4.5 **Experiment No. 5**

Aim:

Write a python Program for Bidirectional Associative Memory with two pairs of vectors.

Objective:

To learn the concept of Bidirectional Associative Memory.

Theory:

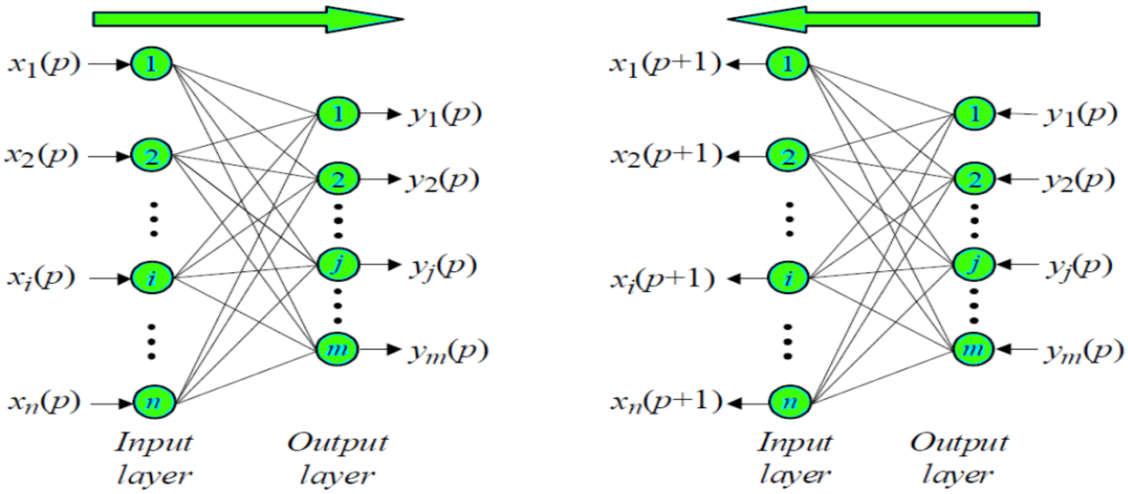
Bidirectional Associative Memory (BAM)

Bidirectional Associative Memory (BAM) is a supervised learning model in Artificial Neural Network. This is ***hetero-associative memory***, for an input pattern, it returns another pattern which is potentially of a different size. This phenomenon is very similar to the human brain. Human memory is necessarily associative. It uses a chain of mental associations to recover a lost memory like associations of faces with names, in exam questions with answers, etc. In such memory associations for one type of object with another, a Recurrent Neural Network (RNN) is needed to receive a pattern of one set of neurons as an input and generate a related, but different, output pattern of another set of neurons.

Why BAM is required?

The main objective to introduce such a network model is to store hetero-associative pattern pairs. This is used to retrieve a pattern given a noisy or incomplete pattern.

BAM Architecture: When BAM accepts an input of ***n***-dimensional vector ***X*** from set ***A*** then the model recalls ***m***-dimensional vector ***Y*** from set ***B***. Similarly when ***Y*** is treated as input, the BAM recalls ***X***.Class Template:



(*a*) Forward direction. (*b*) Backward direction.

Algorithm:

Step 0: Initialize the weights to store 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 pattern, i.e., presenting the input pattern x to X layer 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 activations are not converged.

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

$$y_{inj} = \sum_{i=1}^{n} x_i w_{ij}$$

Applying activations, we obtain

$$y_j = f(y_{inj})$$

Send this signal to the X layer.

Step 5: Update the activations of units in X layer. Calculate the net input,

$$x_{ini} = \sum_{j=1}^{m} y_j w_{ij}$$

Applying activations, we obtain

$$x_i = f(x_{ini})$$

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.

Input: Students to enter the input pattern

e.g. Weight matrix:

```
[[4 0 4]
 [4 0 4]
 [0 4 0]
 [0 4 0]
 [4 0 4]
 [4 0 4]]
```

Output:

```
Testing for input patterns: Set A
Output of input pattern 1
[[1]
 [1]
 [1]]
Output of input pattern 2
[[-1]
 [-1]
 [-1]]
Output of input pattern 3
[[ 1]
 [-1]
 [ 1]]
Output of input pattern 4
[[-1]
 [ 1]
 [-1]]
Testing for target patterns: Set B
Output of target pattern 1
[[1]
 [1]
 [1]
 [1]
 [1]
 [1]]

Output of target pattern 2
[[-1]
 [-1]
 [-1]
 [-1]
 [-1]
 [-1]]


Output of target pattern 3
[[ 1]
 [ 1]
 [-1]
 [-1]
 [ 1]
 [ 1]]


Output of target pattern 4
[[-1]
 [-1]
 [ 1]
 [ 1]
 [-1]
 [-1]]
```

Conclusion:

We have successfully implemented python Program for Bidirectional Associative Memory with two pairs of vectors.

Outcome:

Upon completion of this experiment, students will be able to:

Experiment level outcome (ELO1): Design various applications for storing and retrieving heterogeneous pattern pairs.

Questions:

1. What is associative learning in neural network?
2. Can data be stored directly in associative memory?
3. What are the different types of associative networks?
4. What is a node in associative networks?
5. The bidirectional associative memory is similar in principle to which model?