We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

14

Improved Chaotic Associative Memory for Successive Learning

Takahiro IKEYA and Yuko OSANA
Tokyo University of Technology
Japan

1. Introduction

Recently, neural networks are drawing much attention as a method to realize flexible information processing. Neural networks consider neuron groups of the brain in the creature, and imitate these neurons technologically. Neural networks have some features, especially one of the important features is that the networks can learn to acquire the ability of information processing.

In the filed of neural network, many models have been proposed such as the Back Propagation algorithm (Rumelhart et al., 1986), the Self-Organizing Map (Kohonen, 1994), the Hopfield network (Hopfield, 1982) and the Bidirectional Associative Memory (Kosko, 1988). In these models, the learning process and the recall process are divided, and therefore they need all information to learn in advance.

However, in the real world, it is very difficult to get all information to learn in advance. So we need the model whose learning and recall processes are not divided. As such model, Grossberg and Carpenter proposed the Adaptive Resonance Theory (ART) (Carpenter & Grossberg, 1995). However, the ART is based on the local representation, and therefore it is not robust for damage. While in the field of associative memories, some models have been proposed (Watanabe et al., 1995; Osana & Hagiwara, 1999; Kawasaki et al., 2000; Ideguchi et al., 2005). Since these models are based on the distributed representation, they have the robustness for damaged neurons. However, their storage capacity is very small because their learning processes are based on the Hebbian learning. In contrast, the Hetero Chaotic Associative Memory for Successive Learning with give up function (HCAMSL) (Arai & Osana, 2006) and the Hetero Chaotic Associative Memory for Successive Learning with Multi-Winners competition (HCAMSL-MW) (Ando et al., 2006) have been proposed in order to improve the storage capacity.

In this research, we propose an Improved Chaotic Associative Memory for Successive Learning (ICAMSL). The proposed model is based on the Hetero Chaotic Associative Memory for Successive Learning with give up function (HCAMSL) (Arai & Osana, 2006) and the Hetero Chaotic Associative Memory for Successive Learning with Multi-Winners competition (HCAMSL-MW) (Ando et al., 2006). In the proposed ICAMSL, the learning process and recall process are not divided. When an unstored pattern is given to the

network, the ICAMSL can learn the pattern successively. Moreover, the storage capacity of the proposed ICAMSL is larger than that of the conventional HCAMSL/HCAMSL-MW.

2. Chaotic Neural Network

A neural network composed of the chaotic neurons is called a chaotic neural network (Aihara et al., 1990).
The dynamics of the chaotic neuron i in the neural network composed of N chaotic neurons is represented by the following equation:

$$x_i(t+1) = f\left[\sum_{j=1}^{M} v_{ij} \sum_{d=0}^{t} k_s^d A_j(t-d) + \sum_{j=1}^{N} w_{ij} \sum_{d=0}^{t} k_m^d x_j(t-d) - \alpha \sum_{d=0}^{t} k_r^d x_i(t-d) - \theta_i\right] \quad (1)$$

where $x_i(t)$ shows the output of the neuron i at the time t, M is the number of the external inputs, v_{ij} is the connection weight from the external input j to the neuron i, $A_j(t)$ is the strength of the external input j at the time t, w_{ij} is the connection weight between the neuron i and the neuron j, α is the scaling factor of the refractoriness, k_s, k_m and k_r are the damping factors and θ_i is the threshold of the neuron i. $f(\cdot)$ is the following output function:

$$f(u) = \frac{1}{1 + \exp(-u/\varepsilon)} \quad (2)$$

where ε is the steepness parameter. It is known that dynamic associations can be realized in the associative memories composed chaotic neurons.

3. Multi Winners Self-Organizing Neural Network

Here, we briefly explain the Multi Winners Self-Organizing Neural Network (MWSONN) (Huang & Hagiwara., 1997) which is used in the proposed model.
Figure 1 shows the structure of the MWSONN. The MWSONN consists of the Input/Output Layer and the Distributed Representation Layer, and each layer has 2- dimensional structure. In the learning process of the MWSONN, the distributed representation patterns corresponding to input analog patterns are generated by the multi winners self-organizing process, and the connection weights between layers are trained by the error correction learning. In the recall process of the MWSONN, when a stored pattern or a part of the stored patterns is given to the Input/Output Layer, the corresponding distributed representation pattern appears in the Distributed Representation Layer, and then the corresponding analog pattern appears in the Input/Output Layer.

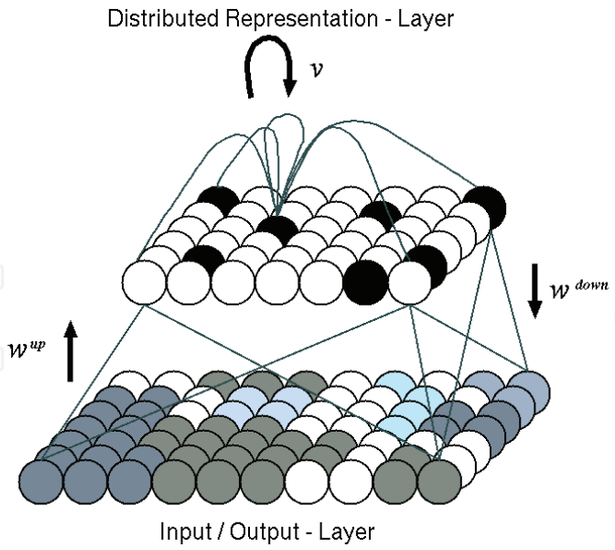


Fig. 1. Structure of MWSONN.

4. Improved Chaotic Associative Memory for Successive Learning

4.1 Outline of ICAMSL

Here, we explain the outline of the proposed Improved Chaotic Associative Memory for Successive Learning (ICAMSL). The proposed ICAMSL has three stages; (1) Pattern Search Stage, (2) Distributed Pattern Generation Stage and (3) Learning Stage.

When an unstored pattern set is given to the network, the proposed ICAMSL distinguishes an unstored pattern set from stored patterns and can learn the pattern set successively. When a stored pattern set is given, the ICAMSL recalls the patterns. When an unstored pattern set is given to the network, the ICAMSL changes the internal pattern for input pattern set by chaos and presents other pattern candidates (we call this the Pattern Search Stage). When the ICAMSL can not recall the desired patterns, the distributed pattern is generated by the multi-winners competition (Huang & Hagiwara., 1997) (Distributed Pattern Generation Stage), and it learns the input pattern set as an unstored pattern set (Learning Stage). Figure 2 shows the flow of the proposed ICAMSL.

4.2 Structure of ICAMSL

The proposed ICAMSL is a kind of the hetero-associative memories. Figure 3 shows a structure of the ICAMSL. This model has two layers; an Input/Output Layer (I/O Layer) composed of conventional neurons and a Distributed Representation Layer (DR Layer) composed of chaotic neurons (Aihara et al., 1990). In this model, there are the connection weights between neurons in the Distribute Representation Layer and the connection weights between the Input/Output Layer and the Distributed Representation Layer.

As shown in Fig.3, the Input/Output Layer has plural parts. The number of parts is decided depending on the number of patterns included in the pattern set. In the case of Fig.3, the Input/Output Layer consists of P parts corresponding to the patterns $1 \sim P$.

In this model, when a pattern set is given to the Input/Output Layer, the internal pattern corresponding to the input patterns is formed in the Distributed Representation Layer. Then,

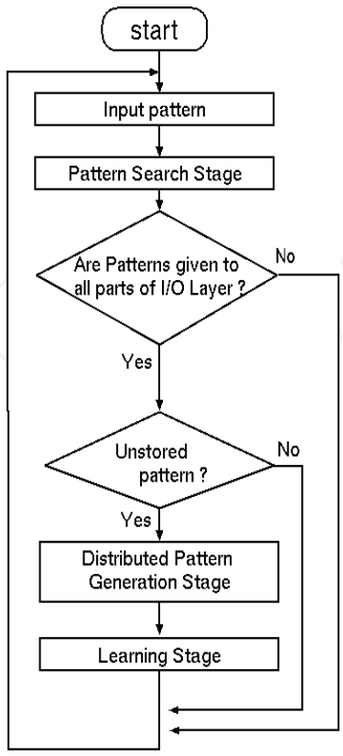


Fig. 2. Flow of Proposed ICAMSL.

in the Input/Output Layer, an output pattern set is generated from the internal pattern. The ICAMSL distinguishes an unstored pattern set from stored patterns by comparing the input patterns with the output pattern.

In this model, the output of the neuron i in the Distributed Representation Layer at the time $t+1$, $x_i(t+1)$ is given by the following equations.

$$x_i(t+1) = \phi\left[\xi_i(t+1) + \eta_i(t+1) + \zeta_i(t+1)\right] \tag{3}$$

$$\begin{aligned} \xi_i(t+1) &= \sum_{j=1}^{M} v_{ij} \sum_{d=0}^{t} k_s^d A_j(t-d) \\ &= k_s \xi_i(t) + \sum_{j=1}^{M} v_{ij} A_j(t) \end{aligned} \tag{4}$$

$$\begin{aligned} \eta_i(t+1) &= \sum_{j=1}^{N} w_{ij} \sum_{d=0}^{t} k_m^d x_j(t-d) \\ &= k_m \eta_i(t) + \sum_{j=1}^{N} w_{ij} x_j(t) \end{aligned} \tag{5}$$

$$\begin{aligned}\zeta_i(t+1) &= -\alpha \sum_{d=0}^{t} k_r^d x_i(t-d) - \theta_i \\ &= k_r \zeta_i(t) - \alpha x_i(t) - \theta_i(1-k_r)\end{aligned} \tag{6}$$

In Eqs. $(3)\sim(6)$, M is the number of neurons in the Input/Output Layer, v_{ij} is the connection weight between the neuron j in the Input/Output Layer and the neuron i in the Distributed Representation Layer, N is the number of neurons in the Distributed Representation Layer, $A_j(t)$ is the external input j to the Input/Output Layer at the time t, w_{ij} is the connection weight between the neuron i and the neuron j in the Distributed Representation Layer, $\alpha(t)$ is the scaling factor of the refractoriness at the time t, and k_s, k_m and k_r are the damping factors. $\phi(\cdot)$ is the following output function:

$$\phi(u_i) = \tanh(u_i / \varepsilon) \tag{7}$$

where ε is the steepness parameter.
The output of the neuron j in the Input/Output Layer at the time t, $x_j^{IO}(t)$ is given as follows.

$$x_j^{IO}(t) = \phi^{IO}\left(\sum_{i=1}^{N} v_{ij} x_i^D(t)\right) \tag{8}$$

$$\phi^{IO} = \begin{cases} 1, & u \geq 0 \\ -1, & u < 0 \end{cases} \tag{9}$$

4.3 Pattern Search Stage

In the Pattern Search Stage, when an input pattern set is given, the ICAMSL distinguishes the pattern set from stored patterns. When an unstored pattern set is given, the ICAMSL changes the internal pattern for the input pattern by chaos and presents the other pattern candidates. Until the ICAMSL recalls the desired patterns, the following procedures are repeated. If the ICAMSL can not recall the desired patterns, when the stage is repeated certain times, the ICAMSL finishes the stage.

4.3.1 Pattern Assumption

In the proposed ICAMSL, only when the input patterns are given to all parts of the Input/Output Layer, the patterns are judged. When the input pattern $A(t)$ is similar to the recalled pattern $x^{IO}(t)$, the ICAMSL can assume that input pattern is one of the stored patterns. The ICAMSL outputs the pattern formed by the internal pattern in the Distributed Representation Layer. The similarity $s(t)$ is defined by

$$s(t) = \frac{1}{M} \sum_{j=1}^{M} A_j(t) x_j^{IO}(t). \tag{10}$$

The ICAMSL regards the input patterns as a stored pattern set, when the similarity rate $s(t)$ is larger than the threshold $s^{th}\left(s(t) \geq s^{th}\right)$.

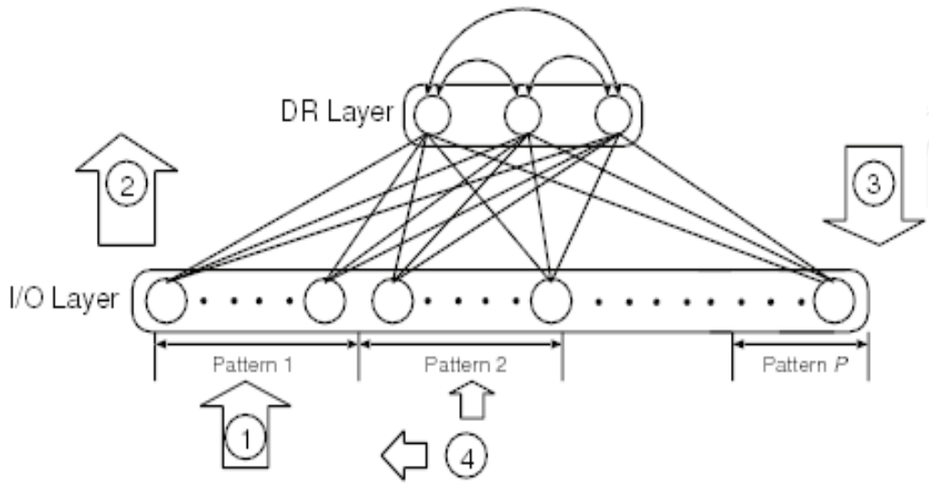


Fig. 3. Structure of Proposed ICAMSL.

4.3.2 Pattern Search

When the ICAMSL assumes that the input patterns are an unstored pattern set, the ICAMSL changes the internal pattern $x^D(t)$ for the input pattern by chaos and presents the other pattern candidates.

In the chaotic neural network, it is known that dynamic association can be realized if the scaling factor of the refractoriness $\alpha(t)$ is suitable (Aihara et al., 1990). Therefore, in the proposed model, $\alpha(t)$ is changed as follows:

$$\alpha(t) = \left((\alpha_{\max}(t) - \alpha_{\min})(1 - s(t)) + \alpha_{\min}\right) / \alpha_{DIV} \tag{11}$$

$$\alpha_{\max}(t) = Mv_{\max} + Nw_{\max} \tag{12}$$

$$v_{\max} = \max\left\{|v_{11}|, \cdots, |v_{ij}|, \cdots, |v_{NM}|\right\} \tag{13}$$

$$w_{\max} = \max\left\{|w_{11}|, \cdots, |w_{ii'}|, \cdots, |w_{NN}|\right\} \tag{14}$$

where $\alpha_{\min}$ is the minimum of α, $\alpha_{\max}(t)$ is the maximum of α at the time t, $s(t)$ is the similarity between the input pattern and the output pattern at the time t (the time when the Pattern Search Stage started), and α_{DIV} is the constant.

4.4 Distributed Pattern Generation Stage

In the Distributed Pattern Generation Stage, the distributed pattern corresponding to the input pattern is generated by the multi-winners competition (Huang & Hagiwara., 1997).

4.4.1 Calculation of Outputs of Neurons in I/O Layer

When the input pattern $A_j(t)$ is given to the Input/Output Layer, the output of the neuron j in the Input/Output Layer x_j^{IO} is given by

$$x_j^{IO} = S_f\left(A_j(t)\right) \tag{15}$$

where $S_f(\cdot)$ is the ramp function and is given by

$$S_f(u) = \begin{cases} u, & u > 0 \\ 0, & u \leq 0 \end{cases}. \tag{16}$$

4.4.2 Calculation of Initial Outputs of Neurons in DR Layer

The output of the neuron i in the Distributed Representation Layer $x_i^{D(0)}$ is calculated by

$$x_i^{D(0)} = C_f\left(\sum_{j=1}^{M} v_{ij} x_j^{IO} - \theta_i\right) \tag{17}$$

where v_{ij} is the connection weight from the neuron j in the Input/Output Layer to the neuron i in the Distributed Representation Layer, θ_i is the threshold of the neuron i in the Distributed Representation Layer and M is the number of neurons in the Input/Output Layer. The output function $C_f(\cdot)$ is given by

$$C_f(u) = \tanh(u / T) \tag{18}$$

where T is the steepness parameter in the sigmoidal function.

4.4.3 Competition between Neurons in DR Layer

The competition dynamics is given by the following equations:

$$x_i^D = C_f(u_i - \theta_i) \tag{19}$$

$$u_i = \sum_{i'=1}^{N} w_{ii'} x_{i'}^D \tag{20}$$

where x_i^D is the output of the neuron i in the Distributed Representation Layer, u_i is the internal state of the neuron i in the Distributed Representation Layer and N is the number of neurons in the Distributed Representation Layer.

4.5 Learning Stage

In the Pattern Search Stage, if the ICAMSL can not recall the desired pattern set, it learns the input pattern set as an unstored pattern set. The Learning Stage has two phases; (1) Hebbian

Learning Phase and (2) anti-Hebbian Learning Phase. If the signs of the outputs of two neurons are same, the connection weight between these two neurons is strengthened. By this learning, the connection weights are changed to learn the input patterns, however the Hebbian learning can only learn a new input pattern set. In the proposed ICAMSL, the anti-Hebbian Learning Phase is employed as similar as the original HCAMSL (Arai & Osana, 2006). In the anti-Hebbian Learning Phase, the connection weights are changed in the opposite direction in the case of the Hebbian Learning Phase. The proposed ICAMSL can learn a new pattern set without destroying the stored patterns by the anti-Hebbian Learning. Figure 4 shows the learning stage of the proposed ICAMSL.

4.5.1 Hebbian Learning Phase

In the Hebbian Learning Phase, until the similarity rate $s(t)$ becomes 1.0, the update of the connection weights is repeated.

The connection weight between the Input/Output Layer and the Distributed Representation Layer v_{ij} and the connection weight in the Distributed Representation Layer $w_{ii'}$ are updated as follows:

$$v_{ij}^{(new)} = v_{ij}^{(old)} + \gamma_v^+ x_i^{D(comp)} A_j(t) \tag{21}$$

$$w_{ii'}^{(new)} = w_{ii'}^{(old)} + \gamma_w^+ x_i^{D(comp)} x_{i'}^{D(comp)} \tag{22}$$

where γ_v^+ is the learning rate of the connection weight v_{ij} in the Hebbian Learning Phase, and γ_w^+ is the learning rate of the connection weight $w_{ii'}$ in this phase.

4.5.2 Give Up Function

When the similarity rate $s(t)$ does not become 1.0 even if the connection weights are updated T_n times, the ICAMSL gives up to memorize the pattern set. If the ICAMSL gives up to learn the pattern set, the anti-Hebbian Learning Phase is not performed.

4.5.3 Anti-Hebbian Learning Phase

The anti-Hebbian Learning Phase is performed after the Hebbian Learning Phase. In this phase, the connection weight v_{ij} and $w_{ii'}$ are changed in the opposite direction in the case of the Hebbian Learning Phase. The anti-Hebbian Learning makes the relation between the patterns is learned without destroying the stored patterns.

In this phase, v_{ij} and $w_{ii'}$ are updated by

$$v_{ij}^{(new)} = v_{ij}^{(old)} + \gamma_v^- x_i^{D(comp)} A_j(t) \tag{23}$$

$$w_{ii'}^{(new)} = w_{ii'}^{(old)} + \gamma_w^- x_i^{D(comp)} x_{i'}^{D(comp)} \tag{24}$$

where $\gamma_v^-\left(\gamma_v^- > \gamma_v^+ > 0\right)$ is the learning rate of the connection weight v_{ij} in the anti-Hebbian Learning Phase, and $\gamma_w^-\left(\gamma_w^- > \gamma_w^+ > 0\right)$ is the learning rate of the connection weight $w_{ii'}$ in this phase.

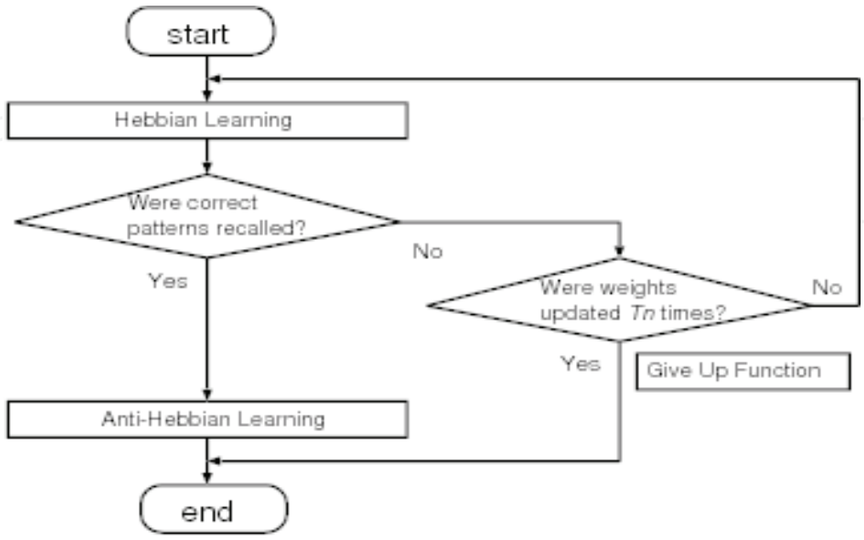


Fig. 4. Learning Stage.

5. Computer Experiment Results

In this section, we show the computer experiment results to demonstrate the effectiveness of the proposed ICAMSL. The computer experiments were carried out under the conditions shown in Table 1.

5.1 Successive Learning and One-to-Many Associations

Figure 5 shows the successive learning and one-to-many associations in the proposed ICAMSL. As seen in Fig.5, the patterns "lion", "mouse" and "penguin" were given to the network at $t = 1$. At $t = 1$, the ICAMSL could not recall the correct patterns because no pattern was stored in the network. During $t = 2 \sim 11$, the ICAMSL changed the internal patterns by chaos and presented the other pattern candidates, however it could not recall the correct patterns. As a result, the ICAMSL regarded the input patterns as unstored patterns, at $t = 13$, the patterns "lion", "mouse" and "penguin" were trained as new patterns. At $t = 14$, the patterns "lion", "mouse" and "duck" were given to the network. At this time, since only the pattern set "lion", "mouse" and "penguin" was memorized in the network, the ICAMSL recalled the patterns "lion", "mouse" and "penguin". During $t = 15 \sim 24$, the ICAMSL changed the internal patterns by chaos and presented the other pattern candidates, however it could not recall the correct patterns. As a result, the ICAMSL regarded the input patterns as unstored patterns, at $t = 28$, the "lion", "mouse" and "duck" were trained as new patterns. At $t = 29$, the patterns "lion" and "mouse" were given to the network, the ICAMSL recalled "lion", "mouse" and "penguin" $(t = 30)$ and "lion", "mouse" and "duck" $(t = 35)$. From these results, we confirmed that the proposed ICAMSL can learn patterns successively and realize one-to-many associations.

Learning Parameters		
the number of pattern searches in Pattern Search Stage		10
initial value of all connection weights		$-1.0 \sim 1.0$
learning rate in Hebbian Learning	γ_v^+, γ_w^+	1.0
learning rate in anti-Hebbian Learning	γ_v^-, γ_w^-	2.0
threshold of similarity rate	s^{th}	1.0
Chaotic Neuron Parameters		
constant for refractoriness	α_{DIV}	25
minimum of scaling facter α	$\alpha_{\min}$	0.0
damping factor	k_s	0.5
damping factor	k_m	0.0
damping factor	k_r	0.95
threshold of neurons	θ	0.0
steepness parameter	ε	1.0
Competition Parameters		
steepness parameter	T	0.0005

Table 1. Experimental Conditions.

5.2 Storage Capacity

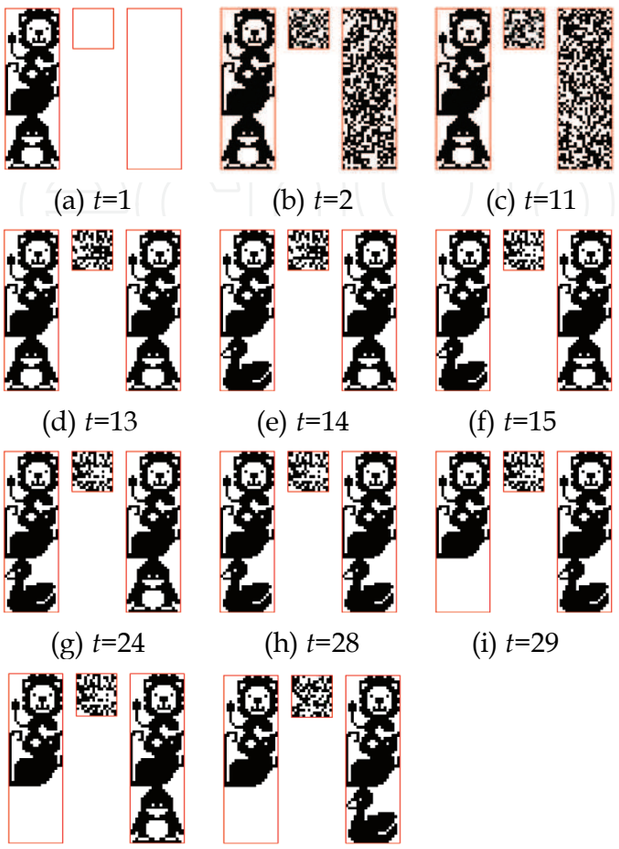
Here, we examined the storage capacity of the proposed ICAMSL. In this experiment, we used the ICAMSL which has 800 neurons (400 neurons for pattern 1 and 400 neurons for pattern 2) in the Input/Output Layer and 225 neurons in the Distributed Representation Layer. We used random patterns to store and Fig.6 shows the average of 100 trials. In this figure, the horizontal axis is the number of stored pattern pairs, and the vertical axis is the perfect recall rate. In this figure, the storage capacities of the model without give up function (HCAMSL-MW) (Ando et al., 2006), the model without the Distributed Pattern Generation Stage (HCAMSL) (Arai & Osana, 2006) and the model without the give up function and the Distributed Pattern Generation Stage are also shown for reference.

From these results, we confirmed that the storage capacity of the proposed ICAMSL is larger than that of the conventional HCAMSL/HCAMSL-MW.

6. Conclusions

In this research, we have proposed the Improved Chaotic Associative Memory for Successive Learning (ICAMSL). The proposed model is based on the Hetero Chaotic Associative Memory for Successive Learning with give up function (HCAMSL) (Arai & Osana, 2006) and the Hetero Chaotic Associative Memory for Successive Learning with Multi-Winners competition (HCAMSL-MW) (Ando et al., 2006). In the proposed ICAMSL, the learning process and recall process are not divided. When an unstored pattern is given to the network, the ICAMSL can learn the pattern successively. We carried out a series of computer experiments and confirmed that the proposed ICAMSL can learn patterns

successively and realize one-to-many associations, and the storage capacity of the ICAMSL is larger than that of the conventional HCAMSL/HCAMSL-MW.



(a) t=1 (b) t=2 (c) t=11

(d) t=13 (e) t=14 (f) t=15

(g) t=24 (h) t=28 (i) t=29

(j) t=30 (k) t=35

Fig. 5. Successive Learning in Proposed Model.

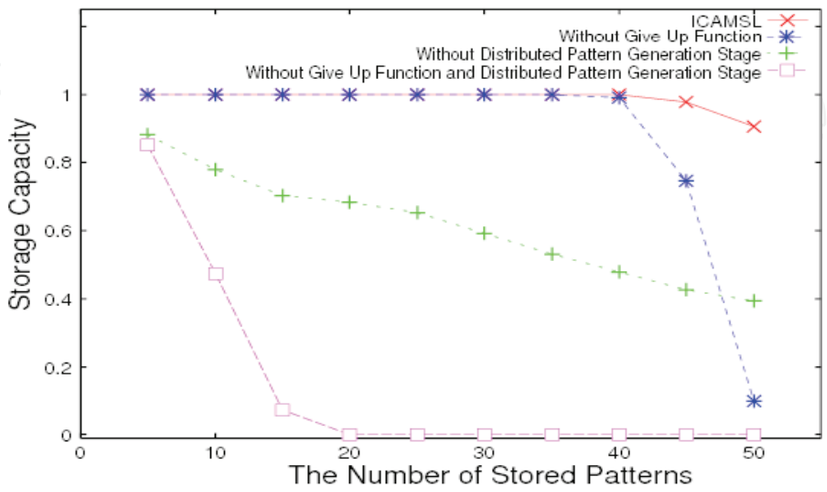


Fig. 6. Storage Capacity.

7. References

K. Aihara, T. Takabe and M. Toyoda. (1990). Chaotic neural networks, Physics Letter A, Vol.144, No.6, 7, pp.333–340

M. Ando, Y. Okuno and Y. Osana. (2006). Hetero chaotic associative memory for successive learning with multi-winners competition, Proceedings of IEEE and INNS International Joint Conference on Neural Networks, Vancouver

T. Arai and Y. Osana. (2006). Hetero chaotic associative memory for successive learning with give up function — One-tomany associations —, Proceedings of IASTED Artificial Intelligence and Applications, Innsbruck

G. A. Carpenter and S. Grossberg. (1995). Pattern Recognition by Self-organizing Neural Networks, The MIT Press

J. J. Hopfield. (1982). Neural networks and physical systems with emergent collective computational abilities, Proceedings of the National Academy of Sciences of the United States of America, 79, pp. 2554–2558

J. T. Huang and M. Hagiwara. (1997). A multi-winners selforganizing neural network, IEEE International Conference on System, Man and Cybernetics, pp. 2499–2504

M. Ideguchi, N. Sato and Y. Osana. (2005). Hetero chaotic associative memory for successive learning and action study of robot, Proceedings of International Symposium on Nonlinear Theory and its Applications, Bruges

N. Kawasaki, Y. Osana and M. Hagiwara. (2000). Chaotic associative memory for successive learning using internal patterns, IEEE International Conference on Systems, Man and Cybernetics

T. Kohonen. (1994). Self-Organizing Maps, Springer

B. Kosko. (1988). Bidirectional associative memories, IEEE Transactions on System, Man and Cybernetics, SMC-18, No. 1, pp. 49–60

Y. Osana and M. Hagiwara. (1999). Successive learning in chaotic neural network, International Journal of Neural Systems, Vol.9, No.4, pp.285–299

D. E. Rumelhart, J. L. McClelland and the PDP Research Group. (1986). Parallel Distributed Processing, Exploitations in the Microstructure of Cognition, Vol.11 : Foundations, The MIT Press

M.Watanabe, K. Aihara and S. Kondo. (1995). Automatic learning in chaotic neural networks, Institute of Electronics, Information and Communication Engineers-A, Vol.J78-A, No.6, pp.686–691

New Developments in Robotics Automation and Control

Edited by Aleksandar Lazinica

ISBN 978-953-7619-20-6
Hard cover, 450 pages
Publisher InTech
Published online 01, October, 2008
Published in print edition October, 2008

This book represents the contributions of the top researchers in the field of robotics, automation and control and will serve as a valuable tool for professionals in these interdisciplinary fields. It consists of 25 chapter that introduce both basic research and advanced developments covering the topics such as kinematics, dynamic analysis, accuracy, optimization design, modelling , simulation and control. Without a doubt, the book covers a great deal of recent research, and as such it works as a valuable source for researchers interested in the involved subjects.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Takahiro Ikeya and Yuko Osana (2008). Improved Chaotic Associative Memory for Successive Learning, New Developments in Robotics Automation and Control, Aleksandar Lazinica (Ed.), ISBN: 978-953-7619-20-6, InTech, Available from: http://www.intechopen.com/books/new_developments_in_robotics_automation_and_control/improved_chaotic_associative_memory_for_successive_learning

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen