A Hierarchical Self-organizing Associative Memory for Machine Learning

Janusz A. Starzyk[1], Haibo He[2], Yue Li[3]

[1] School of Electrical Engineering and Computer Science
Ohio University, OH 45701 USA
[2] Department of Electrical and Computer Engineering
Stevens Institute of Technology, NJ 07030 USA
[3] O_2 Micro Inc., Santa Clara, CA 95054 USA.
{starzyk@bobcat.ent.ohiou.edu, Haibo.He@stevens.edu,
yue.lily.lee@gmail.com}

Abstract. This paper proposes novel hierarchical self-organizing associative memory architecture for machine learning. This memory architecture is characterized with sparse and local interconnections, self-organizing processing elements (PE), and probabilistic synaptic transmission. Each PE in the network dynamically estimates its output value from the observed input data distribution and remembers the statistical correlations between its inputs. Both feed forward and feedback signal propagation is used to transfer signals and make associations. Feed forward processing is used to discover relationships in the input patterns, while feedback processing is used to make associations and predict missing signal values. Classification and image recovery applications are used to demonstrate the effectiveness of the proposed memory for both hetero-associative and auto-associative learning.

Keywords: associative memory, hierarchical structure, self-organization, classification, image recovery.

1 Introduction

Associative memory is of critical importance for machine learning, information representation, signal processing and a wide range of applications. Therefore, it has attracted extensive research in engineering and science. There are two types of associative memories: hetero-associative (HA) memory makes associations between paired patterns, such as words and pictures, while auto-associative (AA) memory associates a pattern with itself, recalling stored patterns from fractional parts of the pattern as in image recovery.

Both types of memories have attracted a significant attention in recent literature. For instance, among HA studies, J. Y. Chang and C. W. Cho proposed adaptive local training rules for second-order asymmetric bidirectional associative memory (BAM) in [1]. Simulation results of this BAM on color graphics adapter (CGA) fonts illus-

trate the effectiveness of this memory. Salih et al. proposed a new approach for bidirectional associative memories (BAM) using feedback neural networks [2]. The perceptron training algorithm was used to solve a set of linear inequalities for the BAM neural network design. In [3], Wang presented a multi-associative neural network (MANN) and showed its application to learning and retrieving complex spatio-temporal sequences. Simulation results show that this system is characterized by fast and accurate learning, and has the ability to store and retrieve a large number of complex sequences of nonorthogonal spatial patterns. Hopfield's paper [4] is a classic reference for auto-associative studies. Since that paper, many research results have been reported. For instance, Vogel presented an algorithm for auto-associative memory in sparsely connected networks [5]. The resulting networks have large information storage capacities relative to the number of synapses per neuron. Vogel et al. derived a lower bound on the storage capacities of two-layer projective networks (P-nets) with binary Hebbian synapses [6]. Recently, Wang et al. proposed an enhanced fuzzy morphological auto-associative memory based on the empirical kernel map [7].

In this paper, we developed a probability based associative memory algorithm and memory architecture that is capable of both hetero-associative (HA) and auto-associative (AA) memory. This paper is organized as follows. In section 2, a new probability based associative learning algorithm is proposed. In section 3, we discuss the network architecture and its associative mechanism. In section 4, classification and image recovery applications are used to illustrate the HA and AA applications of the proposed memory structure. Finally, conclusions are given in section 5.

2 Associative Learning Algorithm

The proposed memory architecture consists of a multilayer array of the processing elements (PE). Its organization follows a general self-organizing learning array concept presented in [8]. Fig. 1 gives the interface model of an individual PE, which consists of two inputs (I_1 and I_2) and one output (O). Each PE stores observed probabilities P_{00}, P_{01}, P_{10} and P_{11}, corresponding to four different combinations of inputs I_1 and I_2 ($\{I_1 I_2\} = \{00\},\{01\},\{10\},\{11\}$), respectively.

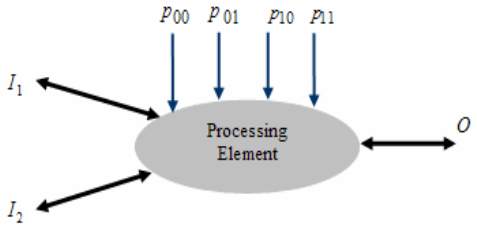

Fig. 1 Individual PE interface model

Fig.2 gives an example of possible distribution of the observed input data points (scaled to the range [0 1]). Probabilities are estimated from $p_{00} = n_{00} / n_{tot}$, $p_{01} = n_{01} / n_{tot}$, $p_{10} = n_{10} / n_{tot}$ and $p_{11} = n_{11} / n_{tot}$, where n_{00}, n_{01}, n_{10} and n_{11} is the number of data points located in $I_1 < 0.5 \,\&\, I_2 < 0.5$, $I_1 < 0.5 \,\&\, I_2 > 0.5$,

$I_1 > 0.5 \& I_2 < 0.5$ and $I_1 > 0.5 \& I_2 > 0.5$, respectively. n_{tot} is the total number of data points defined as $n_{tot} = n_{00} + n_{01} + n_{10} + n_{11}$.

Based on the observed probability distribution p_{00}, p_{01}, p_{10} and p_{11} of an individual PE, each PE decides its output function value F by specifying in its truth table as shown in Table 1.

The output function values f_{00}, f_{01}, f_{10} and f_{11} are decided as follows:

(1) The input, (I_1, I_2), that is associated with the largest probability, $p_{ij}, (i, j = 0,1)$, is assigned a corresponding output function F value of 0.

(2) If the largest probability is less than 0.5, then the input (I_1, I_2), that is associated with smallest probability is also assigned a corresponding F value of 0;

(3) If the sum of the largest and smallest probabilities is less than 0.5, then the input, (I_1, I_2), that is associated with the second-smallest probability $p_{ij}, (i, j = 0,1)$ is also assigned a corresponding F value of 0;

(4) All input combinations not assigned corresponding F value of 0 by the above rules are assigned a corresponding F value of 1.

Table1: Self-determination of function value F

Probability	p_{00}	p_{01}	p_{10}	p_{11}
I_1	0	0	1	1
I_2	0	1	0	1
Function value	f_{00}	f_{01}	f_{10}	f_{11}

The probability that the neuron is active is smaller than 0.5. This type of assignment is motivated by the sparse activity of biological neurons [9]. In addition to biological motivation, lower activities are preferable for efficient power consumption. Probabilities p_{ij} can be efficiently estimated in real time hardware using dynamic probability estimator [10]. Table 2 shows two examples of this self-determination of the function value F.

Table 2: Two examples of setting F value

p_{00}	p_{01}	p_{10}	p_{11}		F			
0.4	0.2	0.3	0.1	⟶	0	1	1	0
0.4	0.05	0.3	0.25	⟶	0	0	1	0

During training, each PE counts its input data points in n_{00}, n_{01}, n_{10} and n_{11} and estimates their corresponding probabilities p_{00}, p_{01}, p_{10} and p_{11}. The objective of the training stage for each PE is to discover the potential relationship between its inputs. This relationship is remembered as the corresponding probabilities and is used to make associations during the testing stage.

Considering the example in Fig. 2, this particular PE finds that most of its input data points are distributed in the lower-left corner ($I_1 < 0.5 \& I_2 < 0.5$). Therefore, if this PE only knows one of the input signal is $I_1 < 0.5$, it will associatively predict that the other signal most likely should also be $I_2 < 0.5$. The developed algorithm

allows all the PEs in the network to make such associations between different input signals.

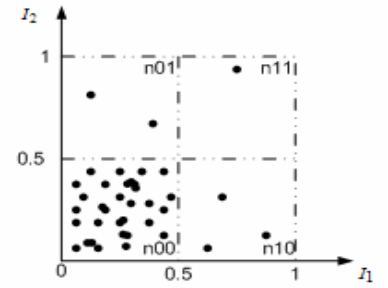

Fig. 2 An example of input space distribution of PE

Fig. 3 illustrates the three types of associations used in the proposed memory model. The undefined signal means its value is equal to 0.5, in such way, 0 and 1 represents the strongest signal. There are three types of associations used in the testing stage to infer the undefined signal value.

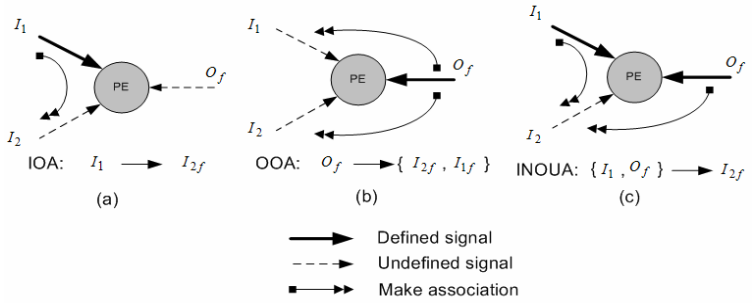

Fig 3 Three types of associations of processing element

(1) Input only association (IOA). If, in the testing stage, one input is defined while the other input and the received output feedback signal O_f from other PEs are undefined (for instance, if $I_1 = 0$, $I_2 = 0.5$ and $O_f = 0.5$ as in Fig. 3(a)), this PE will determine I_2 through association with I_1, driving I_2 towards logic 0.

(2) Output only association (OOA). If both inputs, I_1 and I_2, are undefined, a defined feedback signal, O_f, will determine both inputs (Fig 3(b)). For instance, if $O_f = 0$, based on PE function F= {0, 1, 1, 1}, then this PE will set both inputs, I_{1f} and I_{2f} to 0. (Here we use I_{1f} and I_{2f} to denote the feedback signals of inputs 1 and 2 to distinguish them from the corresponding feed forward signals). On the other hand, if F sets the received output feedback signal to O_f =1, the input feedback values, I_{1f} and I_{2f}, are intermediate and their values will be estimated according to data distribution probabilities.

(3) Input–output association (INOUA). If one input and the output feedback signal, O_f, are defined and the other input is undefined, the PE will set the other input signal according to its observed probabilities, as shown in Fig. 3(c).

This probability based associative learning algorithm can be described as follows:

Case 1: Given the semi-logic values of both inputs $V(I_1)$ and $V(I_2)$, decide the output value $V(O)$

Assume one PE received input values $V(I_1) = m$ and $V(I_2) = n$, then

$$V(O) = \frac{p(I_1 = 1, I_2 = 1, F = 1)}{p(I_1 = 1, I_2 = 1)} \bullet V_{11} + \frac{p(I_1 = 0, I_2 = 1, F = 1)}{p(I_1 = 0, I_2 = 1)} \bullet V_{01} + \frac{p(I_1 = 1, I_2 = 0, F = 1)}{p(I_1 = 1, I_2 = 0)} \bullet V_{10} + \frac{p(I_1 = 0, I_2 = 0, F = 1)}{p(I_1 = 0, I_2 = 0)} \bullet V_{00} \quad (1)$$

where V_{11}, V_{01}, V_{10} and V_{00} are defined as

$$V_{11} = mn;\ \ V_{01} = (1-m)n;$$
$$V_{10} = m(1-n);\ \ \ V_{00} = (1-m)(1-n); \quad (2)$$

and $p(I_1 = 1, I_2 = 1, F = 1)$, $p(I_1 = 1, I_2 = 1)$ etc. are joint probabilities. Case 1 is used when a signal is propagated forward.

Case 2: Given the values of one input, $(V(I_1)$ *or* $V(I_2))$, and an undefined output $V(O)$, decide the value of the other input.

Case 2 corresponds to input-only-association (IOA) when a signal is propagated backwards, as shown in Fig. 3(a). We can use a given $V(I_1)$ to decide an unknown $V(I_2)$ as follows:

$$V(I_2) = \frac{p(I_1 = 1, I_2 = 1)}{p(I_1 = 1)} \bullet V(I_1) + \frac{p(I_1 = 0, I_2 = 1)}{p(I_1 = 0)} \bullet (1 - V(I_1)) \quad (3)$$

where $p(I_1 = 1) = p_{10} + p_{11}$, $p(I_1 = 0) = p_{00} + p_{01}$. In the case in which $V(I_2)$ is given and determines $V(I_1)$, I_1 and I_2 are switched in equation (3).

Case 3: Given the value of the output $V(O)$, decide the value of both inputs $V(I_1)$ and $V(I_2)$.

$$V(I_1) = \frac{p(F = 1, I_1 = 1)}{p(F = 1)} \bullet V(O) + \frac{p(F = 0, I_1 = 1)}{p(F = 0)} \bullet (1 - V(O))$$
$$V(I_2) = \frac{p(F = 1, I_2 = 1)}{p(F = 1)} \bullet V(O) + \frac{p(F = 0, I_2 = 1)}{p(F = 0)} \bullet (1 - V(O)) \quad (4)$$

Case 3 corresponds to output-only-association (OOA) when a signal is propagated backwards as shown in Fig. 3(b). $p(F = 1)$ and $p(F = 0)$ are determined by the output of each PE.

Case 4: Given the values of one input, ($V(I_1)$ or $V(I_2)$, and the output, $V(O)$, decide the other input value, $V(I_2)$ or $V(I_1)$;

Case 4 corresponds to the input-output-association (INOUA) when a signal is propagated backwards (Fig. 3(c)). For example, we can use a given $V(I_1)$ and $V(O)$ to decide $V(I_2)$ as follows:

$$V(I_2) = \frac{p(I_1 = 1, F = 1, I_2 = 1)}{p(I_1 = 1, F = 1)} \bullet \hat{V}_{11} + \frac{p(I_1 = 0, F = 1, I_2 = 1)}{p(I_1 = 0, F = 1)} \bullet \hat{V}_{01} + \frac{p(I_1 = 1, F = 0, I_2 = 1)}{p(I_1 = 1, F = 0)} \bullet \hat{V}_{10} + \frac{p(I_1 = 0, F = 0, I_2 = 1)}{p(I_1 = 0, F = 0)} \bullet \hat{V}_{00} \tag{5}$$

where $\hat{V}_{11}, \hat{V}_{01}, \hat{V}_{10}$ and $\hat{V}_{00}$ are determined in the following way:

$$\hat{V}_{11} = \begin{cases} V(I_1) \bullet V(O) & \begin{cases} X & X & 0 & 1 \\ X & X & 1 & 0 \end{cases} \\ 0 & X \ X \ 0 \ 0 \\ V(I_1) & X \ X \ 1 \ 1 \end{cases} \quad \hat{V}_{10} = \begin{cases} V(I_1) \bullet (1 - V(O)) & \begin{cases} X & X & 0 & 1 \\ X & X & 1 & 0 \end{cases} \\ V(I_1) & X \ X \ 0 \ 0 \\ 0 & X \ X \ 1 \ 1 \end{cases}$$

$$\hat{V}_{01} = \begin{cases} (1 - V(I_1)) \bullet V(O) & \begin{cases} 0 & 1 & X & X \\ 1 & 0 & X & X \end{cases} \\ 0 & 0 \ 0 \ X \ X \\ (1 - V(I_1)) & 1 \ 1 \ X \ X \end{cases} \quad \hat{V}_{00} = \begin{cases} (1 - V(I_1)) \bullet (1 - V(O)) & \begin{cases} 0 & 1 & X & X \\ 1 & 0 & X & X \end{cases} \\ (1 - V(I_1)) & 0 \ 0 \ X \ X \\ 0 & 1 \ 1 \ X \ X \end{cases} \tag{6}$$

The conditions in equation (6) refer to the function value of F for each particular PE, where "X" is a do not care, which means its value can be either '0' or '1'. For example, if one PE received $V(I_1) = m$ and $V(O) = t$, and the function value of this PE is F= {0 1 1 1}, we will get the following results:

$\hat{V}_{11} = m; \quad \hat{V}_{10} = 0; \quad \hat{V}_{01} = (1 - m) * t; \quad \hat{V}_{00} = (1 - m) * (1 - t)$

When $V(I_2)$ and $V(O)$ are given one only needs to switch I_1 and I_2 in equations (5) and (6) to decide $V(I_1)$.

3. Memory Network Architecture

The overall memory network is a hierarchical structure of sparsely connected self-organizing PE's. Each layer of this hierarchical structure contains a number of PEs connected to the primary inputs or to the outputs of other processing elements from lower layers of hierarchy. For n-dimensional input, the network should have at least n/2 PEs in each layer. The required number of layers depends on the problem complexity and may be determined through simulation. In practice, the number of layers grows logarithmically with the size of the input vector.

Each PE in the array can self-organize by dynamically adapting its function in response to the input data. The hierarchical connections are suitable for hardware implementation, time control, and correlate well to complexity of object representation. The further away a PE is from the sensory input, the more abstract and invariant is the representation of objects or their features captured by the PE. Each PE is more likely to connect to other PEs within a short Euclidean distance. This organization is observed in biological memory where neurons tend to have mostly local connections.

(1) Feed forward operation

Fig 5 shows a feed forward network structure for the proposed memory architecture. For simplification, we only illustrate 4 layers with 6 PEs per layer and 6 input

signals. The bold lines from PE 1 to PE 11 and from PE18 to PE21 are two examples of the distant connections.

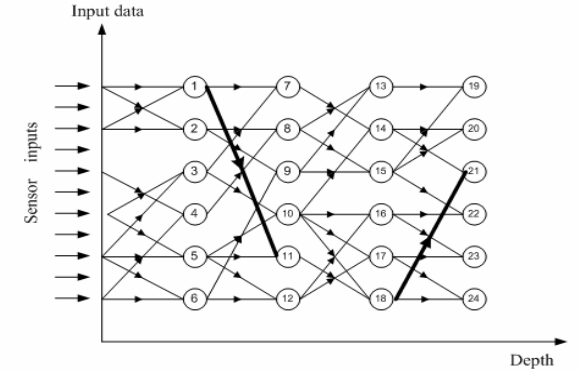

Fig 4: An example of feed forward operation network

During training, all external input data are presented to the network. Each PE counts activities on its inputs to estimate the corresponding probabilities, $p_{ij}, (i, j = 0,1)$, and decide its output function as in case 1 of Section 2. This probability information will be used to make associations in the feedback operation.

(2) Feedback operation

Feedback operation is essential for the network to make correct associations and to recover the missing parts (undefined signals) of the input data. Fig. 5 shows a feedback structure. Assume that signals 1, 2 and 3 are undefined as would be the case in a classification application where all the class ID code inputs are undefined and only the feature input values are available, and in the image recovery application, part of the image could be blocked or undefined. In both cases, the network will use the associations mechanism as discussed in Section 2 to determine these undefined signal values.

In Fig. 5, the shaded PEs are associative and will use associations to recover the undefined values. For instance, PE4 received one defined signal and one undefined signal. In this situation, PE4 will use the IOA to associatively recover this undefined signal based on the information it learned in the training stage. Some associations will also happen in a deeper layer. Considering PE22, it will use IOA to associatively recover the other input signal I_{2f}. This feedback signal will back propagate to PE15 (it will become the O_f for PE15). Therefore, based on the OOA, PE15 will associatively recover both input signals of PE15. In this way, these feedback signals will further back propagate to other hierarchical layers in the network. Therefore, the missing information in the sensor input will be recovered.

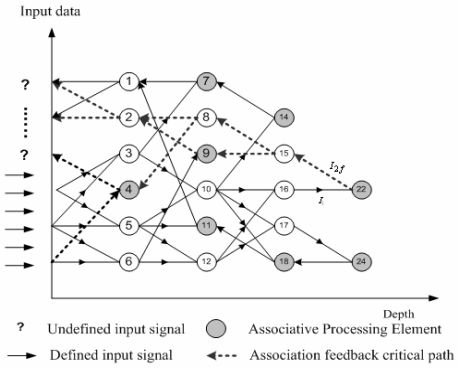

Fig. 5 Example of feedback structure in testing stage

4 Simulation results

The proposed probability based self-organizing associative memory is capable of both hetero and auto-associations. In this section, the Iris database and an image recovery problem are used to illustrate the HA and AA applications.

(1) Hetero-associative memory: Iris database classification

The Iris database [11] developed by R. A. Fisher was used to test the classification performance of the proposed associative memory. We used an N-bits sliding bar coding mechanism to code the input data. Assume that the maximum and minimum values to be coded are $V_{\max}$ and $V_{\min}$ respectively. We set $N - L = V_{\max} - V_{\min}$, where L is the length of the sliding bar. Assume that the value of the scaled feature to be coded is V. In the coded input we set bits numbered from $(V - V_{\min}) + 1$ to $(V - V_{\min}) + L$ to 1s, while the remaining bits were set to 0. The class ID is coded in a similar way using M bit code redundancy. Since there are 3 classes in this database, we use M*3 bits to code the class ID, maximizing their Hamming distance. This was achieved by filling the M bits from position $(C_i - 1) * M$ to $C_i * M$ with 1's, while filling the remaining $M * 2$ bits with 0's. Here $C_i = 1,2$ and 3 for this 3 classes Iris database.

Since there are only 150 instances in the Iris database, the ten-fold cross validation method was used to handle this small sample dataset. Our memory network achieved an overall of 96% correct classification accuracy. Fig. 6(a) shows the associative PEs and their connection structure, and Fig. 6 (b) shows associative PE firing activity for part of the network. The Y-axis represents the input bits, and the X-axis represents the distance from the input (association depth). The associative PEs are represented by circles and their backward propagation paths are marked. The large dots at the input layer represent correctly recognized class ID code bits. It may be seen that only 6 layers are needed for the network to learn the associations in the Iris database.

(2) Auto-associative memory: image recovery

An image recovery problem was used to test the effectiveness of the proposed memory for auto-associative applications. We used the proposed memory to associ-

ate parts of images, and then recall the images from fractional parts of the images. This is necessary for applications where only partial images are available without specifying class identities. Our model can learn features of the training data using unsupervised learning, self-determine the feedback depth, and make correct associations to recover the original images.

We used a 64 x 64 binary panda image [12] to illustrate the auto-associative application of the proposed memory architecture. The panda image is represented by a vector $p_i = (x_1 \quad x_2 \quad ... \quad x_n), n = 4096$, with x_i =1 for a black pixel and x_i =0 for a white pixel. In testing, r% percentage ($r = 10,20$ and 30) of the panda image was randomly blocked. The original panda image and samples of its blocked image are shown in Figs. 7(a) and (b), respectively. Fig. 7(c) shows images recovered through our associative memory. We evaluate image recovery performance by computing the ratios of the number of incorrectly recovered pixels (both erroneous pixels and pixels remaining undefined after recovery) over the total number of pixels. As we can see from Fig. 7, the recovery error bits of our associative memory is range from 0.2% ~ 0.4%.

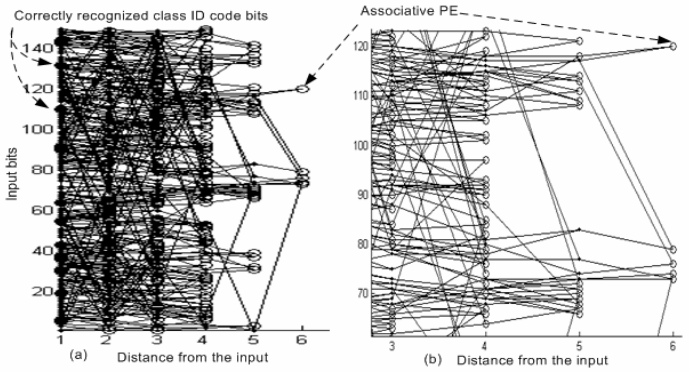

Fig. 6 Associative PEs and their inter connection structure

5. Conclusion

In this paper, we proposed a hierarchical associative memory architecture for machine learning that uses probability based associations. Through the associative learning algorithm, each processing element in the network learns the statistical data distribution, and uses such information for input data association and prediction. Simulation results on both classification and image recovery applications show the effectiveness of the proposed method.

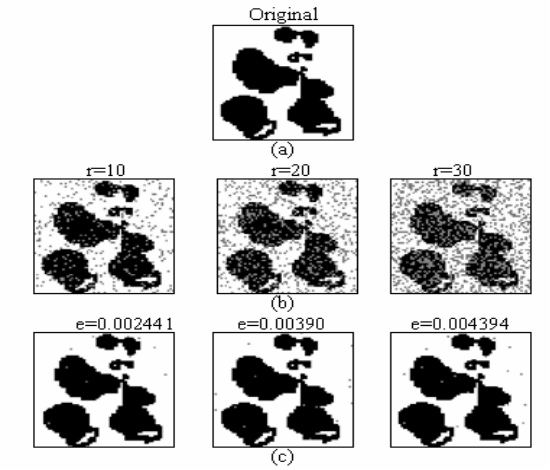

Fig. 7. (a) The original image; (b) Blocked image with r% of undefined values (r= 10, 20 and 30 respectively); (c) Recovered image and the recovery error

6 References

1. Chang, J. Y., Cho, C. W., "Second-order asymmetric BAM design with a maximal basin of attraction," IEEE Trans. on System, Man, and Cybernetics, part A: Systems and Humans, vol. 33, (2003) 421-428
2. Salih, I., Smith, S. H., Liu, D. , "Synthesis approach for bidirectional associative memories based on the perceptron training algorithm," Neurocomputing, vol. 35, (2000) 137-148
3. Wang, L., "Multi-associative neural networks and their applications to learning and retrieving complex spatio-temporal sequences," IEEE Trans. on System, Man, and Cybernetics, part B-Cybernetics, vol. 29, (1999) 73-82
4. Hopfield, J. J., "Neural networks and physical systems with emergent collective computational abilities," in Proc. Nat. Acad. Sci. USA, vol. 79, (1982) 2554-2558
5. Vogel, D., "Auto-associative memory produced by disinhibition in a sparsely connected network," Neural Networks, 5 (11), (1998) 897-908
6. Vogel, D., Boos, W., "Sparsely connected, Hebbian networks with strikingly large storage capacities," Neural Networks, 4 (10), (1997) 671-682
7. Wang, M., Chen, S., "Enhanced EMAM based on empirical kernel map," IEEE Trans. on Neural Network, vol. 16, (2005) 557-563
8. Starzyk, J. A., Zhu, Z., Liu, T.-H., "Self-Organizing Learning Array," IEEE Trans. on Neural Networks, vol. 16, no. 2, (2005) 355-363
9. Triesch, J., "Synergies between intrinsic and synaptic plasticity in individual model neurons," Neural Information Processing System (NIPS), 17 (2004)
10. Starzyk, J. A., Wang, F., "Dynamic Probability Estimator for Machine Learning" IEEE Trans. on Neural Networks, vol.15, no 2, (2004) 298-308
11. Fisher, R. A., "The use of multiple measurements in taxonomic problem," Ann. Eugenics, vol. 7, no. 2, (1936) 179-188
12. Djuric, P. M., Huang, Y., Ghirmai, E., "Perfect sampling: a review and applications to signal processing," IEEE Trans. on Signal Processing, vol. 50, no. 2, (2002) 345 – 356