(s). Unlike the AdNN, which

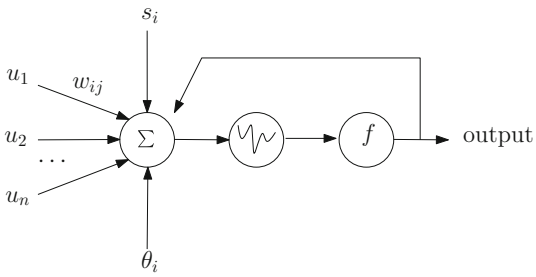

Fig. 2 The structure of an AdNN's single neuron

incorporates all the internal states to achieve the dynamical behavior, the M-AdNN uses two global internal states which are both associated with a *single* neuron, for example, the N^{th} neuron. Thus, Eqs. (1) and (2) get modified to become Eqs. (4) and (5) respectively:

$$\eta_i(t+1) = k_f \eta_N(t) + \sum_{j=1}^{N} w_{ij} u_j(t), \tag{4}$$

$$\xi_i(t+1) = k_r \xi_N(t) - \alpha u_i(t) + a_i. \tag{5}$$

By resorting to this modification, the M-AdNN has two *positive* LEs, namely: $\lambda_N = \ln k_f + \frac{1}{2} \ln N$ and $\lambda_{2N} = \ln k_r + \frac{1}{2} \ln N$, which renders the M-AdNN to be truly chaotic.

Calitoiu and his co-authors also proposed a new approach for modeling the problem of blurring or inaccurate perception, and demonstrated that the quality of a system can be modified without manipulating the quality of the stimulus. This new model has been termed the Mb-AdNN [5]. As opposed to the M-AdNN, where the updates are based on two global states, the updates in the Mb-AdNN are based on the states of the first m neurons. In the interest of brevity, the details of the Mb-AdNN are omitted here.

2.2 Our Previous Work

More recently, in our previous paper [18], we presented a collection of previously unreported properties of the AdNN. We have shown that it goes through a spectrum of characteristics as one of its crucial parameters, α, changes. As α increases, it is first an AM, and it then becomes *quasi*-chaotic. The system is subsequently distinguished by two phases which really do not have clear boundaries of demarcation, where in the former it is *quasi*-chaotic for some patterns, and periodic for others, and in the latter, it exhibits PR properties. It is fascinating that the AdNN also possesses the capability to recognize masked or occluded patterns, and even patterns that are completely inverted.

Later, we investigated the problem of reducing the computational cost of the AdNN and its variants. Because their structures involve a completely connected graph, the computational complexity of the AdNN (and its variants) is quadratic in the number of neurons. In [20], we considered how the computations can be significantly reduced by merely using a linear number of inter-neuron connections. To achieve this, we extracted from the original completely connected graph, *one* of its spanning trees, and then computed the best weights for *this* spanning tree by using a gradient-based algorithm. By a detailed experimental analysis, we showed that the new linear-time AdNN-like network possesses chaotic and PR properties for different settings.

2.3 Overview of Other Chaotic Neural Networks

2.3.1 A Duffing's Equation Based CNN

The CNN based on Duffing's equation (referred to as the Du-CNN in this paper) was initially proposed in [13]. This model can be summarized as following: For a single neuron, the internal state is determined from the variable $x(t)$ of the Duffing's equation, which is defined by:

$$\begin{cases} \frac{dx}{dt} = y \\ \frac{dy}{dt} = \varepsilon - \alpha y + \beta x - \gamma x^3 + f\cos z \\ \frac{dz}{dt} = \omega, \end{cases} \tag{6}$$

where the constant α is a damping coefficient, β and γ are coefficients of the "double well" potential, f and ω are an amplitude and a frequency of a periodic driving force, respectively. ε is a gradient parameter of a driving extended field. It describes a dynamical system that exhibits chaotic behavior. In the Du-CNN, all the neurons are completely connected. The total net input of the i^{th} neuron at time t is given by:

$$I_i(t) = \sum_j w_{ij} u_j(t) + s_i - \theta_i, \tag{7}$$

where w_{ij} is the coupling strength between i^{th} neuron and j^{th} neuron. s_i and θ_i are an external input and a threshold value of i^{th} neuron, respectively. $u_j(t)$ is the output of the j^{th} neuron. The final output of the i^{th} neuron is dictated by a sigmoid function:

$$u_i = f(x_i) = \frac{1}{1 + e^{-x_i/T}}, \tag{8}$$

where T is a temperature-like parameter used to control the uncertainty associated with the firing of the neuron. The reader must note that the output at the next time instant depends significantly on the previous net input. This is achieved by controling the parameter ε in Eq. (6):

$$\varepsilon = \lambda \arctan(I_i(t)). \tag{9}$$

As a result, the structure of the neuron in this NN is slightly different from the AdNN's, as shown in Fig. 3. Compared to Fig. 2, there is one more control function T, which is used to control ε. The nonlinear unit is defined by the Duffing's equation, Eq. (6).

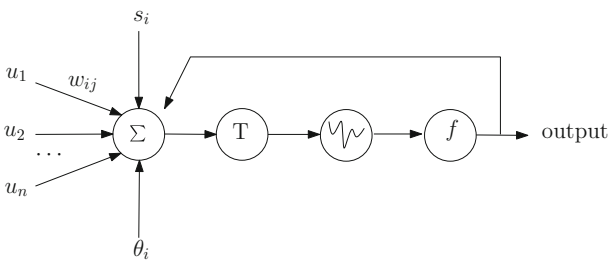

Fig. 3 The structure of a Du-CNN's single neuron

2.3.2 A PWSM Based CNN

Essentially, the CNN based on the Piece-Wise Sin Map (PWSM) is very similar to the CNN illustrated in Sect. 2.3.1. It is also a network with a fully-connected structure. The internal state of a single neuron is determined from the variable $x(t)$ of a PWSM:

$$x_i(t) = g_i(x_i(t)), \tag{10}$$

where the function $g(\cdot)$ is called the Piece-Wise Sin Map defined by:

$$g_i(x) = \begin{cases} g_i^-(x) = \frac{1}{2}\sin[2\{\pi + \sin^{-1}(2\varepsilon_i^-)\}x] & (-0.5 \le x < 0), \\ g_i^+(x) = \frac{1}{2}\sin[2\{\pi + \sin^{-1}(2\varepsilon_i^+)\}x] & (0 \le x \le 0.5), \end{cases} \tag{11}$$

where $\varepsilon_i^{\pm}$ are the values of $|g_i^{\pm}(x)|$ as $x_i = \pm 0.5$. Both ε_i^+ and ε_i^- are positive and satisfy $\varepsilon_i^+ + \varepsilon_i^- = 0.5$. t is a discrete time index with respect to the time evolution of the map. Just as in the case of the Du-CNN, the net input of neuron i is given by Eq. (7). The impact of the net input on the network is also, similar to the Du-CNN, achieved via controlling the parameter ε defined by:

$$\varepsilon_i^{\pm}(t) = \frac{0.25}{1 + \eta I_i^2(t)},$$
$$\varepsilon_i^{\mp}(t) = 0.5 - \varepsilon_i^{\pm}(t).$$

Finally, the output of each neuron is given by a step function:

$$u_i = f(x_i) = \begin{cases} 1, & 0 \le x_i(n) \le 0.5 \\ 0, & -0.5 \le x_i(n) < 0 \end{cases} \tag{12}$$

Obviously, this PWSM-based CNN has exactly the same structure as the Du-CNN. Furthermore, both of these CNNs also use the Hebbian rule to determine the connection weights, as reported in [13,25,26]. In the interest of brevity, further details of this CNN are omitted here.

2.3.3 Time Delayed Differential Equation Based CNNs

Delayed CNNs have been widely investigated in the past decades. They are also a kind of Hopfield-like NN which exhibits rich nonlinear dynamics. A time delayed differential equation-based CNN usually has a general form:

$$\frac{dx_i(t)}{dt} = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f(x_j(t)) + \sum_{j=1}^{n} b_{ij} f(x_j(t - \tau_{ij}(t))) + I_i(t), \tag{13}$$

where n denotes the number of units in the CNN, $x(t) = \{x_1(t), \ldots, x_n(t)\} \in R_n$ is the state vector associated with the neurons, $I = \{I_1, I_2, \ldots, I_n\} \in R_n$ is the external input vector, $f(x(t)) = \{f_1(x_1(t)), f_2(x_2(t)), \ldots, f_n(x_n(t))\} \in R_n$ corresponds to the activation functions of the neurons, $\tau(t) = \tau_{ij}(t)(i, j = 1, 2, \ldots, n)$ are the time delays. $C = diag(c_1, c_2, \ldots, c_n)$ is a diagonal matrix, $A = (a_{ij})_{n\times n}$ and $B = (b_{ij})_{n\times n}$ are the connection weight matrix and the delayed connection weight matrix, respectively. The dynamics of Eq. (13) have been well studied and it has been reported that it can exhibit rich chaotic

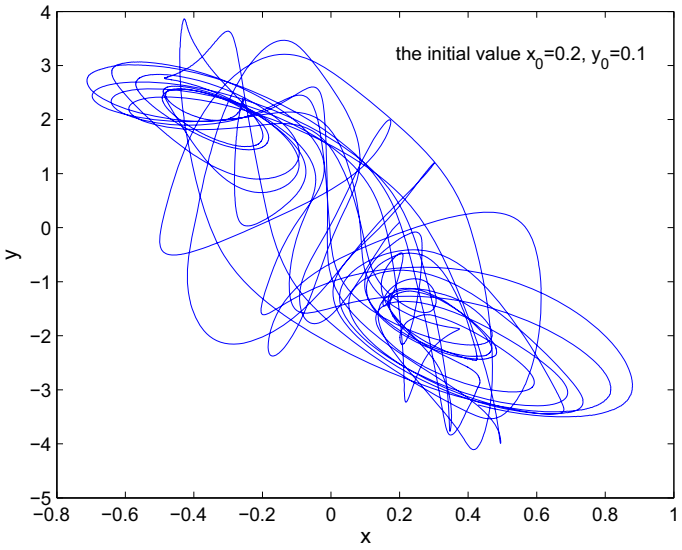

Fig. 4 The trajectories of Eq. (13). In this figure, the values of $x(t)$ and $y(t)$ are calculated by means of the fourth-order Runge-Kutta method. The time span is from 0 to 200 with a total of 30,000 steps

phenomena, for example, if the parameters are: $A = \begin{pmatrix} 2.0 & -0.1 \\ -5.0 & 3.0 \end{pmatrix}$, $B = \begin{pmatrix} -1.5 & -0.1 \\ -0.5 & -2.5 \end{pmatrix}$, $C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

Further, if $f_i(x_i(t)) = tanh(x_i(t))$, $\tau(t) = 1 + 0.1sin(t)$, and $I = 0$, the trajectories of Eq. (13) are shown in Fig. 4, which is an apparent chaotic trajectory.

Summary As we can see from the above discussions, the key point in designing a CNN lies in the manner in which we build a neuron that could possess chaotic properties. Apart from the above mentioned CNNs, other NNs have also been reported (e.g., the models analyzed in [15,24]) using a common method that directly choose a variable of the chaotic equations to indicate a neuron's internal state. In every case, all the neurons are fully inter-connected. Also, the connection weights are determined by a given learning rule, e.g., the Hebbian rule. It has been reported that such models can be well trained to do different tasks such as solving the TSP, PR, AM and so on.

In this paper, we intend to propose a novel and universal way to design CNNs.

3 Preliminaries

A discrete time Present-state/Next-state recurrent NN with n neurons can be described by:

$$\boldsymbol{x}(t+1) = \boldsymbol{f}(\boldsymbol{x}(t)) + \boldsymbol{g}(\boldsymbol{x}(t)) + \boldsymbol{h}(\boldsymbol{I}(t)), \tag{14}$$

where $\boldsymbol{x} \in \mathcal{R}^n$, $\boldsymbol{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$ is the network state vector at time step t. $\boldsymbol{f}$, $\boldsymbol{g}$ and $\boldsymbol{h}$ are continuous differentiable functions as follows:

1. $\boldsymbol{f}(\boldsymbol{x}) = [f_1(x_1), f_2(x_2), \ldots, f_n(x_n)]^T$ is a self-feedback function;
2. $\boldsymbol{g}(\boldsymbol{x}) = [g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_n(\boldsymbol{x})]^T$ is a coupling function;
3. $\boldsymbol{h}(\boldsymbol{I}(t)) = [h_1(I_1(t)), h_2(I_2(t)), \ldots, h_n(I_n(t))]^T$ is an extra stimulus transfer function.

$\boldsymbol{I}(t) = [I_1(t), I_2(t), \ldots, I_n(t)]^T$ is an extra stimulation.

To simplify the model, we let $f_i(\cdot) = f_j(\cdot)$, $g_i(\cdot) = g_j(\cdot)$ and $h_i(\cdot) = h_j(\cdot)$ for every pair, i and j.

The model given by Eq. (14) characterizes most of discrete recurrent NNs such as the discrete Hopfield network, a single neuron of which is typically described by:

$$x_i(t+1) = kx_i(t) + \alpha \sum_{j=1}^{n} w_{ij} y_j(t) + I_i, \tag{15}$$

where k is a constant refractory factor, and $y_i(t)$ is an output function of the state $x_i(t)$ at time t. By comparing Eqs. (14) and (15), we can observe that the Hopfield model is a very special case of our "universal" model.

Before we proceed, we state the following definition and proposition, so that the claims that follow can be justified.

Definition 1 An $n \times n$ real matrix A is positive definite (denoted as $A > 0$) if $Z^T AZ > 0$ for all non-zero column vectors Z with real entries ($Z \in \mathcal{R}^n$), where Z^T denotes the transpose of Z.

Proposition 1 *For any given symmetric diagonal dominant matrix $A = [a_{ij}]$, $a_{ij} = a_{ji}$, A is positive definite if $a_{ii} > 0$ for all $i = 1, 2, \ldots, n$.*

Proof This can be proven easily by the definition of positive definiteness. If $A = [a_{ij}]$ with $a_{ii} > 0$, A is diagonal dominant if $a_{ii} > \sum_{j=1, j\neq i}^{n} |a_{ij}|$ for all $i, j = 1, 2, \ldots, n$. Now consider $Z = [z_1, z_2, \ldots, z_n]^T$, with $Z \neq 0$. Then:

$$\begin{aligned} Z^T AZ &= \sum_{i=1}^{n} \left(z_i \sum_{j=1}^{n} a_{ji} z_j \right) = \sum_{i=1}^{n} \sum_{j=1}^{n} z_i z_j a_{ij} \\ &= \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} z_i z_j a_{ij} + \sum_{i=1}^{n} z_i^2 a_{ii}. \end{aligned}$$

Thus,

$$\begin{aligned} Z^T AZ &= \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} z_i z_j a_{ij} + \sum_{i=1}^{n} z_i^2 a_{ii} \\ &> \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} z_i z_j a_{ij} + \sum_{i=1}^{n} z_i^2 \sum_{j=1, j\neq i}^{n} |a_{ij}| \\ &= |a_{12}|(z_1 \pm z_2)^2 + |a_{13}|(z_1 \pm z_3)^2 + \cdots \\ &\quad + |a_{n(n-1)}|(z_n \pm z_{n-1})^2 \\ &\geq 0. \end{aligned} \tag{16}$$

Hence the result. □

Theorem 1 *Let A and B be real symmetric matrices. Then the roots $\{\kappa_i\}$ of the characteristic equation $det[A - \kappa B] = 0$ satisfy $\kappa_i \geq 1$, $(i = 1, 2, \ldots, n)$ if $A > 0$, $B > 0$ and $A - B \geq 0$.*

Proof A is real and symmetric. Thus there exists an orthogonal matrix Q such that $Q^T AQ = \Lambda_a$ where $\Lambda_a = diag(a_1, a_2, \ldots, a_n)$, where a_i $(i = 1, 2, \ldots, n)$ are the eigenvalues of A. Obviously, $a_i > 0$ for all $i = 1, 2, \ldots, n$ because A is positive definite.

Let $C = diag(c_1, c_2, \ldots, c_n)$ where $c_i = \sqrt{\frac{1}{a_i}}$. Clearly, we have $C^T Q^T AQC = I$ where I is the Identity matrix. If we denote $P = QC$, we get:

$$P^T AP = I. \tag{17}$$

Similarly, B is real and symmetric, and so there exists an orthogonal matrix H such that $H^T BH = \Lambda_b$ where $\Lambda_b = diag(b_1, b_2, \ldots, b_n)$, where $b_i > 0$ $(i = 1, 2, \ldots, n)$ are the eigenvalues of B. Consequently, we obtain

$$P^T BP = P^T H \Lambda_b H^T P = (H^T P)^T \Lambda_b (H^T P). \tag{18}$$

We now denote $R = H^T P$, and thus:

$$P^T BP = R^T \Lambda_b R. \tag{19}$$

Since H, Q and C are invertible, R is also invertible. As a result,

$$(R^T)^{-1} P^T BPR^{-1} = \Lambda_b. \tag{20}$$

On the other hand, observe that

$$R^T R = P^T HH^T P = P^T P = C^T Q^T QC = \Lambda_a^{-1}. \tag{21}$$

According to Eqs. (17) and (21), we get

$$(R^T)^{-1} P^T APR^{-1} = (R^T R)^{-1} = \Lambda_a. \tag{22}$$

If we now denote $(PR^{-1}) = S$, it implies that $S^T AS = \Lambda_a$ and $S^T BS = \Lambda_b$, where S is also invertible. One can easily verify that:

$S^T S = (PR^{-1})^T (PR^{-1}) = (PP^{-1}(H^T)^{-1})^T (PP^{-1}(H^T)^{-1}) = I$, which indicates the matrix S is orthogonal. Consequently:

$S^T (A - B)S = diag(a_1 - b_1, a_2 - b_2, \ldots, a_n - b_n)$.

We now use the fact that $A - B \geq 0$, which means that $a_i \geq b_i$ for $i = 1, 2, \ldots, n$. Therefore,

$det[A - \kappa B] = 0 \Rightarrow det[S^T AS - S^T \kappa BS] = 0$

$\Rightarrow det[S^T (A - \kappa B)S] = 0$

$\Rightarrow \kappa_i = a_i / b_i \geq 1$, proving the result. □

4 A Framework for CNNs Design

We are now in the position to discuss how we can force the system described by Eq. (14) to yield chaotic properties.

First of all, we do not expect the states of the network to tend towards infinity (i.e., unbounded values), because if they did the network would be "unusable". We thus constrain the self-feedback, the coupling function and the external stimulus uniform to be bounded, and so, for all $\boldsymbol{x}(t) \in \mathcal{R}^n$:

1. $||\boldsymbol{f}(\boldsymbol{x}(t))||_\infty \leq \epsilon$,
2. $||\boldsymbol{g}(\boldsymbol{x}(t))||_\infty \leq G$, and
3. $||\boldsymbol{h}(\boldsymbol{I}(t))||_\infty \leq H$.

We also set, $\boldsymbol{g}(\boldsymbol{0}) = \boldsymbol{0}$ which means there is no "coupling stimulation" while a neuron's net input is 0.

As a result, for any given initial point $\boldsymbol{x}(0)$, the trajectory:

$||\boldsymbol{x}(t)||_\infty \leq ||\boldsymbol{f}(\boldsymbol{x}(t))||_\infty + ||\boldsymbol{g}(\boldsymbol{x}(t))||_\infty + ||\boldsymbol{h}(\boldsymbol{I}(t))||_\infty \leq \epsilon + G + H$,

which implies that the network states are limited.

Our goal is to find a set of $\boldsymbol{f}(\cdot)$, $\boldsymbol{g}(\cdot)$ and $\boldsymbol{h}(\cdot)$ so that at least one the LEs of Eq.(14) is positive, which, in turn, would imply chaotic behavior. That is:

$$0 < c \leq \lambda_i(\boldsymbol{x}(0)) < \infty, \quad i \in \{1, 2, \ldots, n\}, \tag{23}$$

where c is a given constant.

Indeed, we can succeed in making the system described by Eq. (14) to be chaotic, as formalized in Theorem 2.

Theorem 2 *The system described by Eq. (14) is chaotic if $M_t = [\Gamma_t^T \Gamma_t] - e^{2tc} I$ is diagonal dominant where $\Gamma_t = J_t \cdot \Gamma_{t-1}$, Γ_t^T denotes the transpose of Γ_t, J_t is the Jacobian matrix of system described by Eq. (14) at time t, I is an $n \times n$ Identity matrix, and c is a given positive constant.*

Proof First of all, we mention that the existence of a single positive LE implies chaos. Thus, we intend to prove that the dynamical system described by Eq. (14), indeed, possesses at least a single positive LE by using the principles of induction and deduction.

We know that the Jacobian matrix is involved in calculating the LEs. The Jacobian matrix of Eq. (14) at point $\boldsymbol{x}(t)$ is defined as:

$$J_t = \boldsymbol{f}'(\boldsymbol{x}(t)) + \boldsymbol{g}'(\boldsymbol{x}(t)) + \boldsymbol{h}'(\boldsymbol{I}(t)), \tag{24}$$

Let us assume that the external stimulus is independent of $\boldsymbol{x}$, which is a very reasonable assumption. Thus $\boldsymbol{h}'(\boldsymbol{I}(t)) = \boldsymbol{0}$. Then, the equality Eq. (24) equals:

$$J_t = \boldsymbol{f}'(\boldsymbol{x}(t)) + \boldsymbol{g}'(\boldsymbol{x}(t)) = \Lambda_{\boldsymbol{x}(t)} + \boldsymbol{g}'(\boldsymbol{x}(t)), \tag{25}$$

where $\Lambda_{\boldsymbol{x}(t)} = diag\{f'(x_1(t)), f'(x_2(t)), \ldots, f'(x_n(t))\}$ is a diagonal matrix.

Let $\boldsymbol{g}'_t$ and Λ_t denote $\boldsymbol{g}'(x(t))$ and $\Lambda_{\boldsymbol{x}(t)}$ at time t respectively. In the interest of simplicity, we let $f(x_i(t)) = (-1)^m(\sigma(t)x_i(t) - 2m\epsilon)$, $m = 0, \pm 1, \pm 2, \ldots$ (which is called the "sawtooth" function[1], as shown in Fig. 5). Consequently:

$|f(x_i(t))| = |(-1)^m(\sigma(t)x_i(t) - 2m\epsilon)| \leq \epsilon$ and $|f'(x_i(t))| = \sigma(t)$.

Thus, when $t = 0$,

$$\begin{aligned} \Gamma_0^T \Gamma_0 &= J_0^T J_0 = (\Lambda_0 + \boldsymbol{g}'_0)^T \cdot (\Lambda_0 + \boldsymbol{g}'_0) \\ &= (\boldsymbol{g}'_0)^T \boldsymbol{g}'_0 + \Lambda_0[\boldsymbol{g}'_0 + (\boldsymbol{g}'_0)^T] + \Lambda_0^2. \end{aligned} \tag{26}$$

Observe that $[\Gamma_0^T \Gamma_0]$ is symmetric and Λ_0 is diagonal[2]. We can thus, certainly, find a large $\sigma(0)$ so as to make $[\Gamma_0^T \Gamma_0]$ diagonal dominant. According to Proposition 1, we know that a symmetric diagonal dominant matrix with a positive diagonal implies positive definiteness, which means that $[\Gamma_0^T \Gamma_0] > 0$ and that $[\Gamma_0^T \Gamma_0]^{-1} > 0$.

At any time t, we are going to demonstrate that an appropriate $\sigma(t)$ will lead to $M_t = [\Gamma_t^T \Gamma_t] - e^{2tc} I$ to be diagonal dominant.

When $t = 1$,

$$\begin{aligned} M_1 &= \Gamma_1^T \Gamma_1 - e^{2c} I \\ &= (J_1 \Gamma_0)^T \cdot J_1 \Gamma_0 - e^{2c} I \\ &= \Gamma_0^T (\Lambda_1 + \boldsymbol{g}'_1)^T \cdot (\Lambda_1 + \boldsymbol{g}'_1) \Gamma_0 - e^{2c} I \\ &= \left(\Lambda_1^2 \Gamma_0^T \Gamma_0 - e^{2c} I\right) + \Lambda_1 \Gamma_0^T \left[\boldsymbol{g}'_1 + (\boldsymbol{g}'_1)^T\right] \Gamma_0 + \Gamma_0^T (\boldsymbol{g}'_1)^T \boldsymbol{g}'_1 \Gamma_0. \end{aligned} \tag{27}$$

[1] The authors of [6] also use this function and a modulus function to achieve the self-feedback.

[2] Please note that $\Lambda_t = \sigma(t) \cdot I$, and thus $\Lambda_0 g'_0 = g'_0 \Lambda_0$.

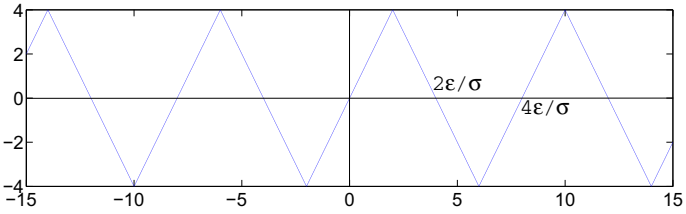

Fig. 5 An example of the sawtooth function. In this figure, $\sigma = 2$ and $\epsilon = 4$

The first item, $\Gamma_0^T \Gamma_0$ is diagonal dominant. Further, the second and third items are symmetric, and thus a proper $\sigma(1)$ can force the matrix given by Eq. (27) to be diagonal dominant, which implies that M_1 is positive definite as per Proposition 1. Thus,

$$M_1 = \Gamma_1^T \Gamma_1 - e^{2c} I = \left[(J_1 \Gamma_0)^T \cdot J_1 \Gamma_0 \right] - e^{2c} I \tag{28}$$

$$= \Gamma_0^T J_1^T J_1 \Gamma_0 - e^{2c} I > 0. \tag{29}$$

Since $\Gamma_0^T \Gamma_0 > 0$ and $\left[\Gamma_0^T \Gamma_0 \right]^{-1} > 0$, the inequality Eq. (29) can be rewritten as:

$$J_1^T J_1 - e^{2c} \left[\Gamma_0 \Gamma_0^T \right]^{-1} > 0. \tag{30}$$

Consequently, $J_1^T J_1 > 0$.

In a similar way, we may choose a serial $\{\sigma(t)\}$ so that the following conditions could be fulfilled:

$$\Gamma_{t-1}^T \Gamma_{t-1} > 0. \tag{31}$$

$$J_t^T J_t - e^{2tc} \left[\Gamma_{t-1} \Gamma_{t-1}^T \right]^{-1} > 0. \tag{32}$$

$$J_t^T J_t > 0. \tag{33}$$

Consider the inequality Eq. (32). If we let $A = J_t^T J_t$, $B = e^{2tc} \left[\Gamma_{t-1} \Gamma_{t-1}^T \right]^{-1}$, we observe that $A > 0$ and $A - B > 0$, as per Theorem 1, and all the eigenvalues of the matrix $A - \kappa B$ are no less than unity. Let us now substitute J_t and J_t^T by $\Gamma_t \Gamma_{t-1}^{-1}$ and $[\Gamma_{t-1}^{-1}]^T \Gamma_t^T$ respectively. Then

$$\begin{aligned} |A - \kappa B| &= \left| J_t^T J_t - \kappa e^{2tc} \left[\Gamma_{t-1} \Gamma_{t-1}^T \right]^{-1} \right| \\ &= \left| \left[\Gamma_{t-1}^{-1} \right]^T \Gamma_t^T \Gamma_t \Gamma_{t-1}^{-1} - \kappa e^{2tc} \left[\Gamma_{t-1} \Gamma_{t-1}^T \right]^{-1} \right| \\ &= e^{2tc} \left| \left[\Gamma_{t-1}^{-1} \right]^T \cdot \left[e^{-2tc} \Gamma_t^T \Gamma_t - \kappa I \right] \cdot \Gamma_{t-1}^{-1} \right|, \end{aligned} \tag{34}$$

which means that all the roots of the characteristic equation $\left| e^{-2tc} \Gamma_t^T \Gamma_t - \kappa I \right| = 0$ are no less than unity. In other words, all the eigenvalues of $\Gamma_t^T \Gamma_t$ are no less than e^{2tc}. Therefore, according to the definition of LEs,

$$\begin{aligned} \lambda_i &= \lim_{t \to \infty} \frac{1}{2t} \log \left| \mu_i \left[\Gamma_t^T \Gamma_t \right] \right| \\ &\geq \lim_{t \to \infty} \frac{1}{2t} \log(e^{2tc}) = c > 0. \end{aligned} \tag{35}$$

We have thus proven that the system described by Eq. (14) is truly chaotic since $\lambda_i > 0$. □

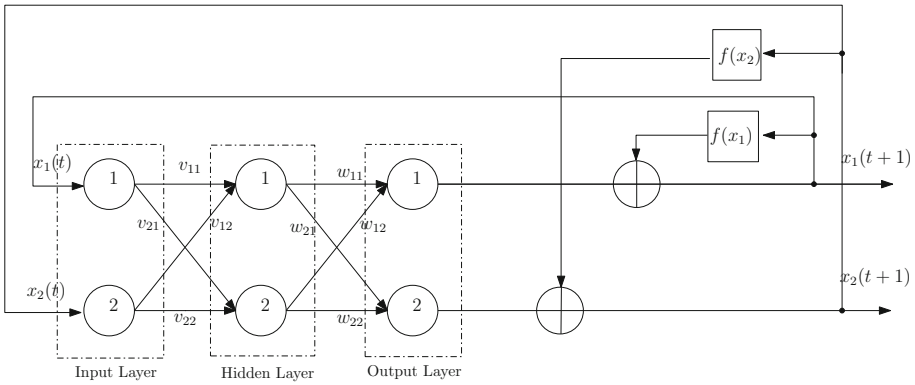

Fig. 6 The topological structure of the CNN model defined by Eq. (36). It consists of an input layer, an output layer, a hidden layer and a self-feedback component. In each layer, only 2 neurons are plotted in the interest of simplicity

5 Experimental Results

Based on the analysis above, we can obtain one of the simplest CNN models that can be characterized by:

$$x_i(t+1) = f(x_i(t)) + \sum_{j=1}^{n} w_{ij} \tanh\left(\sum_{k=1}^{n} v_{jk} x_k(t)\right), \tag{36}$$

where $f(\cdot)$ is a sawtooth function. W and V are randomly generated[3] between 0 and 1. The total discrete time length is $T = 2000$.

$$W = \begin{pmatrix} 0.21 & 0.03 & 0.05 \\ 0.55 & 0.61 & 0.49 \\ 0.63 & 0.36 & 0.19 \end{pmatrix}, \quad V = \begin{pmatrix} 0.12 & 0.19 & 0.28 \\ 0.21 & 0.04 & 0.54 \\ 0.15 & 0.64 & 0.70 \end{pmatrix}. \tag{37}$$

The topological structure of this CNN model is shown in Fig. 6.

It possesses rich dynamical properties. As one can verify:

1. $|g_i(\boldsymbol{x})| = |\sum_{j=1}^{n} w_{ij} \tanh(\sum_{k=1}^{n} v_{kj} x_j(t))| \leq n$,
2. $|\frac{\partial g_i(\boldsymbol{x})}{\partial x_j}| \leq w_{ij}$,
3. $|f(x)| = |(-1)^m(\sigma x - 2m\epsilon)| \leq \epsilon$, and
4. $|f'(x)| = \sigma$ and $g(\mathbf{0}) = 0$.

We can visualize the system's dynamical behavior by plotting the phase diagram of three neurons: $x_1(t)$, $x_2(t)$ and $x_3(t)$. These phase diagrams are shown in Figs. 7 and 8.

We explain the system dynamics as follows:

1. Let $\epsilon = 0$. In this setting, there is no self-feedback according to the sawtooth function definition. In this case, as shown in Fig.7, no chaos phenomenon is observed. Instead, the trajectory converges at a period-n orbit, as shown in Fig. 7.

[3] In this part of the paper, we only concentrate on the design of CNNs and we do not discuss their applications. This is why we randomly generate W and V here. However, W and V can be also weighted by appropriate learning algorithms for different applications.

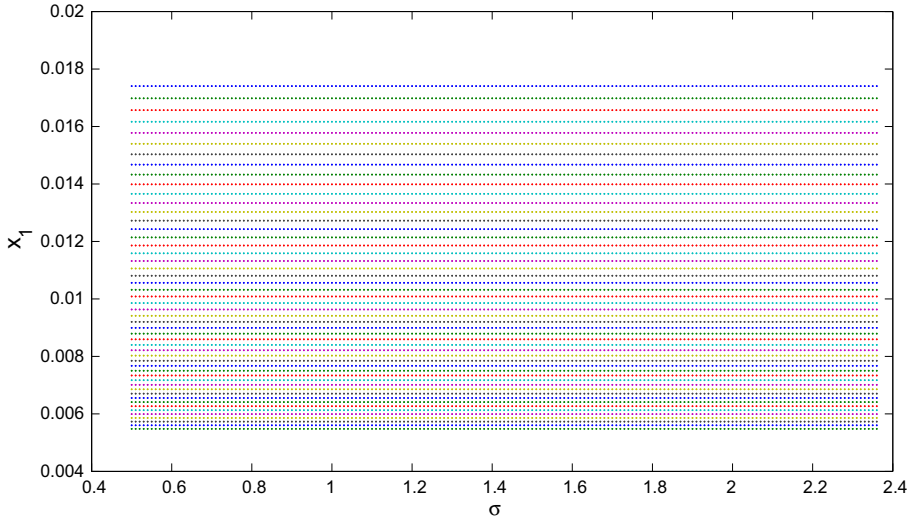

Fig. 7 The network's trajectory converges to a serial period-n points when there is no self-feedback (when $\epsilon = 0$). Each *dotted-line* implies a period-n point

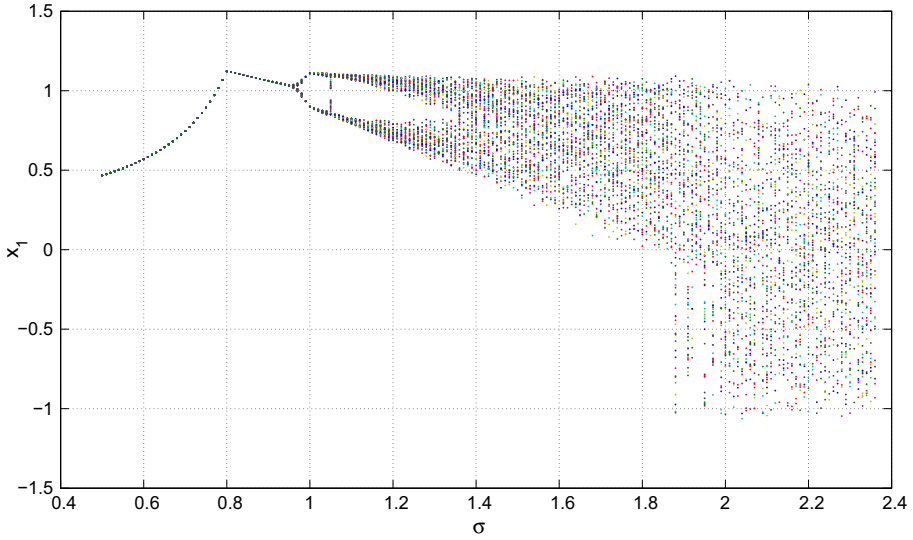

Fig. 8 Double-period bifurcation and chaos happen at different values of σ

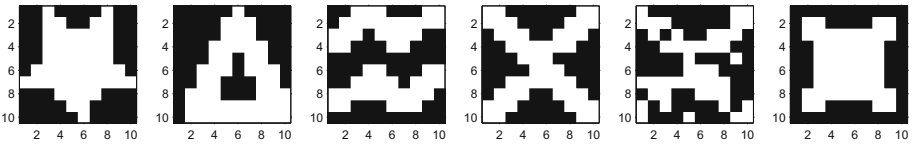

Fig. 9 The 10×10 patterns used by Adachi et al. The first four patterns are used to train the network. The fifth patterns are obtained from the fourth pattern by including 15 % noise. The sixth pattern is the untrained pattern

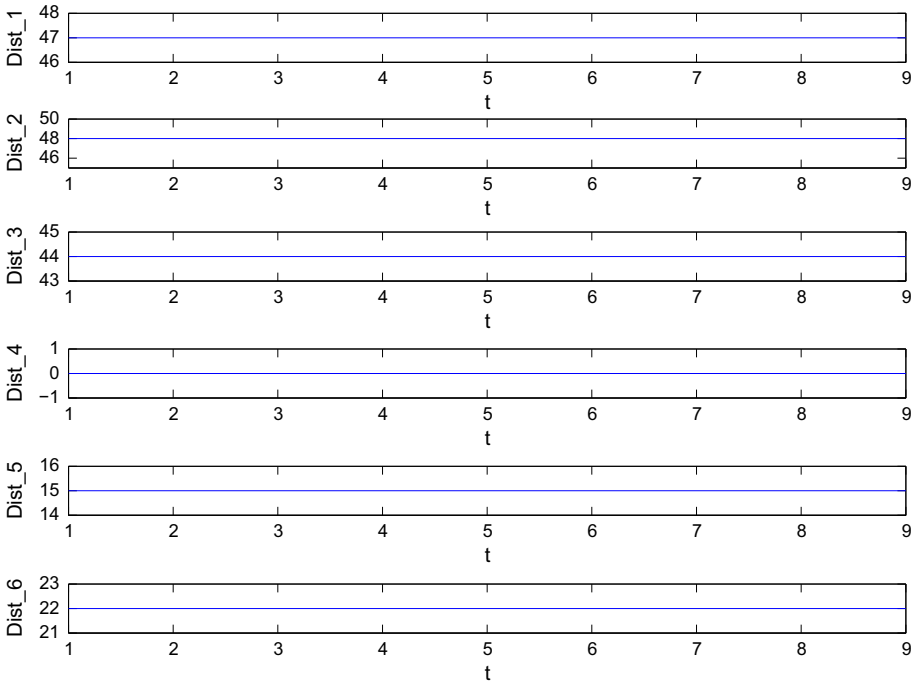

Fig. 10 PR properties: The Hamming distance between the output and the trained patterns. The input patterns are the fourth pattern of Fig. 9

2. Let ϵ increase, e.g., $\epsilon = 1.0$. Here, the phase space of the network varies with the value of σ:
   (a) If $\sigma < 0.95$, there is still no chaos, the trajectory converges at a fixed point.
   (b) A double-period bifurcation happens when $\sigma \approx 0.95$. We must point out that it is not easy to calculate the exact point of σ where this bifurcation happens since the network is a high dimensional chaotic system. $\sigma \approx 0.95$ can be observed from Fig. 8.
   (c) Chaos windows appear while σ increases, e.g., $\sigma = 1.4$. As we can verify, in this case, such values of σ, W and V could lead to Eqs. (31), (32) and (33) to be diagonal dominant, which implies chaos as per Theorem 2.

6 Applications of the Designed Model

6.1 Chaotic Pattern Recognition

We shall now report the PR properties of the designed model specified by Eq. (36). These properties have been gleaned as a result of examining the Hamming distance between the input pattern and the patterns that appear at the output. The experiment was conducted with the data sets used in [1], which is given in Fig. 9, and are referred to as the *Adachi* data sets. The patterns were described by 10×10 pixel images, and the networks thus had 100 neurons.

Before we proceed, we emphasize that there is a marked difference between the basic principles of achieving PR using CNNs and when one uses a classical NN. Traditionally, a classical NN will "stay" at a certain *known* pattern if the input can be recognised. As opposed

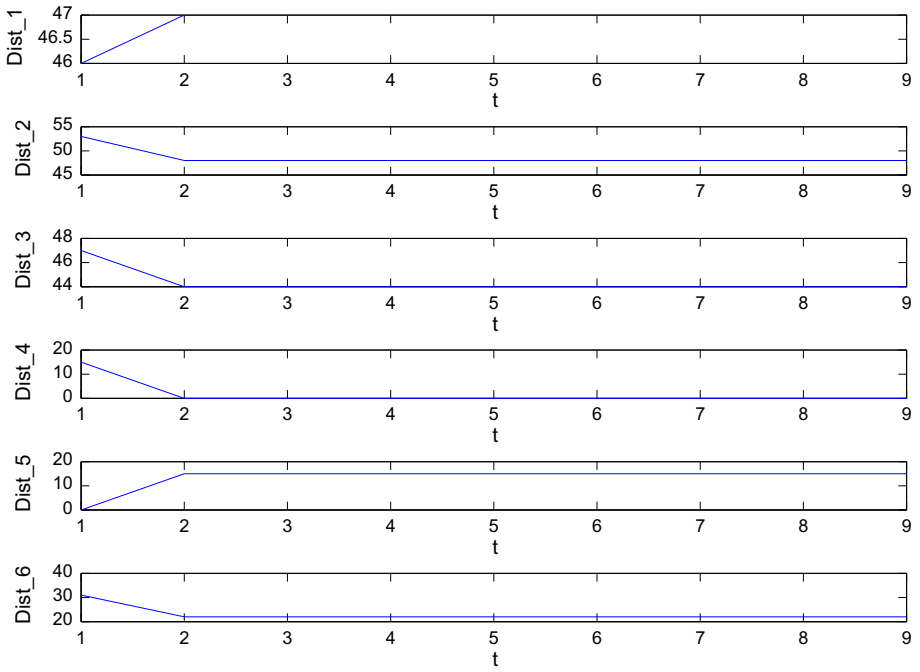

Fig. 11 PR properties: The Hamming distance between the output and the trained patterns. The input pattern is the fifth pattern of Fig. 9

to this, if the input cannot be recognised, the network outputs an *unknown* pattern, implying that the pattern is not one of the trained patterns. Observe that in both these situations, the outputs of network are stable. However, the basic principle of Chaotic PR is significantly different. In the ideal setting we would have preferred the CNN to be chaotic when it is exposed to untrained patterns, and the output to appear stable when it is exposed to trained patterns.

To obtain the desired PR properties of the model described by Eq. (36), the parameters were set as follows: w is the synaptic weight obtained by the Hebbian rule, v is set, for simplicity, as the Identity matrix, and the transfer function is a sigmoid function. We enumerate three cases as below:

1. The initial input of the network is a known pattern, say P4.
   The Hamming distance converges to 0 immediately, which implies that the output converges to the input pattern, as shown in Fig. 10. Obviously, we can conclude that the input pattern can be recognised successfully if it is one of the known patterns.
2. The initial input of the network is a noisy pattern, in this case P5, which is a noisy version of P4.
   It is interesting to see that the output converges to the original pattern P4 instead of the initial input P5 after only one step. That is, even if the initial input contains some noise, the network is still able to recognize it correctly (Fig. 11).
3. The initial input of the network is an unknown pattern, P6.
   From Fig. 12 we see that the output does not converge to any known/unknown pattern.
   From the above three figures we can conclude the following: If a CNN is appropriately defined, it converges immediately when presented with known patterns or if the input

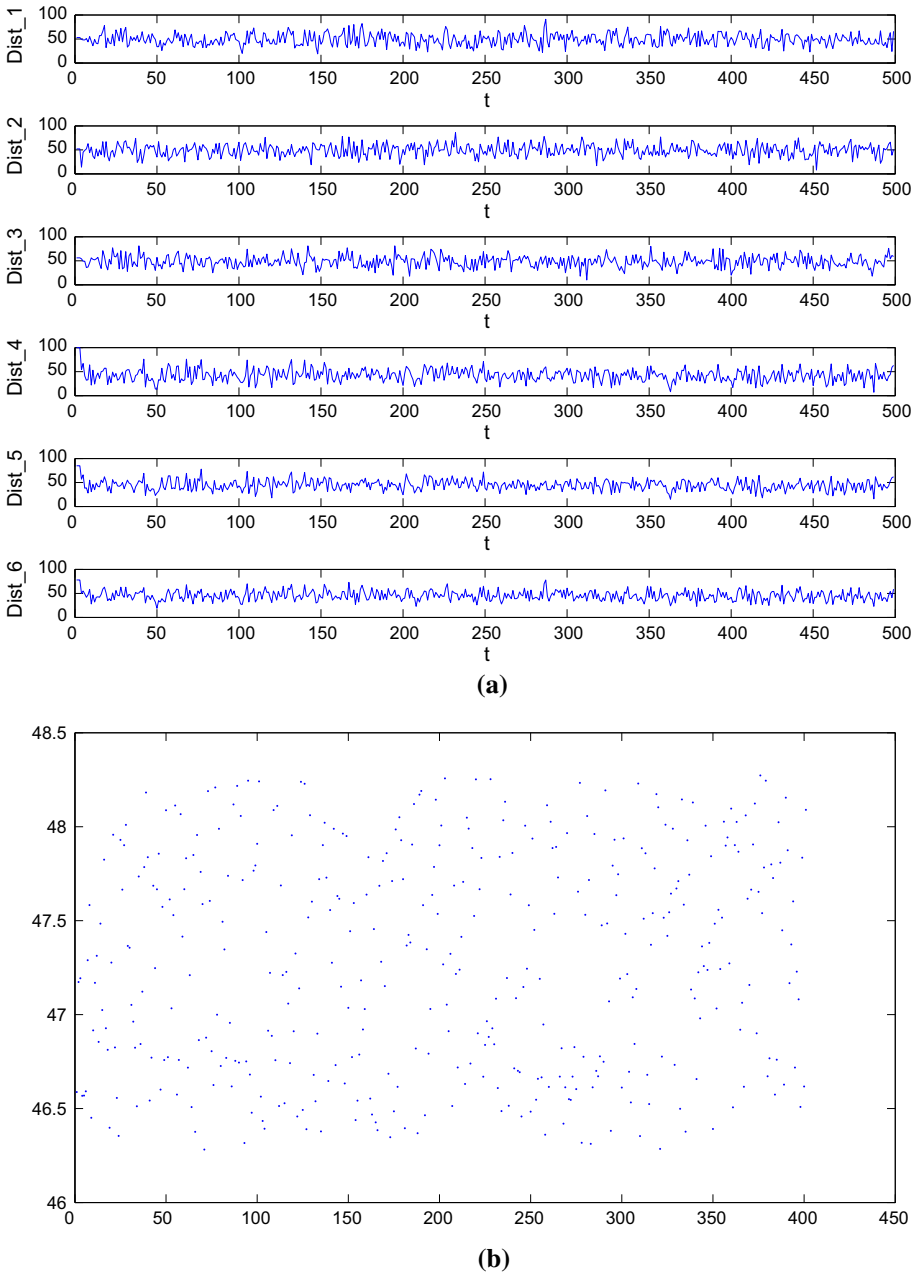

Fig. 12 PR properties: **a** The Hamming distance between the output and the trained patterns. The input pattern is the sixth pattern of Fig. 9. The Hamming distance does not imply convergence. **b** The chaotic trajectory of a randomly chosen neuron

is a noisy version of a trained pattern. On the other hand, it demonstrates chaos when presented with unknown patterns. In this sense, we confirm that the proposed CNN possesses chaotic PR properties.

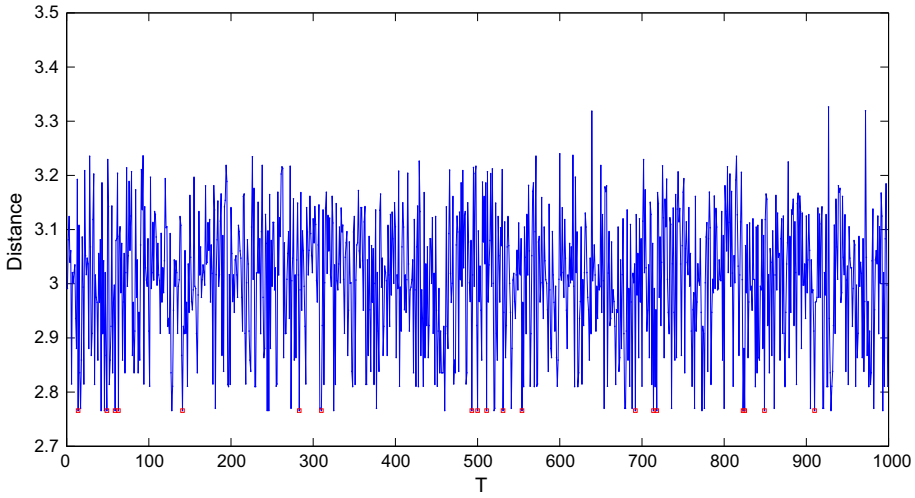

Fig. 13 The solution (minimum distance summation) of the TSP varying with the evolution of the network in 1000 steps

6.2 Solving the Traveling Salesman Problem

In this section, we report the results of utilizing the CNN to solve the Traveling Salesman Problem (TSP). We adopt the same coordinates for the cities as used in [14]. In the interest of brevity, we omit the details of how we apply CNNs to solve TSP. Actually, we used the exact same network topology, weights and energy function as Hopfield et al. used in [13,14,25]. The only difference was the dynamics of neurons. To be specific, all the neurons' states were updated according to the model given by Eq. (36), which has been proven to possess chaotic properties.

As per Fig. 13, we find that during the network's evolution, the optimal solution was located 16 times (labeled here by red squares) within 1000 iterations. With regard to its searching ability (37 times within 38,558 iterations reported in [25]), we affirm that our newly proposed model performs better, even though it is a very general model.

7 Conclusions

In this paper we have investigated how to design a chaotic neural network (CNN) by appropriately applying a self-feedback function to a recurrent neural network. By means of Jacobian matrix analysis, diagonal dominant matrix and Lyapunov analysis, we have proved that a two-layer recurrent NN with a carefully-devised feedback function can lead to chaotic behavior. Numerical results have been presented that are based on experiments conducted by using the sawtooth function as the self-feedback function and a hyperbolic tangent function as the coupling function. The results show that our general CNN model is able to present rich periodic and chaotic properties by choosing appropriate control parameters. The applications of the paper for PR and to solve the TSP have also been included.

The future work spawning from this paper would be to train this model by using suitable learning algorithms so that it can be applied in various areas including pattern recognition, associate memory, cryptography etc.

Acknowledgements The author is extremely grateful to the anonymous Referees of the initial version of this paper for their valuable comments. Their comments significantly improved the quality of this paper. He is also sincerely grateful to Prof. B. J. Oommen from Carleton University in Canada, for being his friend and colleague, and for his valuable feedback. This work is supported by National Natural Science Foundation of China (Grant No. 61300093) and Fundamental Research Funds for the Central Universities in China (Grant No. ZYGX2013J071).

References

1. Adachi M, Aihara K (1997) Associative dynamics in a chaotic neural network. Neural Netw 10(1):83–98
2. Aihara K, Takabe T, Toyoda M (1990) Chaotic neural networks. Phys Lett A 144(6–7):333–340
3. Araujo FR, Bueno LP, Campos MA (2007) Dynamic behaviors in chaotic bidirectional associative memory. J Intell Fuzzy Syst 18(5):513–523
4. Calitoiu D, Oommen BJ, Nussbaum D (2007) Desynchronzing of chaotic pattern recognition neural network to model inaccurate parception. IEEE Trans Syst Man Cybern Part B 37(3):692–704
5. Calitoiu D, Oommen BJ, Nussbaum D (2007) Periodicity and stability issues of a chaotic pattern recognition neural network. Pattern Anal Appl 10(3):175–188
6. Chen G, Lai D (1996) Feedback control of lyapunov exponents for discrete-time dynamical systems. Int J Bifurc Chaos Appl Sci Eng 6(7):1341–1350
7. Chua L, Yang L (1988) Cellular neural networks: applications. IEEE Trans Circuits Syst 35(10): 1273–1290
8. Chua L, Yang L (1988) Cellular neural networks: Theory. IEEE Trans Circuits Syst 35(10):1257–1272
9. Datta A, Talukdar V, Knoar A, Jain LC (2009) A neural network based approach for protein structural class prediction. J Intell Fuzzy Syst 20(1–2):61–71
10. Freeman WJ (1992) Tutorial on neurobiology: from single neurons to brain chaos. Int J Bifurcation Chaos Appl Sci Eng 2:451–482
11. Garcia-Orellana C, Marcias M, Serrano-Perez A, Gonzalez-Velasco HM, Gallardo-Caballero R (2002) A comparison of pca, ica and ga selected features for cloud field classification. J Intell Fuzzy Syst 12(3–4):213–219
12. Ghosh-Dastidar S, Adeli H (2009) Spiking neural network. Int J Neural Syst 19(4):295–308
13. Hiura E, Tanaka T (2007) A chaotic neural network with duffing's equation. In: Proceedings of international joint conference on neural networks, Orlando, Florida, USA, pp 997–1001
14. Hopfield J, Tank D (1985) Neural computation of decision in optimization problems. Biol Cybern 52: 141–152
15. Li SY (2011) Chaos control of new mathieu-van der pol systems by fuzzy logic constant controllers. Appl Soft Comput 11(8):4474–4487
16. Li YT, Deng SJ, Xiao D (2011) A novel hash algorithm construction based on chaotic neural network. Neural Comput Appl 20(1):133–141
17. Oliver JL, Tortosa L, Vicent JF (2011) An application of a self-organizing model to the design of urban transport networks. J Intell Fuzzy Syst 22(2–3):141–154
18. Qin K, Oommen BJ (2008) Chaotic pattern recognition: the spectrum of properties of the adachi neural network. In: Lecuture Notes in Computer Science, Florida, USA, vol. 5342, pp 540–550
19. Qin K, Oommen BJ (2009) Adachi-like chaotic neural networks requiring linear-time computations by enforcing a tree-shaped topology. IEEE Trans Neural Netw 20(11):1797–1809
20. Qin K, Oommen BJ (2009) An enhanced tree-shaped adachi-like chaotic neural network requiring linear-time computations. In: The 2nd international conference on chaotic modeling, simulation and applications, Chania, Greece, pp 284–293
21. Qin K, Oommen BJ (2012) The entire range of chaotic pattern recognition properties possessed by the adachi neural network. Intell Decis Technol 6:27–41
22. Qin K, Oommen BJ (2014) Logistic neural networks: their chaotic and pattern recognition properties. Neurocomputing 125:184–194
23. Qin K, Oommen BJ (2015) On the cryptanalysis of two cryptographic algorithms that utilize chaotic neural networks. Math Probl Eng. doi:10.1155/2015/468567
24. Suzuki H, Imura J, Horio Y, Aihara K (2013) Chaotic boltzmann machines. Sci Rep 3:1–5
25. Tanaka T, Hiura E (2003) Computational abilities of a chaotic neural network. Phys Lett A 315(3–4): 225–230
26. Tanaka T, Hiura E (2005) Dynamical behavior of a chaotic neural network and its applications to optimization problem. In: The international joint conference on neural network, Montreal, Canada, pp 753–757

27. Timotheou S (2010) The random neural network: a survey. Comput J 53(3):251–267
28. Tsui APM, Jones AJ (1999) Periodic response to external stimulation of a chaotic neural network with delayed feedback. Int J Bifurcation Chaos 9(4):713–722
29. Yu WW, Cao JD (2006) Cryptography based on delayed chaotic neural networks. Phys Lett A 356(4–5):333–338