BIDIRECTIONAL ASSOCIATIVE MEMORY ARCHITECTURE

The architecture of BAM network consists of two layers of neurons which are connected by directed weighted pare interconnections. The network dynamics involve two layers of interaction. The BAM network iterates by sending the signals back and forth between the two layers until all the neurons reach equilibrium. The weights associated with the network are bidirectional. Thus, BAM can respond to the inputs in either layer.

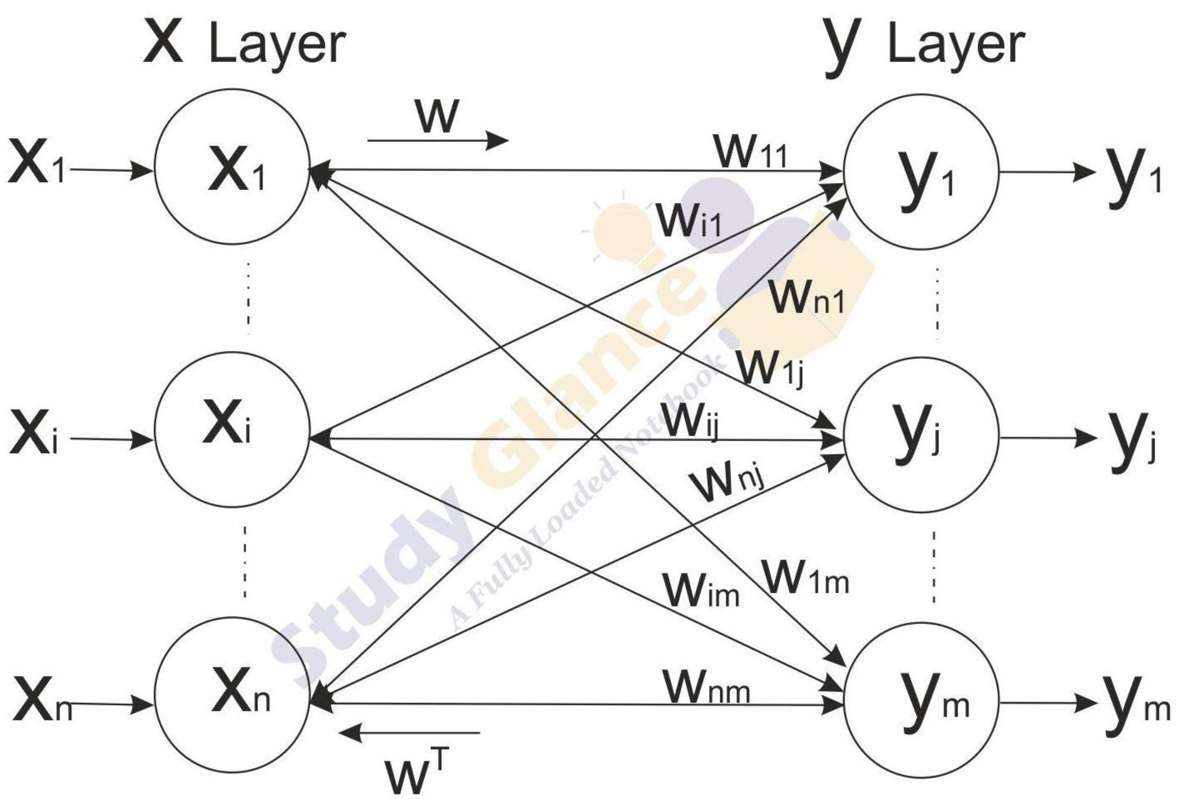

Figure shows a BAM network consisting of n units in X layer and m units in Y layer. The layers can be connected in both directions(bidirectional) with the result the weight matrix sent from the X layer to the Y layer is W and the weight matrix for signals sent from the Y layer to the X layer is W^T. Thus, the Weight matrix is calculated in both directions.

Determination of Weights

Let the input vectors be denoted by s(p) and target vectors by t(p). p = 1, ... , P. Then the weight matrix to store a set of input and target vectors, where

$s(p) = (s_1(p), .. , s_i(p), ... , s_n(p))$
$t(p) = (t_1(p), .. , t_j(p), ... , t_m(p))$

can be determined by Hebb rule training a1gorithm. In case of input vectors being binary, the weight matrix $W = \{w_{ij}\}$ is given by

$$w_{ij}=\sum_{p=1}^{P}[2s_i(p)-1][2t_j(p)-1]$$

When the input vectors are bipolar, the weight matrix $W = \{w_{ij}\}$ can be defined as

$$w_{ij}=\sum_{p=1}^{P}[s_i(p)][t_j(p)]$$

The activation function is based on whether the input target vector pairs used are binary or bipolar

Activation Function for the Y-Layer

1. With Binary Input Vectors

y_j <u>*With Binary Input Vectors*</u>

$$y_j = \begin{cases} 1, & \text{if } y_{in\,j} > 0 \\ y_j, & \text{if } y_{in\,j} = 0 \\ 0, & \text{if } y_{in\,j} < 0 \end{cases}$$

$\hat{y}_j$ <u>*With Bipolar Input Vectors*</u>

$$y_j = \begin{cases} 1, & \text{if } y_{in\,j} > \theta_j \\ y_j, & \text{if } y_{in\,j} = \theta_j \\ -1, & \text{if } y_{in\,j} < \theta_j \end{cases}$$

Activation Function for the X-Layer

1. <u>*With Binary Input Vectors*</u>

$$x_i = \begin{cases} 1, & \text{if } x_{in\,j} > 0 \\ x_i, & \text{if } x_{in\,j} = 0 \\ 0, & \text{if } x_{in\,j} < 0 \end{cases}$$

$\hat{x}_i$ <u>*With Bipolar Input Vectors*</u>

$$x_i = \begin{cases} 1, & \text{if } x_{in\,i} > \theta_j \\ x_i, & \text{if } x_{in\,i} = \theta_i \\ -1, & \text{if } x_{in\,j} < \theta_i \end{cases}$$

Testing Algorithm for Discrete Bidirectional Associative Memory

Step 0: Initialize the weights to srore p vectors. Also initialize all the activations to zero.

Step 1: Perform Steps 2-6 for each testing input.

Step 2: Ser the activations of X layer to current input pauern, i.e., presenting the input pattern x to X layer and similarly presenting the input pattern y to Y layer. Even though, it is bidirectional memory, at one time step, signals can be sent from only one layer. So, either of the input patterns may be the zero vector

Step 3: Perform Steps 4-6 when the acrivacions are not converged.

Step 4: Update the activations of units in Y layer. Calculate the net input,

yinj=∑i=1nxiwij

Applying ilie activations, we obtain

yj=f(yinj)

Send this signal to the X layer.

Step 5: Updare the activations of unirs in X layer. Calculate the net input,

xini=∑j=1myjwij

Applying ilie activations, we obtain

xi=f(xini)

Send this signal to the Y layer.

Step 6: Test for convergence of the net. The convergence occurs if the activation vectors x and y reach equilibrium. If this occurs then stop, Otherwise, continue.