UNIT- III (DIVYA BHARDWAJ) (1)

(NEURAL NETWORKS & FUZZY LOGIC) (32.)

Syllabus:- Bidirectional Associative Memories: Structure, retrieving a stored association, encoding associations, memory capacity. ART: ART architecture, ART classification operation, ART implementation and characteristics of ART.

→ Pattern Association:

- Pattern association is the process of forming association between related patterns.
- The pattern that has to be associated may be of the same type or of a different type.

(A) Associative Memory Networks :-

→ Associative memory net can be seen as a simplified model of a human brain which can associate similar patterns.

→ These are kind of neural network that work on the basis of pattern association, which means they can store different patterns and at the time of giving an output they can produce one of the stored patterns by matching them with the given input pattern.

→ Associative neural nets are single layer in which the weights are determined to store an asset of pattern associations.

→ They are of two kinds

Auto-associative Net	Hetero-Associative Net
If the input vector pair is <u>same</u> as the output vector pair then it results in an auto-associative Net	If the input vector pair is <u>different</u> from that of the output vector pair, then it results in a hetero-associative Net

The gives [wh

→ The associative net involves the common methods of training used for a single layer net. They are <u>Hebb rule</u> and <u>delta rule</u>.

i) <u>Hebb rule for pattern association</u> :- [Unsupervised Algorithm]

- Consider the input sample $S = (S_1 --- S_i --- S_n)$ and the output vector $t = (t_1 ---- t_j ---- t_m)$

- The <u>outer product of two vectors</u> can ~~be~~ also be used to find the weights based on Hebb rule.

- $$ST = \begin{bmatrix} S_1 \\ \vdots \\ S_i \\ \vdots \\ S_n \end{bmatrix} [t_1 ---- t_j --- t_m]$$

- ST is simply the matrix product of $n \times 1$ matrix $S = S^T$ and $1 \times m$ matrix $T = t$.

- The multiplication of the above matrices gives the weight matrix to store the association [which is found using Hebb rule]. ②

- In general, we can write the weight determination formula as,

$$\omega = \sum_{p=1}^{P} s^T(p)\, t(p)$$

ii) Delta Rule for pattern association :- [Supervised Algorithm]

- Delta learning rule is an iterative learning rule.
- This may be stated as "the adjustment made to a synaptic weight of a neuron is proportional to the product of the error signal and the input signal of the synapse".

- Delta rule for single output unit.

$$\Delta w_i = \alpha\,(t - y_{in})\, x_i$$

- Delta rule for several output unit.

$$\Delta w_{ij} = \alpha\,(t_j - y_{inj})\, x_i$$

Scanned with CamScanner

(B) HETERO ASSOCIATIVE MEMORY NEURAL NETWORKS :

→ They are static networks.

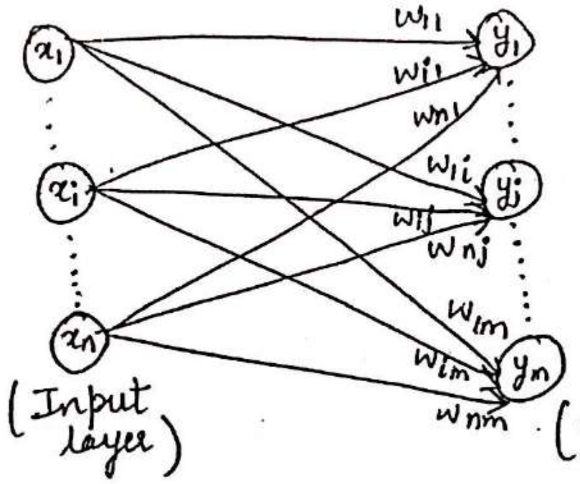
→ Delay operations can be done using hetero associative networks.

→ Weights may be found using Hebb rule or delta rule.

→ The net in the case finds an appropriate output vector that corresponds to an input vector x that can be either one of the stored patterns or the patterns, which may or may not be similar.

→ Architecture of Hetero Associative Net :

- The architecture resembles a single layer feed forward network.
- It consists of only one layer of weighted inter connections.
- There exist 'n' no of input neurons in the input layer and 'm' no of output neurons in output layer.
- The training process is based on Hebb rule.
- This is a fully Interconnected network, wherein inputs and outputs are different, hence it is called hetero-associative network.



** Note:

Training Algorithm for Hetero Associative is same as Auto Associative (Refer Page: 1)

** Take $j = 1 \text{ ---- } m$

Figure: Architecture of Hetero Associative Neural Net

→ Application Algorithm :- (Testing Algorithm)

Step-I: Weights are initialized using Hebb or delta rule

Step-II: For each input vector do steps 3 to 5.

Step-III: Set the activations for input layer units equal to current vector x_i.

Step-IV: Compute net input to the output units.

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

Step-V: Determine the activation of the output units

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

Note:-

Hetero associative networks use supervised learning (when delta Rule is used) in the sense that teacher supplies the proper output to associate with. and unsupervised learning also when Hebb Rule is used.

B) **Imp AUTO ASSOCIATIVE MEMORY NETWORK :-

→ In this the training input and target output vector should be same.

→ The performance of the net is better for bipolar vectors than the binary vectors.

→ In this, the weights on the diagonal are set to zero (weights with no self connection).
[setting the weights to zero improves the network ability to generalize or increase the plausibility of the net]. (acceptability)

Note: Learning in auto-associative networks is unsupervised because they just take input and try to organise a useful representation based on inputs, when Hebb Rule used and Supervised when delta rule is used

Architecture :

→ The architecture resembles a single layer feed forward network.

→ There exist 'n' no of input neurons in the input layer and 'n' no of output neurons in the output layer.

→ The training process is based on the Hebb learning rule.

→ This is a fully interconnected network, wherein the inputs and the outputs are same, hence called an auto associative network.

Training Algorithm :

- The parameters used in this net are as follows:

  $S(p)$ = sample vector $(S_1(p), S_2(p), ---- S_n(p))$

  X = Input vector $(x_1(p), x_2(p) --- x_n(p))$

  Y = Output vector $(y_1(p), y_2(p) ---- y_n(p))$

  W_{ij} = Weights between j^{th} output unit and i^{th} input unit.

- Initially, the weights of the auto associative net are taken as zero.

- This is a pattern associative supervised learning network wherein both the inputs and outputs are known; as a result activations are set for both the input and output units.

- Then the final weights are calculated based on Hebb learning rule.

Step-I: Initialize all weights, $i = 1 --- n$, $j = 1 ---- n$

$$W_{ij} = 0;$$

Step-II :- For each vector to be stored follow steps 3-4.

Step-III : Set activation for each input unit $i = 1 ---- n$

$$x_i = S_i$$

Step-IV :- Set activation for each output unit $j = 1 ---- n$

$$y_j = S_j.$$

Step-V: Adjust the weight for $i = 1$ to n and $j = 1$ to n

$$\boxed{W_{ij}(new) = W_{ij}(old) + x_i y_j}$$

The weights can also be determined from Hebb learning as,

$$\boxed{W = \sum_{p=1}^{P} s^T(p) \cdot s(p)}$$

→ Application Algorithm :- [Testing Algorithm]

Step-I :- Initialize the weights.

Step-II :- For each testing input follow steps 3 to 5.

Step-III : Set the activation of the input units equal to the input vector.

Step-IV : compute net input to each output unit: $j = 1, \ldots n$;

$$\boxed{y_{inj} = \sum x_i w_{ij}}$$

Step-V: Apply activation function $(j = 1, \ldots n)$

$$y_j = f(y_{inj}) = \begin{cases} 1 & \text{if } y_{inj} > 0 \\ 0 & \text{if } y_{inj} \leq 0 \end{cases}.$$

Storage Capacity :-

- An important criterion for an associative network is the no. of patterns it can store.
- The capacity of an auto associative net depends on the no. of components in the stored vector.

- More no. of vectors can be stored if they are mutually orthogonal.
- $n-1$ mutually orthogonal bipolar vector each with n components can be stored using the sum of the outer product weight matrices.
- Two vectors a and b are orthogonal if
  $$\sum a_i b_i = 0$$

** Imp

C BI-DIRECTIONAL ASSOCIATIVE MEMORY : (BAM)

Concept :-

→ Kosko developed BAM in the year 1988!

→ It is a hetero associative recurrent neural network consisting of two layers

→ The net associates a set of patterns by summing bipolar correlation matrices.

→ This type of net consists of two layers of neurons, which are connected by means of directional weighted connection paths.

→ The iteration is carried out by sending signals back and forth between two layers until all neurons reach equilibrium.

→ The net can respond to input on either layer.

→ The layers in this case are referred as X-layer and Y-layer instead of input and output layer, because the weights are bidirectional and the algorithm alternates between updating the activations of each layer.

→ Three forms of BAM are:
i) Binary.
ii) Bipolar.
iii) continuous.
} → Architecture is same for all.

⇒ Architecture of BAM :

→ The hetero associative BAM network has 'n' units in X-layer and 'm' units in Y-layer.

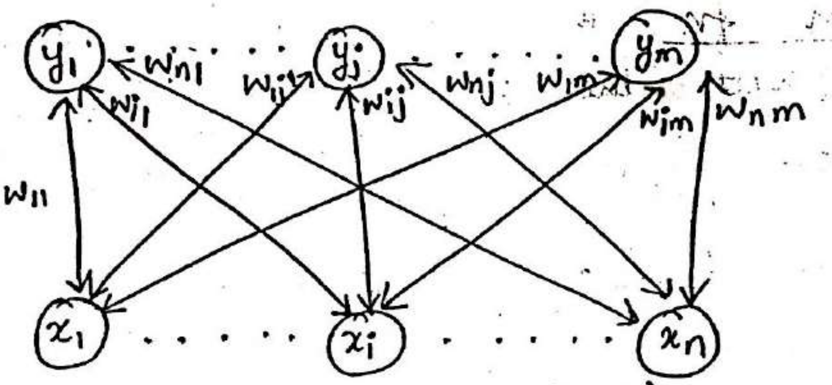
→ The connections between the layers are bidirectional i.e if the weight matrix for signals sent from X-layer to Y-layer is W, then the weight matrix for signal sent from Y-layer to X-layer is W^T

→ This architecture resembles a single layer feed forward network.

→ It is a fully interconnected network, wherein the inputs and the outputs are different.

→ The most important feature of BAM is that there exist weighted interconnections from both X-layer to Y-layer and vice-versa.

→ The weights are adjusted between X-layer to Y-layer and also from Y-layer to X-layer.



(Architecture of BAM)

⇒ Types of BAM Network :

- Discrete BAM
- Continuous BAM.

Note: BAM networks can use supervised or unsupervised learning. If it uses delta rule (involves target) hence using supervised learning. If it uses Hebb rule (without target) using unsupervised learning

a) DISCRETE BAM :

→ It can use either binary or bipolar form

→ In both case (binary + bipolar) weights are found from the sum of Hebb outer product of bipolar form of training vector pairs.

→ Activation fn used is step activation function with, the possibility of non-zero threshold.

→ Activation fn for Binary input vector :-

Y Layer

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

X Layer

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

Note:: If net input is equal to threshold value, the activation fn decides to leave the activation of that unit at its previous value.

b) CONTINUOUS BAM :

→ It has the capability to transfer the input smoothly and continuously into the respective output in the range between [0, 1]

→ It uses logistic sigmoid function as activation fn.

→ Y- layer : Activation fn is given by :

$$\boxed{f(y_{inj}) = \frac{1}{1 + \exp(-y_{inj})}}$$

if bias is included then net input

$$y_{inj} = b_j + \Sigma x_i w_{ij}$$

→ X- layer :

$$\boxed{f(x_{ini}) = \frac{1}{1 + \exp(-x_{ini})}}$$

Net input : $x_{ini} = b_i + \Sigma y_j w_{ij}$

→ Note : storage capacity of BAM is min(n, m)

where n = no of units in x- layers.

m = no of units in y- layers

→ Application Algorithm: (Testing)

→ From the training process, obtain the final w[eights]

→ The input patterns are presented to both X-layer and Y-layer.

→ The net input and the activations to the Y-layer are calculated.

→ Then the signals are sent to the X-layer and here net input and activations are found.

→ In this manner, BAM is tested for its performance.

→ The algorithm for BAM is given below:

Step-I: Initialize the weight to store a set of P vectors. Initialize all activations to 0.

Step-II: For each testing input follow steps 3 to 8.

Step-III: Set activations of X-layer to current input pattern.

Step-IV: Input pattern is presented to Y-layer.

Step-V: While activations are not converged follow steps 6 to 8.

Step-VI: Activation unit in Y-layer and net input are computed as:

Scanned with CamScanner

compute net input $y_{inj} = \sum_i W_{ij} x_i$.
compute activations $y_i = f(y_{inj})$.
send signals to X-layer.

step-VII: Update activation unit in X-layer.
Net input is computed as $x_{ini} = \sum_j W_{ij} y_j$.
Then compute activations $x_i = f(x_{ini})$
Send signals to Y-layer.

Step-VIII: Test for convergence.
⇒ The stopping condition may be that the activation vectors x and y have reached equilibrium.

→ Hamming Distance: (used for error correction and error detection)

The difference between the no of bits in two binary or bipolar vectors x_1 and x_2 is called the hamming distance between the vectors.

Eg: Let
x_1 = 1 1 0 1 1 0 1 0
x_2 = 0 1 0 0 1 0 0 1

$HD(x_1, x_2) = 4$

Average HD $= \frac{1}{n} HD(x_1, x_2) \Rightarrow \frac{4}{8} \Rightarrow 0.5$

→ Storage Capacity :-

→ An important criterion for an associative network is the no of patterns or pattern pairs it can store

→ capacity depends on the no of components in the stored vector.

→ More no. of vectors can be stored if they are mutually orthogonal.

→ n-1 mutually orthogonal bipolar vector each with n components can be stored using the sum of outer product weight matrices.

→ $\boxed{\text{The two vectors } a \text{ and } b \text{ are orthogonal if } \Sigma a_i b_i = 0}$

(D) ADAPTIVE RESONANCE THEORY (ART) :-

→ This network was developed by Stephen Grossberg and Gail Carpenter in 1987.

→ It is based on competition and uses unsupervised learning.

→ ART is always open to new learning (adaptive) without losing the old patterns (resonance).

→ Basically, ART network is a <u>vector classifier</u> which accepts an input vector and classifies it into one of the categories depending upon which of the stored pattern it resembles the most.

→ If the input pattern does not match any stored pattern, a new category is created by storing pattern.

→ <u>ART nets</u> are designed to be <u>both stable and plastic</u>.

Plasticity: The ability of a net to learn a new pattern equally well at any stage of learning is called plasticity. (The previous patterns should not be modified).

→ ART is classified into two types:

ART-1
Designed for clustering binary vectors.

ART-2
Designed for clustering continuous valued vectors

→ <u>Operating Principal of ART</u> :-

The main operation of ART classification can be divided into 3 phases.
- Recognition phase
- Comparison Phase
- Search Phase.

i) Recognition Phase: The input vector is compared with the classification (Recognition layer. neurons with weights) presented at every node. The output of the neuron becomes "1" if it best matches with the classification applied; otherwise it becomes "0".

ii) Comparison Phase: In this phase, a comparison of the input vector to the comparison layer vector is done. If degree of similarity is less than vigilance parameter causes reset.

Vigilance parameter: It is used for controlling degree of similarity.

iii) Search phase:

→ If no reset signal is generated, the match is adequate and the classification is finished.
→ Otherwise, the stored pattern must be sent to seek a correct match.
→ The process is repeated for neurons until a stored pattern is found that matches the specified tolerance or all the stored patterns have been tried and all are in mismatch with the input vector.

Basic Operations: (For both ART1 or ART2)

i) Learning Trail :- It consists of presentation of one input pattern to the network.

→ Before the pattern is presented:

a) The activation of all units should be zero.

b) The f_2 units are made inactive.

ii) Controlling the degree of similarity :-

→ It is controlled by vigilance parameter.

→ It is denoted as ρ (range: 0 to 1)

iii) Reset Mechanism state:

→ The mechanism differs for ART1 and ART2.

→ Its function is to control the state of each node in F_2 layer.

→ The f_2 layer node is present in any one of the 3 states mentioned here.

a) Active - "ON". The unit in F_2 is on. Its activation is given by d, where $d = 1$ for ART1 & $d = 0$ to 1 for ART2

b) Inactive: "OFF". The unit in F_2 is off. Its activation = 0, but it is available to participate in next competition.

Scanned with CamScanner

c) Inhibit : "OFF'. The unit in F_2 is off. Its activation $= 0$, and is prevented from participation in further competition during the presentation of current input vector.

→ Learning in ART :-

Fast learning	Slow learning
i) It is used in ART 1 network.	i) It is used in ART 2 network.
ii) The weight changes during the resonance period occur more rapidly.	ii) The weight changes during the resonance period occur slowly.
iii) The network is stabilized each time the pattern has to choose the correct cluster unit where it is presented.	iii) The weight produced are continuously changing. So the network will not be stabilized. It will be stabilized, only after presenting a large no of inputs.

2) ART 1 :

It is a type of ART, which is designed to cluster binary vectors.

- Architecture of ART 1 :-

It consist of following two units → computational unit → supplement unit.

- Computational Unit : It is made up of the following.

a) Input unit (F_1 layer): It is divided into 2 portions

$F_1(a)$ layer (Input portion)

$F_1(b)$ layer (Interface portion)

→ The input portion just presents the input vector given.

→ No processing occurs

→ It is connected to $F_1(b)$ layer.

→ This portion combines the signal from the input portion with that of F_2 layer.

→ $F_1(b)$ layer is connected to F_2 Layer through bottom up weights b_{ij} and F_2 layer is connected to $F_1(b)$ layer through top down weights t_{ji}

b) Cluster Unit (F_2 layer) :

→ This is a competitive layer.

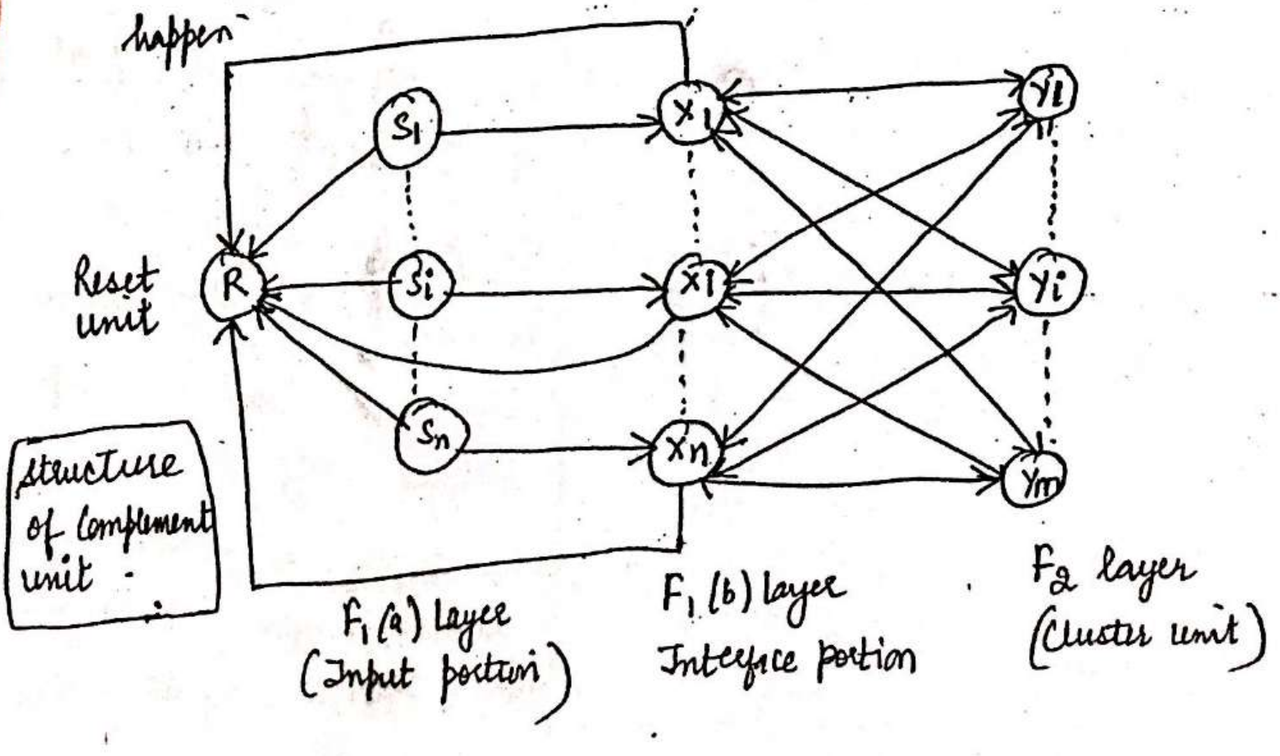
→ The unit having the largest net input is selected to learn the input pattern.

→ The activation of all other cluster unit are set to 0.

c) Reset Mechanism :-

→ The work of this mechanism is based upon the similarity between the top - down weight and the input vector.

→ If the degree of similarity is less than the vigilance parameter, then the cluster is not allowed to learn the pattern and a reset would happen.



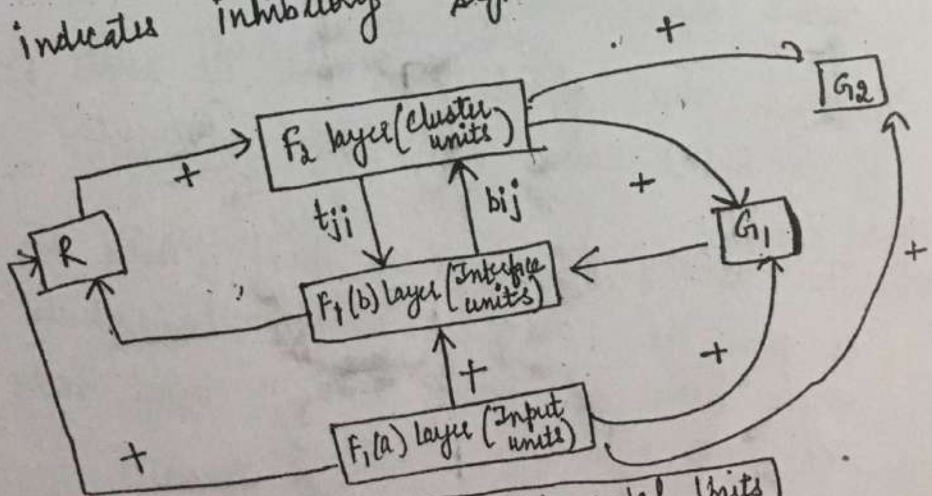
• Supplemental Units :-

→ Difficulty posed in Computational units (F_1, F_2 and reset units) is that these units are required to respond differently at different stages of the process.

→ Moreover, the issue with Reset mechanism is that the layer F_2 must have to be inhibited under certain conditions and must also be available when some learning happens.

→ That is why, two supplemental units namely, G_1 and G_2 is added along with reset unit R. They are called Gain control units. These units receive and send signals to the other units present in the network.

'+' indicates excitatory signal
'–' indicates inhibitory signal.



Structure of Supplemental Units

Each of these units $\rightarrow (F_1(a), F_1(b), F_2)$ also receives two excitatory signals in order to be 'on'. Hence due to the the requirement, is called two - third rule.

ALGORITHM :-

Working :-

→ Binary input vector is presented to $F_1(a)$ layer and is then received by $F_1(b)$.

→ The $F_1(b)$ layer sends the activation signal to F_2 layer over weighted interconnection path.

→ Each F_2 unit will calculate the net input.

→ The unit with the largest net input will be the winner that will have activation $d=1$. All the other units will have activation as zero.

→ That wining unit alone will learn the current input pattern.

→ Then the signal ~~sent~~ will be sent from F_2 to $F_1(b)$ using top down weights.

→ The norm of the vector $\|x\|$ will be calculated. [It gives the no of components in which top-down weight vector for wining unit + input vector S are both 1].

Depending upon the ratio of norm of x to norm of S $\left(\frac{\|x\|}{\|s\|}\right)$, the weights of the winning cluster unit are adjusted.

→ The whole process may be repeated until either a match is found or all neurons are inhibited.

→ The ratio $\frac{\|x\|}{\|s\|}$ is called match ratio.

Norm is a function that assigns a positive length or size to each vector in vector space

→ The parameters used in the algorithm are:

n - no of components in the input vector.

m - Max no of clusters that can be formed.

b_{ij} - bottom-up weights (from $F_1(b)$ to F_2 unit).

t_{ji} - top-down weights (from F_2 to $F_1(b)$ units).

ρ - vigilance parameter.

s - binary input vector.

x - activation vector for interface ($F_1(b)$ layer (binary))

$\|x\|$ - norm of vector x.

L - learning Rate.

→ The training algorithm of ART-1 network :-

Step-1 :- Initialize parameters

$L > 1$ and $0 < p \leq 1$.

Initialize weights $0 < b_{ij}(0) < \frac{L}{L-1+n}$, $t_{ji}(0) = 1$.

Step-2 : While stopping condition is false, perform 3-14.

Step-3 :- For each training input, do steps 4-13.

Step-4 : Set activations of all F_2 units to zero.

Set activations of $F_1(a)$ units to input vector s.

Step-5 : Compute the norm of s :

$$\|s\| = \sum_i s_i$$

Step-6 : Send Input signal from $F_1(a)$ to $F_1(b)$ layer

$$x_i = s_i .$$

Step-7 : For each F_2 node that is not inhibited

If $y_j \neq -1$ then $y_j = \sum_i b_{ij} x_i$

Step-8 : while reset is true, perform steps 9-12.

Step-9 : Find J such that $y_J > y_j$ for all nodes j.

If $y_j = -1$, then the node is inhibited

Scanned with CamScanner

Step-10: Again calculate the activation of $F_1(b)$. (14)

$$\boxed{x_i = s_i t_{ji}}$$

Step-11: Compute the norm of vector x:

$$\|x\| = \sum_i x_i$$

Step-12: Test for reset.

If $\frac{\|x\|}{\|s\|} <$ vigilance parameter ρ, then inhibit node j and continue step-8 again.

If $\frac{\|x\|}{\|s\|} \geq$ vigilance parameter ρ, then Proceed to step-13.

Step-13: Update the weights for node J.

$$\boxed{\begin{aligned} b_{ij(new)} &= \frac{L x_i}{L-1+\|x\|} \\ t_{ji(new)} &= x_i \end{aligned}}$$

(15)

Step-14:- The stopping condition for algorithm must be checked and it may be as follows—

→ Do not have any change in weight.

→ Reset is not performed for units.

→ Maximum no of epoch reached.

Points to be noted :-

→ In winner selection, if there is a tie, take J to be the smallest index.

→ Also t_{ji} is either 0 or 1, and once it is set to 0 during learning, it can never be set back to 1, because of stable learning method.

→ The parameters used have the typical values as shown below.

Parameter	Range	Typical value
L	$L > 1$	2
P	$0 < P \leq 1$	0.9
b_{ij}	$0 < b_{ij}(0) < \frac{L}{L-1+n}$	$\frac{1}{1+n}$
t_{ji}	$t_{ji}(0) = 1$	1

(F) ART 2 :

→ ART 2 accepts continuous valued vectors.

→ The difference between ART 2 and ART 1 reflects the modifications need to accomodate patterns with continuous valued components.

→ ART 2 has a highly complex F_1 units.

→ The F_1 units of ART 2 posses combination of normalization and noise suppression, along with the comparison of weights needed for reset mechanism.

→ ART 2 has two types of continuous valued inputs:

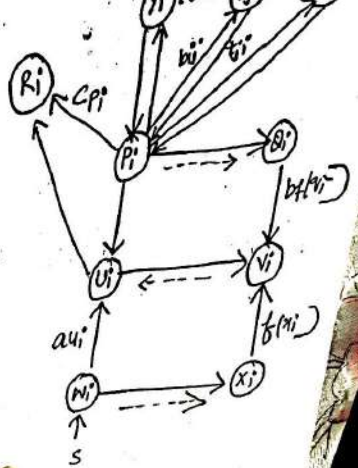
Noisy binary signal
[can operate with the fast learning type of data)

Continuous
[Suitable for slow learning mode].

→ Architecture of ART 2 Network :

→ From the architecture, it is seen that, the F_1 layer has six types of units (W, X, U, V, P & Q).

→ Between all the units say, W and X, P and Q, U and V there exists supplemental unit, which receives signals from respective units, calculates its norm and sends to the other units.



→ The receiving unit receives both inhibitory and excitatory signals from the sending units through supplemental units.

→ The X and Q units are connected to V units.

→ The transformation occuring in the signal is indicated by symbols indicated in connection.

→ The P unit path is connected to the cluster units by bottom up and top down weights.

→ The symbol ----→ indicates normalization.

→ The V units perform the operation of $F_1(a)$ layer as in ART 1

→ P units perform the operation of $F_1(b)$ Layer interface portion as in ART 1.

Concept of Training Algorithm :-

i) As the learning trail starts, all activations are set to zero.

ii) The computation cycle starts with the computation of activation of unit U_i.

iii) Then the signal is sent from U_i to W_i and P_i. The activations of W_i and P_i are then computed

iv). W_i sums the signals from U_i and S_i
P_i sums signals from U_i and top down signal (in case if F_2 is active)

v) X_i and Q_i activations are the normalized vector of W_i and P_i respectively.

vi) Before signal is sent to V_i, activation is calculated on each unit.

vii) Finally, V_i sums the signals it receives from X_i and Q_i.

viii) This entirely forms one cycle updation of F_1 Layer.

ix) When the activations of F_1 Layer reaches equilibrium, P units send their signals to F_2 layer, and winner unit is selected based on competition.

x) The reset mechanism checks for a reset whenever it receives a signal from P, since the further computations are based on the value of that signal.

xi) This signal is going to be most recent signal the unit R_i had received from U_i.

xii) After the check for reset is finished, the cluster unit may be rejected or accepted.

xiii) Based on this, the learning process starts.

xiv) ART2 can perform

Slow learning
→ Only one iteration of the weight update equations occurs on each learning trail.

fast learning
→ the weight update continue until the weights reach equilibrium on each trail.

→ ART2 Training Algorithm [No need to go in details]

→ ART Applications:

- It describes a no of neural network models which use supervised and unsupervised learning methods and address problems such as pattern recognition and prediction.