International Symposium on Nonlinear Theory and its Applications
Xi'an, PRC, October 7–11, 2002

A Solution of the Binding Problem for Unstored Patterns with Chaotic Neurodynamics

Yuko OSANA† and Kazuyuki AIHARA‡

† Department of Information Technology
Tokyo University of Technology
1404–1 Katakura-cho, Hachioji-shi,
Tokyo, 192-0982, Japan
TEL: +81-426-37-2111, FAX:+81-426-37-2118
E-mail: osana@cc.teu.ac.jp

‡ Department of Complexity Science and Engineering, Graduate School of Frontier Sciences,
University of Tokyo
7–3–1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
TEL: +81-3-5841-6910, FAX: +81-3-5841-8594
E-mail: aihara@sat.t.u-tokyo.ac.jp

Abstract

In this paper, we propose a chaotic associative memory for binding problem and apply it for unstored patterns. In our previous work[1], we have proposed the chaotic associative memory for binding problem and showed that it can bind the stored patterns. In this research, we apply the proposed model not only to stored patterns but also to unstored patterns. The proposed model is based on chaotic neural networks and the bi-directional theory.

1. Introduction

Since Hubel and Wisel's physiological experiments of visual cortex in 1960's, a lot of physiological facts on visual system have been found. In the fields of computational theory of brain and artificial neural networks, a lot of models of visual system have been proposed.

Although a lot of models of visual system have been proposed , how to bind the information which are once separated as elements is still an unsolved problem. This is called "binding problem". Some hypothesis have been proposed to solve this problem, such as the synchronous firing theory, the dynamical cell assembly hypothesis, the bi-directional theory and the feature integration theory. Based on the above theories, some models have been proposed, such as the neocognitron selective attention model, the model based on the dynamical cell assembly hypothesis and the model based on the feature integration theory.

On the other hand, chaotic behavior of the real neurons has been observed in various physiological experiments, recently. In particular, it is considered that chaos plays an important role in memory and learning in a human brain. In order to mimic the real neurons, a chaotic neuron model has been proposed. It is known that the dynamic (chaotic) association is realized in the associative memories composed of the chaotic neurons.

In our previous work[1], we have proposed the chaotic associative memory for binding problem and showed that it can bind the stored patterns. In this paper, based on the model in ref.[1], we propose a chaotic associative memory for binding problem and apply it for unstored patterns. The proposed model is based on chaotic neural networks and the bi-directional theory.

2. Bi-directional Theory

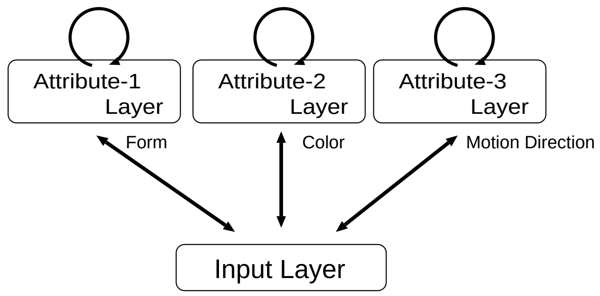
The bi-directional theory was presented by Kawato *et al.*[2]. This theory is based on the fact that bi-directional connections between regions exist in the cortex. According to this theory, information of objects processed in different higher modules return to the primary map through backward connections. In this way, higher modules are connected through the primary region instead of direct connections, and thus may give a solution to the binding problem.

3. Chaotic Associative Memory for Binding Problem

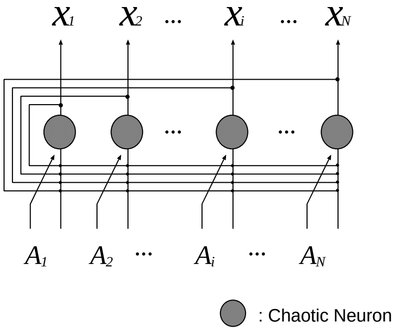
3.1. Structure of Proposed Model

Figure 1 shows a structure of the proposed model. As shown in Fig.1, the proposed model has an Input Layer and some Attribute Layers. Each layer consists of Chaotic Associative Memory (CAM)[3]. The CAM is composed of chaotic neurons[4], and can separate superimposed patterns and deal with many-to-many associations.

In the proposed model, the dynamics of the ith neuron



(a) Proposed Model



(b) Chaotic Associative Memory

Figure 1: Structure of Proposed Model.

in the Input Layer is given as follows:

$$\begin{aligned} x_i^{IN}(t+1) &= f\Bigg[V\sum_{d=0}^{t} k_e^d A_i(t-d) \\ &+ \sum_{j=1}^{N(0)}\sum_{d=0}^{t} k_m^d v_{ij}^{IN} x_j^{IN}(t-d) \\ &+ \sum_{l=1}^{L}\sum_{j=1}^{N(l)}\sum_{d=0}^{t} k_m^d w_{ij}^{IN-AT(l)} x_j^{AT(l)}(t-d) \\ &- \alpha\sum_{d=0}^{t} k_r^d x_i^{IN}(t-d) + a_i^{IN}\Bigg] \end{aligned} \qquad (1)$$

where $x_i^{IN}(t)$ is the output of the ith neuron in the Input Layer at the time t, V is the connection weight from an external input to a neuron in the Input Layer, $A_i(t)$ is strength of the ith external input at the time t, $N(0)$ is the number of neurons in the Input Layer, v_{ij}^{IN} is the connection weight between the ith neuron and the jth neuron in the Input Layer, L is the number of Attribute Layers, $N(l)$ is the number of neurons in the lth Attribute Layer, $w_{ij}^{IN-AT(l)}$ is the connection weight between the ith neuron in the Input Layer and the jth neuron in the lth Attribute Layer, $x_j^{AT(l)}(t)$ is the output of the jth neuron in the lth Attribute Layer, and a_i^{IN} is the bias of the ith neuron in the Input Layer, which value is fixed to 0 in this paper, k_e, k_m and k_r are the damping factors of external input, mutual interaction and refractoriness respectively, and α is the scaling factor of the refractoriness. $f(\cdot)$ is the following output function:

$$f(u) = \tanh(u/\varepsilon) \qquad (2)$$

where ε is the steepness parameter.

The dynamics of the jth neuron in the lth Attribute Layer is represented by the following equation:

$$\begin{aligned} x_j^{AT(l)}(t+1) &= f\Bigg[\sum_{i=1}^{N(l)}\sum_{d=0}^{t} k_m^d v_{ij}^{AT(l)} x_i^{AT(l)}(t-d) \\ &+ \sum_{i=1}^{N(0)}\sum_{d=0}^{t} k_m^d w_{ij}^{IN-AT(l)} x_i^{IN}(t-d) \\ &- \alpha\sum_{d=0}^{t} k_r^d x_j^{AT(l)}(t-d) + a_j^{AT(l)}\Bigg] \end{aligned} \qquad (3)$$

where $v_{ij}^{AT}(l)$ is the connection weight between the ith neuron and the jth neuron in the lth Attribute Layer and $a_j^{AT(l)}$ is the bias of the jth neuron in the $l$the Attribute Layer, which value is fixed to 0 in this paper.

3.2. Learning Process

Here, we explain the learning process of the proposed model. The proposed model is trained by Hebbian Learning, Quick Learning algorithm[5] which has been proposed to improve the storage capacity of BAM and so on as same as the conventional associative memories. Associative memories trained by Hebbian Learning or Quick Learning algorithm can not memorize and recall a training set including common terms because the stored common data cause superimposed patterns.

In order to memorize a training set including common terms, we must convert it into a form which doesn't include any common terms. For this purpose, each training data is memorized together with its own contextual information in the proposed model.

The learning process of the proposed model has two phases; (1) learning of connection weights for auto-associations in each layer and (2) learning of connection weights for hetero-associations between the Input Layer and each Attribute Layer.

Step1: Learning of connection weights for auto-associations

In Step1, the connection weights for auto-associations in each layer are trained. As for the training set shown in Table 1, for example, "*circle*", "*triangle*" and "*rectangle*" are stored in

Table 1: An Example of Stored Data.

	Input Layer	Attribute-1 (Form)	Attribute-2 (Color)	Attribute-3 (Direction)
data-1	pattern 1	circle	red	left
data-2	pattern 2	circle	green	right
data-3	pattern 3	circle	blue	up
data-4	pattern 4	circle	red	down
data-5	pattern 5	triangle	green	left
data-6	pattern 6	triangle	blue	right
data-7	pattern 7	triangle	red	up
data-8	pattern 8	triangle	green	down
data-9	pattern 9	triangle	blue	left
data-10	pattern 10	rectangle	red	right
data-11	pattern 11	rectangle	green	up
data-12	pattern 12	rectangle	blue	down

the Attribute-1 Layer (Form Layer), "*red*", "*green*" and "*blue*" are stored in the Attribute-2 Layer (Color Layer), "*left*", "*right*", "*up*" and "*down*" are stored in the Attribute-3 Layer (Motion Direction Layer), and patterns associated with attribute patterns are stored in the Input Layer.

Step2: Learning of connection weights for hetero-associations

In Step2, the connection weights between the Input Layer and each Attribute Layer are trained. As for the training set shown in Table 1, the connection weights are trained using the following training data.

Input Layer $\leftrightarrow$ Attribute-1 Layer

$\{(pattern1,\ circle),\ (pattern2,\ circle),\ \cdots,(pattern12,\ rectangle)\}$

Input Layer $\leftrightarrow$ Attribute-2 Layer

$\{(pattern1,\ red),\ (pattern2,\ green),\ \cdots,\ (pattern12,\ blue)\}$

Input Layer $\leftrightarrow$ Attribute-3 Layer

$\{(pattern1,\ left),(pattern2,\ right),\ \cdots,(pattern12,\ down)\}$

Here, $patternM$ is the pattern corresponding to the Mth data. For example, $pattern1$ is the pattern corresponding to $\{circle, red, left\}$, and $pattern2$ is that corresponding to $\{circle, green, right\}$ and so on.

3.3. Recall Process

In the recall process of the proposed model, only the patterns without contextual information are given to the network because we can assume that contextual information is usually unknown except for designer. In the proposed model which stores the above training set, when a pattern associated with some attribute patterns is given to the Input Layer, the corresponding attribute patterns are recalled in Attribute Layers, because of searching dynamics of the chaotic neural network anchored by the input pattern. When a superimposed pattern composed of patterns is given to the Input Layer, the corresponding attribute patterns are also successively recalled according to temporal separation by dynamics of the network.

When a superimposed patterns including unstored patterns (for example, the pattern corresponding (*circle*, *red*, *right*)) is given to the Input Layer, we can expect that the corresponding attribute patterns can be separated and recalled according to recall ability of associative memory.

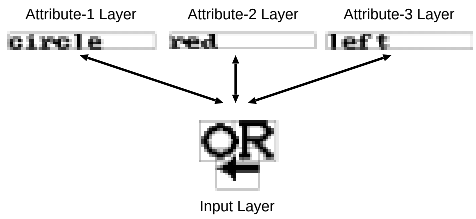


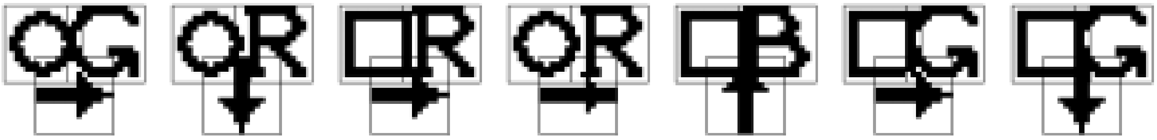
Figure 2: An Example of Stored Data (data-1).

4. Computer Simulation Results

The computer simulations were carried out using the training set shown in Table 1 and Fig. 2 (binary data).

4.1. Result when an Unstored Pattern was Given

We examined the recall ability of the proposed model for stored and unstored patterns. For example, when the pattern corresponding to "*circle*", "*red*" and "*right*" was given to the Input Layer as an external input, although the proposed model does not store the pattern, it could recall the desired patterns "*circle*", "*red*"



(a) pattern 2 (b) pattern 4 (c) pattern 10 (d) pattern A (e) pattern B (f) pattern C (g) pattern D

pattern A : *circle, red, right* pattern B : *rectangle, blue, up*
pattern C : *rectangle, green, right* pattern D : *rectangle, green, down*

Figure 3: Recalled patterns in Input Layer.

Table 2: Recalled number of times for superimposed pattern during $t = 0 \sim 49$.

Input Pattern		Recalled Pattern		
		Desired Patterns		Other Patterns
Pattern (1)	Pattern (2)	Pattern (1)	Pattern (2)	
pattern 1 (*circle, red, left*)	pattern 11 (*rectangle, green, up*)	12	8	0
pattern 1 (*circle, red, left*)	pattern B (*rectangle, blue, up*)	21	6	0
pattern 2 (*circle, green, right*)	pattern C (*rectangle, green, right*)	15	8	0
pattern A (*circle, red, right*)	pattern D (*rectangle, green, down*)	14	8	2*

$*$ (*circle, green, right*)

and "*right*" in the Attribute Layers. This is because that the proposed model stores the patterns which are similar to the inputted pattern such as pattern 2 (for "*circle*", "*green*", "*right*"), pattern 4 (for "*circle*", "*red*", "*down*") and pattern 10 (for "*rectangle*", "*red'*", "*right*"). Figure 3 shows the recalled patterns in the Input Layer. The pattern shown in Fig. 3(d) is not correct, but is similar to the correct pattern, so the proposed model could recall the correct pattern in the Attribute Layers.

4.2. Result when a Superimposed Pattern Composed of Two Patterns was Given

Table 2 shows the result when a superimposed pattern composed of two patterns was given. In Table 2, the patterns A∼D show unstored patterns. As shown in Table 2, the proposed model could recall desired patterns even if the superimposed pattern including unstored patterns.

5. Conclusions

In this paper, we have proposed the chaotic associative memory for binding problem and apply it for unstored patterns. The proposed model is based on chaotic neural networks and the bi-directional theory. We carried out computer simulations and showed that the binding can be realized in the proposed model.

References

[1] Y. Osana and K. Aihara: "A solution of the binding problem with chaotic neurodynamics," NOLTA, Zao, 2001.

[2] M. Kawato, H. Hayakawa and T. Inui: "A forward-inverse optic model of reciprocal connections between visual areas," Network: Computation in Neural Systems, 4, pp. 415–422, 1993.

[3] Y. Osana and M. Hagiwara, "Separation of superimposed pattern and many-to-many associations by chaotic neural networks", IJCNN, Anchorage, **1**, pp. 514-519, 1998.

[4] K. Aihara, T. Takabe and M. Toyoda, "Chaotic neural networks," Physics Letters A, **144**, No. 6,7, pp. 333–340, 1990.

[5] M. Hattori, M. Hagiwara and M.Nakagawa, "Quick learning for bidirectional associative memory," IEICE Japan, **E77-D**, No. 4, pp. 385–392, 1994.