2010 International Symposium on Nonlinear Theory and its Applications
NOLTA2010, Krakow, Poland, September 5-8, 2010

Associative Dynamics of Color Images in a Chaotic Neural Network

Makito Oku† and Kazuyuki Aihara†,‡

†Department of Mathematical Informatics, Graduate School of Information Science and Technology, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
‡Institute of Industrial Science, The University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan
Email: oku@sat.t.u-tokyo.ac.jp

Abstract—
Recently, we have succeeded in numerically simulating a large-scale chaotic neural network with 1 million units. This success will open new possibilities for applications of chaotic neural networks from the viewpoints of academic interest as well as new media art. In this paper, we report a way to deal with color images by using a large-scale chaotic neural network. In the proposed method, color images are converted to binary sequences, modified slightly by inverting some bits, and stored in the network. The results of numerical simulations show that chaotic alternations among stored patterns as well as their reverse patterns can be observed within a certain range of parameters. We also compare four different coding schemes of color information, which changes the appearance of chaotic dynamics.

1. Introduction

The recent development of computing technologies allows us to handle much larger neural networks than before. One of the leading studies in this direction is the "Blue Brain Project" [1]. The first subgoal of this project, that is, the simulation of a single cortical column, has been achieved; in this simulation, 10,000 biologically detailed neuron models are used. On the other hand, Izhikevich has performed a simulation of the entire neocortex and thalamus by using 1 million neuron models that require few computational resources but retain the realistic physiological properties of neurons [2].

Last year, our group succeeded in simulating a large-scale chaotic neural network with 1 million units [3]. A characteristic feature of the chaotic neural network model [4, 5, 6] is that the network's state chaotically wanders among multiple attractor states. The model is used for the phenomenological modeling of the dynamical associative memory or the endogenous perceptual alternation as well as for efficient optimization methods [6, 7, 8, 9]. Although large-scale chaotic neural networks have been investigated in the context of optimization problem solving [10] or electrical circuit implementation [11], no one has investigated a large-scale dynamical associative memory model.

We found in our previous study [3] that the network exhibits chaotic itinerancy among the stored patterns if we preprocess the patterns to modify their statistics. We also found that even an incompletely retrieved image—the network output differs from the correct image at many pixels—can evoke a clear perception of the original image. Because of these features, a large-scale dynamical associative memory model will have new possibilities for applications from the viewpoints of academic interest as well as new media art.

In this study, we store color images in the network and investigate whether chaotic itinerancy occurs. In addition, we are also interested in the appearance of the output of the network. Since an associative memory model can store only binary patterns, the conversion of color images to binary sequences is required. We first attempt the most simple and direct way of conversion and then carry out other three conversion methods.

2. Chaotic Neural Network Model

First, we will explain the chaotic neural network model [4, 5, 6]. In this study, we consider a recurrent neural network with N units. The external input is constant in both the time and the spatial domains. Each unit has two internal state variables η_i and ζ_i and one output variable x_i. If we adopt the vector representation, $\boldsymbol{\eta} = \{\eta_1, \ldots, \eta_N\}^{\mathrm{T}}$, $\boldsymbol{\zeta} = \{\zeta_1, \ldots, \zeta_N\}^{\mathrm{T}}$, and $\boldsymbol{x} = \{x_1, \ldots, x_N\}^{\mathrm{T}}$, the system's dynamics can be described as follows:

$$\boldsymbol{\eta}(t+1) = k_f \boldsymbol{\eta}(t) + W\boldsymbol{x}(t), \tag{1}$$

$$\boldsymbol{\zeta}(t+1) = k_r \boldsymbol{\zeta}(t) - \alpha \boldsymbol{x}(t) + \boldsymbol{a}, \tag{2}$$

$$\boldsymbol{x}(t+1) = \boldsymbol{f}(\boldsymbol{\eta}(t+1) + \boldsymbol{\zeta}(t+1)). \tag{3}$$

Here, W denotes the $N \times N$ weight matrix; $k_f, k_r \in [0, 1]$, the time constants; $\alpha \geq 0$, the strength of the refractoriness; and $\boldsymbol{a}$, which is a vector of constant values $a \geq 0$, the external input. The first internal state variable $\boldsymbol{\eta}$ changes in response to the input from other units. The other internal state variable $\boldsymbol{\zeta}$ depends on the output of each unit. The output $\boldsymbol{x}$ is defined by a nonlinear function of the summation of the internal state variables. Here, $\boldsymbol{f}$ is an operation that applies the sigmoid function below to each element of the argument vector:

$$f(y) = \frac{1}{1 + \exp(-y/\epsilon)}. \tag{4}$$

Next, we explain the associative memory model. In general, the associative memory model has two phases: the encoding phase and the retrieval phase. In the encoding phase, K binary patterns $\boldsymbol{s}^1, \ldots, \boldsymbol{s}^K \in \{-1, 1\}^N$ are given. For simplicity, we assume that each pattern contains equal numbers of 1 and -1. Then, the weight matrix is determined by the autocorrelation matrix of the patterns:

$$W = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{s}^k (\boldsymbol{s}^k)^{\mathrm{T}}. \tag{5}$$

In the retrieval phase, one of the stored patterns with a slight perturbation is given as an initial state. Then, the corresponding stored pattern is recovered in finite steps. This phenomenon is called pattern completion. Each stored pattern corresponds to an equilibrium point in the phase space.

The dynamical associative memory model differs from the associative memory model in that the system's state is attracted to a stored pattern for a short period, but leaves the pattern after a while, and then visits other patterns. The chaotic neural network model exhibits such behavior in some parameter regions. Neither the order of visiting nor the duration of each stay is predictable although the system's dynamics is deterministic. This phenomenon is called chaotic itinerancy.

Many previous works on the dynamical associative memory model use approximately 100 units. In order to perform large-scale simulations, one of the major problems is the requirement of a considerably large amount of memory capacity for representing the weight matrix, which increases in $\mathrm{O}(N^2)$. However, the all-to-all connection regime used in the associative memory model has high redundancy; even if we remove some of the network connections, the qualitative property of the dynamics can be retained. Therefore, we use a partially connected network in our simulations. The weight matrix $W = \{w_{ji}\}$ is changed as follows:

$$w_{ji} = \begin{cases} \frac{1}{K} \sum_{k=1}^{K} s_j^k s_i^k & e_{ji} \in E \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

where e_{ji} denotes a connection from unit i to unit j, and E denotes the set of all the connections. Notice that if K is even, we can get rid of the connections whose values are 0.

3. Conversion of Color Images to Binary Sequences

In our previous study [3], binary images of 1,000 × 1,000 pixels were transformed to 10^6-dimensional vectors by concatenating all rows and stored in the network. Although binary images are useful for investigating the dynamical associative memory model, applications of the model to multivariate images such as gray-scale images or color images should be appreciated from a practical point of view. Henceforth, we only consider color image processing since the same techniques as those proposed here can be applied to gray-scale images in a straightforward manner.

The basic color image representation is *RGB* (red, green, and blue). *RGB* values are normally represented by integer values from 0 to 255. Each value is encoded in 8 bits; 24 bits represent the complete information of a pixel. By concatenating these 24-bit sequences, we can directly convert the 24-bit *RGB* color images to binary sequences of length 24 × (the number of pixels).

The obtained binary sequences of 1 and 0 are transformed to those of 1 and -1, and then their statistics are adjusted by inverting the minimum number of bits (see Table 1). This preprocessing balances the numbers of 1 and -1 in each pattern as well as equalizes the amount of overlaps among patterns.

In addition to the abovementioned conversion method, we investigated two other color spaces. First, we investigated *YIQ*, which is used by the TV system in the U.S. and Japan. The *YIQ* values of an image are obtained from the *RGB* values of the image by using the linear transformation below [12]:

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.2990 & 0.5870 & 0.1140 \\ 0.5957 & -0.2745 & -0.3213 \\ 0.2115 & -0.5226 & -0.3111 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}, \tag{7}$$

where *RGB* values are rescaled from 0 to 1. $Y \in [0, 1]$, $I \in [-0.5958, 0.5957]$, and $Q \in [-0.5226, 0.5226]$. The first component Y provides the luminance value, and the other two components describe the chrominance information.

In our simulations, the *YIQ* values that are converted from the original *RGB* values are rescaled from 0 to 255 and digitized. Then, we can obtain a 24-bit representation per pixel as in the case of *RGB* images.

Next, we investigated *HSV* (hue, saturation, and value), also known as *HSB* (hue, saturation, and brightness), which is commonly used in image processing because it is more natural to human vision. In the determination of *HSV*, we consider a cylindrical coordinate because hue is a circular quantity. Let $M = \max(R, G, B), m = \min(R, G, B)$, and $C = M - m$. If $R \neq G \neq B$ holds (thus $M, C \neq 0$), the *HSV* color space is defined as follows [13]:

$$H = \begin{cases} 60°(G - B)/C \bmod 360° & (M = R) \\ 60°(B - R)/C + 120° & (M = G) \\ 60°(R - G)/C + 240° & (M = B) \end{cases}, \tag{8}$$

$$S = C/V, \tag{9}$$

$$V = M. \tag{10}$$

In addition, $S = 0$ if $C = 0$, $V \neq 0$, and $V = 0$ if $M = 0$. $H \in [0°, 360°)$, $S \in [0, 1]$, and $V \in [0, 1]$. Usually a certain value is assigned if a value is undefined ($H = 0$ if $C = 0$, for example). As in the case of the *YIQ* values, *HSV* values are rescaled from 0 to 255 and digitized in our simulations in order to obtain a binary representation.

We also investigated another encoding method. The color space is *RGB*, but gray code is used instead of binary

Table 1: List of adjusted statistics of the stored patterns

Statistics	Target value
$\sum_{i=1}^{N} s_i^k$	0
$\sum_{i=1}^{N} s_i^k s_i^l \quad (k \neq l)$	$0.08N$
$\sum_{i=1}^{N} s_i^k s_i^l s_i^m \quad (k \neq l \neq m)$	$-0.08N$

code for transforming the 0–255 integer values into 8-bit binary sequences. In gray code, every pair of successive values differs in only one bit. A binary code representation of an integer n can be converted to a gray code representation by $n \oplus \lfloor n/2 \rfloor$, where $\oplus$ denotes an XOR operation.

Network output can be decoded in the opposite manner. If the recovered RGB values are outside the specified range, we reset the values larger than 255 to 255 and the negative values to 0.

4. Simulations and Results

In the following simulations, we set $k_f = 0.8$, $k_r = 0.9$, $a = 6.4$, $\alpha = 12$, and $\epsilon = 0.015$. Each unit receives input from 100 units selected at random. These connections include those of value 0, which are removed before carrying out the simulations. As an initial condition, η_i takes a random value that is uniformly distributed in $[0, 1]$, and ζ_i is set to 0.

The source code of the program is written in the C programming language with Message Passing Interface (MPI). The program is run on a cluster of eight Linux server machines that have a single 3.2–3.6 GHz processor and 2.0 GB RAM each. It takes approximately 1 s to compute one step of simulation.

First, we use four color images represented in RGB space, as shown in Fig. 1, and convert them to binary sequences by using binary code. All images have 256×256 pixels. The RGB values range from 0 to 255. The number of units, or the length of the binary sequences, is 1,572,864.

After the preprocessing, 1.1–3.9% (Ave. 2.6%) of the bits in each binary sequence are inverted. The root-mean-square (RMS) error per color component of a pixel is 0.67.

Figure 2 shows a typical time series of the decoded network output. The four stored patterns as well as their reverse patterns are retrieved alternately. A few cycles of oscillation between a stored pattern and its reverse one is also observed.

To analyze the time series of the network output $\boldsymbol{x}(t)$, we calculate the Hamming distances between the digitized network output and the four binary sequences (see Fig. 3). The downward and upward peaks correspond to the retrievals of the stored patterns and the reverse ones, respectively. A similar chaotic behavior is observed even in the run of 10^5 steps. The maximum Lyapunov exponent is estimated to be 0.67.

Next, we apply the other three coding schemes: YIQ space with binary code, HSV space with binary code, and RGB space with gray code. After preprocessing, the RMS error is 1.58, 1.23, and 0.70 for the YIQ, HSV, and gray code cases, respectively. A chaotic behavior similar to the previous case is observed in all the three cases. The appearance of the retrieved images decoded from the network output is also similar to that in the previous case for the stored patterns, but qualitatively different for the reverse patterns except in the YIQ case (see Fig. 4).

5. Discussions

The results of numerical simulations, as shown in Figs. 2 and 3, confirmed that chaotic alternations among stored patterns as well as their reverse patterns can be observed if we store color images in the network. The positive maximum Lyapunov exponent is another evidence for the existence of chaos, and the lifetime of chaotic behavior in the considered network appeared to be significantly longer than that of a smaller network [6].

Figure 1: Original color images for stored patterns. From left to right, “Mandrill”, “Lena”, “Peppers”, and “Tree”.

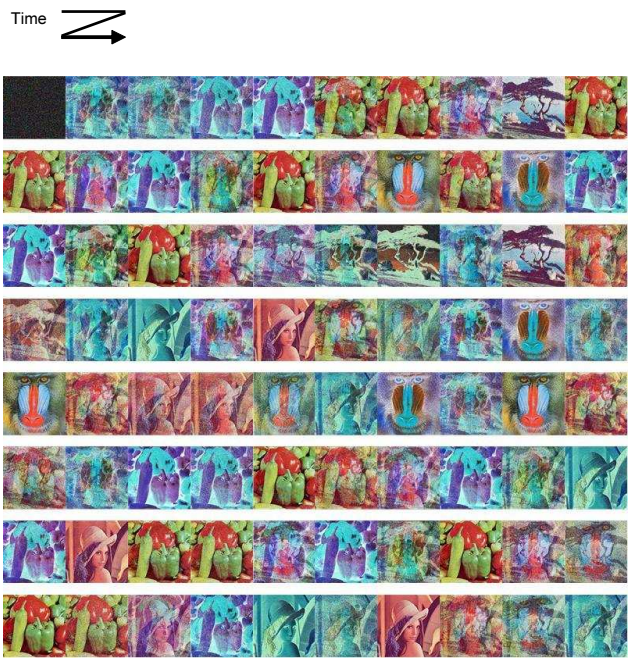

Figure 2: An example of time series of the decoded network output displayed with 10 step intervals.

In addition, it appeared that the color images decoded from the network output, as shown in Fig. 2, can evoke a clear perception of the original images. These observations are also consistent with the results of our previous work in which binary images were used as stored patterns [3].

We found that the use of different color coding schemes can change the appearance of chaotic dynamics, as shown in Fig. 4. This would be because the bit-wise reverse operation may have a qualitatively different effect under different coding schemes in general case.

The reason why YIQ does not change the appearance can be explained as follows. To rescale the YIQ values obtained by using Eq. 7, we multiply these values with a scaling matrix and add a translation vector. The reverse operation applied to the rescaled values has the same effect as in RGB space. The same is true for few other linear transformations of RGB space such as YUV and $I_1I_2I_3$. However, a linear transformation in general does not have such a property.

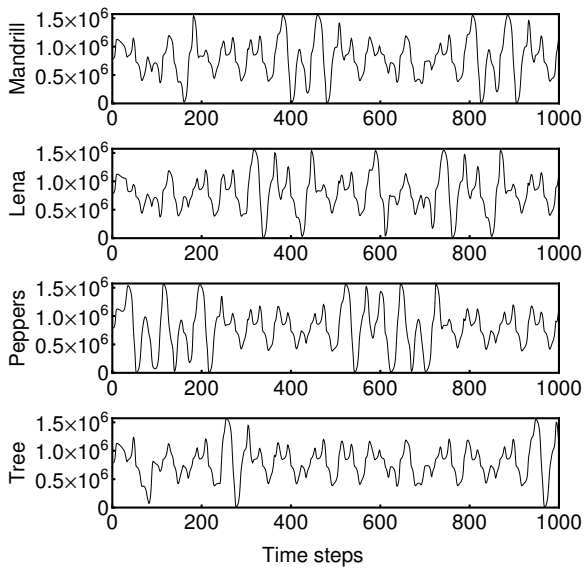

Figure 3: The Hamming distances between the digitized network output and the four binary sequences.

Figure 4: Reverse patterns for different encoding schemes. From left to right, RGB with binary code (used in Figs. 2 and 3), YIQ with binary code, HSV with binary code, and RGB with gray code.

6. Conclusion

In conclusion, we stored 24-bit RGB color images of 256×256 pixels in a large-scale chaotic neural network with approximately 1.6 million units. To convert the color images represented by integer values from 0 to 255, we used an 8-bit binary representation. We observed chaotic alternations among the stored images as well as their reverse images. We also found that the use of different color coding schemes could change the appearance of chaotic dynamics. Finally, as the future work, we plan to store as many images as possible in the network.

Acknowledgments

This research is partially supported by the Japan Society for the Promotion of Science, a Grant-in-Aid for JSPS Fellows (21·937).

References

[1] H. Markram, *Nat. Rev. Neurosci.*, **7**:153–160, 2006.

[2] E. M. Izhikevich and G. M. Edelman, *Proc. Natl. Acad. Sci. U.S.A.*, **105**:3593–3598, 2008.

[3] M. Oku, K. Iwayama, K. Tokuda, H. Suzuki, and K. Aihara, *IEICE Technical Report*, **109**(269):55–59, 2009 (in Japanese).

[4] K. Aihara, in *Bifurcation phenomena in nonlinear systems and theory of dynamical systems*, edited by H. Kawakami (World Scientific, Singapore, 1990) pp. 143–161 .

[5] K. Aihara, T. Takabe, and M. Toyoda, *Physics Letters A*, **144**(6-7):333–340, 1990.

[6] M. Adachi and K. Aihara, *Neural Netw*, **10**:83–98, 1997.

[7] L. Chen and K. Aihara, *Neural Netw.*, **8**(6):915–930, 1995.

[8] N. Nagao, H. Nishimura, and N. Matsui, *Neural Processing Letters*, **12**(3):267–276, 2000.

[9] Y. Kakimoto and K. Aihara, *New Mathematics and Natural Computation*, **5**(1):123–134, 2009.

[10] M. Hasegawa, T. Ikeguchi, and K. Aihara, *Neural Netw.*, **15**:271–283, 2002.

[11] Y. Horio, K. Aihara, and O. Yamamoto, *IEEE Trans. Neural Netw.*, **14**(5)1393–1404, 2003.

[12] P. Shih and C. Liu, *Int. J. Pattern Recogn. Artif. Intell.*, **19**(7):873–894, 2005.

[13] A. Smith, *ACM SIGGRAPH Computer Graphics*, **12**(3):12–19, 1978.