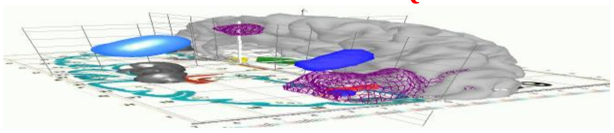


Neural Networks and Fuzzy Logic (630514)

Lecture 13

Supervised Learning in Neural Networks (Part 6)

ANN as Heteroassociative Memory (**Bidirectional associative memory**)

- The Hopfield network represents an **auto-associative** type of memory; it can retrieve a corrupted or incomplete memory but ***cannot associate this memory with another different memory***.
- Human memory is essentially associative. We use a chain of mental associations to recover a lost memory: We associate the faces with names, letters with sounds, etc.
- To associate one memory with another, we need a recurrent neural network capable of accepting an input pattern on one set of neurons and producing a related, but different, output pattern on another set of neurons.
- Bidirectional associative memory (**BAM**), first proposed by Bart Kosko.
- **BAM** is **hetero-associative**, meaning given a pattern it can return another pattern which is potentially of a different size.
- **BAM** is **content-addressable memory**.
- The Bidirectional associative memory maps **bipolar** binary vectors

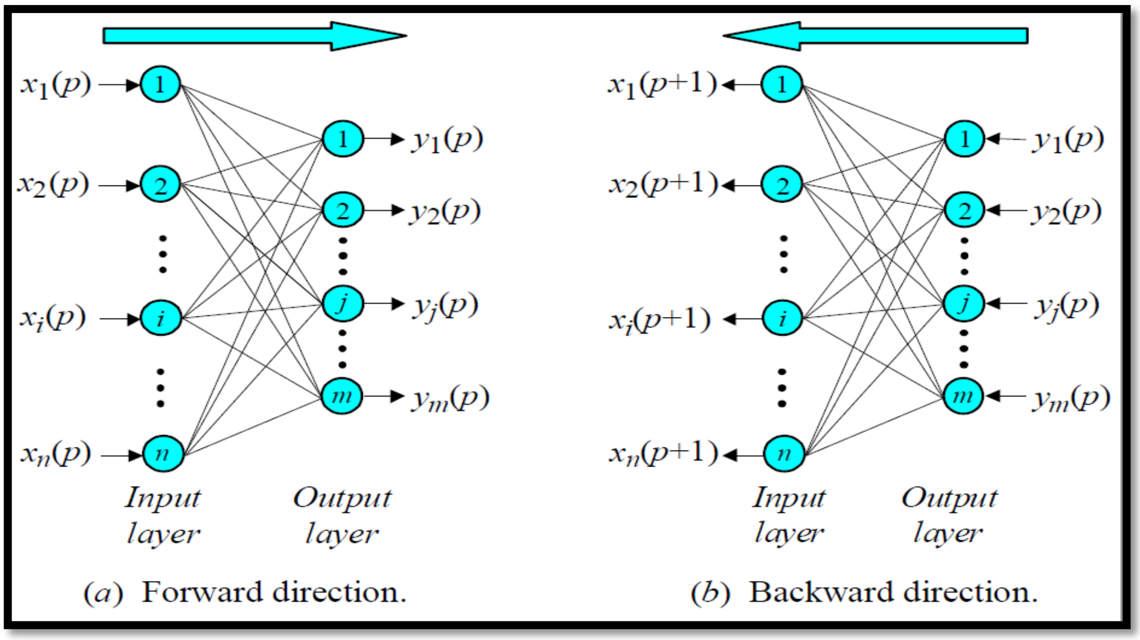
$$\mathbf{a} = [a_1 \quad a_2 \quad \dots \quad a_n]^t, \qquad a_i = \pm 1 \ , \quad i = 1, 2, \dots, n,$$

Input vectors

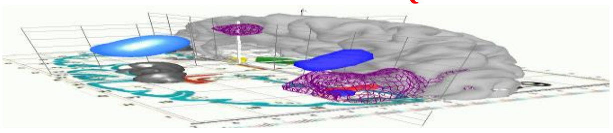
$$\mathbf{b} = [b_1 \quad b_2 \quad \dots \quad b_m]^t, \qquad b_i = \pm 1, \quad i = 1, 2, \dots, m,$$

Output vectors Or vice versa.

- The BAM ***correlation encoding scheme*** is extended to a general ***Hebbian learning law***.



BAM operation



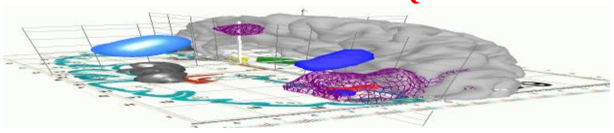
- The basic idea behind the **BAM** is to store pattern pairs so that when n-dimensional vector **X** from set **A** is presented as input, the BAM recalls m-dimensional vector **Y** from set **B**, but when **Y** is presented as input, the BAM recalls **X**. A BAM network is one with two fully connected layers, in which the links are all bi-directional (The neurons in input layer are fully interconnected to the neurons in the output layer. There is no interconnection among neurons in the same layer. The weight from input layer to output layer is same as the weights from output layer to input layer). A BAM network may be trained, or its weights may be worked out in advance. It is used to map a set of vectors **Xi** (input layer) to a set of vectors **Yi** (output layer).
- To develop the BAM, we need to create a **correlation matrix** for each pattern pair we want to store. The correlation matrix is the matrix product of the input vector **X**, and the transpose of the output vector $\mathbf{Y}^{\mathbf{T}}$. The **BAM weight (memory)** matrix is the sum of all correlation matrices, that is

$$W = \sum_{m=1}^{M} X_m Y_m^T \qquad \mathbf{1}$$

Where **M** is the number of pattern pairs to be stored in the BAM. The weight matrix associates all **X-Y** pairs.
- If a BAM network is used to implement an auto-associative memory then the input layer is the same as the output layer, i.e., there is just one layer. This network can be used to retrieve a pattern given a noisy or incomplete pattern.
- **Stability of the BAM:** The BAM is ***unconditionally stable***. This means that any set of associations can be learned without risk of instability.
- The **BAM** is capable of **error correction.**
- The output is updated ***asynchronously*** (for a given time, only a single neuron is allowed to update its output).
- A content-addressable memory is a type of memory that allows for the recall of data based on the degree of similarity between the input pattern and the patterns stored in memory. It refers to a memory organization in which the memory is accessed by its content as opposed to an explicit address like in the traditional computer memory system. Therefore, this type of memory allows the recall of information based on partial knowledge of its contents.

Bidirectional associative memory training algorithm:

1) **Storage (learning):** In the learning step for BAM we need to find weight matrix between, M pairs of patterns (fundamental memories) are stored in the synaptic weights of the network according to the equation **1**



Assume

Set A: $X_1 = \begin{bmatrix}1\\1\\1\\1\\1\\1\end{bmatrix}$ $X_2 = \begin{bmatrix}-1\\-1\\-1\\-1\\-1\\-1\end{bmatrix}$ $X_3 = \begin{bmatrix}1\\1\\-1\\-1\\1\\1\end{bmatrix}$ $X_4 = \begin{bmatrix}-1\\-1\\1\\1\\-1\\-1\end{bmatrix}$

Set B: $Y_1 = \begin{bmatrix}1\\1\\1\end{bmatrix}$ $Y_2 = \begin{bmatrix}-1\\-1\\-1\end{bmatrix}$ $Y_3 = \begin{bmatrix}1\\-1\\1\end{bmatrix}$ $Y_4 = \begin{bmatrix}-1\\1\\-1\end{bmatrix}$

The input layer = 6 neurons and the output layer = 3 neurons

The weight matrix W

$$W = \begin{bmatrix}1\\1\\1\\1\\1\\1\end{bmatrix}\begin{bmatrix}1 & 1 & 1\end{bmatrix} + \begin{bmatrix}-1\\-1\\-1\\-1\\-1\\-1\end{bmatrix}\begin{bmatrix}-1 & -1 & -1\end{bmatrix} + \begin{bmatrix}1\\1\\-1\\-1\\1\\1\end{bmatrix}\begin{bmatrix}1 & -1 & 1\end{bmatrix} + \begin{bmatrix}-1\\-1\\1\\1\\-1\\-1\end{bmatrix}\begin{bmatrix}-1 & 1 & -1\end{bmatrix}$$

$$= \begin{bmatrix}4 & 0 & 4\\4 & 0 & 4\\0 & 4 & 0\\0 & 4 & 0\\4 & 0 & 4\\4 & 0 & 4\end{bmatrix}$$

2) Testing

- We need to confirm that the BAM is able to
  - recall Y_m when presented X_m.
  - recall X_m when presented Y_m

  Using

$$Y_m = sign\,(W^T X_m), \qquad m = 1.2, \dots, M$$

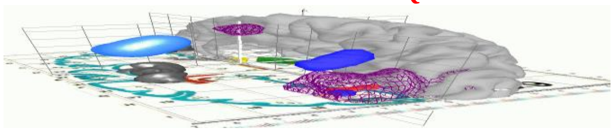
$$Y_1 = sign\,(W^T X_1) = sign\left\{\begin{bmatrix}4 & 4 & 0 & 0 & 4 & 4\\0 & 0 & 4 & 4 & 0 & 0\\4 & 4 & 0 & 0 & 4 & 4\end{bmatrix}\begin{bmatrix}1\\1\\1\\1\\1\\1\end{bmatrix}\right\} = \begin{bmatrix}1\\1\\1\end{bmatrix}$$

And using

$$X_m = sign\,(W Y_m), \qquad m = 1.2, \dots, M$$

$$X_3 = sign\,(W Y_3) = sign\left\{\begin{bmatrix}4 & 0 & 4\\4 & 0 & 4\\0 & 4 & 0\\0 & 4 & 0\\4 & 0 & 4\\4 & 0 & 4\end{bmatrix}\begin{bmatrix}1\\-1\\1\end{bmatrix}\right\} = \begin{bmatrix}1\\1\\-1\\-1\\1\\1\end{bmatrix}$$

All four pairs are recalled perfectly



3) Retrieval

- Present an unknown vector (**probe**) X (corrupted or incomplete version of a pattern from set **A** or **B**) to the BAM and retrieve a stored association:

$$X \neq X_m, \quad m = 1, 2, \ldots, M$$

  - **Initialize** the BAM:

$$X(0) = X, \quad p = 0$$

  - **Calculate** the BAM output at iteration p:

$$Y(p) = sign[W^T X(p)]$$

  - **Update** the input vector $X(p)$:

$$X(p+1) = sign[WY(p)]$$

  - **Repeat** the iteration until convergence, when input and output remain **unchanged**.

Example:

$X = (-1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1)$ *---single error pattern*

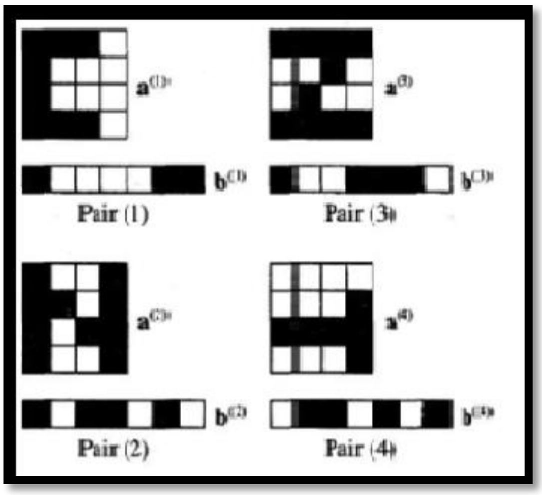
This probe applied as the BAM input and produces Y_1 , the Y_1 is then used as input and the BAM retrieves the vector X_1

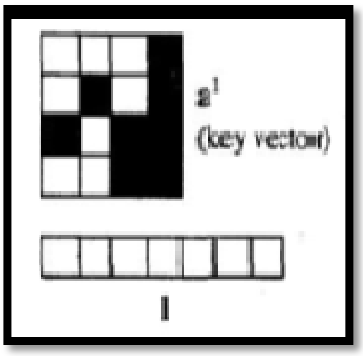
Problems of BAM Network

- **Storage capacity of the BAM**: The maximum number of associations to be stored in the BAM should ***not exceed the number of neurons in the smaller layer***.
- **Incorrect convergence**. The BAM may not always produce the closest association.

Example

Consider four **16-pixel** bit maps of letter characters should be associated to **7-bit binary** vectors as below:





$a^{(1)} = [1\ 1\ 1\ -1\ 1\ -1\ -1\ -1\ 1\ -1\ -1\ -1\ 1\ 1\ 1\ -1]^T$;

$b^{(1)} = [1\ -1\ -1\ -1\ -1\ 1\ 1]^T$;

$a^{(2)} = [1\ -1\ -1\ 1\ 1\ 1\ -1\ 1\ 1\ -1\ 1\ 1\ 1\ -1\ -1\ 1]^T$;

$b^{(2)} = [1\ -1\ -1\ 1\ 1\ 1\ -1]^T$;

$a^{(3)} = [1\ 1\ 1\ 1\ -1\ -1\ 1\ -1\ -1\ 1\ -1\ -1\ 1\ 1\ 1\ 1]^T$;

$b^{(3)} = [1\ -1\ 1\ 1\ -1\ 1\ -1]^T$;

$a^{(4)} = [-1\ -1\ -1\ -1\ -1\ -1\ -1\ 1\ 1\ 1\ 1\ 1\ -1\ -1\ -1\ 1]^T$;

$b^{(4)} = [-1\ 1\ 1\ -1\ 1\ -1\ 1]^T$;