한국 퍼지 및 지능시스템학회 논문지 1999, Vol. 9, No. 6 (1999)571~576

Neural Network Cubes (N-Cubes) for Unsupervised Learning in Gray-Scale Noise

Hoon Kang and Won-Hee Lee

Department of Electrical & Electronics Engineering, Chung-Ang University

ABSTRACT

We consider a class of auto-associative memories, namely, N-Cubes (Neural-network Cubes) in which 2-D gray-level images and hidden sinusoidal 1-D wavelets are stored in cubical memories. First, we develop a learning procedure based upon the least-squares algorithm. Therefore, each 2-D training image is mapped into the associated 1-D waveform in the training phase. Second, we show how the recall procedure minimizes errors among the orthogonal basis functions in the hidden layer. As a 2-D image corrupted by noise is applied to an N-Cube, the nearest one of the originally stored training images would be retrieved in the recall phase. Simulation results confirm the efficiency and the noise-free properties of N-Cubes.

1. Introduction

Most of the associative memories deal with a pattern-matching problem. In this paper, we address a cubical structure of associative memories, an N-Cube(Neural-network Cube) which stores training images in 2-D and retrieves those in the optimal sense. It is a class of auto-associative memories where training images are matched to sinusoidal functions in 1-D. These sinusoidal functions forms orthogonal basis vectors with which each training image constitute an energy local minimum.

Related works are Kosko's BAM(Bidirectional Associative Memories)[1], Kohonen's optimal linear associative memories[2], Wang's coding strategies in BAM[3], Kang's MANN(Multilayer Associative Neural Networks)[4,7], and Wang's MBDS(Modified Bidirectional Decoding Strategy)[5]. These works are based upon unsupervised learning in the case of bipolar or binary data. However, in the case of continuous gray-level images, more cautious treatments are necessary in order to guarantee robust pattern classification. An extension of MBDS, a CMBDS(Continuous MBDS) is proposed in Lee *et al.*[6] where ten gray-scale is used. Now, we propose an auto-associative memory with two layers in which arbitrary gray-scale images can be encoded and decoded even if the applied 2-D patterns are perturbed by noise.

Basically, two phases of processes are involved in N-Cubes, the training phase and the recalling phase. First, in the training phase, synaptic weights are determined from the Cartesian product of each element between a 2-D training image, x_m, which is given and a 1-D sinusoidal waveform, y_m, generated by the appropriate frequency and angle. The discrete orthogonal basis functions, y_m, have the following properties:

<1> $$y_m(k) = \sin\left(\frac{2\pi n_m}{N_w}k + \phi_m\right)$$

where

$$n_m = \left\lceil \frac{m}{2} \right\rceil, \quad \phi_m = \begin{cases} 0 & \text{(if m is odd)} \\ \frac{\pi}{2} & \text{(if m is even)} \end{cases}$$

<2> $$\langle y_i, y_j \rangle = 0 \qquad \text{if } i \neq j$$

i.e., $$\sum_{k=1}^{N_w} y_i(k) y_j(k) = 0$$

Second, in the recalling phase, the input-output relation Σ of N-Cubes can be stated as follows:

$$\Sigma : net_y(k) = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} x(i, j) \cdot C(i, j, k) \tag{1}$$

$$y(k) = g[\beta_1 \text{net}_y(k) - \theta_1] \tag{2}$$

where x and y are the input matrix and the output vector of an N-Cube, respectively; C is the $N_x \times N_y \times N_z$ weight cube of an N-Cube which is updated at each learning step; and net_y is the net input to the nonlinear sigmoid function g(hyper tangent function) whose parameters are β_1, θ_1.

2. Main Results: Architecture of N-Cubes

The learning algorithm in N-Cubes minimizes the following quadratic function $J_1(C_{(i,j,k)})$ which is a cubic version of the least-squares algorithm:

$$J_1(C_{(i,j,k)}) = \frac{1}{2}\sum_{m=1}^{M}\left(x_m(i,j) - \sum_{z=1}^{N_w}(C(i,j,z)y_m(z))\right)^2 + \frac{1}{2}((C_{(i,j,k)}-C_0)^T P_0^T (C_{(i,j,k)}-C_0)) \quad (3)$$

Note that the learning elements of an input matrix x_m or an output vector y_m take real values between -1 and +1, and the weight cube C is updated as a new training input-output pair is stored in an N-Cube. Also, the nonlinear threshold function is defined as

$$g(y) = \tanh \beta(y-\theta),$$

$$\frac{dg(y)}{dy} = \beta \mathrm{sech}^2\beta(y-\theta) \quad (4)$$

Fig. 1 shows the architecture of N-Cubes where the adaptive and the recall mechanisms for the only one pixel are presented for simplicity.

2.1 Training Algorithm of N-Cubes

The training algorithm for N-Cubes is stated as follows:

▶ Training Algorithm:

· Initialize

<1> $m = 0$, $C_0 = \bigcirc$, $P_0 = I$
where $C_m \in \mathbf{R}^{Nx} \times \mathbf{R}^{Ny} \times \mathbf{R}^{Nw}$
$P_m \in \mathbf{R}^{Nw} \times \mathbf{R}^{Nw}$

· Repeat

<2> $m = m + 1$, and let $n_m = \left\lceil \frac{m}{2} \right\rceil$

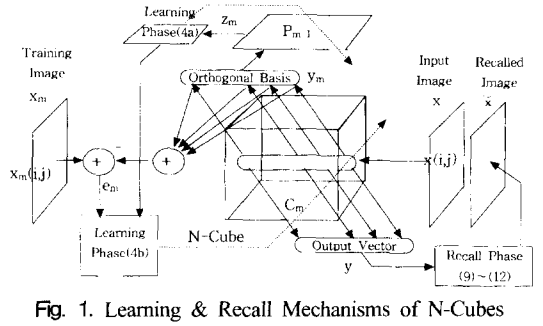


Fig. 1. Learning & Recall Mechanisms of N-Cubes

and $\phi_m = \begin{cases} 0 & \text{(if m is odd)} \\ \frac{\pi}{2} & \text{(if m is even)} \end{cases}$

<3> provide a training image
$x_m(i,j)$, $(i = 1, 2, \cdots N_X;\ j = 1, 2, \cdots N_Y)$
and the associated basis function
$y_m(k)$, $(k = 1, 2, \cdots N_w)$

$$y_m(k) = \sin\left(\frac{2\pi n_m}{N_w}k + \phi_m\right)$$

where $x_m \in \boldsymbol{R}^{Nx} \times \mathbf{R}^{Ny}$ and $y_m \in \boldsymbol{R}^{Nw}$

<4> define $z_m = P_{m-1}y_m \in \boldsymbol{R}^{Nw}$
and $\delta_m = 1 + y_m^T z_m$

<5> $P_m = P_{m-1} - z_m z_m^T/\delta_m$

<6> for all i, j & k compute

$$e_m(i,j) = x_m(i,j) - \sum_{z=1}^{Nw} C_{m-1}(i,j,z)\cdot y_m(z) \quad (5)$$

$$C_m(i,j,k) = C_{m-1}(i,j,k) + e_m(i,j)\cdot z_m(k)/\delta_m \quad (6)$$

· end

· Until {no more training images}

Note that $-1 < x_m(i,j) < 1$ is the (i,j)th element of the (m)th training matrix, and $y_m(k)$ is the (k)th element of the (m)th training vector. Eq.(5,6) is derived from thefollowing procedure. First, we define parameter vector ϕ_m, θ_m, Φ_M, X_M :

$$\phi_m = \begin{bmatrix} \phi_m(1) \\ \phi_m(2) \\ \vdots \\ \phi_m(N_w) \end{bmatrix} = \begin{bmatrix} y_m(1) \\ y_m(2) \\ \vdots \\ y_m(N_w) \end{bmatrix} = y_m$$

$$\theta_{m-1}^{(i,j)} = \begin{bmatrix} \theta_{m-1}^{(i,j)}(1) \\ \theta_{m-1}^{(i,j)}(2) \\ \vdots \\ \theta_{m-1}^{(i,j)}(N_w) \end{bmatrix} = \begin{bmatrix} C_{m-1}(i,j,1) \\ C_{m-1}(i,j,2) \\ \vdots \\ C_{m-1}(i,j,N_w) \end{bmatrix}$$

$$X_M^{(i,j)} = \begin{bmatrix} x_1(i,j) \\ x_2(i,j) \\ \vdots \\ x_M(i,j) \end{bmatrix}$$

$$\Phi_M = [\phi_1\ \phi_2\ \cdots\ \phi_M]^T = \begin{bmatrix} \phi_1^T \\ \phi_2^T \\ \vdots \\ \phi_M^T \end{bmatrix}$$

so that $\sum_{z=1}^{N_W} C_{m-1}(i, j, z)y_m(z) = \phi_m{}^T \theta_{m-1}^{(i,j)}$

Eq. (3) is redefined as follows:

$$J_1(\theta^{(i,j)}) = \frac{1}{2}[X_M^{(i,j)} - \Phi_M\theta^{(i,j)}]^T[X_M^{(i,j)} - \Phi_M\theta^{(i,j)}] + \frac{1}{2}((\theta^{(i,j)} - \theta_0^{(i,j)})^T P_0^{-1}(\theta^{(i,j)} - \theta_0^{(i,j)}))$$

For update θ, the required condition is $\frac{\partial J_m(\theta)}{\partial \theta} = 0$

$$\frac{\partial J_1(\theta^{(i,j)})}{\partial \theta^{(i,j)}} = [\Phi_M^T\Phi_M + P_0^{-1}]\theta^{(i,j)} - [P_0^{-1}\theta_0^{(i,j)} + \Phi_M^T X_M^{(i,j)}] = 0$$

If the last updated one is $\theta_M = \theta$, by substituting θ_M for $\theta_m (m = 1\sim M)$ in general

$$\theta_m^{(i,j)} = \theta_{m-1}^{(i,j)} + \frac{P_{m-1}\phi_m[x_m(i,j) - \phi_m^T\theta_{m-1}^{(i,j)}]}{1 + \phi_m^T P_{m-1}\phi_m}$$

$$\begin{bmatrix} C_m(i,j,1) \\ C_m(i,j,2) \\ \vdots \\ C_m(i,j,N_w) \end{bmatrix} = \begin{bmatrix} C_{m-1}(i,j,1) \\ C_{m-1}(i,j,2) \\ \vdots \\ C_{m-1}(i,j,N_w) \end{bmatrix} + \frac{z_m\left[x_m(i,j) - \sum_{z=1}^{N_w} C_{m-1}(i,j,z)y_m(z)\right]}{1 + \phi_m^T P_{m-1}\phi_m}$$

For the (i, j, k)th element

$$C_m(i, j, k) = C_{m-1}(i, j, k) + \frac{z_{m(k)}\left[x_m(i,j) - \sum_{z=1}^{N_w} C_{m-1}(i,j,z)y_m(z)\right]}{1 + \phi_m^T P_{m-1}\phi_m}$$

2.2 Recall Process of N-Cubes

In the recalling phase, when the input image x is applied to an N-Cube, it is obvious that one should find the matched output vector y_n minimizing the sum of squared errors from the corresponding output vector y. Therefore, we define the performance index J_2

$$J_2(n, \phi) = \frac{1}{2}\sum_{k=1}^{N_w}\left\{y(k) - \sin\left(\frac{2\pi n}{N_w}k + \phi\right)\right\}^2 \quad (7)$$

where $n = 1, 2, 3, \cdots$ and $\varphi =$ either 0 or $\pi/2$. To find n, φ that minimizes $J_2(n, \phi)$ i.e., $\min_{n,\phi} J_2(n, \phi)$, the following conditions must hold,

$$\frac{\partial J_2}{\partial \phi} = \sum_{k=1}^{N_w}\left\{y(k) - \sin\left(\frac{2\pi n}{N_w}k + \phi\right)\right\}\cdot\cos\left(\frac{2\pi n}{N_w}k + \phi\right) = 0 \quad (8)$$

$$\frac{\partial J_2}{\partial n} = \sum_{k=1}^{N_w}\left\{y(k) - \sin\left(\frac{2\pi n}{N_w}k + \phi\right)\right\}\cdot\frac{2\pi k}{N_w}\cos\left(\frac{2\pi n}{N_w}k + \phi\right) = 0 \quad (9)$$

Here, from the properties of discrete sinusoidal waves,

$$\sum_{k=1}^{N_w}\sin\left(\frac{2\pi n}{N_w}k + \phi\right)\cdot\cos\left(\frac{2\pi n}{N_w}k + \phi\right) = \sum_{k=1}^{N_w}\frac{1}{2}\sin\left[2\left(\frac{2\pi n}{N_w}k + \phi\right)\right] = 0 \quad (10)$$

$$\sum_{k=1}^{N_w}k\cdot\sin\left(\frac{2\pi n}{N_w}k + \phi\right)\cdot\cos\left(\frac{2\pi n}{N_w}k + \phi\right) = \sum_{k=1}^{N_w}\frac{1}{2}\sin\left[2\left(\frac{2\pi n}{N_w}k + \phi\right)\right] = 0 \quad (11)$$

Hence, the above two conditions reduce to

$$\sum_{k=1}^{N_w}y(k)\cdot\cos\left(\frac{2\pi n}{N_w}k + \phi\right) = 0 \quad (12)$$

$$\sum_{k=1}^{N_w}k\cdot y(k)\cdot\cos\left(\frac{2\pi n}{N_w}k + \phi\right) = 0 \quad (13)$$

The recall process is summarized as follows:

▶ Recall Process:

<1> provide an image x perturbed by noise

<2> let $\beta_1 = \frac{1}{96}, \theta_1 = -0.8\tanh\left(\frac{50}{m^2}\right)$

and $\beta_2 = 1$, $\theta_2 = 0$

<3> $net_y(k) = \sum_{i=1}^{N_x}\sum_{j=1}^{N_y} x(i,j)\cdot C(i,j,k)$

$y(k) = \tanh[\beta_1 net_y(k) - \theta_1]$

$(x \rightarrow \text{cube} \rightarrow y)$ (14)

<4> $net_x(i,j) = \sum_{k=1}^{N_x} y(k)\cdot C(i,j,k)$

$\tilde{x}(i, j) = \tanh[\beta_2 net_x(i, j) - \theta_2]$

$(y \rightarrow \text{cube} \rightarrow \text{intermediate } x)$ (15)

<5> find m that minimizes

$$J_2(m) = \frac{1}{2}\sum_{k=1}^{N_w}\{y(k)-y_m(k)\}^2 \quad (16)$$

i.e., find n and φ minimizing

$$J_2(n,\phi) = \frac{1}{2}\sum_{k=1}^{N_w}\left\{y(k)-\sin\left(\frac{2\pi n}{N_w}k+\phi\right)\right\}^2$$

<6> let $m = m_0$ (for the minimum J2) then

$$net_x(i,j) = \sum_{k=1}^{N_w} y_{m_0}(k)\cdot C(i,j,k)$$

$$\tilde{x}(i,j) = \tanh[\beta_2 net_x(i,j) - \theta_2]$$

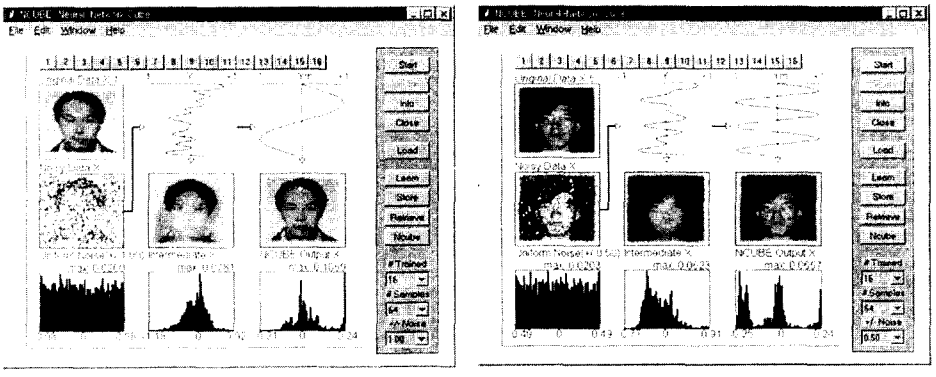
$(y \rightarrow \text{cube} \rightarrow x)$ (17)

3. Simulation Results for Image Recognition

In the simulation, the resolution of training images is chosen to be 64×64 with 256 gray level and the size of the orthogonal basis function is varied. (i.e., $N_x = 64$, $N_y = 64$, $N_w = 16,32,48,64...$)

The selection of the resolution of elements, rm N_w, in the orthogonal basis functions is important because the

Fig. 2. 16 Training Images (x1~x16)



(a) Noisy data in x_5 (Uniform Noise : 0.5) (b) Noisy data in x_6 (Uniform Noise : 1.0)

Fig. 3. Noise-free Recall of the applied images x and the associated N-Cube outputs

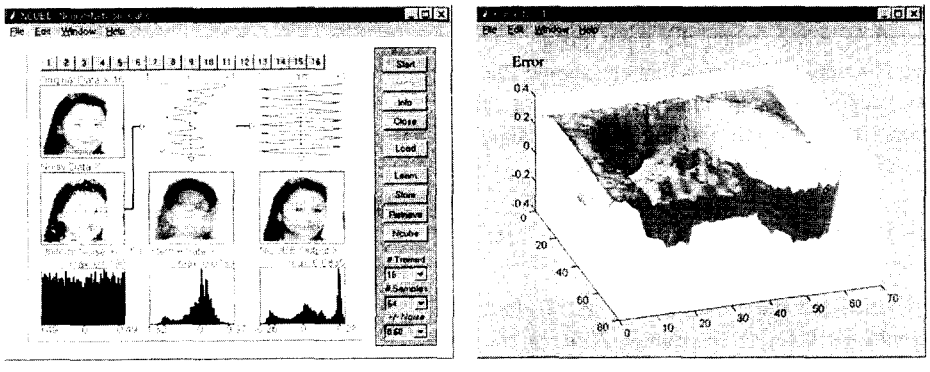
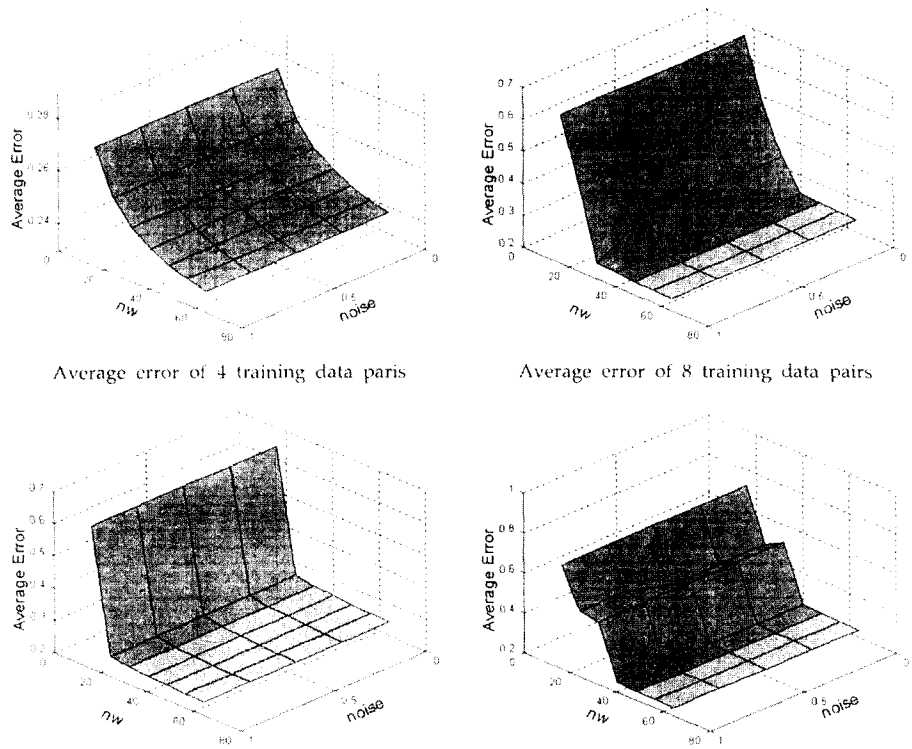


Fig. 4. Pixel-by-pixel errors in $\tilde{x}$ with respect to x_{16} Average error of 4 training data paris



Average error of 4 training data paris

Average error of 8 training data pairs

Average error of 12 training data pairs

Average error of 16 training data pairs

Fig. 5. (noise, N_w) vs. average errorgraph

stored data in an N-Cube should have enough discriminating grades for successful retrieval. The number of training pairs is also varied among 4, 8, 12, 16. Each pixel in the 256 gray level is first transformed into the values between -1 and +1 both for learning and for recalling. Fig. 2 represents the 16 training images used for the simulation. In Fig. 3, the test images are blurred by the uniform noise (-0.5~0.5) & (-1.0~1.0) and three bottom graphs in each result show error distributions of x with respect to the original images. Fig. 4 represents the recall result and the pixel-by-pixel errors between the 16th training image x_{16} and the N-Cube output $\tilde{x}$. If a error of the recall data is smaller then ± 0.25, we consider it as a successful retrieval. Fig. 5, shows the average error of a training pairs, as noise and N_w are varying. The average error only depends on rm N_w and has noise free characteristic

4. Conclusions

We introduce a cubical neural network structure of the auto-associative memory, an N-Cube, which performs learning and recalling 2-D images simultaneously. Primarily, N-Cubes utilize two performance criteria, J_1 for learning and J_2 for successful recall. Moreover, N-Cubes have the properties of robustness under noise, gray-level preserving within the minimum grade-errors, and large storage capacity. It is very promising that N-Cubes will be applied to a variety of applications such as image restoration, noise-free pattern recognition, control problems, function approximation, and so forth.

References

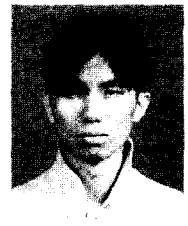
[1] B. Kosko, "Bidirectional Associative Memories", *IEEE Trans. Syst.* Man & Cybern., Vol. 18, pp. 49-60, Jan/Feb, 1988.

[2] T. Kohonen, Self-Organization and Associative Memory, New York: Springer-Verlag, 1984.

[3] Y. F. Wang, J. B. Cruz, and J. H. Mulligan Jr., "Two coding strategies for bidirectionalassociative memory", *IEEE Trans.* Neural Networks, Vol. 1, pp.

81-92, Mar., 1990.
[4] H. Kang, "Multi-layer Associative Neural Networks (M.A.N.N.) : Storage Capacity vs. Perfect Recall', *IEEE Transactions on Neural Networks*, Vol. 5, No. 5, pp. 812-822, Sep., 1994.
[5] W. J. Wang and D. L. Lee, "A Modified Bidirectional Decoding Strategy Based on the BAM Structure', IEEE Trans. Neural Networks, Vol. 4, No. 4, pp. 710-717, Jul., 1993.
[6] D. L. Lee and W. J. Wang, "Neighbor-Layer Updating in MBDS for the Recall of Pure Bipolar Patterns in Gray-Scale Noise", Vol. 6, No. 6, pp. 1478-1489, Nov., 1995.
[7] H. Kang, "Multi-layer Associative Neural Networks (M.A.N.N.) : Storage Capacity vs. Noise-free Recall', selected conference papers in Neural Networks Theory, Technology, and Applications, (ed.) P. K. Simpson, IEEE Technology Update Series, IEEE, pp. 215-221, 1996.

강 훈 (Hoon Kang)
제 9권 4호 참조



이 원 희 (Won-Hee Lee)
1998년 2월 : 중앙대학교 제어계측학과 (공학과)
1999년 : 중앙대학교 제어계측학과 대학원 석사과정
관심분야 : 신경회로망, 퍼지시스템 및 제어, 인공생명, 셀룰라 오토마타, 진화연산