Chaotic Modeling and Simulation (CMSIM) 4: 583–590, 2013

Chaotic Neural Networks with a Random Topology Can Achieve Pattern Recognition*

Ke Qin[1] and B. J. Oommen[2,3]

[1] University of Electronic Science & Technology of China, Chengdu, China. 611731
(E-mail: qinke@uestc.edu.cn)
[2] Carleton University, Ottawa, ON, Canada. K1S 5B6
[3] University of Agder, Postboks 509, 4898 Grimstad, Norway.
(E-mail: oommen@scs.carleton.ca)

Abstract. This paper confirms the fascinating result that we can design chaotic Neural Networks (NNs) that have a *random* topology and that these NNs can achieve *chaotic* Pattern Recognition (PR). What we imply by this is that the NN yields a strong *periodic* or *more frequent* signal when a pattern is recognized, and in between two consecutively recognized patterns, none of the trained patterns are recalled. Finally, and most importantly, if an untrained pattern is presented, the system yields a chaotic signal. The basic model that we use here is the Adachi Neural Network (AdNN), which we modify in a random manner. The AdNN is a fascinating NN which has been shown to possess chaotic properties, and to also demonstrate Associative Memory (AM) and PR, and some of its variants have also been used to obtain other PR phenomena, including blurring. All these NNs require a quadratic number of computations in the training phase. This computation was reduced to be linear in [1] by resorting to a Maximum Spanning Tree topology, and a gradient search method. In this paper, we mainly consider the issue of how the network topology can be modified by involving randomized connections so as to render the new network much closer to "*real*" NNs. At the same time, we require that the newly obtained network still displays PR characteristics. To achieve this, we first construct a random network by means of the E-R model and then address the problem of computing the weights for the new network. This is done by constraining the the modified random connection-based NN to have approximately the same input-output characteristics using a gradient-based algorithm. Through a detailed experimental analysis, we show that the new random AdNN-like network possesses PR properties for appropriate settings. As far as we know, such a random AdNN has not been reported, and our present results are novel.

Keywords: Chaotic Neural Networks, Chaotic Pattern Recognition, Adachi-like Neural Networks, Random Networks.

1 Introduction

The goal of the field of *Chaotic* Pattern Recognition (PR) systems can be summarized as follows: We do not intend a chaotic PR system to report the identity

*A preliminary version of this paper was presented at CHAOS'13, the 2013 Chaotic Modeling and Simulation International Conference, Istanbul, Turkey, in June 2013.

Received: 27 March 2013 / Accepted: 8 September 2013

ISSN 2241-0503

of a testing pattern with a "class proclamation" indicating the class to which the pattern belongs. Rather, what we want to achieve is to have the chaotic PR system give a strong *periodic* or *more frequent* signal when a pattern is recognized. Furthermore, between two consecutively recognized patterns, none of the trained patterns must be recalled. Finally, and most importantly, if an untrained pattern is presented, the system must give a chaotic signal.

The use of Artificial Neural Networks (ANNs) is one of the four best approaches for PR. However, one of the limitations of most ANN models is the dependency on an external stimulation. Once an output pattern has been identified, the ANN remains in that state until the arrival of a new external input. This is in contrast to real biological NNs and the brain, which exhibit sequential memory characteristics. Indeed, once a pattern is recalled from a memory location, the brain is not "stuck" in it; it is also capable of recalling other associated memory patterns without being prompted by any additional external inputs. This ability to "jump" from one memory state to another *in the absence of a stimulus* is one of the hallmarks of the brain, which is *one phenomenon that a chaotic PR system has to emulate.*

This paper deals with the Adachi Neural Network AdNN [2], which possesses a spectrum of very interesting chaotic, AM and PR properties, as described in [1,3–7,9–12]. The fundamental problem associated with the AdNN and its variants are their quadratic computational requirements. We shall show that by using the E-R model and an effective gradient search strategy, this burden can be significantly reduced, and yet be almost as effective with regard to the chaotic and PR characteristics.

We are currently working on reducing the complexity of the AdNN and the associated computations by invoking the so-called "small-world" model.

2 Limitations of the Current Schemes

Adachi *et al* and Calitoiu *et al* have done a lot of ground-breaking work in this area [2–4], and we have built on these results in various avenues [3–5], including that of designing a NN that can yield *ideal* chaotic PR [8]. Generally speaking, the computational burden of the family of AdNNs is excessive, rendering it impractical. Besides this, most of current NNs have a regular topology, e.g., a completely connected graph or a neighbor-coupled graph. This is in contrast with "real" NNs which usually have irregular topologies, e.g., a random graph, a small-world graph or even a scale-free graph. The contribution of this paper is to present a novel NN which is connected in a randomized AdNN way, which we refer to as the "Random-AdNN".

3 Designing the Random-AdNN

3.1 The Topology of the Random-AdNN

To present the new characteristics of the Random-AdNN, we shall first arrive at a topology with randomly-chosen edges. Such a modified random AdNN is obtained in two steps. Firstly, we connect the neurons by using the E-R model. The second step involves the computation of the weights associated with this new structure, which we will address subsequently.

Algorithm 1 Topology_Random-AdNN

Input: N, the number of neurons in the network, and a set of P patterns which the network has to "memorize".

Output: The topology and initial weights of the Random-AdNN.

Method:

1: Create a fully-connected graph $\mathcal{G}$ with N vertexes which represents the AdNN.
2: For each edge, we delete it with a fixed probability, p_d.
3: Continue this process for all the $\binom{N}{2}$ edges.
4: Compute the initial weights of the edges of $\mathcal{G}$, $\{w_{ij}\}$ as follows:
$w_{ij} = \frac{1}{P}\sum_{s=1}^{P}(2x_i^s - 1)(2x_j^s - 1)$, where x_i^s is the i^{th} component of the s^{th} pattern.
5: If there is no edge between vertex i and j, $w_{ij} = 0$;

End Algorithm Topology_Random-AdNN

3.2 The Weights of the Random-AdNN: Gradient Search

Since we have removed most of the "redundant" edges from the completely-connected graph by using the E-R model, it is clear that the NN at hand will not adequately compare with the original AdNN. Thus, our next task is to determine a new set of weights so as to force the Random-AdNN to retain some of its PR properties, namely those corresponding to the trained patterns. We briefly explain below (the details are omitted in the interest of space, and one can refer to [13] for more details) the process for achieving this.

The Random-AdNN is defined by the following equations:

$$x_i^R(t+1) = f(\eta_i^R(t+1) + \xi_i^R(t+1)), \tag{1}$$

$$\eta_i^R(t+1) = k_f\eta_i^R(t) + \sum_{e_{ij}\in T} w_{ij}^{R^*} x_j^R(t), \tag{2}$$

$$\xi_i^R(t+1) = k_r\xi_i^R(t) - \alpha x_i^R(t) + a_i. \tag{3}$$

where $\{w_{ij}^{R^*}\}$, x_i^R, ξ_i^R and η_i^R are the weights, outputs, and state variables of the Random-AdNN respectively, and have similar meanings to $\{w_{ij}\}$, x_i, ξ_i and η_i of the AdNN.

In order to find the optimal values of $\{w_{ij}^{R^*}\}$, we define the square error between the original output of the AdNN and new output at the n^{th} step as:

$$E_p = \frac{1}{2}\sum_{i=1}^{N}(x_i^{A,p} - x_i^{R,p}(n))^2, \tag{4}$$

where $x_i^{A,p}$ and $x_i^{R,p}$ imply the outputs of the i^{th} neuron when the p^{th} pattern is presented to the AdNN network and the Random-AdNN network respectively. The overall global error is $E = \sum_{p=1}^{P} E_p$, where there are P training patterns.

In order to adjust w_{ij}^R to obtain the smallest global error E, we consider the gradient, Δw_{ij}^R, and move w_{ij}^R by an amount which equals Δw_{ij}^R in the direction where the error is minimized. This can be formalized as follows:

$$\Delta w_{ij}^R = -\beta\frac{\partial E}{\partial w_{ij}^R} = -\beta\frac{\partial\sum_{p=1}^{P} E_p}{\partial w_{ij}^R} = -\beta\sum_{p=1}^{P}\frac{\partial E_p}{\partial x_i^{R,p}(n)}\cdot\frac{\partial x_i^{R,p}(n)}{\partial w_{ij}^R}$$

$$= \beta \sum_{p=1}^{P} (x_i^{A,p} - x_i^{R,p}(n)) \cdot \frac{1}{\varepsilon} \cdot x_i^{R,p}(n) \cdot (1 - x_i^{R,p}(n)) \cdot x_j^{R,p}(n), \qquad (5)$$

where β is the learning rate of the gradient search. The formal algorithm which achieves the update can be found [14].

The results of a typical numerical experiment which proceeds along the above gradient search on the Adachi data set (shown in Fig. 5) are displayed in Fig. 1 and 3. In these, we have chosen the learning rate β to be 0.05. To clarify issues, we catalogue our experiments for three specific cases, namely when the probability p_d for deleting an edge is 0.9, 0.5 and 0.1 respectively.

If p_d is 0.9, the total error E and average values of Δw_{ij}^R do not converge to 0, as shown in Fig. 1. However, as p_d decreases, e.g., 0.5, then E and Δw_{ij}^R converge to 0, as shown in Fig. 3 (a) and (b). If p_d is even less, E and Δw_{ij}^R also converge to 0 but with a faster rate, as shown in Fig. 3 (c) and (d). This phenomenon can be easily explained: The larger the value of p_d, the smaller is the number of edges and vice versa. Thus, if $p_d = 0$, it means that the Random-AdNN is exactly the same as the original AdNN. On the other hand, if $p_d = 1$, it means that all the vertexes are isolated and remain as disconnected units. Of course, the "fitting" effect that we obtain by the approximate graph, the Random-AdNN, is more precise as the number of edges increases.

The experimental results obtained for the LOVE data set (also shown in Fig. 5) are quite similar, and are displayed in Fig. 2 and 4.

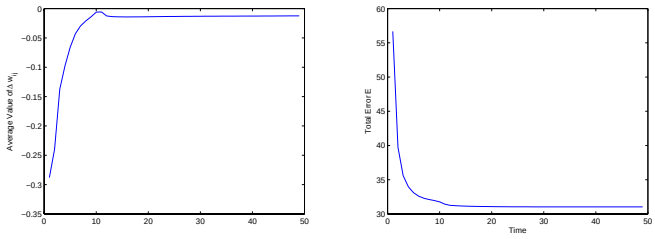


Fig. 1. For the Adachi data set: The figure on the left shows the variation of the average of Δw_{ij}^L (averaged over all values of i and j) over the first 50 iterations of the gradient search scheme. The average converges to a value arbitrarily close to zero after 12 time steps. The figure on the right shows the variation of the global error over the same time frame. Observe that this quantity does not converge to *zero*.

The Lyapunov analysis of the Random-AdNN is also available, but omitted here in the interest of space. It can be found in [13].

4 Chaotic and PR Properties of the Random-AdNN

We now briefly report the PR properties of the Random-AdNN. These properties have been discovered as a result of examining the Hamming distance between the input pattern and the patterns that appear in the output. The experiments were conducted using two data sets described below.

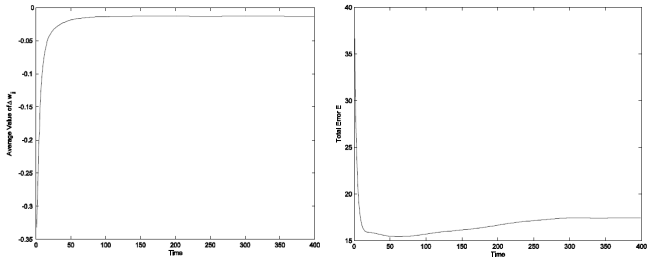


Fig. 2. For the LOVE data set: The figure on the left shows the variation of the average of Δw_{ij}^L over the first 400 iterations of the gradient search scheme. The average converges to a value arbitrarily close to zero after 50 time steps. The figure on the right shows the variation of the global error over the same time frame, which does not converge to *zero* either.

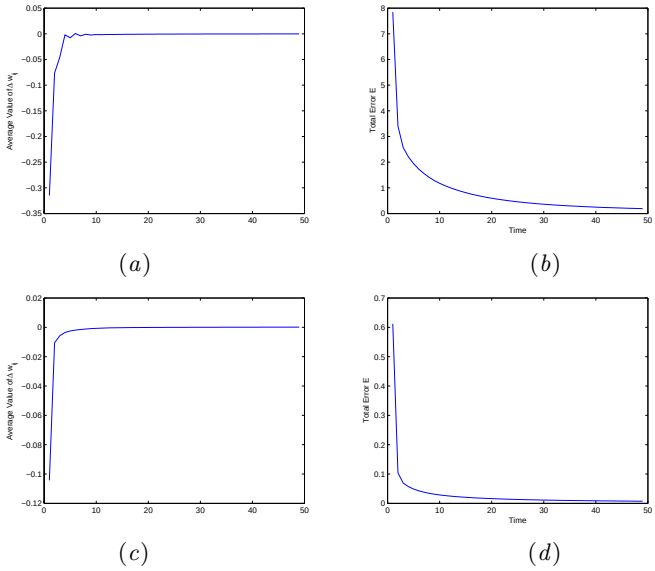
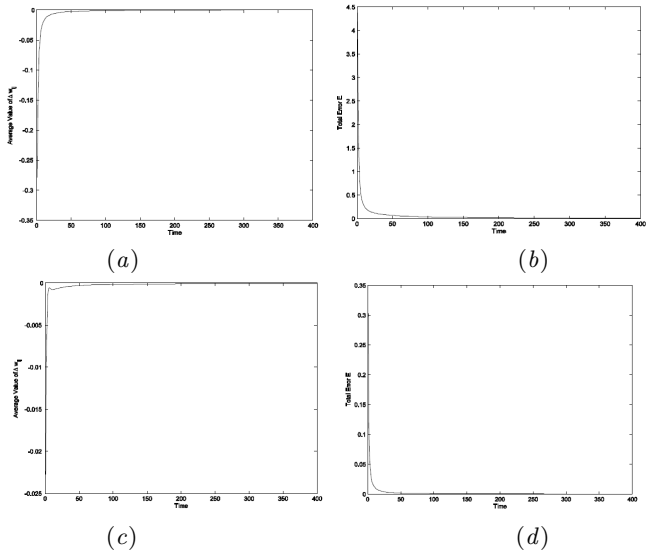


Fig. 3. For the Adachi data set: The figures show the variation of the average of Δw_{ij}^L and the global error over the same time frame. The probability of edge deletion is $p_d = 0.5$ (for (a) and (b)) and $p_d = 0.1$ (for (c) and (d)) respectively.

In the ideal setting we would have preferred the Random-AdNN to be chaotic when exposed to untrained patterns, and the output to appear periodically or more frequently when exposed to trained patterns. Besides yielding this phenomenon, the Random-AdNN also goes through a chaotic phase and a PR phase as some of its parameters change.

We summarize the results for the Random-AdNN, obtained by using different settings of p_d. The others parameters are: $k_f = 0.2$, $k_r = 1.02$, $\alpha = 10$, $\varepsilon = 0.015$, $\beta = 0.05$.

From these tables we see clearly that, the Random-AdNN is able to "resonate" the input patterns with the corresponding output patterns. Consider Table 1 (a) as an example. If the input is P1, then the network outputs P1 accordingly, and



(a) (b) (c) (d)

Fig. 4. For the LOVE data set: The figures show the variation of the average of Δw_{ij}^L and the global error over the same time frame. The probability is $p_d = 0.5$ (for (a) and (b)) and $p_d = 0.1$ (for (c) and (d)) respectively.

(a) (b)

Fig. 5. The patterns used by Adachi *et al* (a) and Inoue *et al* (b). The first four patterns in (a) and (b) are used to train the network. The fifth pattern in (a) is obtained from the fourth pattern by including 15% noise. The sixth pattern in (a) and the fifth pattern in (b) are the untrained patterns.

at the same time, no other trained patterns appear in the output sequence. Even when a noisy pattern is presented to the system, e.g., P5, which is a noisy pattern of P4 with 15% noise, the network still "resonates" P4 instead of P5 in the output sequence. Furthermore, if the input is an untrained pattern, e.g., P6, then none of the trained patterns will be recalled. Observe that even the input pattern P6, will itself be retrieved only a few times, which is much less than the other diagonal entries in the table, i.e., when the inputs are P1 – P4. The difference between (a) – (c) is that in Table (c), the network "resonates" the input patterns more frequently than in (a) and (b). This is because when $p_d = 0.1$, the Random-AdNN is almost the same as the original AdNN since the Random-AdNN has most of the edges of the AdNN. However, in this case, the Random-AdNN also needs a quadratic number of computations, which is computationally much more intensive than for the case when $p_d = 0.9$. In this regard, we comment that $p_d = 0.9$ is good enough for PR, which has only a very small computational burden. By a simple computation we can see that the expected degree for each vertex of the Random-AdNN is only $N(1-p_d) = 10$ for the Adachi data set, which implies that

Table 1. The frequency of the Hamming distance between the input and the output patterns for the Random-AdNN. The probability p_d is 0.9, 0.5, 0.1 for (a), (b), (c) respectively.

$p_d = 0.9$		Input Patterns					
		P1	P2	P3	P4	P5	P6
Retrieved Patterns	P1	151	0	0	0	0	0
	P2	0	422	0	0	0	0
	P3	0	0	161	0	0	0
	P4	0	0	0	106	177	0
	P5	0	0	0	10	2	0
	P6	0	0	0	0	0	46

(a)

$p_d = 0.5$		Input Patterns					
		P1	P2	P3	P4	P5	P6
Retrieved Patterns	P1	202	0	0	0	0	0
	P2	0	285	0	0	0	0
	P3	0	0	234	0	0	0
	P4	0	0	0	211	206	0
	P5	0	0	0	4	3	0
	P6	0	0	0	0	0	33

(b)

$p_d = 0.1$		Input Patterns					
		P1	P2	P3	P4	P5	P6
Retrieved Patterns	P1	238	0	0	0	0	0
	P2	0	331	0	0	0	0
	P3	0	0	258	0	0	0
	P4	0	0	0	237	189	0
	P5	0	0	0	9	20	0
	P6	0	0	0	0	0	34

(c)

the computational load has been greatly reduced when compared to the original AdNN, which has a vertex degree of 99.

5 Conclusions

In this paper we have concentrated on the field of Chaotic Pattern Recognition (PR), which is a relatively new sub-field of PR. Such systems, which have only recently been investigated, demonstrate chaotic behavior under normal conditions, and resonate when it is presented with a pattern that it is trained with. The network that we have investigated is the Adachi Neural Network (AdNN) [2], which has been shown to possess chaotic properties, and to also demonstrate Associative Memory (AM) and Pattern Recognition (PR) characteristics. In this paper we have considered how the topology can be modified so as to render the network much closer to "real" neural networks. To achieve this, we have changed the network structure to be a random graph, and then computed the best weights for the new graph by using a gradient-based algorithm. By a detailed experimental suite, we showed that the new Random-AdNN possesses chaotic and PR properties for different settings.

Acknowledgements: The authors are grateful for the National Natural Science Foundation of China (grant No.61300093) and the Natural Sciences and Engineering Research Council of Canada.

References

1.Qin, K., Oommen, B.J.: Adachi-like chaotic neural networks requiring linear-time computations by enforcing a tree-shaped topology. IEEE Transactions on Neural

Networks **20**(11) (2009) 1797–1809
2.Adachi, M., Aihara, K.: Associative dynamics in a chaotic neural network. Neural Networks **10**(1) (1997) 83–98
3.Calitoiu, D., Oommen, B.J., Nussbaum, D.: Desynchronizing a chaotic pattern recognition neural network to model inaccurate perception. IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics **37**(3) (2007) 692–704
4.Calitoiu, D., Oommen, B.J., Nussbaum, D.: Periodicity and stability issues of a chaotic pattern recognition neural network. Pattern Analysis and Applications **10**(3) (2007) 175–188
5.Qin, K., Oommen, B.J.: Chaotic pattern recognition: The spectrum of properties of the Adachi neural network. In: Lecuture Notes in Computer Science. Volume 5342., Florida, USA (2008) 540–550
6.Chen, L., Aihara, K.: Global searching ability of chaotic neural networks. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications **46**(8) (1999) 974–993
7.Qin, K., Oommen, B.J.: An enhanced tree-shaped adachi-like chaotic neural network requiring linear-time computations. In: Proceedings of CHAOS'09, the 2009 Chaotic Modeling and Simulation International Conference, Chania, Greece (2009) 284–293
8.Qin, K., Oommen, B.J.: Ideal chaotic pattern recognition is achievable: The Ideal-M-AdNN - its design and properties. In: Transactions on Computational Collective Intelligence (2013) 22–51
9.Luo, G.C., Ren, J.S., Qin, K.: Dynamical associative memory: The properties of the new weighted chaotic Adachi neural network. IEICE Transactions on Information and Systems **E95d**(8) (2012) 2158–2162
10.Qin, K., Oommen, B.J.: Networking logistic neurons can yield chaotic and pattern recognition properties. In: IEEE International Conference on Computational Intelligence for Measure Systems and Applications, Ottawa, Canada (2011) 134–139
11.Hiura, E., Tanaka, T.: A chaotic neural network with Duffing's equation. In: Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA (2007) 997–1001
12.Qin, K., Oommen, B.J.: The entire range of chaotic pattern recognition properties possessed by the Adachi neural network. Intelligent Decision Technologies **6**(1) (2012) 27–41
13.Qin, K.: Generic Analysis of Chaotic Neural Networks and Their Applications in Pattern Recognition and Crypto-systems. PhD thesis. (2010) University of Electronic Science and Technology of China, Chengdu, China.
14.Qin, K., Oommen, B.J.: *Chaotic* pattern recognition using the Adachi neural network modified in a random manner. In: Proceedings of CHAOS'13, the 2013 Chaotic Modeling and Simulation International Conference, Istanbul, Turkey (2013) 540–550.