Q.No 9. Bidirectional Associative Memory (BAM) :- BAM is a (2 Marks)

hetero associative, content-addressable memory consist of 2 layers.

→ It is similar to linear associator but connections are bidirectional that is

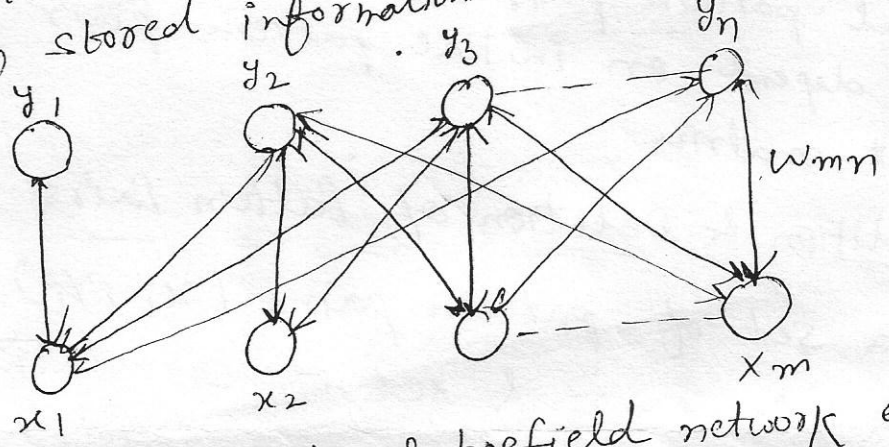
$$w_{ij} = w_{ji} \quad \text{for} \begin{cases} i = 1, 2 \dots m \\ j = 1, 2 \dots n \end{cases}$$

i.e BAM allows forward & backward flow of information.

→ In BAM the units in both layers serve as both input & output units depending on the direction of propagation.

→ Propagating the signals from X layer to Y layer makes the units in X layer to act as input units while units in Y layer act as output units. Same is true for other direction because memory is bidirectional.

→ BAM model can perform both auto & hetero associative recall of stored information.



→ Like is linear associator & hopfield network encoding in BAM can be carried out by using

$$W_k = x_k^T y_k \quad \text{(to store a single associated pattern pair)}$$

$$W = \alpha \sum_{k=1}^{p} W_k \quad \text{(to store simultaneously several associated pattern pairs)}$$

This process is generally called storage or encoding.

→ After encoding N/w can be used for decoding. The decoding process is as follows:

1) Input pattern can be applied either on X layer or on the Y layer.
2) When input pattern is applied, N/w will propagate the input pattern to other layer allowing the units in other layer to compute their o/p value.
3) Pattern that was produced by other layer is then propagated back to original layer & in original layer compute the O/P value
4) This process is repeated until further propagation & computations do not result in a change in the associates of units in both layers when final pattern pair is one of the stored associated pattern pair.
5) Final pattern pair that will be produced by the N/w depend on initial pattern pair & Connection weight matrix.

* Addition & Deletion of Pattern Pairs:

Given a set of pattern pairs (x_i, y_i) for $i = 1, 2, \ldots n$ & set of correlation Matrix M.

→ a new pair (x', y') can be added
→ an existing pair (x_j, y_j) can be deleted from memory model.

Addition: Add a new pair (x', y') to existing correlation Matrix M, then new matrix M_{new} is given by

$$M_{new} = x_1^T y_1 + x_2^T y_2 + \cdots + x_n^T y_n + x'^T y'$$

- **Deletion:** Subtract the matrix corresponding to an existing pair (X_j, Y_j) from matrix M, then correlation Matrix is given by:

$$M_{new} = M - (X_j^T Y_j)$$

* The addition & deletion of information is similar to the functioning of the system as a human memory exhibiting learning & forgetfulness.

* **Discrete BAM:** N/w propagates an input pattern X → to Y layer & Y layer compute the net input using:

Input is calculate using $y_i = \sum_{i=1}^{m} x_i w_{ij}$ $\quad i = 1, 2 \ldots m$, $j = 1, 2 \ldots m$

o/p " " $$y_i(t+1) = \begin{cases} +1 & \text{if } y_i > 0 \\ y_i(t) & \text{if } y_i = 0 \\ -1 & \text{if } y_i < 0 \end{cases}$$

pattern Y produced on Y layer is then propagated back to X layer & X layer compute value.

Input $$x_j = \sum_{i=1}^{n} y_i w_{ij}$$

o/p $$x_i(t+1) = \begin{cases} +1 & \text{if } x_i > 0 \\ x_i(t) & \text{" } x_i = 0 \\ -1 & \text{" } x_i < 0 \end{cases}$$

Energy function for DBAM $$E = -\sum_{i=1}^{m} \sum_{j=1}^{n} x_i w_{ij} y_j$$

* Continuous BAM: In this, the units use hyperbolic tangent O/P function. The units in X-layer have an extra external input I_i while units in Y-layer have an extra " " J_j. for $i = 1, 2 -- m$ & $j = 1, 2 -- m$.

These extra input lead to a modification in the computation of net inputs

$$x_i = \sum_{j=1}^{n} y_i w_{ij} + I_i$$

$$y_i = \sum_{j=1}^{m} x_i w_{ij} + J_i$$

Que-7 What is associative memory [2 marks]

Associative Memory :- Is a storehouse of associated patterns which are encoded in some form.

→ When the storehouse is triggered or incited with a pattern, the Associated pattern pair is recalled or o/p.

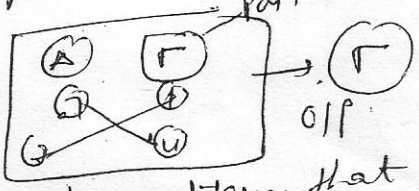
→ The input pattern could be an Exact Replica of stored pattern or a Distorted or Partial Repratal of stored patter.

OR

. An associative memory is a content-addressable structure that maps a set of input patterns to a set of o/p patterns.

→ A content-addressable structure is a type of memory that allows the recall of data stored on the degree of similarity b/w input pattern & pattern stored in memory.

Two Type [Auto associative
Hetero "



Auto A → retrives a previously stored pattern that most closely resembles the curred pattern (not only in content

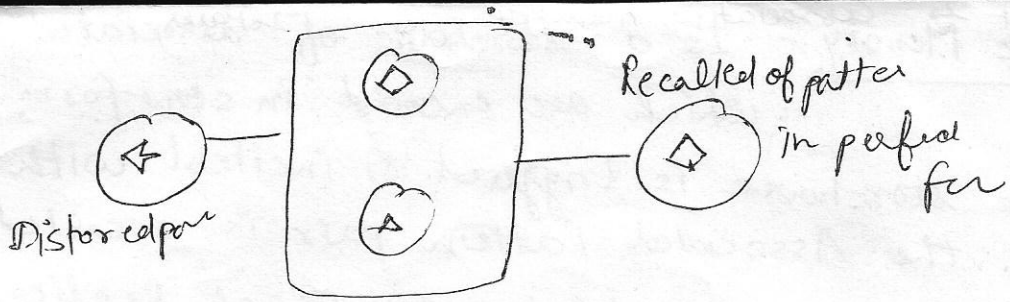
Hetero → diff from input pattern but possibly also type & format.

→ Neural N/w are used to implement these associative memory models called NAM (Neural associative memory)

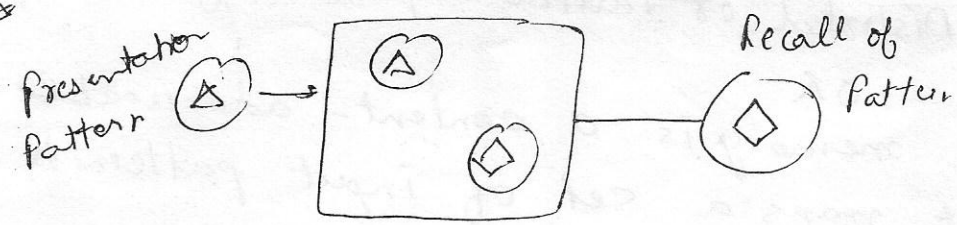
Use of : Auto → Image Refinement that a distored or partial pattern, the whole patten in it perfect form ca be recall. Also known a Autocorrelators.

Het → Association of patt.
also called Heterocorrelators

Ans



Hey



aman.sodhi91@gmail

* Recurrent Autoassociative Memory :- In order to produce an improved association, there is a need for suppressing the output noise at memory output in linear associative memory. This can be done by recycling of output to input & by thresholding the o/p.

→ The repetive process of recycling of the output to the input through the N/W can be performed by a Recurrent Neural Network (RNN).

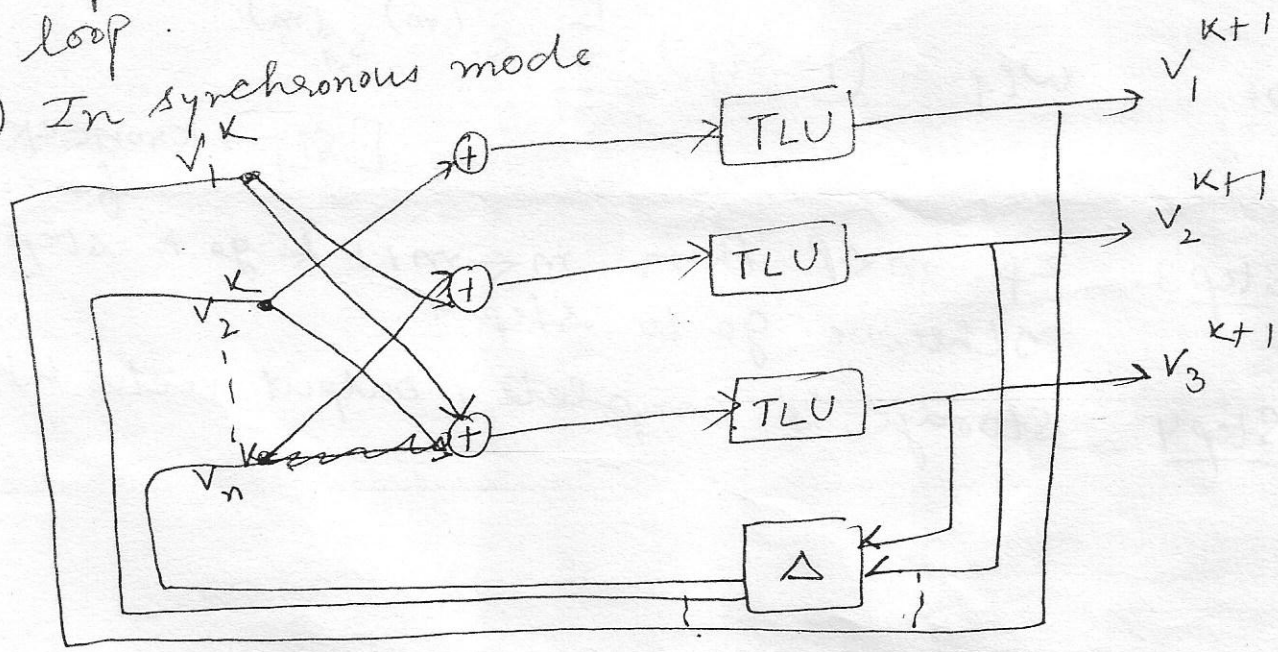
→ RNN is similar to linear associative memory. But it has feedback is dynamical.

There are 2 modes of operation of updation

Asynchronous Synchronous

1) Under the Asynchronous update mode, only one neuron is allowed to compute or change state at a time & then all outputs are delayed by a time u, that is produced by unit delay element in the feedback loop.

2) In synchronous mode



Explanation of figure dots represent the neurons. TLU threshold logic unit & these are used because associative memories are updated in discrete time.

$\Delta \rightarrow$ Unit delay in time.

Storage & Retrieval Algorithm :- Storage algorithm is called encoding & retrieval algorithm is called decoding process.

Given : P bipolar binary vector $\{ s^{(1)}, s^{(2)} \cdots s^{(P)} \}$

where s^m is $n \times 1$ for $m = 1, 2 \cdots P$

Initializing Vector v^0 is $n \times 1$.

Storage Algorithm

Step 1 : Weight matrix W is $(n \times n)$; $W \leftarrow 0$, $m \leftarrow 1$

Step 2 : vector s^m is stored. Therefore the storage algo. for calculating the weight matrix is

$$W \leftarrow s^m s^{m(t)} - I$$

or $$w_{ij} = (1 - s_{ij}) \sum_{m=1}^{P} s_i^{(m)} s_j^{(m)}$$

[$s_{ij} \rightarrow$ Kronecker func.)

Step 3 : If $m < P$ then $m \leftarrow m+1$ & go to step 2, otherwise go to step 4.

Step 4 : Storage is complete, output vector W.

* Retrieval Algorithm :-

Step 1 :- Cycle counter K is initialized $K \leftarrow 1$. Within the cycle counter i is initialized $i \leftarrow 1$ & n/w is initialized as $V \leftarrow V^0$.

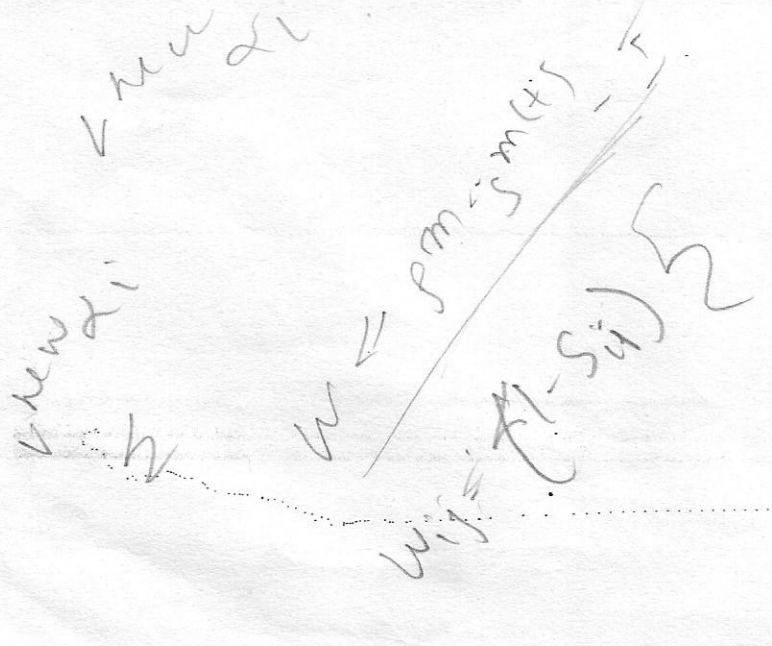
Step 2: Integers $1, 2, \ldots n$ are arranged in an ordered random sequences $\alpha_1, \alpha_2, \ldots \alpha_n$. If randomized sequence of updation is not used then $\alpha_1 = 1$, $\alpha_2 = 2$ ~ $\alpha_n = n$ or $\alpha_i = i$ at each update cycle.

Step 3: Neuron i is updated by computing $V^{new}_{\alpha_i}$.

$$net_{\alpha_i} = \sum_{j=1}^{n} W_{\alpha_i j} V_j$$

Step 4: If $i < n$ then $i \leftarrow i+1$ & go to step 3, otherwise go to step 5.

Step 5: If $V^{new}_{\alpha_j} = V_{\alpha_i}$ for $i = 1, 2 \ldots n$



Que-8: Define Single & Multi layer perceptron and their algorithm. 20 Marks

Classification Models :- A Neural Network classifies a given object presented to it according to the o/p ~~fired~~ activation. For binary outputs, 1 corresponds to one class & 0 corresponds to the other. (6)

2 types of classification models are discussed below :

1) Single layer perceptron
2) Multilayer perceptron

The Perceptron :- The perceptron is a program that learns concepts, i.e it can learn to respond with true (1) or false (0) for inputs we present to it, by repeatedly "studying" examples presented on it.

Example: for e.g, how do you teach a child to recognize what a chair is? You show him examples telling him ("This is a chair, that one is not a chair" ..... until the child learns the concept of what a chair is. This is exactly the idea behind the perceptron.

Single layer perceptron :- A single layer perceptron consists of an input layer & an output layer. An output unit will assume the value 1 if the sum of weighted inputs is greater than its threshold i.e

$$\sum w_{ji} x_i > \theta_j \qquad w_{ji} \text{ weight from } i \text{ to } j$$

$x_i \rightarrow$ input
$\theta_j \rightarrow$ threshold

There are 2 classes A & B. If $\sum w_{ji} x_i > \theta_j$ then object will be classified as class A otherwise class B. Explain

Linear Separability → As discussed in notes of Unit-4

* The Perceptron Algorithm

1) Weight Initialization :-
set all weights & node thresholds to small random numbers.

2) Calculation of Activation:
The activation level O_j of an output unit.

$$O_j = F_h(W_{ji} X_i - O_j)$$

F_h = Hard limiting fun.

$$F_h(a) = \begin{cases} 1 & \text{if } a > 0 \\ 0 & \text{Else} \end{cases}$$

3) Weight Training:

a) Adjust weights $\boxed{W_{ji}(t+1) = W_{ji}(t) + \Delta W_{ji}}$

$W_{ji}(t)$ → weight from unit i to unit j at time t.

ΔW_{ji} → weight adjustment.

b) Weight change may be computed by Delta learning rule $\Delta W_{ji} = \eta \delta_j X_i$

$\delta_j = T_j - O_j$

δ_j → Error at unit j

T_j → desired output activation

O_j → actual " " at output unit j.

c) Repeat the iterations will convergence.

* Limitations :- First, the output values of a perceptron can take on only one of two values (T or F).

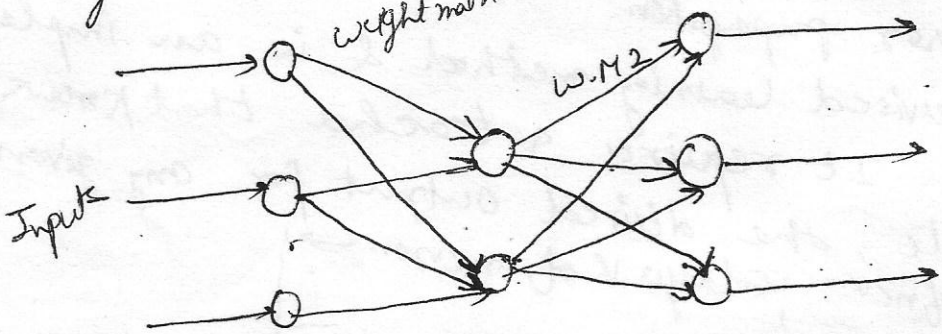
→ Second perceptrons can only classify linearly separable sets of vectors.

→ If a straight line or plane can be drawn to separate the input vectors into their correct categories the input vectors are linearly separable & the perceptron will find the solution. If the vectors are not linearly separable learning will never reach a point where all vectors are classified properly.

→ The most famous inability to solve problems with linearly nonseparable is boolean exclusive-or problem for these multilayer perceptron is used.

* Multilayer Perceptron :- An MLP is a network of simple neurons called perceptrons. It was introduced by Rosenblatt in 1958.

→ It is a feedforward neural N/w with at least one hidden layer. It can deal with nonlinear classification problem.



Neuron layer → 1 input layer, 1 or more hidden layer, 1 output layer.

Input value types → Binary

Activation func. → Sigmoid

Learning Method → Supervised

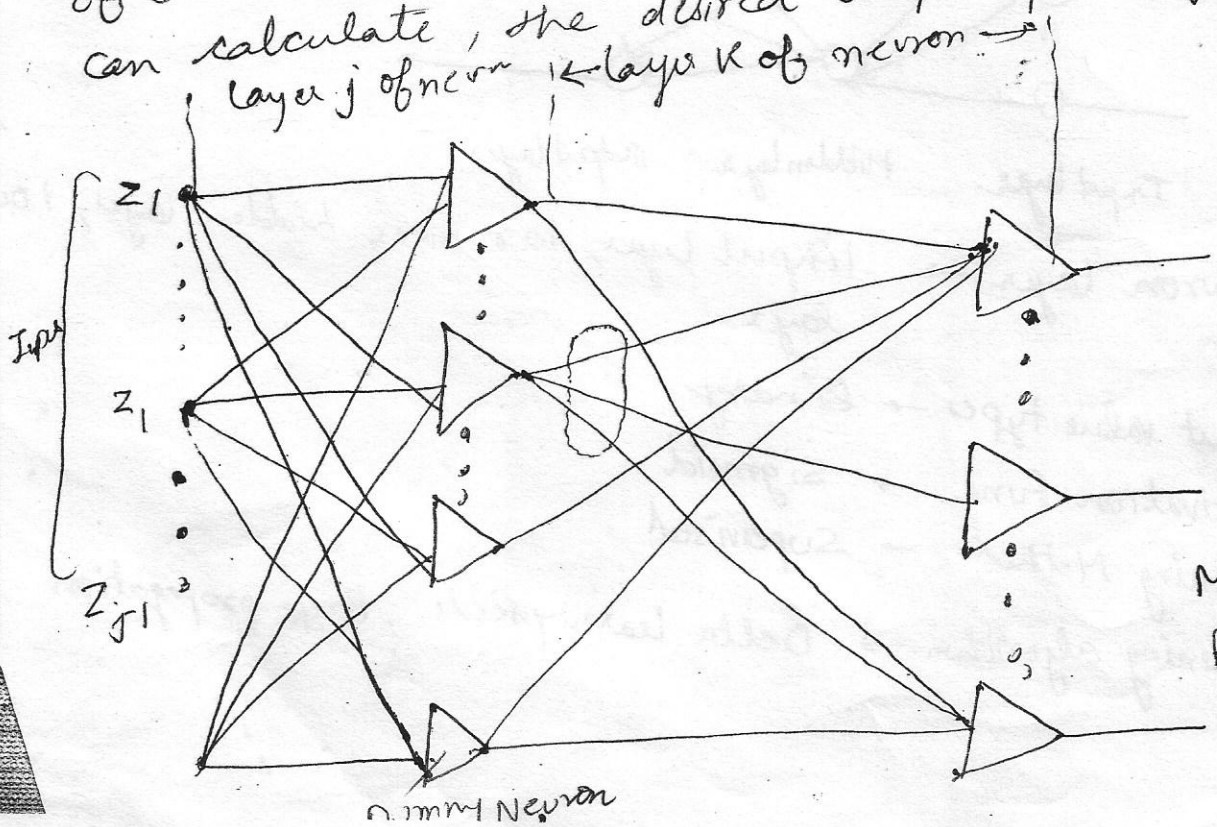
Learning algorithm → Delta learning Rule, Back propagation

* Neural Network learning :- The neural network has also been called the "connectionist". It contains a large no. of neuron like procesing elements & a large no. of weighted connections b/w the elements. The weights on the connections encode the knowledge of a N/w. The various learning algorithm are:

1. Backpropagation (Generalized Delta Learning Rule)
2. Reinforcement learning
3. ART Networks

1. Backpropagation (Generalized Delta Learning Rule) :-

- Backpropagation Network is Multilayer feedforward N/w with diff. transfer function in ANN & a more powerful learning rule. This learning rule is called "Backpropagation" which is a kind of gradient descent technique with backward error propagation.
- It is a Supervised learning method & is an implementation of Delta Rule. It requires a teacher that knows, or can calculate, the desired output for any given input



Backpropagation Algorithm

1) Weight Initialization – Set all the weights & node threshold to small random nos.

2) Calculation of Activation

The activation level O_j of hidden & o/p unit is determined by

$$O_j = F\left(\sum w_{ji} O_i - \theta_j\right)$$

$\theta_j \rightarrow$ threshold

$F \rightarrow$ sigmoid func.

$$F(a) = \frac{1}{(1+e^{-a})}$$

a = weight * input.

3) Weight Training :- a) Adjust weights by

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}$$

b) Weight change is computed by

$$\Delta w_{ji} = \eta \delta_j O_i.$$

$\Delta \rightarrow$ trial independent learning rate.

$\delta \rightarrow$ Error gradient at unit j.

c) Error gradient

$$\delta_j = O_j(1-O_j)(T_j - O_j)$$

$T_j \rightarrow$ desired o/p activation

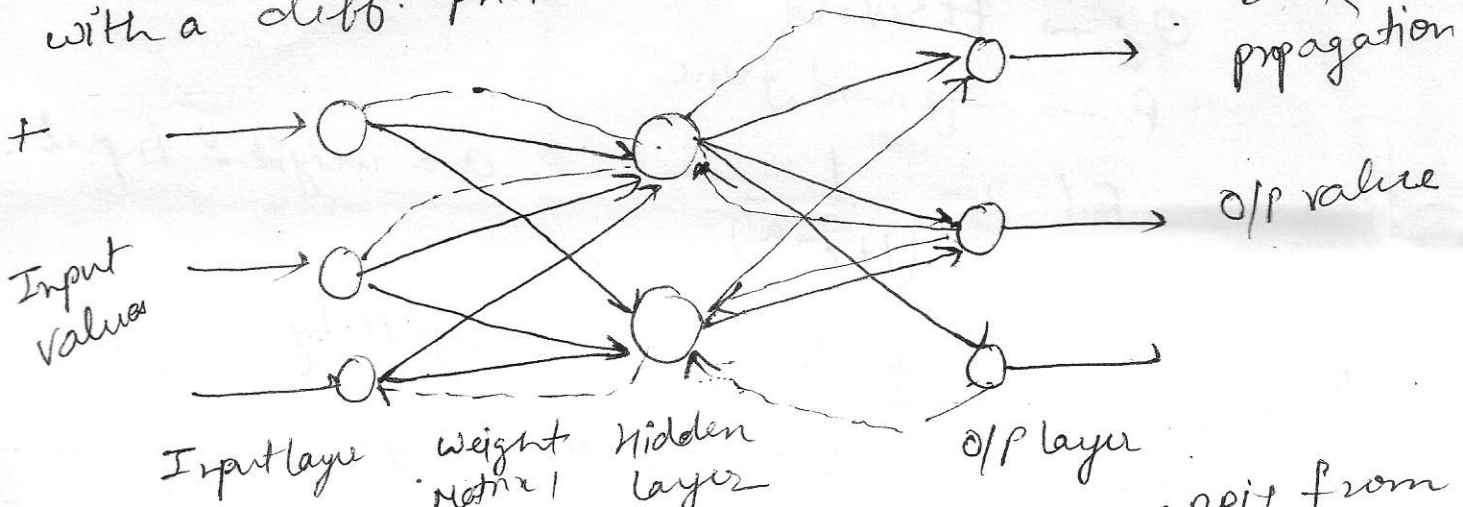
$O_j \rightarrow$ Actual

d) for hidden

$$\delta_j = O_j(1-O_j)\sum \delta_k w_{kj}.$$

e) Repeat iteration until convergence in terms of the selected error criterion.

Some situation where a BP NN might be good idea:
→ A large amount of input/output data is available, but you are not sure how to relate it to the o/p.
→ The sol. to the problem may change over time, within the bounds of the given input & output parameter.
→ Most common application is in image processing.
Some eg would be: identifying hand-written characters, matching a photograph of a person's face with a diff. photo in dB.



→ The backpropagation network learns a mapping from a set of input patterns to a set of o/p patterns. This N/w can be designed & trained to accomplish a wide variety of mappings. This ability comes from the nodes in the hidden layer which learns to respond to the feature found in the input pattern.
→ As the N/w is trained with diff. examples the N/w has the ability to generalize over similar features found in diff. patterns.
→ ∴ the hidden units must be trained to extract a sufficient set of general features applicable to all types of instances. To achieve this goal, the N/w should not be overtrained.

Factors Affecting Backpropagation Training :-

1) Initialization of Weights - The initialization affects the complete solution. If all the weights start out with equal weight values & if the sol. requires that unequal weights be developed, the N/w may not be trained properly. There are 2 methods for training an ANN
   - → Self-organizing ANN
   - → Back propagation ANN
2) Learning constant - Careful selection of step size(η) is often necessary to ensure the smooth convergence. However optimum value of η depends on problem being solved & there is no single learning constant value suitable for different training cases.

$$\eta = \frac{1.5}{N_1^2 + N_2^2 + \cdots + N_n^2}$$

N_i → no. of patterns

3) Steepness of Activation function : Both choice & shape of the activation fun. strongly affect the speed of N/w learning.

$$f(net) = \frac{2}{(1+\exp(-\lambda net))} - 1$$

Then $f'(net) = 2\lambda \exp(-\lambda net) / (1+\exp(-\lambda net))^2$

It reaches max value of $\frac{1}{2}\lambda$ at $net = 0$

Generally λ is kept at standard value of 1 & to control the learning speed using learning constant.

4) N/w size :- The N/w should have such a size as to capture the structure of the data & to model the problem. With some prior knowledge about the problem structure, one can form a good estimate of the proper N/w size.

5) Threshold value : A neuron generates the O/P if the weighted sum of the input exceeds the threshold value.

Associative Memory

What is Associative Memory ?

- An associative memory is a content-addressable structure that maps a set of input patterns to a set of output patterns.
- A content-addressable structure is a type of memory that allows the recall of data based on the degree of similarity between the input pattern and the patterns stored in memory.
- There are two types of associative memory : auto-associative and hetero-associative.
- An auto-associative memory retrieves a previously stored pattern that most closely resembles the current pattern.
- In a hetero-associative memory, the retrieved pattern is in general, different from the input pattern not only in content but possibly also in type and format.
- Neural networks are used to implement these associative memory models called NAM (Neural associative memory).

1. Associative Memory

An associative memory is a content-addressable structure that maps a set of input patterns to a set of output patterns. A content-addressable structure refers to a memory organization where the memory is accessed by its content as opposed to an explicit address in the traditional computer memory system. The associative memory are of two types : auto-associative and hetero-associative.

- An **auto-associative memory** retrieves a previously stored pattern that most closely resembles the current pattern.
- In **hetero-associative memory**, the retrieved pattern is in general different from the input pattern not only in content but possibly also in type and format.

4.1 Description of Associative Memory

An associative memory is a content-addressable structure that allows, the recall of data, based on the degree of similarity between the input pattern and the patterns stored in memory.

- **Example : Associative Memory**

The figure below shows a memory containing names of several people.

If the given memory is content-addressable,

Then using the erroneous string "Crhistpher Columbos" as key is sufficient to retrieve the correct name "Christopher Colombus."

In this sense, this type of memory is robust and fault-tolerant, because this type of memory exhibits some form of error-correction capability.

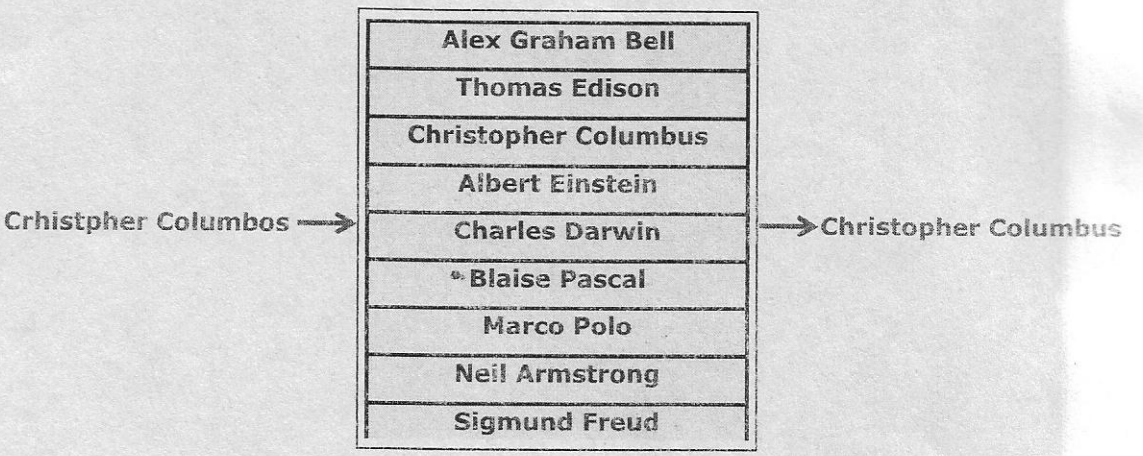


Fig. A content-addressable memory, Input and Output

Note : An associative memory is accessed by its content, opposed to an explicit address in the traditional computer memory system. The memory allows the recall of information based on partial knowledge of its contents.

[Continued from previous slide]

- Associative memory is a system that associates two patterns (X, Y) such that when one is encountered, the other can be recalled. The associative memory are of two types : auto-associative memory and hetero-associative memory.

Auto-associative memory

Consider, y[1], y[2], y[3], ... y[M], be the number of stored pattern vectors and let y(m) be the components of these vectors, representing features extracted from the patterns. The auto-associative memory will output a pattern vector y(m) when inputting a noisy or incomplete version of y(m).

Hetero-associative memory

Here the memory function is more general. Consider, we have a number of key-response pairs {c(1), y(1)}, {c(2), y(2)}, ... , {c(M), y(M)}. The hetero-associative memory will output a pattern vector y(m) if a noisy or incomplete verson of the c(m) is given.

- Neural networks are used to implement associative memory models. The well-known neural associative memory models are :
  - **Linear associater** is the simplest artificial neural associative memory.
  - **Hopfield model** and **Bidirectional Associative Memory (BAM)** are the other popular ANN models used as associative memories.

These models follow different neural network architectures to memorize information.

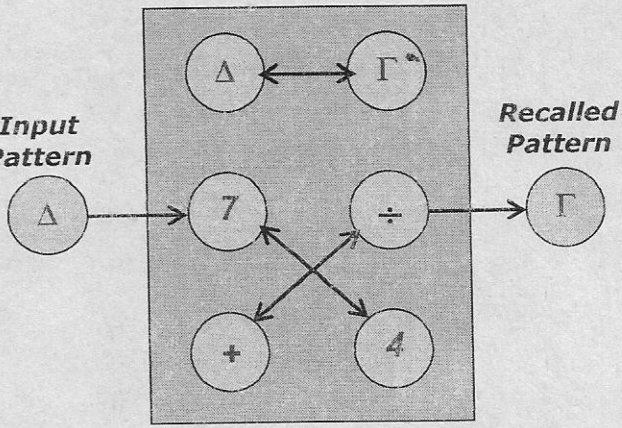
4.2 Working of Associative Memory

- **Example**

An associative memory is a storehouse of associated patterns which are encoded in some form.

- When the storehouse is triggered or excited with a pattern, then the associated pattern pair is recalled or appears at the output.
- The input could be an exact or distorted or partial representation of a stored pattern.

Fig below illustrates the working of an associated memory.



The associated pattern pairs

$(\Delta, \Gamma), (\div, +), (7, 4)$.

The association is represented by the symbol ↔

The associated pattern pairs are stored the memory.

Fig. Working of an associated memory

When the memory is triggered with an input pattern say Δ then the associated pattern Γ is retrieved automatically.

4.3 Associative Memory - Classes

As stated before, there are two classes of associative memory:

- auto-associative and
- hetero-associative memory.

An auto-associative memory, also known as auto-associative correlator, is used to retrieve a previously stored pattern that most closely resembles the current pattern;

A hetero-associative memory, also known as hetero-associative correlator, is used to retrieve pattern in general, different from the input pattern not only in content but possibly also different in type and format.

Examples

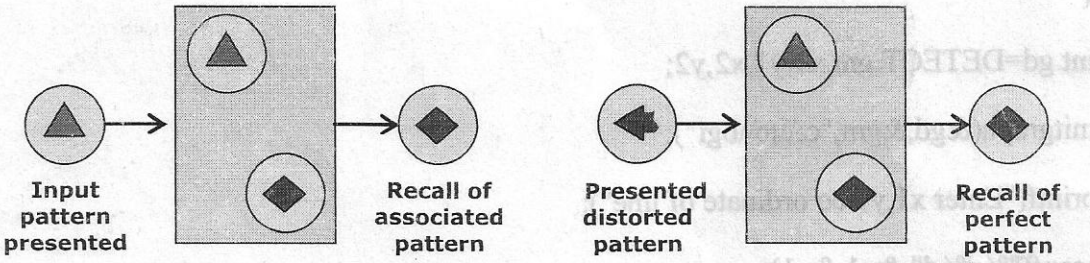


Fig. Hetero and Auto Associative memory Correlators

Back-Propagation Network

What is BPN ?

30(a)

- A single-layer neural network has many restrictions. This network can accomplish very limited classes of tasks.

  Minsky and Papert (1969) showed that a two layer feed-forward network can overcome many restrictions, but they did not present a solution to the problem as "how to adjust the weights from input to hidden layer" ?

- An answer to this question was presented by Rumelhart, Hinton and Williams in 1986. The central idea behind this solution is that the errors for the units of the hidden layer are determined by back-propagating the errors of the units of the output layer.

  This method is often called the Back-propagation learning rule.

  Back-propagation can also be considered as a generalization of the delta rule for non-linear activation functions and multi-layer networks.

- Back-propagation is a systematic method of training multi-layer artificial neural networks.

30(b)

1. Back-Propagation Network – Background

Real world is faced with a situations where data is incomplete or noisy. To make reasonable predictions about what is missing from the information available is a difficult task when there is no a good theory available that may to help reconstruct the missing data. It is in such situations the Back-propagation (Back-Prop) networks may provide some answers.

- A BackProp network consists of at least three layers of units :
  - an **input layer**,
  - at least one intermediate **hidden layer**, and
  - an **output layer**.

- Typically, units are connected in a feed-forward fashion with input units fully connected to units in the hidden layer and hidden units fully connected to units in the output layer.

- When a BackProp network is cycled, an input pattern is propagated forward to the output units through the intervening input-to-hidden and hidden-to-output weights.

- The output of a BackProp network is interpreted as a classification decision.

[Continued in next slide]

- With BackProp networks, learning occurs during a training phase. The steps followed during learning are :
  - each input pattern in a training set is applied to the input units and then propagated forward.
  - the pattern of activation arriving at the output layer is compared with the correct (associated) output pattern to calculate an error signal.
  - the error signal for each such target output pattern is then back-propagated from the outputs to the inputs in order to appropriately adjust the weights in each layer of the network.
  - after a BackProp network has learned the correct classification for a set of inputs, it can be tested on a second set of inputs to see how well it classifies untrained patterns.
- An important consideration in applying BackProp learning is how well the network generalizes.

NN Unit-3

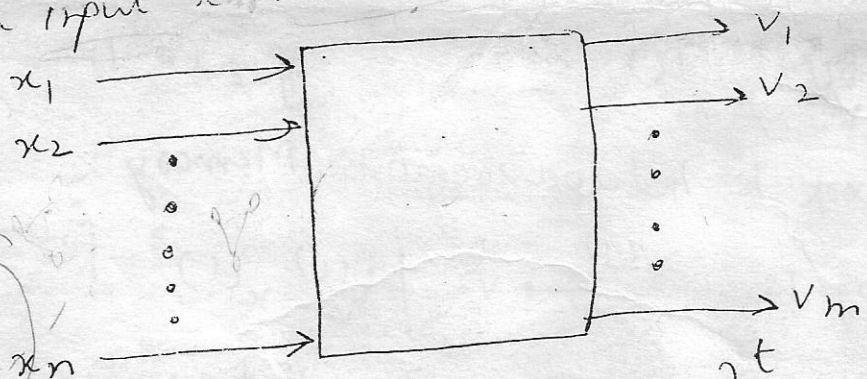
Introduction :- In this unit, we will interpret the system evolution as a movement of an input pattern toward another stored pattern called a prototype or stored memory.

→ Also look at building association for pattern retrieval & restoration.

→ Also study the dynamics of discrete-time system. Neural networks of this class are called associative memories.

(Associative Memory :- An efficient associative memory can store a large set of patterns as memories during recall, the memory is excited with a key pattern (search argument) containing a portion of information about a particular member of a stored pattern set. This particular stored prototype can be recalled through association of the key pattern & information memorized.

Basic Concepts :- fig. shows a general block diagram of associative memory performing an associative mapping of an input x into an output vector v.



$$x = [x_1\ x_2\ \cdots\ x_n]^t$$

$$v = [v_1\ v_2\ \cdots\ v_m]^t$$

→ The system shown maps vectors x to vectors v, in the pattern space R^n & output space R^m by performing the transformation.

$$V = M[x] \qquad \text{(1)}$$

$M \rightarrow$ Nonlinear Matrix-type operator. It has different meaning for each of memory models.

→ for dynamic memories, M also involves time variable. Thus V is available at memory o/p at a later time than the input has been applied.

→ The algorithm allowing the computation of M is called recording or storage algorithm.

→ The mapping in eq (1) on a key vector x is called a retrieval.

Let us assume that the memory has certain prototype vectors stored in such a way that once a key input has been applied, an output produced by the memory & associated with the key is memory response.

Assuming that there are p stored pairs of associator defined as

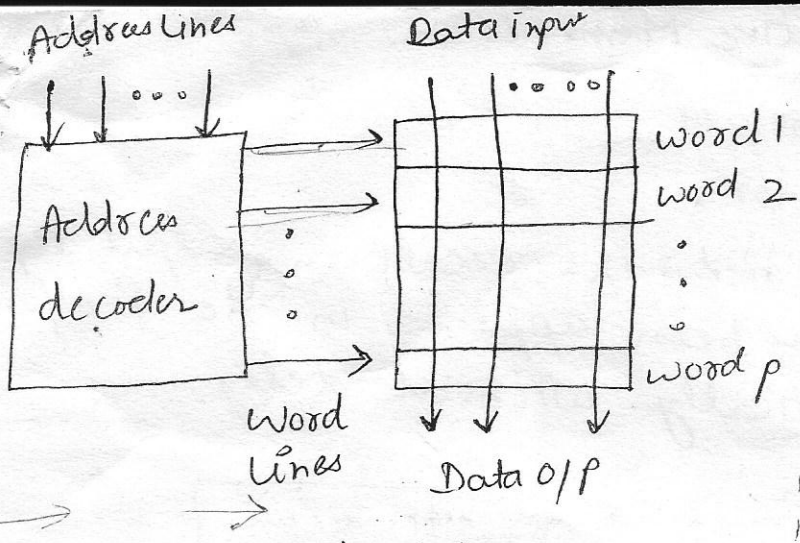
$$x^{(i)} \longrightarrow v^{(i)} \quad \text{for } i = 1, 2, \cdots p$$

$$\& \quad v^{(i)} \neq x^{(i)} \quad \text{for } i = 1, 2 \cdots p$$

Network is heteroassociative Memory.

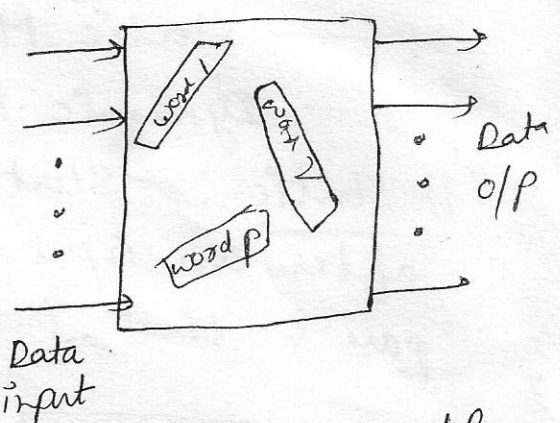
If mapping is $x^{(i)} \longrightarrow v^{(i)} \mid v^{(i)} = x^{(i)}$ for $i = 1, 2, \cdots p$

then Memory is autoassociative.



address-addressable Memory

fig (a)



Content addressable Memory

fig (b)

* Associative memory which uses neural N/w concepts bears very little resemblance to digital computer memory.

Let us compare two diff. addressing modes which are commonly used for data retrieval.

1) address-addressable Memory :- In digital computers, data are accessed when their correct addresses in the memory are given.

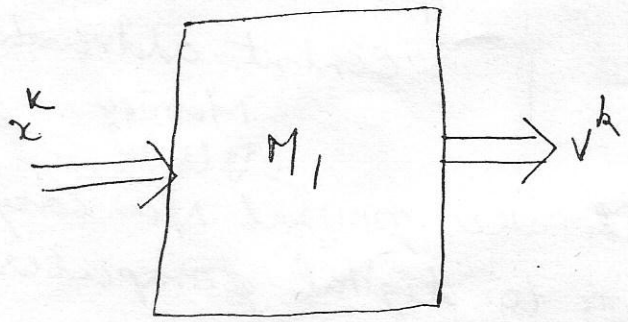
fig (a) shows a typical memory organization, data have input & output lines & a word line accesses & activates the entire word row of binary cells containing word data bits. This activation takes place whenever the binary address is decoded by the address decoder. The addressed word can be either "read" or replaced during the "write" operation.

2) Content addressable Memory :- The word in this memory are accessed based on the content of key vector. The mapping is implemented through dense connections. Sometimes involve feedback operation

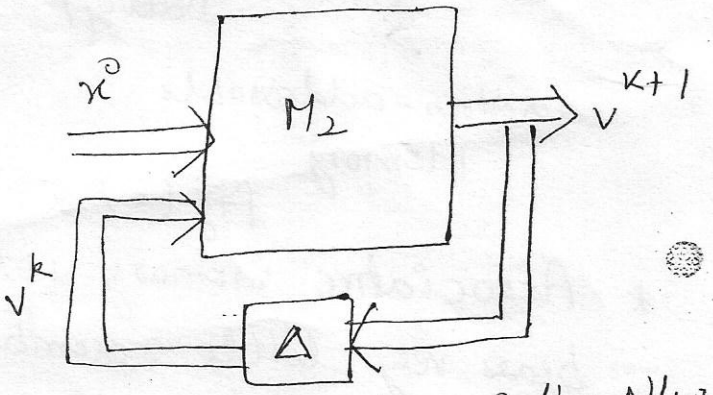
* Classes of Associative Memories :-

- Static Memories
- Dynamic Memories

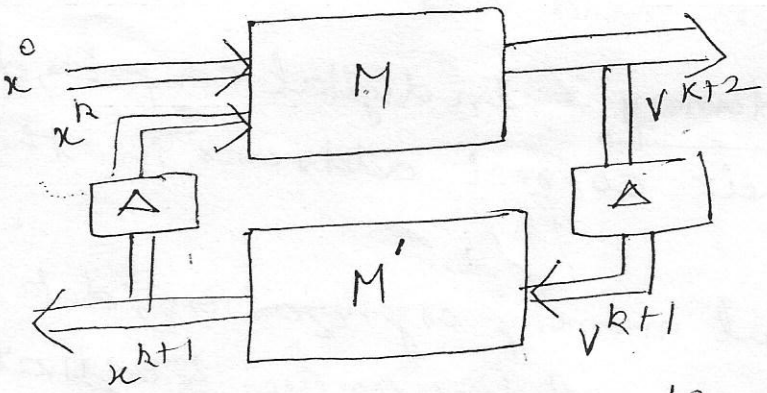
1) Static :- Static networks recall an output response after an input has been applied in one feedforward pass & ~~that~~ theoretically without delay.



Feedforward N/w
(a)



Recurrent Autoassociative N/w
(b)



c) Recurrent Heteroassociative N/w

→ The Static N/ws implement a feedforward operation of mapping without a feedback. Sometimes called non-recurrent. Static memory with block diagram shown in fig (a) performs the mapping as in Equation.

$$v^K = M_1[x^K]$$

K → Index of Recursion

M_1 → operator Symbol.

2) Dynamic Memory N/ws :- Dynamic Memory N/ws exhibit dynamic evolution in the sense that they converge to an equilibrium state acc. to recursive formula

$$v^{k+1} = M_2[x^k, v^k]$$

The block diagram is in fig (b).

3) Recurrent Heteroassociative N/w: fig (c) shows the block diagram of Recurrent Heteroassociative memory that operates with a cycle of 2Δ.

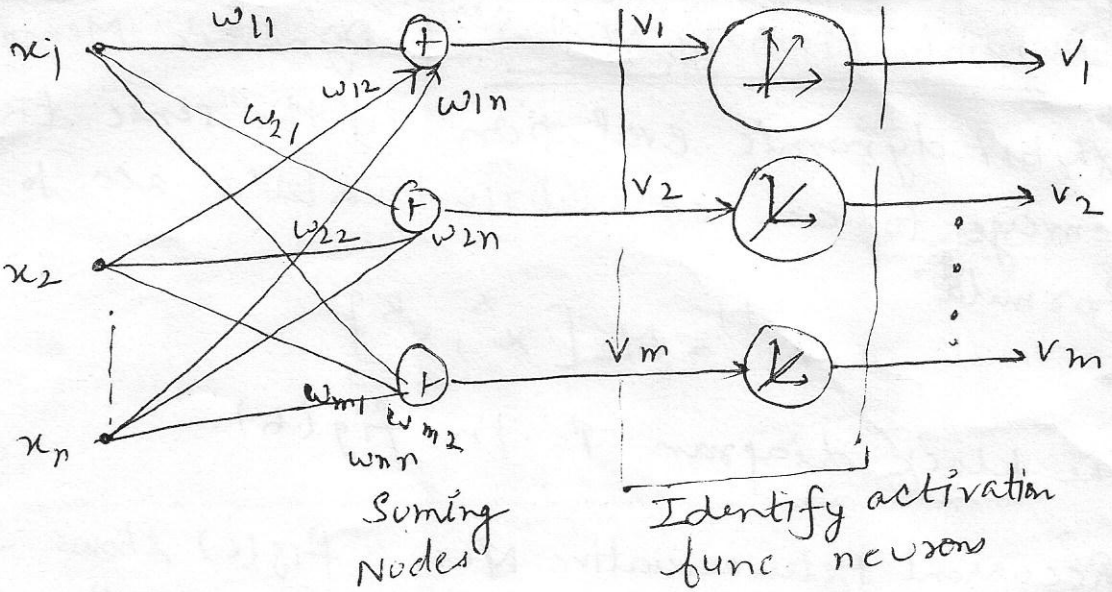
* Linear Associator :- Traditional associative memories are of the feedforward. The task required for the associative memory is to learn the association within p vector pairs $\{x^{(i)}, v^{(i)}\}$, for $i = 1, 2, \dots p$.

→ for the linear associative memory, an input pattern x is presented & mapped to the o/p by simply performing matrix multiplication operation

$$v = Wx$$

where x, v, W are matrics of size $n \times 1$, $m \times 1$, $m \times n$ resp.

The linear associative N/w diagram can be drawn as :



Linear Associator

Related terms of Associative Memory :-

(1) **Encoding :-** The process of constructing the connection weight matrix is called encoding. During encoding for an associated pattern pair (x_k, y_k), the weight values of the correlation matrix w_k are computed as:

$$(w_{ij})_k = (x_i)_k (y_j)_k \text{ where}$$

$(x_i)_k \rightarrow$ ith component of pattern x_k

$(y_j)_k \rightarrow$ jth " " y_k

for $i = 1, 2 \cdots m$ & $j = 1, 2 \cdots n$.

Constructing of the connection weight matrix w is accomplished by summing up the individual correlation matrices w_k i.e

$$w = \alpha \sum_{k=1}^{p} w_k$$

where as $\alpha \rightarrow$ Proportionality or normalizing constant.

(2) **Retrieval or Recollection :-** After memorization, the process of retrieving a stored pattern, given an input pattern is called decoding.

Given an input pattern x, the decoding or recollection is accomplished by :

→ first compute the net input to the o/p units using

$$\text{Inputs}_j = \sum_{i=1}^{m} x_i w_{ij}$$

where input j is weighted sum of the input or activation value of node j, for $j = 1, 2 \cdots n$.

Then determine the units o/p using a bipolar o/p func:

$$y_i = \begin{cases} +1 & \text{if input}_j \geq \theta \\ -1 & \text{otherwise} \end{cases}$$

where θ_j is the threshold value of o/p neuron j.

(3) Errors & Noise :- The input pattern may contain error & Noise or may be an incomplete version of some previously encoded pattern.

→ When a corrupted input pattern is presented, the N/w will retrieve the stored pattern that is closest to actual input pattern.

→ The presence of noise or errors results only in a more decrease rather than total degradation in the performance of the N/w.

Thus, associative memories are robust & fault tolerant because of many processing elements performing highly parallel & distributed computation.

4) Performance Measures :- The memory capacity & content-addressability are the measures of associative memory performance for correct retrieval. These two performance measures are related to each other.

- Memory capacity refers to the max. number of associated pattern pairs that can be stored & correctly retrieved
- Content addressability is the ability of the N/w to retrieve the correct stored pattern.

Total stored pairs in the memory is p & the associations are defined as;

$$x^{(i)} \rightarrow v^{(i)} \quad \text{for } i = 1, 2 \dots p$$

&

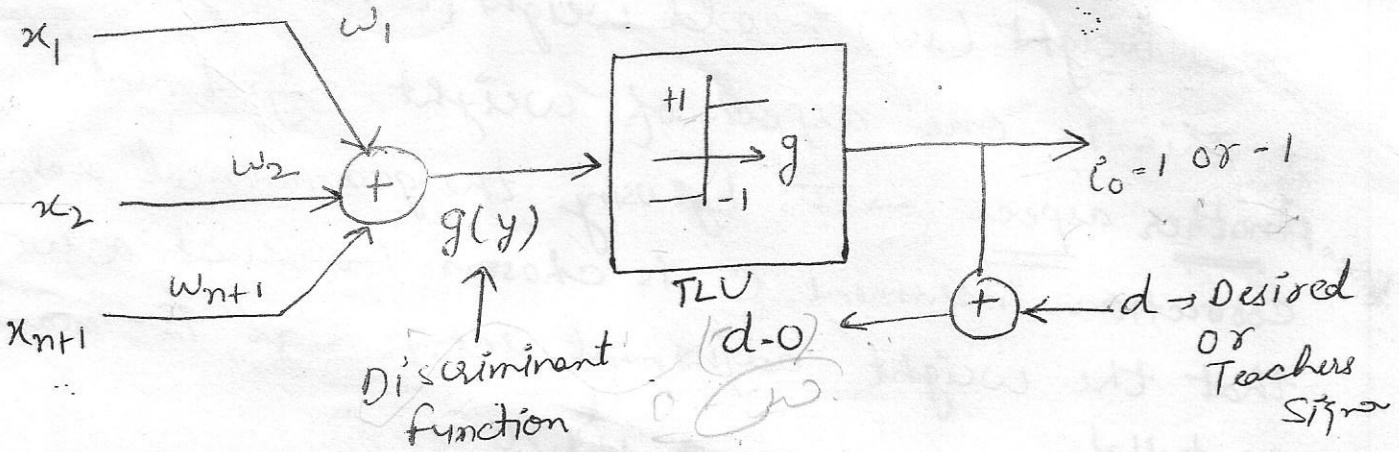
$$v^{(i)} \neq x^{(i)} \quad \text{for } i = 1, 2 \dots p$$

The N/w can be termed as hetero associative memory.

If $x^{(i)} \rightarrow v^{(i)}$

$v^{(i)} = x^{(i)}$ for $i = 1, 2 \dots p$ called auto associative memory.

* Training & Classification: Using the discrete perceptron:



Linear Dichotomizer using TLU Based Perceptron

TLU → Threshold Logical Unit.

Dichotomizer: Classifier that divides the pattern space into 2 classes (Di means 2). i.e x_1 & x_2 only.

* Training Sequence :- Sample pattern vectors $x_1, x_2, \ldots x_n$ called training sequence i.e sequence in which ANN is trained. Response is provided by the teacher & it specifies the classification information for each input vector.

→ The N/w learns from the experience by comparing the targeted correct response with the actual response to classify structure & the weight is adjused after each incorrect response based on error value generated

→ In this discrete perceptron concept the formula for adjused weights is $w^2 = w^1 + \Delta w(old)$

$$w' = w \pm cy$$

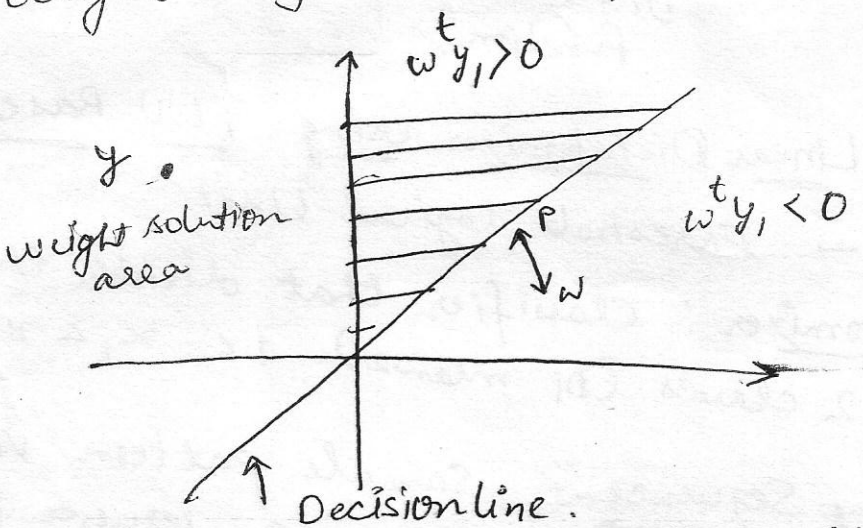
c → constant & $c > 0$ called correction increment

+ve → sign for pattern class 1, -ve → for class 2.

If a correct classification takes place under this rule, no adjustment of weight is made, then new weight (w') = old weight (w).

→ This is one aspect of weight adjustment.

Another aspect → Is by using the geometrical relationship correction increment C is chosen in such a way that the weight adjustment setup size is meaningful controlled.



Let p is distance of a point w' from the plane $w^t y = 0$ in (n+1) dimensional Euclidean space & it is calculated acc. to formula..

$$p = \frac{\pm w'^t y}{\|y\|}$$

p → distance & is always a non-negative scalar.

∴ can be written as

$$p = \frac{w'^t y}{\|y\|}$$

Now C must be selected such that corrected weight vector w^2 based on $w' = w \pm cy$ dislocated on the decision hyperplane $w'^t y = 0$ which is the decision hyperplane used for particular correction step:

$$w^{2t} y = 0$$

$$(w' \pm cy)^t y = 0$$

$$(w'^t + cy^t) y = 0$$

$$w'^t y \pm cy^t y = 0$$

$$cy^t y = \mp w'^t y$$

$$c = \mp \frac{w'^t y}{y^t y}$$

Since c is correction constant & is positive.

$$c = \frac{|w'^t y|}{y^t y}$$

* Single Discrete Perceptron Training Algorithm:

Given there are P training pairs.

$\{x_1, d_1, x_2, d_2 \text{ --- } x_p, d_p\}$ where

x_i is $(n \times 1)$ & d_i is (1×1), $i = 1, 2, \text{---} P$

Input vector $y_i = \begin{bmatrix} x_i \\ 1 \end{bmatrix}$ for $i = 1, 2, \text{---} P$

Step 1 $c > 0$ is chosen

Step 2 weights are initialized at w at small random values & $k \leftarrow 1$, $p \leftarrow 1$, $E \leftarrow 0$ where k is training step.

p is step counter within the training step or cycle.

Step 3 Training cycle starts.

Input is presented & output is computed.

$y \leftarrow y_p$, $d \leftarrow d_p$, $o \leftarrow \text{sgn}(w^t y)$

Step 4 Weights are updated.

$w \leftarrow w + \frac{1}{2} c(d-o) y$

Step 5 Cycle error is computed

$E \leftarrow \frac{1}{2}(d-o)^2 + E$ Previous cycle error

Step 6 If $p < P$ then $p \leftarrow p+1$, $k \leftarrow k+1$ & go to step 3, otherwise go to step 7.

Step 7 Training is completed. If $E = 0$, terminate the training, output weight & k.

If $E > 0$, then $E \leftarrow 0$, $p \leftarrow 1$, new training step or iteration starts, starting from step 3.

* **Single Layer Continuous Perceptron Networks for Linear separable classifications (SCPTA)**

Given P training pairs

$\{x_1, d_1, x_2, d_2 \text{---} x_p, d_p\}$

Input vectors are used

$y_i = \begin{bmatrix} x_i \\ 1 \end{bmatrix}$ for $i = 1, 2, \text{---} P$

where x_i is $(n \times 1)$, d_i is (1×1), $i = 1, 2 \text{---} P$

$p \rightarrow$ step counter

$k \rightarrow$ training step

1. Choose $\eta > 0$, λ (steepness constant) $= 1$,

$E_{max} > 0$.

2. Weight are initialized at w at small random value, counters & error are initialized.

$k \leftarrow 1$, $p \leftarrow 1$, $E \leftarrow 0$

3. Training starts → input y is presented & o/p is computed i.e

$y \leftarrow y_p$, $d \leftarrow d_p$, $o \leftarrow f(w^t y)$

4. Weights are updated:

$$w^{k+1} \leftarrow w^k + \frac{1}{2}\eta(d^k - o^k)(1 - o^{k^2})y$$

5. Error E is computed

$$E^{k+1} \leftarrow \frac{1}{2}(d^k - o^k)^2 + E^k$$

6. ~~Error &~~ If $p < P$, then $p \leftarrow p+1$, $k \rightarrow k+1$. & repeat step 3, otherwise go to step 7.

7. Training cycle is completed.

If $E < E_{max}$ terminate the session.

If $E \geq E_{max}$ then $E \leftarrow 0$, $p \leftarrow 1$ start new train cycle & go to step 3.

Que 9 Define the terms → Classification, Model and Decision Regions. (20 Marks) (1)

Unit - 2

Ans: Classification Model, Features & Decision Regions :-

1) Basic Meaning of Classifier :- Assume that a set of 8 points $a_0, \dots a_7$ in three dimensional space is available where

$a_0 = (-1, -1, -1)$
$a_1 = (-1, -1, 1)$
$a_2 = (-1, 1, -1)$
$a_3 = (-1, 1, 1)$
$a_4 = (1, -1, -1)$
$a_5 = (1, -1, 1)$
$a_6 = (1, 1, -1)$
$a_7 = (1, 1, 1)$

Elements of this set needs to be classified into two categories :-

(1) first category can be set of points with two or more +ve ones.

(2) second category contains all the remaining points that do not belong to first category.

∴ first category consists of a_3, a_5, a_6, a_7 & remaining points lie in second category.

This is the function of classifier that is to classify or to divide the patterns in different categories

→ One of the most important application of ANN is pattern classification :

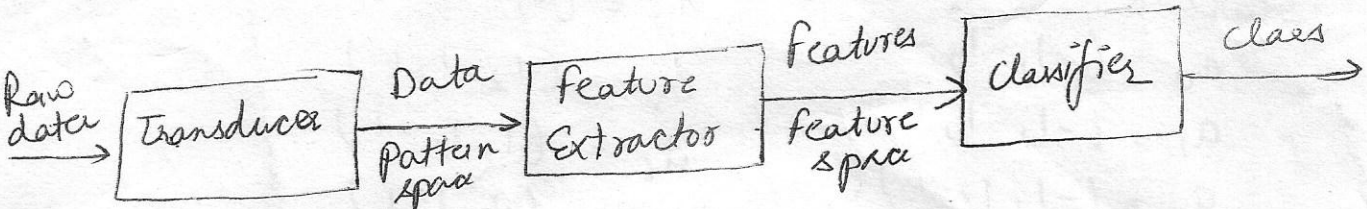
Pattern :- Pattern is the quantitative description of an object, event.

Patterns can be classified as spatial or temporal patterns

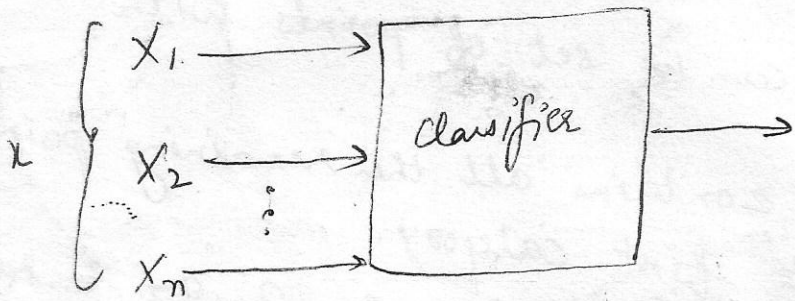
1) Spatial :- Ex. are pictures, whether maps video games etc

2) Temporal :- Ex. are speech signal, ECG etc. These patterns involve the ordered sequence of data appearing in time.

* Pattern Classification :- The main goal of pattern classification is to assign a physical object, event or phenomenon to one of the prespecified classes (categories) for e.g We classify various objects into different categories like living, non-living things etc.



a) Block diagram of classification system.



b) General diagram of pattern classifier.

→ Transducer generally convert one form of energy to another form of energy. Here the input is raw data space $x = [x_1, x_2 -- x_n]$ & o/p is transducer is data in pattern space that belong to a certain category that is to $= 1, 2 -- R$. converted data at the o/p can be compressed. Compressed data is called features.

→ Classifier at the end, convert or map the feature to the particular class.