A CODING PROCEDURE FOR REAL-VALUED PATTERNS USING BIDIRECTIONAL ASSOCIATIVE MEMORY

M. J. Abedin*

College of Computer and Information Sciences
King Saud University
P. O. Box 51178, Riyadh 11543, Saudi Arabia

الخلاصـة :

يُمكن استخدام ذاكرة ارتباطية ثنائية الاتجاه (BAM) من النوع الذي استخدمه كوسكو Kosko، لتخزين واستدعاء أزواج الأنماط الثنائية / ثنائية الأقطاب. وقد تَـمَّ استخدام خوارزمية تعلم تكرارية تسمى PRLAB لاستدعاء المضمون لجميع أزواج الأنماط وذلك حتى النهاية القصوى (العظمى) لسعة ذاكرة التخزين الارتباطية المذكورة. وفي هذا البحث فإننا نبادر باقتراح نظام ترميـز بـارع لتخزين واستدعاء الأنماط الثنائية ذات القيم الحقيقية باستخدام ذاكرة من نـوع (BAM). وتعتمد الخوارزمية المقترحة على الخوارزمية التكرارية PRLAB، والتي تمكّـن من طبيق واستخدام الذاكرة من نوع (BAM) في البيئات الصناعية حيث توجد بكثرة الأنماط التناظرية التماثلية للإشـارات. وقد تم تطوير خوارزمية لتحويل أزواج الأنماط ذات القيم الحقيقية إلى أنماط ثنائية / ثنائية القطب تمكننا من تطبيق هذه الطريقة على الذاكـرة (BAM) المنفصلة (المتقطعة). وبهذه الطريقة يمكـن تطبيق خوارزمية التعليم PRLAB في تخزين واستدعاء أزواج الأنماط ذات القيم الحقيقية بنـجاح. وقد تم إعطاء أمثلة لبيان قابلية الترميز الجديدة للتطبيق على الأنماط ذات القيم الحقيقية بوضوح كاف للاستخدام في أي تطبيق عملي .

*Address for correspondence:
P. O. Box 51178
Riyadh 11543, Saudi Arabia

ABSTRACT

A bidirectional associative memory (BAM), reported by Kosko, can be used to store and recall binary/bipolar pattern pairs. An iterative learning algorithm called PRLAB has been used for guaranteed recall of all training pairs up to the maximum storage capacity of a BAM. In this paper we propose a new coding strategy for storage and recall of real-valued patterns using BAM. The proposed algorithm is based on the PRLAB iterative learning algorithm, which enables the BAM to be applied in industrial environments where analog patterns are commonly encountered. An algorithm has been developed for converting real-valued pattern pairs into binary/bipolar patterns, thus enabling it to be applied to a discrete BAM. The PRLAB learning algorithm is then applied to store and recall the real-valued pattern pairs successfully. Examples are given demonstrating the applicability of the new encoding mechanism for real-valued patterns with sufficient resolution to be used in any practical application.

A CODING PROCEDURE FOR REAL-VALUED PATTERNS USING BIDIRECTIONAL ASSOCIATIVE MEMORY

1. INTRODUCTION

The human information processing system consists of the biological brain. The basic building block of the nervous system is the neuron that handles intercommunication of information among various parts of the body. The neuron can be considered as a threshold unit that collects inputs from other neurons and produces an output only if the sum of the inputs exceeds an internal threshold value. Artificial neural systems (ANSs) are mathematical models of the theorized mind and brain activity [1]. ANSs are created by interconnecting many of the simple 'neurons' into a network. The primary function of ANS is to act as a memory. ANSs store different types of patterns and they perform different types of recall. ANSs act as content addressable memory (CAM) or associative memory. CAMs map data to addresses while associative memories map data to data. The two primary ANS mapping mechanisms are auto-association and hetero-asssociation. An ANS is autoassociative if its memory, W, stores vectors (patterns) $A_1, A_2 \ldots A_m$ and is hetero-associative if W stores the pattern pairs $(A_1, B_1), (A_2, B_2) \ldots (A_m, B_m)$.

An ANS is said to be learning while there is a change in the memory W *i.e.*, $dW/dt \neq 0$. ANS learning methods can be classified into two categories: supervised and unsupervised learning. Supervised learning is a process that incorporates an external teacher and/or global information. Unsupervised learning is a process that incorporates no external teacher and relies upon only local information. Examples of supervised learning are error correction learning, reinforced learning, stochastic learning, while examples of unsupervised learning are Hebbian learning and competitive learning [2].

Hebbian learning, named after Donald Hebb, illustrates a simple correlation learning that adjusts the connection weights according to the correlation (multiplication) of the values of two processing elements (PEs) *i.e.* neurons. Hebbian correlation can be mathematically defined as

$$dw_{ij}/dt = -w_{ij} + a_i b_j \tag{1}$$

where w_{ij} represents the connection strength from neuron a_i to neuron b_j. An extension of simple Hebbian learning is the signal Hebbian learning where the correlation of the activation is filtered through a sigmoid or signal function and is described by the equation

$$dw_{ij}/dt = -w_{ij} + S(a_i)S(b_j) \tag{2}$$

where $S(\ )$ is the sigmoid function. The most common sigmoid function is the logistic function $S(x) = (1+e^{-x})^{-1}$.

ANS recall corresponds to the decoding of the stored contents which may have been encoded in a network previously. Assume that a set of patterns can be stored in a network. Later if the network is presented with a pattern similar to a member of the stored set, it may associate the input with the closest stored pattern. The process is called autoassociation. Association of input patterns can also be stored in a heteroassociative variant. In heteroassociative processing, the association between pairs of patterns is stored. The heteroassociative recall mechanism can be considered a function $g()$ that takes W (memory) and A_i (cue) as input and returns B_i (response). This relationship can be illustrated by the equation $g(A_i, W) = B_i$. Details of other learning and recall mechanisms are available in references [1,2].

Kosko [3] proposed a neural network model of bidirectional associative memory (BAM) that consists of two layers of neurons with feedbacks and symmetric synaptic connections between layers. The discrete BAM behaves as a heteroassociative pattern matcher that encodes arbitrary spatial patterns using Hebbian learning. The encoding procedure places the association at system energy minima. The discrete BAM has been studied extensively [3–8]. However, these neural models have not been utilized for a wide range of applications because of their inability to encode a large number of pattern pairs and the restriction to binary or bipolar pattern pairs. The restriction on input data has limited applicability of BAM. In applications such as medical diagnosis, robot tracking and control, and industrial fault diagnosis where patterns in continuous time are encountered, an appropriate coding scheme is desired.

In this paper we present the A/D converted pattern encoding mechanism that enables the discrete BAM to encode continuous patterns. The BAM encoding and recall mechanism developed can be applied to analog signal patterns encountered

in real life. The encoding mechanism can be applied with any desired accuracy to encode analog signal patterns and recall them successfully. This removes the difficulty of encoding real-valued patterns by discrete BAM. Also the use of pseudo-relaxation learning algorithm for BAM (PRLAB) enables it to store and recall patterns up to the maximum theoretical capacity of BAM [5].

The Rest of this paper is organized as follows. A review of the basic concepts of BAM encoding with enhancements of BAM capacity and recall by different proposed mechanisms are provided in Section 2. Section 3 provides the mechanism of real valued BAM encoding and its limitations. In Section 4 the A/D converted BAM encoding mechanism is developed. In Section 5 simulation results and discussions on the proposed algorithm are made. An example illustrating the capability of the proposed learning and recall mechanism is given in Section 6 while Section 7 concludes the paper.

2. DISCRETE BAM

BAM concepts were first introduced by Kosko [3]. A BAM stores and recalls associations (A_i, B_i) that are learned by summing correlation matrices. Consider an $n \times m$ BAM with n neurons in the first layer and m neurons in the second layer. The BAM stores and recalls p discrete pattern pairs

$$(A_1, B_1), (A_2, B_2) \ldots . (A_p, B_p) \text{ where}$$

$$A_i = (a_{i1}, a_{i2}, \ldots a_{in})$$

$$B_i = (b_{i1}, b_{i2}, \ldots b_{im}).$$

Here a_{ij} and b_{ij} are either ON or OFF. (In binary mode ON = 1, OFF = 0; in bipolar mode ON = 1, OFF = –1).

Kosko used the correlation matrix:

$$M = \sum_{i=1}^{p} X_i^T Y_i \tag{3}$$

where $X_i = (x_{i1}, x_{i2}, \ldots x_{in})$

$Y_i = (y_{i1}, y_{i2}, \ldots y_{im})$ and

$x_{ij}\,(y_{ij})$ is the bipolar form of $a_{ij}\,(b_{ij})$.

The use of such a correlation matrix for storing information in a neural network model is often called the first-order correlation encoding [8]. The correlation matrix M superimposes the information of several patterns on the same memory medium. However, unless the training vectors are orthogonal, the superposition may introduce noise in the system and the recall of all training pairs is not guaranteed [5].

Kosko [3] has shown that the BAM energy function E for the pair (α, β) and correlation matrix M is:

$$E = -\alpha M \beta^T. \tag{4}$$

At each cycle of decoding the energy E given by Equation (4) is lowered. Furthermore the pairs (α, β) and those resulting from subsequent iterations remain finite. Thus if we wish to retrieve one of the nearest (A_i, B_i) pairs from the network where any (α, β) pair is presented, a finite sequence $(\alpha', \beta'), (\alpha'', \beta'')$ ... will be encountered until an equilibrium point (α_F, β_F) is reached where:

$$\beta' = \phi(\alpha M)$$

$$\alpha' = \phi(\beta' M^T). \tag{5}$$

The function $\phi()$ is defined as follows:

$$\phi(F) = G = (g_1, g_2 \ldots, g_n)$$

$$F = (f_1, f_2, \ldots, f_n)$$

$$g_i = 1, f_i > 0$$

$$g_i = \begin{cases} 0 \text{ (binary)} \\ f < 0 \\ -1 \text{ (binary)} \end{cases}$$

g_i = previous g_i, $f_i = 0$.

Thus for the final pattern (α_F, β_F) the energy

$$E = -\alpha_F M \beta_F^T \tag{6}$$

is a local minimum.

If the energy, E, evaluated using the coordinates of the pair (A_i, B_i) *i.e.* $E = -A_i M B_i^T$ does not constitute a local minimum, then the point cannot be recalled even though one starts with $\alpha = A_i$. Kosko's encoding method does not ensure that the stored pairs are at local minima [6]. Depending on the applications, it may be important to guarantee recalling of a particular pair or several training pairs. Wang *et al.* [4] introduced a multiple training concept to ensure recalling of a desired pair in a set of training pairs.

If multiple training of order q is directed to the training pair (A_i, B_i), one augments the M matrix by the matrix P which is defined as:

$$P = (q-1) X_i^T Y_i \,. \tag{7}$$

The new value of the energy E' evaluated at the point (A_i, B_i) then becomes:

$$E'(A_i, B_i) = -A_i M B_i^T - (q-1) A_i X_i^T Y_i B_i^T \,. \tag{8}$$

The augmentation of M in the manner adds $(q-1)$ more pairs located at (A_i, B_i) to the existing N pairs. The energy E' can be decreased to an arbitrary low value by suitable choice of q. The multiple training method can guarantee that one training pair will always be recalled [4]. However, there is no guarantee that all training pairs will be recalled. For a guaranteed recall of all training pairs the dummy augmentation method is proposed in [4].

The dummy augmentation method requires the users to generate dummy elements during the coding phase of BAM. In the dummy augmentation method strictly noise-free sets (dummy sets) are appended to (A_i, B_i). Consider training pairs $\left[\{(A_i, B_i)\}, i = 1 \text{ to } N\right]$ and a strictly noise-free set with N elements $\left[\{D_i\}, i = 1 \text{ to } N\right]$,

where

$$D_i = (d_{i1}, d_{i2} \ldots d_{in}). \tag{9}$$

A new set of training pairs

$$\left[A_i \underbrace{|D_i|\ldots|D_i}_{K} \right], \left[B_i \underbrace{|D_i|\ldots|D_i}_{K'} \right] \Bigg\}_{i=1}^{N} \tag{10}$$

is formed with K D_is appended to A_i and K' D_is appended to B_is. It is shown in [4] that by appropriate choice of D_i, K, and K', it is possible to guarantee recall of all patterns. It is to be noted that dummy augmentation algorithm requires the generation of strictly noise free dummy sets to be augmented to the patterns. It is not always possible to generate allowable dummy sets which can guarantee complete recall of all training pairs.

Wang *et al.* [6] presented the idea of linear programming/multiple training (LP/MT) method, which determines weights that satisfy the conditions to recall all training pairs.

Suppose *MT* is used for all pairs in the training set, then the correlation matrix becomes:

$$M = \sum_{i=1}^{N} q_i X_i Y_i . \tag{11}$$

A pair $P_i = (A_i, B_i)$ can be recalled using the generalized correlation matrix in (7) if and only if

$$\sum_{j=1}^{N} q_i \eta_{ij} \geq 0$$

for every neighbor in both fields A and B where $q_j > 0$ for all $i = 1,2 .. N$ and η_{ij} is:

$$\eta_{ij} = A_i X_j^T Y_j B_i^T - A'^i X_j^T Y_j B_i'^T . \tag{12}$$

The physical meaning of η_{ij} is the energy difference between P_i' and P_i caused by P_j. If $\eta_{ij} < 0$ then the neighbor P_i' achieves more negative energy than P_i from P_j, such that P_i is relatively more unstable; if $\eta_{ij} > 0$ then the neighbor P_i achieves more negative energy than P_i' from P_j, such that P_i is relatively more stable.

All training pairs in $T = \left(\{P_i | i = 1,2 \ldots N\}; P_i = (A_i, B_i)\right)$ can be recalled using a generalized correlation matrix if and only if the positive real weights satisfy

$$\Sigma\, q_j \eta_{ij} \geq 0,$$

where $q_j > 0$ for all $i = 1,2, \ldots N$ and η_{ij} is defined in (12).

An iterative learning algorithm called pseudo-relaxation learning algorithm for BAM (PRLAB) which ensures recall of all training pairs has been introduced by Oh and Kothari [9]. This algorithm updates the weights based on a mathematical technique called the relaxation method.

Let W_{ij} be the connection strength between the i-th neuron in the first layer and the j-th neuron in the second layer of a $N \times M$ BAM. Let θ_{Xi} be the threshold for the i-th neuron in the first layer and θ_{Yj} be the threshold for the j-th neuron in the second layer.

Let $T = \{(X^{(k)}, Y^{(k)})\}_{k=1 \ldots p}$ be a set of training vector pairs where

$$X^{(k)} \in \{-1,1\}^N \text{ and } Y^{(k)} \in \{-1,1\}^M .$$

The vectors in T are guaranteed to be recalled if the following system of linear inequalities are satisfied for all K=1,2 ... P.

$$\left(\sum_{i=1}^{N} W_{ij} X_i^{(k)} - \theta_{Yj} \right) Y_j^{(k)} > 0 \text{ for } j = 1 \ldots M \tag{13}$$

$$\left(\sum_{j=1}^{M} W_{ij} Y_j^{(k)} - \theta_{Xi} \right) X_i^{(k)} > 0 \text{ for } i = 1 \ldots N . \tag{14}$$

PRLAB examines each training pair $X^{(k)}$, $Y^{(K)}$ one by one systematically and updates the weights and threshold values if inequalities (13) or (14) are not satisfied. It iterates through the training pairs as follows.

$$\Delta W_{ij} = -(\lambda/(1+M))\left[S_{X_i}{}^{(k)} - \xi X_i{}^{(k)}\right]Y_i{}^{(k)} \quad \text{if} \quad S_{X_i}{}^{(k)} X_i{}^{(k)} \le 0$$

$$\Delta\theta_{X_i} = +\lambda/(1+M)\left[S_{X_i}{}^{(k)} - \xi X_i{}^{(k)}\right]Y_i{}^{(k)} \quad \text{if} \quad S_{X_i}{}^{(k)} X_i{}^{(k)} \le 0$$

and for the neurons in the second layer

$$\Delta W_{ij} = -\lambda/(1+N)\left[S_{Y_j}{}^{(k)} - \xi Y_j{}^{(k)}\right]X_i{}^{(k)} \quad \text{if} \quad S_{Y_j}{}^{(k)} Y_j{}^{(k)} \le 0$$

$$\Delta\theta_{Y_j} = +\lambda/(1+N)\left[S_{Y_j}{}^{(k)} - \xi Y_j{}^{(k)}\right]X_i{}^{(k)} \quad \text{if} \quad S_{Y_j}{}^{(k)} Y_j{}^{(k)} \le 0$$

where:

$$S_{X_i}{}^{(k)} = \sum_{j=1}^{M} W_{ij} Y_j{}^{(k)} - \theta_{X_j}\,; \tag{15}$$

and

$$S_{Y_j}{}^{(k)} = \sum_{i=1}^{N} W_{ij} X_i{}^{(k)} - \theta_{Y_j}\,. \tag{16}$$

The relaxation factor λ is a constant between 0 and 2 and the normalizing constant ξ must be positive. PRLAB always finds a solution in finitely many steps if the inequalities (13) and (14) are not satisfied. Note that the initial values of the weights and threshold are chosen at random. It has been demonstrated in [10] that a quick learning for BAM by using the PRLAB is possible if the BAM is initially formed by the correlation matrix W in the first stage as:

$$W = \sum_{K=1}^{P} X^{(k)T} Y^{(k)}\,. \tag{17}$$

In the second stage, the BAM is trained by the PRLAB as defined above. As the correlation matrix learning (Hebbian learning) is used in the first stage, a reduction of learning epochs is expected in the proposed algorithm. The term epoch is used to denote a learning iteration. Each training pattern is presented once during an epoch and the learning speed is measured in number of epochs. If a pattern is not already memorized, weights are adjusted in successive epochs.

Our discussion so far is concerned with the encoding and recall of binary (bipolar) BAM. As most of the research carried out on BAM are restricted in binary or bipolar pattern pairs, the application of BAM in practical use is very much limited. However, improved encoding mechanisms [4, 6, 8] ensured the recall of all training pairs which was not possible in the original BAM. Also, some improvement has been done on the increment of the size of BAM *i.e.* to construct a BAM capable of recalling a large number of training pairs [8].

In the following sections we shall mainly concentrate on the encoding and recall of real valued patterns as to make the BAM useful for applications such as fault diagnosis, robot control, medical diagnosis *etc*. The success behind these depends on the successful encoding and recall of real valued pattern pairs by using BAM.

3. REAL VALUED BAM ENCODING

The adaptive BAM (ABAM) [5] is an extension of BAM where the correlation matrix elements are updated by Hebbian learning according to:

$$dw_{ij}/dt = \alpha\left[-w_{ij} + S(a_i)S(b_i)\right]. \tag{18}$$

where w_{ij} represents the connection strength from neuron a_i to neuron b_j, $S()$ is the sigmoid function and α is a small positive constant controlling the learning rate.

ABAM recall uses the same processing sequence as the BAM but the recall equations are now changed to:

$$da_i/dt = -a_i + \sum S(b_j)w_{ij} + I_i \tag{19}$$

$$db_j/dt = -b_j + \sum S(a_i)w_{ij} + J_j . \tag{20}$$

The ABAM theorem assumes the global stability of recall process. The ABAM is able to process analog patterns in continuous time. But, it has the same storage limitation of BAM. An additional limitation is the memory plasticity. Any pattern pair presented for an extended period of time is "burned in" to the point that no other patterns can be stored in the memory [1]. However, despite its potential the ABAM has not yet been applied owing to its limitations.

In [11] the weighted matrix product coding procedure has been introduced that can be applied for real valued analog patterns. Such patterns can be discretized and their real valued amplitudes can be expressed in binary form. An analog pattern pair is considered as a pair of real valued vectors $P_i \in [0,1]^n$ and $Q_i \in [0,1]^p$. In order to encode it into a correlation matrix these real valued vectors can be approximated by an averaged sum of k binary valued vectors as

$$P_i = (1/k)\sum a_{ij}, \quad i = 1,2,3 \ldots n$$

$$Q_j = (1/k)\sum b_{ij}, \quad j = 1,2,3 \ldots p.$$

The real valued vectors can be written as $(n \times k)$ and $(p \times k)$ binary valued data matrices A_i and B_i. The correlation matrix in terms of these data matrices can be derived as

$$W_i = wA_i^T B_i \quad \text{where } w = (1/k^2).$$

The weighted matrix-product law sums up the binary correlation matrices for each pattern:

$$W = w\sum A_i^T B_i . \tag{21}$$

We can represent the weighted-matrix product law more compactly as:

$$W = wA^T B \tag{22}$$

where $A^T = \left[A_1^T : A_2^T : \ldots A_m^T\right]$

$$B^T = \left[B_1^T : B_2^T : \ldots B_m^T\right]$$

and $w = 1/k^2$.

Note that encoding one real valued vector pair results in encoding k binary valued vector pairs. Since the storage limit of binary-valued BAM is $\min(n,p)$, it is desirable to make n and p large enough if the value of k is large. Also, for better approximation of the real-valued patterns, the value of k should be large enough, which in turn complicates the arbitrary decomposition of P_i and Q_i into binary patterns. In the next section we shall generalize the real-valued pattern encoding, which can be used for up to any desirable range of resolution. Also, it will simplify the mechanism of formation of A_i and B_i while real-valued pattern pairs P_i and Q_i are presented.

4. A/D CONVERTED PATTERN ENCODING MECHANISM

Let P_i and Q_i be the real valued pattern pairs,

where $P_i = \{p_{i1}, p_{i2} \ldots, p_{il}\}$ and $Q_i = \{q_{i1}, q_{i2}, \ldots, q_{im}\}$.

Each of p_{ij} and q_{ij} are *A/D* converted and expressed as

$$p_{ij} = \sum_{k=0}^{n} U_k 2^k \tag{23}$$

$$q_{ij} = \sum_{k=0}^{n} V_k 2^k , \tag{24}$$

the values of U_i and V_i being $\in$ [0,1].

By choosing n large enough the necessary resolution can be obtained. For example, if $n = 6$ the resolution can be 0.01 and for $n = 9$ the resolution is 0.001.

Let (P_i, Q_i) be the real valued pattern pair represented by:

$$P_i = \{p_{i1}, p_{i2}, \ldots, p_{il}\} \tag{25}$$

$$Q_i = \{q_{i1}, q_{i2}, \ldots, q_{im}\} \tag{26}$$

real valued vectors.

When converted into binary form each p_{ij} and q_{ij} becomes

$$\begin{aligned} p_{i1} &= \{U_{i1,0}, U_{i1,1}, \ldots, U_{i1,n}\} \\ p_{i2} &= \{U_{i2,0}, U_{i2,1}, \ldots, U_{i2,n}\} \\ &\ldots\ldots \\ p_{il} &= \{U_{il,0}, U_{il,1}, \ldots, U_{il,n}\} \end{aligned} \tag{27}$$

and

$$\begin{aligned} q_{i1} &= \{V_{i1,0}, V_{i1,1}, \ldots, V_{i1,n}\} \\ q_{i2} &= \{V_{i2,0}, V_{i2,1}, \ldots, V_{i2,n}\} \\ &\ldots\ldots \\ q_{im} &= \{V_{im,0}, V_{im,1}, \ldots, V_{im,n}\}. \end{aligned} \tag{28}$$

Let (A_i, B_i) be the binary representation of (P_i, O_i). (A_i, B_i) can be written as

$$A_i = \{U_{i1,0}U_{i1,1} \ldots U_{i1,n}, U_{i2,0} \ldots U_{i2,n} \ldots U_{i1,0}U_{i1,} \ldots U_{i1,n}\} \tag{29}$$

and

$$B_i = \{V_{i1,0}V_{i1,1} \ldots V_{i1,n}, V_{i2,0} \ldots V_{i2,n} \ldots V_{im,0}V_{im,1} \ldots V_{im,n}\}. \tag{30}$$

Each of the vector A_i has $(n + 1)l$ elements and B_i has $(n + 1)m$ elements of binary 0s and 1s. An added advantage of *A/D* converted encoding of BAM is that there is an $(n + 1)$ fold enlargement of the size *i.e.* both A_i and B_i vectors are now expanded $(n + 1)$ times than the original P_i and Q_i vectors. This automatically increases the number of BAM interconnections and hence the storage capacity of the BAM. Once the vector pairs (A_i, B_i) are formed the next step is to convert the (A_i, B_i) pairs into their bipolar forms.

Let (X_i, Y_i) be the bipolar representation of (A_i, B_i). Then the correlation matrix W can be formed using Equation (3) *i.e.*

$$W = \sum_{i=1}^{p} X_i^T Y_i . \tag{31}$$

If the number of patterns encoded are small the original BAM recall mechanism as stated in [3] can be used to recall the trained patterns. As Kosko's original BAM suffers from limitations in recalling a large number of patterns we used the PRLAB recall mechanism to encode a large number of patterns. The mechanism of encoding and recall of real valued patterns by the A/D converted recall method can be summarized in the following steps.

Step 1. Convert each pattern pair (P_i, Q_i) to its binary form (A_i, B_i)

Step 2. Produce the bipolar form (X_i, Y_i) of the binary pattern pair (A_i, B_i).

Step 3. Form the correlation matrix W from the bipolar pattern pairs $\left(X_i^{(k)}, Y_i^{(k)}\right)$.

Step 4. Use the PRLAB encoding mechanism to from the final correlation matrix satisfying the required conditions.

Step 5. Use the BAM recall mechanism to decode desired patterns.

Step 6. Convert the bipolar pattern pairs back to binary and then to analog patterns *i.e.* (P_i, Q_i).

5. SIMULATION RESULTS AND DISCUSSION

In this section we shall demonstrate the ability of *A/D* converted BAM encoding mechanism for successful encoding and recall of real valued pattern pairs. Table 1 shows the simulation results of the learning and recall of random real valued patterns for both the original and the PRLAB BAM. It is clear that the capacity of the PRLAB mechanism is much better than the original BAM. It can recall real valued pattern pairs successfully in the absence of any noise in the patterns. It is also shown that the original BAM has a very limited capacity of recalling real valued patterns. For example a 98×98 BAM can store and recall up to 8 pattern pairs successfully (Table 1) while a PRLAB BAM of the same size can store and recall successfully up to 98 real valued pattern pairs. Thus we can reach up to the maximum storage capacity of a BAM [3] using the PRLAB recall mechanism.

In Table 2 we demonstrated the capability of the proposed algorithm to store and recall randomly generated patterns using the PRLAB mechanism. We took 20 arbitrary pattern pairs each with 9 real-valued numbers in the range [0,1] and with resolution up to 0.001 (10 bits). We converted each pattern into $9*10 = 90$ binary (bipolar) numbers. So, we used a BAM of

Table 1. Original BAM and PRLAB Learning and Recall Capacity for Random Real Valued Pattern Pairs.

BAM size ($M \times N$)	Number of Real Valued Pattern Pairs	Original BAM (Pairs Recalled Correctly)	PRLAB (Pairs Recalled Correctly)
50×50	50	5	50
64×64	64	6	64
70×70	70	7	70
80×80	80	8	80
98×98	98	9	98

size 90×90 to encode and decode the patterns. In Table 2, rows 1 and 2 represent the first pattern pair (input real-valued vectors $P1$ and $Q1$). All other 19 input pattern pairs are shown in successive pairs of rows, *i.e.* rows 3 and 4 for pattern pair 2 and rows 5 and 6 for pattern pair 3 and so on. The BAM is trained by the PRLAB mechanism with the 20 input pattern pairs. The BAM recall mechanism is then used to recall any desired pattern. All 20 pattern pairs are recalled successfully without any error in recalling. The PRLAB mechanism has been tested for storing and recalling pattern pairs up to the maximum capacity of a BAM and found to be working perfectly without any difficulty. The number of learning epochs required for recalling all pattern pairs successfully lies in the range 10–15 with an average of 12 epochs [10].

We have also studied the proposed learning mechanism for real valued patterns in the presence of Gaussian noise. Random Gaussian noise (normal distribution) having the same mean and standard deviation ($m = \sigma$) was introduced in a certain

Table 2. Randomly Generated Real Valued Pattern Pairs Successfully Learnt and Recalled by a (90 × 90) PRLAB BAM.

LABEL	1	2	3	4	5	6	7	8	9
P1	0.524	0.106	0.968	0.208	0.011	0.319	0.521	0.382	0.331
Q1	0.851	0.543	0.901	0.457	0.174	0.861	0.068	0.426	0.665
P2	0.050	0.779	0.709	0.655	0.887	0.647	0.912	0.923	0.575
Q2	0.532	0.743	0.393	0.594	0.984	0.754	0.392	0.432	0.439
P3	0.947	0.836	0.068	0.366	0.991	0.847	0.603	0.364	0.301
Q3	0.750	0.415	0.310	0.102	0.642	0.479	0.094	0.980	0.258
P4	0.348	0.387	0.694	0.256	0.430	0.292	0.927	0.675	0.992
Q4	0.149	0.318	0.966	0.840	0.957	0.070	0.803	0.200	0.377
P5	0.894	0.891	0.030	0.939	0.410	0.940	0.550	0.730	0.597
Q5	0.950	0.189	0.409	0.945	0.714	0.234	0.897	0.284	0.335
P6	0.535	0.947	0.018	0.671	0.937	0.822	0.764	0.748	0.021
Q6	0.154	0.848	0.847	0.778	0.954	0.776	0.877	0.584	0.177
P7	0.202	0.849	0.706	0.985	0.658	0.207	0.202	0.378	0.964
Q7	0.603	0.446	0.768	0.471	0.446	0.348	0.984	0.909	0.293
P8	0.187	0.202	0.107	0.444	0.982	0.682	0.885	0.492	0.636
Q8	0.316	0.295	0.851	0.912	0.225	0.438	0.825	0.789	0.782
P9	0.389	0.539	0.916	0.218	0.420	0.844	0.117	0.999	0.638
Q9	0.453	0.377	0.213	0.389	0.939	0.700	0.395	0.603	0.070
P10	0.414	0.210	0.197	0.130	0.920	0.385	0.601	0.576	0.608
Q10	0.156	0.851	0.414	0.821	0.953	0.752	0.904	0.042	0.215
P11	0.743	0.945	0.469	0.421	0.930	0.901	0.262	0.831	0.755
Q11	0.933	0.853	0.876	0.247	0.973	0.123	0.932	0.444	0.324
P12	0.977	0.968	0.446	0.436	0.616	0.509	0.447	0.758	0.540
Q12	0.089	0.959	0.417	0.576	0.747	0.340	0.960	0.677	0.492
P13	0.377	0.205	0.534	0.237	0.086	0.222	0.361	0.952	0.101
Q13	0.990	0.797	0.053	0.764	0.924	0.412	0.375	0.851	0.055
P14	0.509	0.270	0.097	0.543	0.552	0.747	0.098	0.891	0.818
Q14	0.487	0.609	0.670	0.196	0.436	0.147	0.192	0.907	0.725
P15	0.884	0.542	0.761	0.723	0.855	0.577	0.382	0.701	0.163
Q15	0.766	0.180	0.967	0.401	0.574	0.154	0.416	0.274	0.068
P16	0.977	0.737	0.401	0.347	0.170	0.534	0.763	0.118	0.446
Q16	0.823	0.102	0.592	0.423	0.097	0.958	0.134	0.100	0.378
P17	0.633	0.091	0.753	0.251	0.198	0.756	0.503	0.401	0.911
Q17	0.386	0.682	0.452	0.356	0.901	0.578	0.013	0.126	0.173
P18	0.130	0.328	0.987	0.053	0.559	0.704	0.189	0.338	0.702
Q18	0.501	0.628	0.685	0.948	0.527	0.650	0.550	0.352	0.448
P19	0.154	0.398	0.990	0.691	0.292	0.838	0.626	0.665	0.421
Q19	0.572	0.018	0.977	0.774	0.952	0.634	0.203	0.168	0.596
P20	0.899	0.618	0.341	0.868	0.985	0.783	0.898	0.492	0.728
Q20	0.036	0.813	0.550	0.977	0.901	0.474	0.696	0.497	0.266

percentage of the patterns. We have studied the original and the PRLAB BAM in presence of up to 30% noise introduced. Once a certain pattern has been selected for the introduction of noise, all elements of the same pattern are shifted up or down according to the normal distribution with $m = \sigma$. The BAM recall capabilities, in presence of noise, are shown in Figures 1–4. Figures 1–2 are drawn for the original BAM while Figures 3 and 4 are for the PRLAB BAM. It can be seen in Figures 1 and 2 that the introduction of random noise degrades the original BAM performance drastically. In case of the PRLAB BAM (Figures 3 and 4), with a small number of patterns (5 to 10 patterns), the performance degrades slightly when random noise is introduced. Thus a PRLAB BAM can also be used to recall noisy patterns when the number of patterns are small.

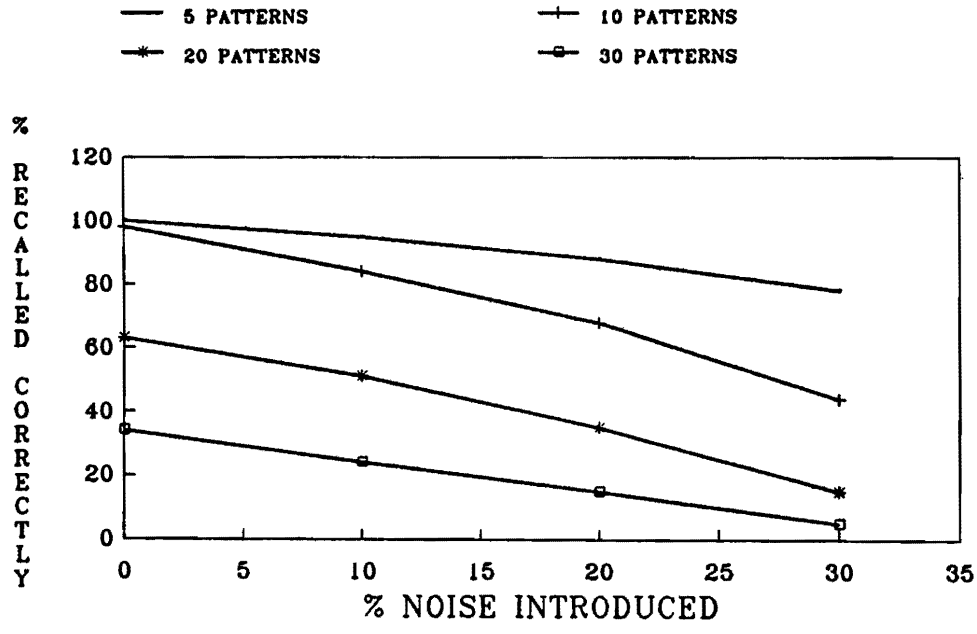

Figure 1. Recall Capacity of Original BAM in Presence of Noise (BAM size 84 × 84).

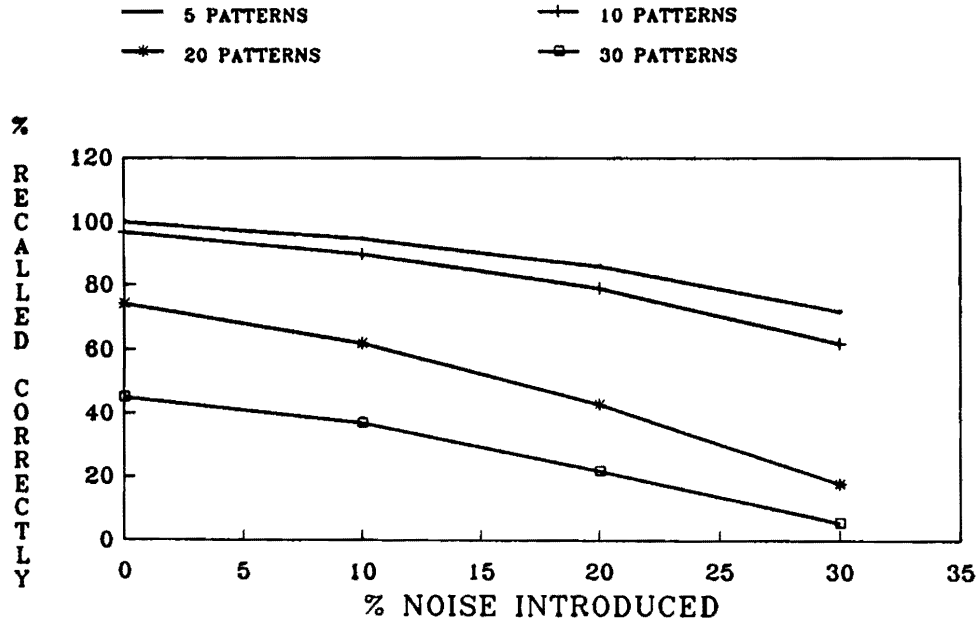

Figure 2. Recall Capacity of Original BAM in Presence of Noise (BAM size 98 × 98).

6. AN EXAMPLE

In this section we give an example demonstrating the use of the new BAM encoding method for storing and recalling real-valued pattern pairs applying the A/D converted pattern encoding mechanism. We used a 42 × 42 BAM to store and recall 10 real-valued pattern pairs given in Table 3. Each pattern has 6 real numbers (real numbers P_i, $Q_i \in [0-1]$), where each real number can have a resolution of 0.01. First of all, the real-valued patters are converted into their binary and then to bipolar forms (steps 1, 2). Table 4 shows the binary representation of the real-valued pattern pairs. We omitted the decimal point from each real number while converted to binary. As the BAM processes the data in binary/bipolar form only, the omission

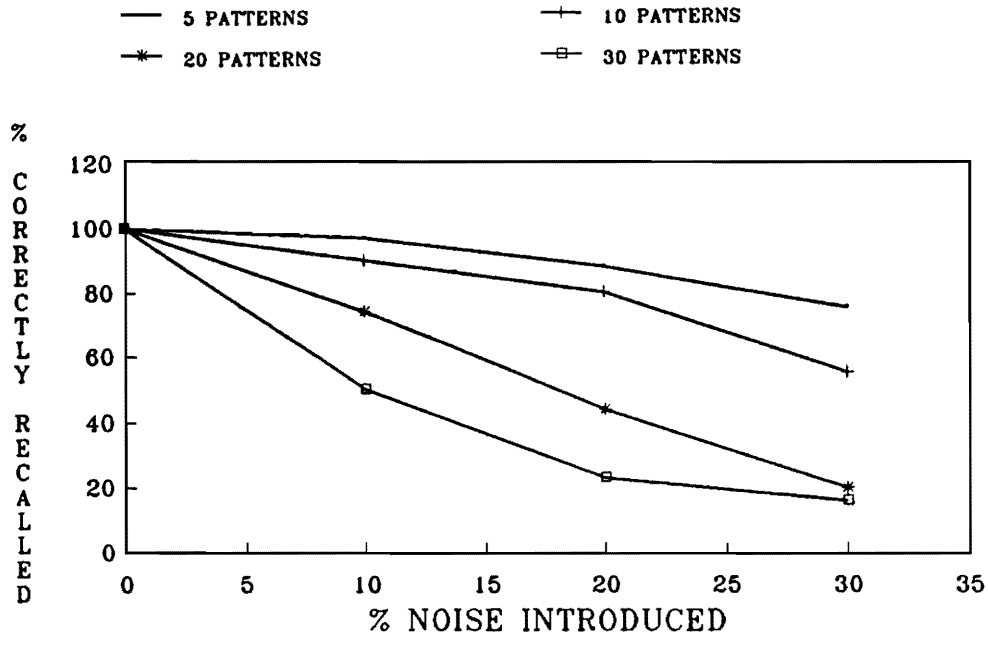

Figure 3. Recall Capacity of PRLAB BAM in Presence of Noise (BAM size 84 × 84).

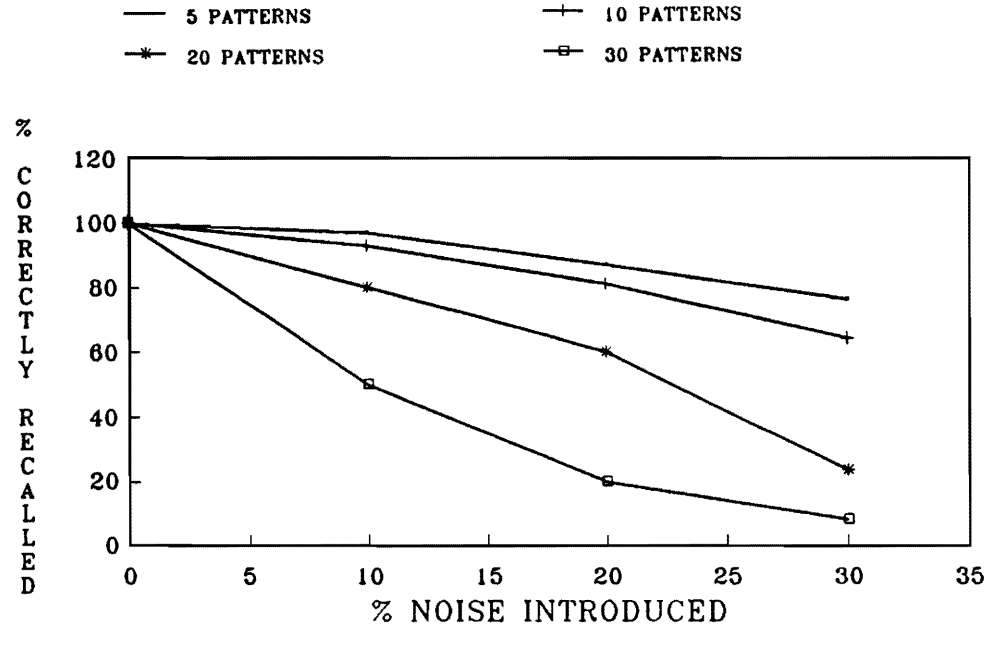

Figure 4. Recall Capacity of PRLAB BAM in Presence of Noise (BAM size 98 × 98).

of the decimal point does not effect the BAM processing. Here the omission of the decimal point simply means the premultiplication of all real numbers by a factor of 100. Once the BAM correctly recalls the input patterns, the output data can again be returned back as real numbers within the range [0–1] after division by 100. Once the input patterns are ready in their bipolar form, the initial correlation matrix W is formed using the bipolar pattern pairs and taking the help of Equation 3 (step 3). After the initial correlation matrix W is formed, the PRLAB learning mechanism is used to determine the final correlation matrix through successive iterations satisfying the required conditions set in Equations(13, 14). This

Table 3. Input Pattern Pairs (P_i, Q_i) Used for the (42 × 42) PRLAB BAM.

P1	1.00	0.80	0.70	0.67	0.50	0.50
Q1	0.33	0.66	0.99	0.50	0.35	0.16
P2	0.00	0.16	0.33	0.66	0.85	1.00
Q2	0.67	0.32	0.34	0.18	0.10	0.00
P3	0.15	0.44	0.90	0.50	0.31	0.11
Q3	0.00	0.43	0.54	0.55	0.67	0.00
P4	0.32	0.56	0.54	0.56	0.12	0.10
Q4	0.67	0.89	0.99	0.50	0.22	0.14
P5	0.21	0.32	0.34	0.50	0.40	0.80
Q5	0.56	0.67	0.00	0.00	0.34	0.56
P6	0.50	0.10	0.40	0.60	0.70	0.55
Q6	0.76	0.32	0.98	0.99	0.01	0.01
P7	0.33	0.67	0.71	0.80	0.98	1.00
Q7	0.22	0.34	0.56	0.45	0.10	0.20
P8	0.00	0.43	0.23	0.43	0.10	0.05
Q8	0.10	0.30	0.40	0.60	0.80	0.90
P9	0.45	0.21	0.80	0.99	0.52	0.11
Q9	0.00	0.10	0.20	0.10	0.01	0.00
P10	0.43	0.56	0.78	0.77	0.45	0.55
Q10	0.23	0.30	0.60	0.21	0.34	0.54

Table 4. Binary Input Pattern Pairs to the (42 × 42) PRLAB BAM.

A1	1100100	1010000	1000110	1000011	0110010	0110010
B1	0100001	1000010	1100011	0110010	0100011	0010000
A2	0000000	0010000	0100001	1000010	1010101	1100100
B2	1000011	0100000	0100010	0010010	0001010	0000000
A3	0001111	0101100	1011010	0110010	0011111	0001011
B3	0000000	0101011	0110110	0110111	1000011	0000000
A4	0100000	0111000	0110110	0111000	0001100	0001010
B4	1000011	1011001	1100011	0110010	0010110	0001110
A5	0010101	0100000	0100010	0110010	0101000	1010000
B5	0111000	1000011	0000000	0000000	0100010	0111000
A6	0110010	0001010	0101000	0111100	1000110	0110111
B6	1001100	0100000	1100010	1100011	0000001	0000001
A7	0100001	1000011	1000111	1010000	1100010	1100100
B7	0010110	0100010	0111000	0101101	0001010	0010100
A8	0000000	0101011	0010111	0101011	0001010	0000101
B8	0001010	0011110	0101000	0111100	1010000	1011010
A9	0101101	0010101	1010000	1100011	0110100	0001011
B9	0000000	0001010	0010100	0001010	0000001	0000000
A10	0101011	0111000	1001110	1001101	0101101	0110111
B10	0010111	0011110	0111100	0010101	0100010	0110110

Table 5. Output Pattern Pairs after Successful Recall by the (42 × 42) PRLAB BAM.

P1	1.00	0.80	0.70	0.67	0.50	0.50
Q1	0.33	0.66	0.99	0.50	0.35	0.16
P2	0.00	0.16	0.33	0.66	0.85	1.00
Q2	0.67	0.32	0.34	0.18	0.10	0.00
P3	0.15	0.44	0.90	0.50	0.31	0.11
Q3	0.00	0.43	0.54	0.55	0.67	0.00
P4	0.32	0.56	0.54	0.56	0.12	0.10
Q4	0.67	0.89	0.99	0.50	0.22	0.14
P5	0.21	0.32	0.34	0.50	0.40	0.80
Q5	0.56	0.67	0.00	0.00	0.34	0.56
P6	0.50	0.10	0.40	0.60	0.70	0.55
Q6	0.76	0.32	0.98	0.99	0.01	0.01
P7	0.33	0.67	0.71	0.80	0.98	1.00
Q7	0.22	0.34	0.56	0.45	0.10	0.20
P8	0.00	0.43	0.23	0.43	0.10	0.05
Q8	0.10	0.30	0.40	0.60	0.80	0.90
P9	0.45	0.21	0.80	0.99	0.52	0.11
Q9	0.00	0.10	0.20	0.10	0.01	0.00
P10	0.43	0.56	0.78	0.77	0.45	0.55
Q10	0.23	0.30	0.60	0.21	0.34	0.54

involves step 4 in our algorithm. The BAM recall mechanism is then applied to correctly recall all real valued pattern pairs (step 5). The recalled pattern pairs, after conversion to real numbers $(P_i, Q_i \varepsilon[0-1])$, are shown in Table 5 (step 6).

It is to be noted that the original correlation matrix W formed by Equation 3 is not sufficient to recall the pattern pairs correctly unless the PRLAB learning mechanism is applied to form the final correlation matrix. The above example demonstrates steps (1–6) described in section 4 earlier for the encoding and accurate recall of real valued pattern pairs with the PRLAB BAM encoding mechanism. The mechanism can be used to learn and correctly recall any number of real-valued pattern pairs up to the storage capacity of the BAM. The BAM storage capacity equals min(n,m), where there are n neurons in the first layer and m neurons in the second layer [3].

7. CONCLUSIONS

The encoding of real-valued vector pairs into a BAM has been considered in this paper. The *A/D* converted BAM encoding mechanism has been developed and applied. Examples are cited to demonstrate the ability of BAM encoding and recall for real-valued pattern pairs. The capability of real-valued pattern encoding and recall will be useful in many practical applications like industrial signal encoding, fuzzy pattern encoding *etc.* Further work is directed to extend the algorithm for more realistic applications.

REFERENCES

[1] P. K. Simpson , " A Review of Artificial Neural Systems I: Foundations", in *CRC Critical Reviews in Artificial Intelligence*. San Diego; General Dynamics, Electronics Division, May, 1988.

[2] Jacek M. Zurada, *Introduction to Artificial Neural Systems*. St. Paul: West Publishing Co. 1992.

[3] B. Kosko, "Bidirectional Associative Memories", *IEEE Transactions on Systems, Man and Cybernetics*, **18** (1988), pp. 49–60.

[4] Y. F. Wang, J. B. Cruz, and J. H. Mulligan, "Two Coding Strategies for Bidirectional Associative Memory", *IEEE Transactions on Neural Networks*, **1** (1990), pp. 81–89.

[5] B . Kosko, *Neural Network and Fuzzy Systems: a Dynamic Systems Approach to Machine Intelligence*. Englewood Cliffs, N.J.: Prentice-Hall, 1992.

[6] Y. F. Wang, J. B. Cruz, and J. H. Mulligan, "Guaranteed Recall of all Training Pairs for BAM", *IEEE Transactions on Neural Networks*, **2** (1991), pp. 559–567.

[7] G. Mathai and B. R. Upadhyan, "Performance Analysis and Applications of Bidirectional Associative Memory to Industrial Spectral Signatures", *Proceedings, International Joint Conference on Neural Networks -89,* 1989, pp. 33–37.

[8] P. K. Simpson, "Higher-Ordered and Interconnected Bidirectional Associative Memories", *IEEE Transactions on Systems, Man and Cybernetics,* **20(3)** (1990), pp. 637–653.

[9] H. Oh and S. C. Kothari, "Adaptation of the Relaxation Method for Learning in Bidirectional Associative Memory", *IEEE Trans. on Neural Networks,* **5,** (1994), pp. 576–583.

[10] M. Hattori, M. Hagiwara, and M. Nakagawa, " Quick Learning of Bidirectional Associative Memory", *International Joint Conference on Neural Networks-93,* (1993), pp. II297–II300,

[11] M. J. Abedin and S. I. Ahson, "A New Coding Procedure for Bidirectional Associative Memory", *Journal of Microcomputer Applications,* **16** (1993), pp. 189–195.

Paper Received 25 July 1995, Revised 20 April 1996; Accepted 26 March 1997.