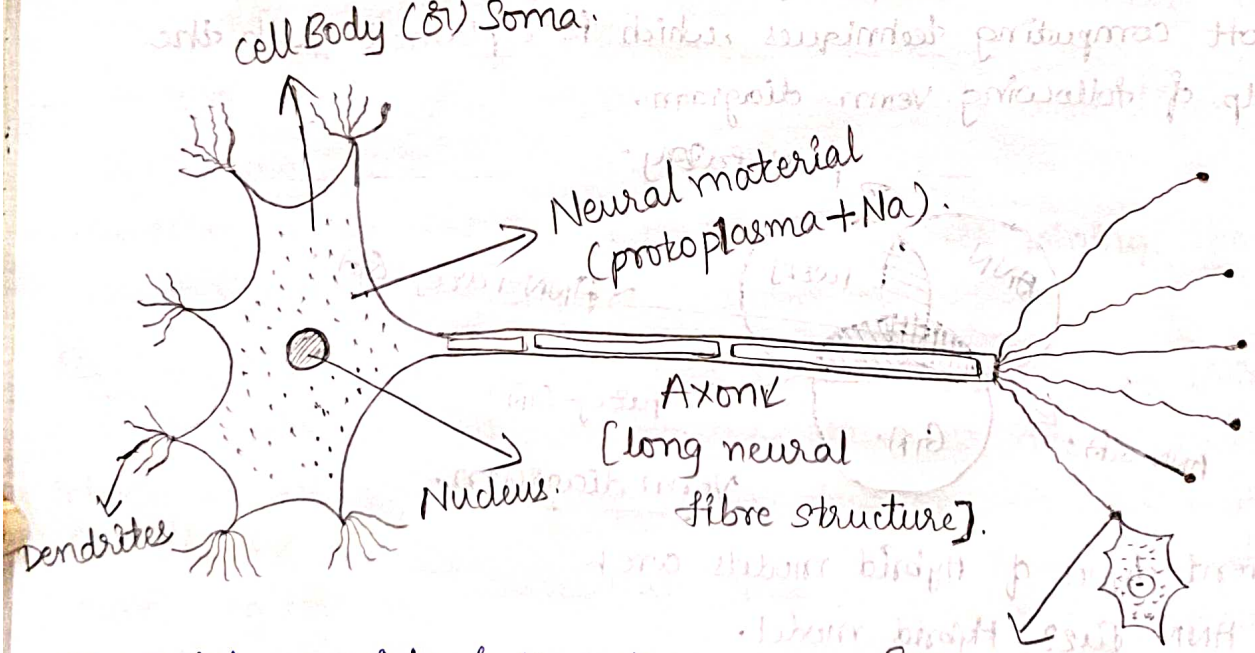
UNIT-2:-

* ANN & its Paradiagrams:

→ Biological Neural Network (Human Brain).
→ NN modes (Perceptron model).
→ Architectures of NN.
→ Learning rules.
→ Back Bopagation NN.
→ Kaheninis self organising network.
→ Hop field network.
→ Applications of ANN.

- **Biological NN:-** Biological NN are nothing but human brain. The average weight of human brain is 1.5 kg & is the interconnection of 10^{11} number of biological neurons involved in 10^{14} number of interconnections. The average weight of each biological neuron is 1.5×10^{-9} g.

The biological neuron can be shown as given below:-



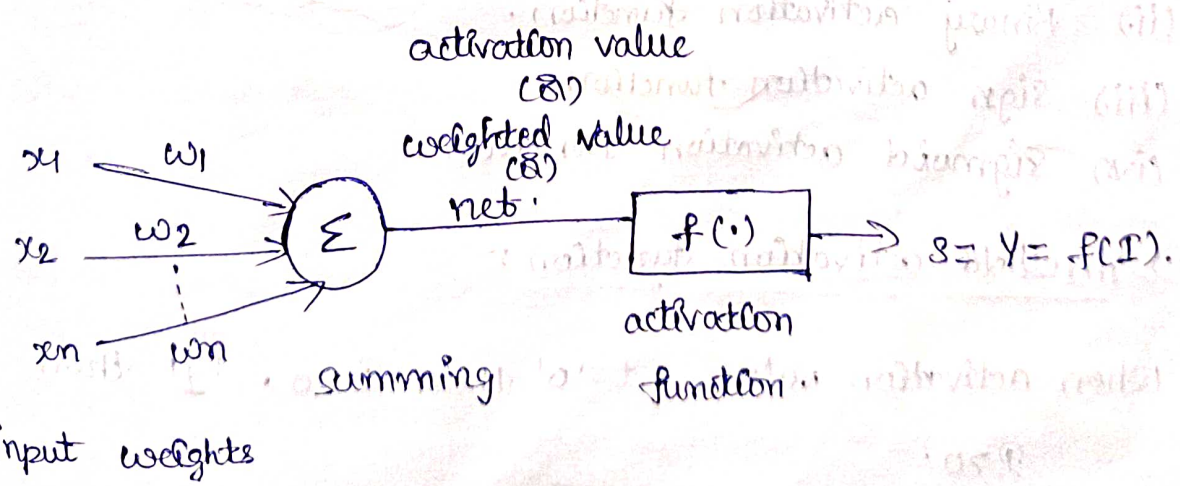
- It mainly consists of 4 parts:-

(i) Cell Body (or) Somma.
(ii) Dendrites.
(iii) Axon.
(iv) Synaptix function / synapsis.

* Cell Body:- It contains nucleus and neural materials. Neural materials contains proto plasms (-ve charge) and sodium ions (+ve charge). The presents of +ve and -ve charges inside cell body naturally possesses & contains some potential around 60 milli volts. The cell body receives information from other neurons from dendrites.

-> Dendrites:- These are tree type neural fibre structures attached to cell body. Once signals are received to the cell body, sums of all the signals which will rise the potential. If the rised potential is more than the threshold value neuron is said to be activated / fixed & transfers information through a long neural fiber structure called as Axon. The end of the axon is stranded (or) sub-stranded in to the 1000's of neural fiber structures where new neurons will be connected at synaptic junction. The size of cell body is around 10-80 micro meters. The diameter of axon is several micrometers. The length of the synaptic junction is 200 nano meters, the speed of the signal transferred from one neuron to another neuron is 0.2 to 10 m/s.

* Artificial neuron model (perceptron):-



Artificial neurons are designed to function just like a biological neuron. Different scientists represent the artificial neuron models in different ways. The most popular artificial neuron models are developed by a scientist called Rojen Blatt, which is popularly known as perceptron model as given in above figure.

$x_1, x_2, \ldots\ldots, x_n$ are inputs, $w_1, w_2, w_3, \ldots\ldots, w_n$ are weights which reflects already stored information. 'I' is the activation value / weighted value net value $f(I)$ is the activation function. $S = Y = f(I)$ is the output.

Artificial neuron is represented as summing point which is sums up all the signals received and is called as Activation function.

$$I = x_1w_1 + x_2w_2 + \ldots\ldots + x_nw_n.$$

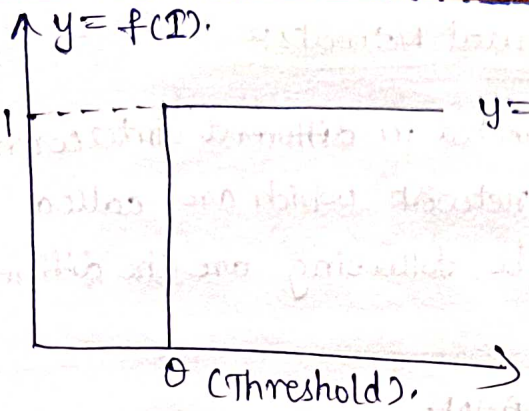
The output of the neuron is calculated by processing the activation value through activation function.

→ Activation function :- The activation function is used to find the output of the artificial neuron, the following are the different activation function.

(i.) Threshold (or) Step activation function.
(ii.) Binary activation function.
(iii.) Sign activation function.
(iv.) Sigmoid activation function.

→ Threshold activation function :-

When activation value get '0' then $I < 0$, '1' then $I > 0$.



$y = F(I) = 1$ if $I \geq 0$.

$y = F(I) = 0$ if $I < 0$.

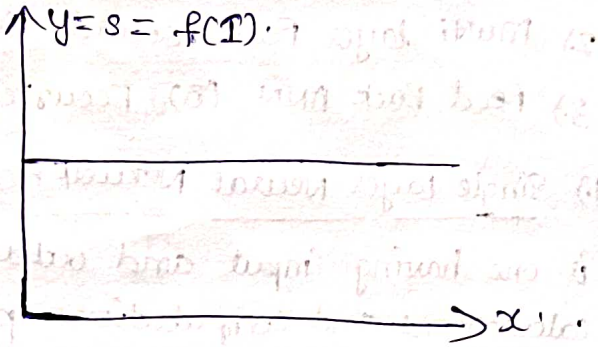
- Binary activation function :-

$s = 1$.

$s = 0$.

$y = f(I) = 1$ if $I \geq 0$.

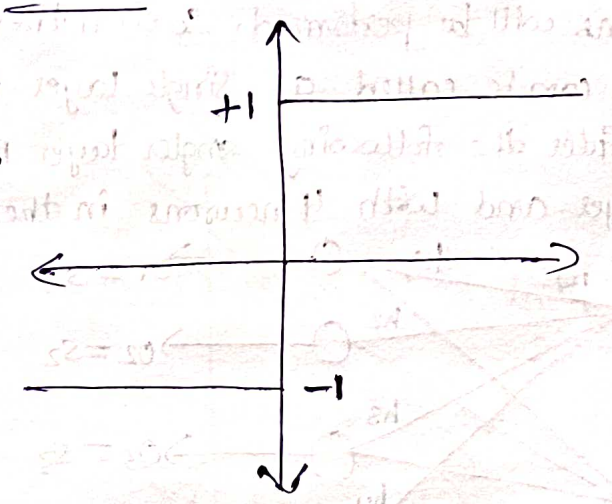
$y = f(I) = 0$ if $I = 0$.



- Sign activation function :-

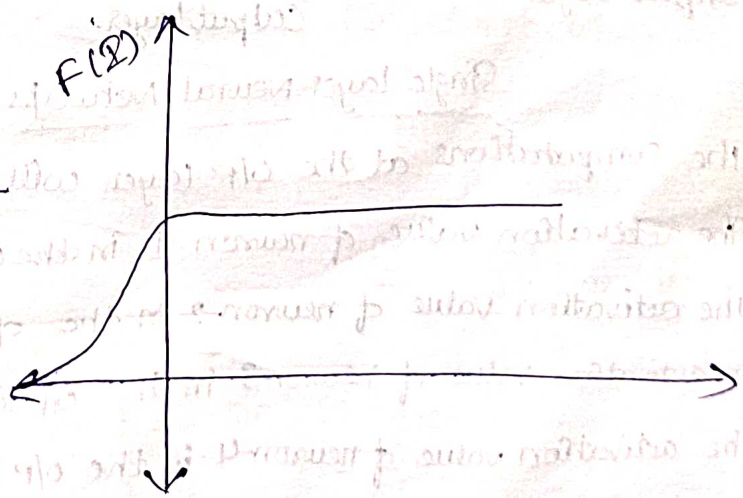
$y = f(I) = +1, \; I > 0$.

$y = f(I) = -1, \; I < 0$.



- Sigmoid activation function :-

$$y = f(I) = \frac{1}{1 + e^{-I}}.$$

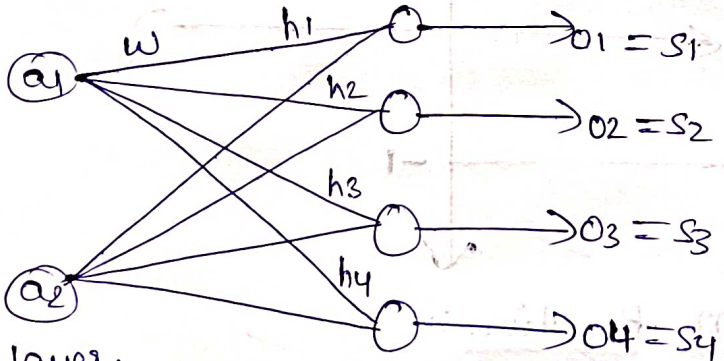


* Architectures of Artificial Neural Networks:-

The artificial neurons are connected in different interconnections form the artificial neural network which are called as Architectures of ANN. The following are the different architectures of ANN;

1.) Single layer Feed forward ANN.
2.) Multi layer Feed forward ANN.
3.) Feed Back ANN (or) Recurrent Neural network.

1.) **Single layer Neural Network:-** A single layer feed forward network is one having input and output layers. At the I/P layer no calculations and computations performed, just transfers the inputs to output layer. At the output layer, calculations and computations will be performed. So even there are I/P and O/P layers, this can be called as Single layer Feed forward network. Let us consider the following single layer ANN with 2 neurons in I/P layer and with 4 neurons in the O/P layer.



Single layer Neural Networks.

The computations at the O/P layer will be calculated as follows;

- The activation value of neuron 1 in the O/P layer $h_1 = a_1w_{11} + a_2w_{21}$
- The activation value of neuron-2 in the O/P layer $h_2 = a_1w_{12} + a_2w_{22}$
- The activation value of neuron-3 in the O/P layer $h_3 = a_1w_{13} + a_2w_{23}$
- The activation value of neuron-4 in the O/P layer $h_4 = a_1w_{14} + a_2w_{24}$

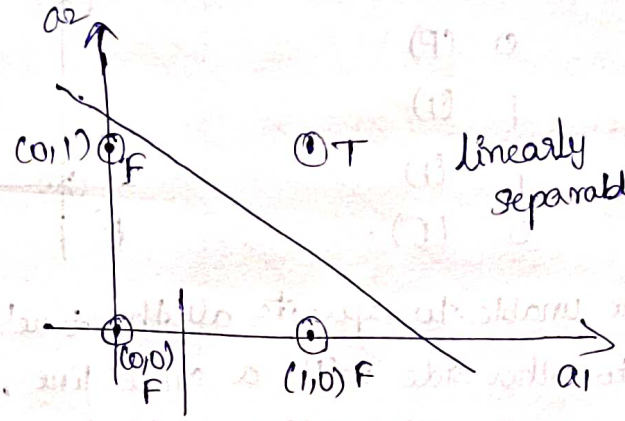
Consider sigmoidal activation function;

$$S_1 = \frac{1}{1+e^{-h_1}}, \quad S_2 = \frac{1}{1+e^{-h_2}}, \quad S_3 = \frac{1}{1+e^{-h_3}}, \quad S_4 = \frac{1}{1+e^{-h_4}}.$$

The single layer feed forward network can be used to design different logical operations like and, or, NAND, NOR, etc.....

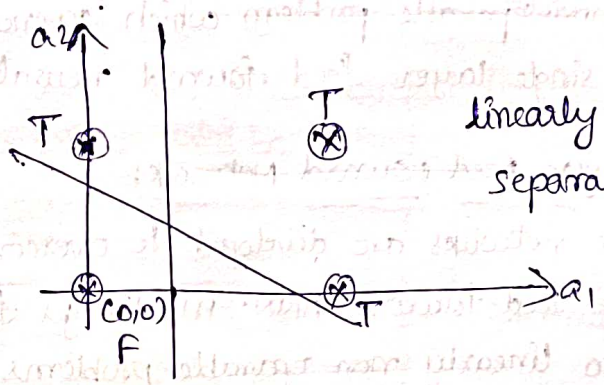
- AND:-

a_1	a_2	S
0	0	0 → F
0	1	0 → F
1	0	0 → F
1	1	1 → T



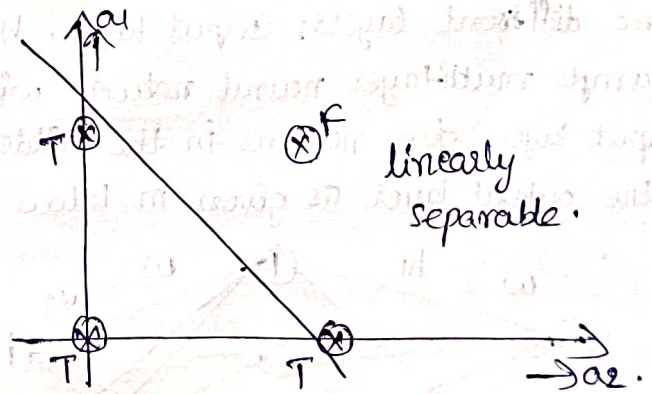
→ OR:-

a_1	a_2	S
0	0	0 → F
0	1	1 → T
1	0	1 → T
1	1	1 → T



→ NAND:-

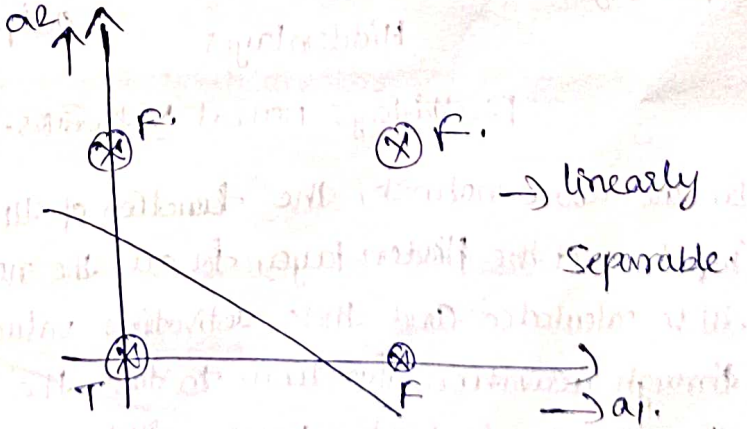
a_1	a_2	S	
0	0	1	(T)
1	0	1	(T)
0	1	1	(T)
1	1	0	(F)



Linearly separable and can be implemented using single layer feed forward network.

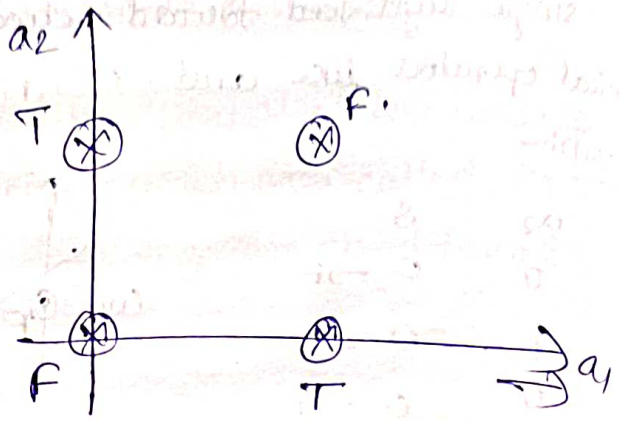
→ NOR:-

a_1	a_2	S	
1	1	0	(F)
1	0	0	(F)
0	1	0	(F)
0	0	1	(T).



- Exclusive-OR (XOR):-

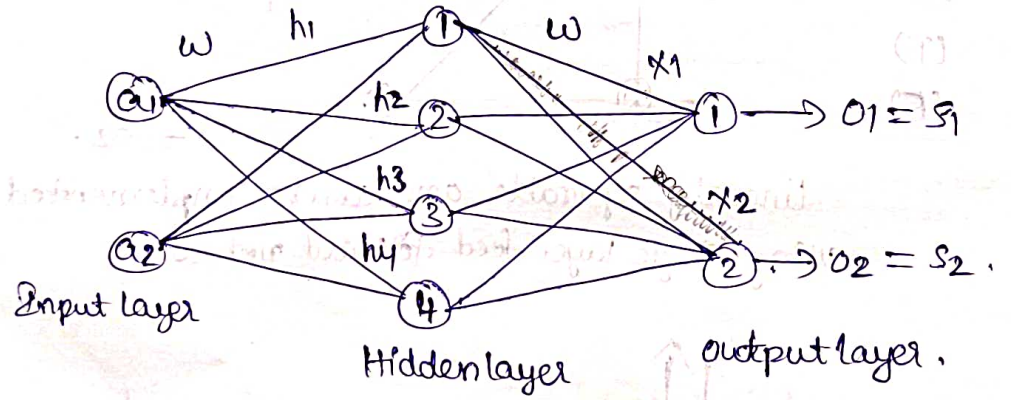
a_1	a_2	$s = \bar{a_1}a_2 + a_1\bar{a_2}$
0	0	0 (F)
0	1	1 (T)
1	0	1 (T)
1	1	0 (F).



- As we are unable to separate all the true's to one side and all the Falses to other side with a single line, it is called as linearly non-separable problem which cannot be designed with the help of single layer feed foward neural network.

2.) Multi-layer Feed Foward Network:-

These networks are developed to overcome the limitations of single layer feed foward ANN. Multilayer feed foward ANN can also be used to linearly non-separable problems. This network contains three different layers; Input layer, Hidden layer and output layer. A sample multilayer neural network with two neurons in the input layer, four neurons in the hidden layer and two neurons in the output layer as given in below figure.



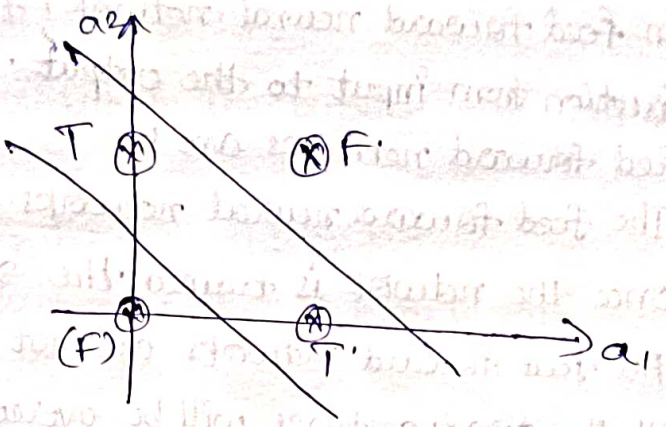
Multi-layer Neural Networks.

- In the above network, the function of IIP layer is to feed the inputs. At the Hidden layer, for all the neurons, activation values will be calculated and these activation values has to be processed through activation function, to find the output of each neuron. The outputs of hidden layer will be the inputs of the output layer. Again at the output layer, at each neuron activation values will be calculated which are processed through the activation function to find the outputs. In the above problem, as we

are using two computational layers; so we can use two linear separables to solve X-OR problem.

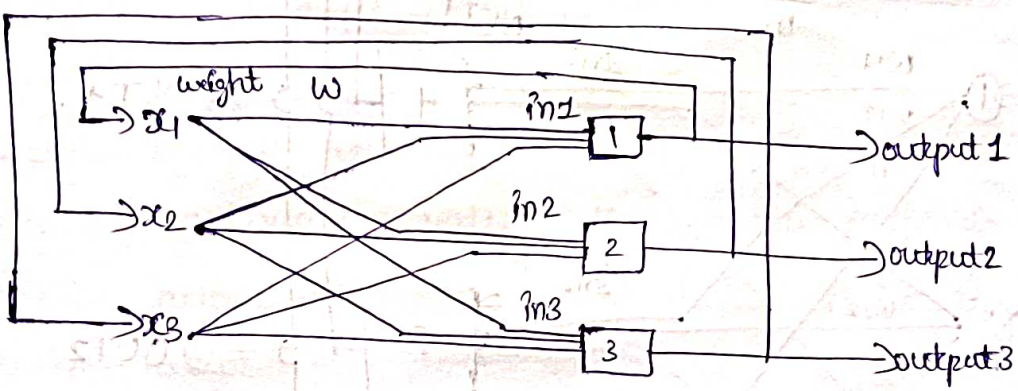
- Exclusive-OR:

a_1	a_2	S.
0	0	0 (F)
0	1	1 (T)
1	0	1 (T)
1	1	0 (F).



3) Feedback ANN (or) Recurrent ANN:-

In the feedback neural network, the output is fed back to the input so that information will be stored.



In the above network, initial inputs will be given to the output layer and the outputs of output layer fed back to the input, so that it will act as memory cell.

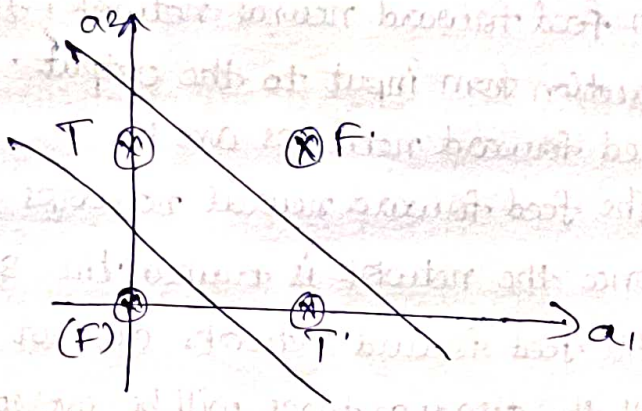
* Learning Rules of ANN:-

Learning & Training is nothing but updating & changing the weights in order to match the actual output with the expected output (or) minimizing the error.

are using two computational layers; so we can use two linear separables to solve X-OR problem.

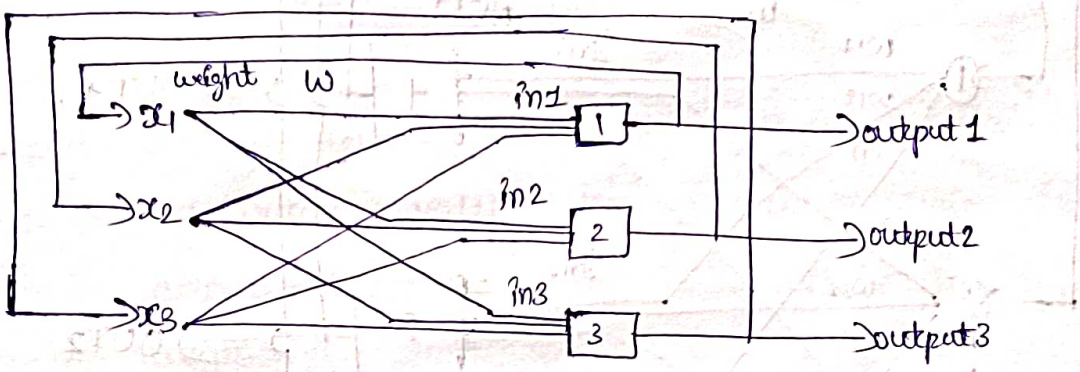
- Exclusive-OR:

a_1	a_2	S.
0	0	0 (F)
0	1	1 (T)
1	0	1 (T)
1	1	0 (F).



3) Feedback ANN (or) Recurrent ANN:-

In the feedback neural network, the output is fed back to the input so that information will be stored.



In the above network, initial inputs will be given to the output layer and the outputs of output layer fed back to the input, so that it will act as memory cell.

* Learning Rules of ANN:-

Learning & Training is nothing but updating & changing the weights in order to match the actual output with the expected output (or) Minimizing the error.

* Feedback ANN (or) Recurrent networks (or) Hopfield Networks :-

In feed forward neural network, the signal flows in forward direction from input to the output. Different disadvantages of feed forward networks are :-

1.) The feed forward neural networks are unconditionally stable.
2.) Once the network is trained, the state of the network is fixed.
3.) All feed forward networks does not have memory.
4.) All the disadvantages will be overcome by feedback (or) Recurrent network which is as given below :-

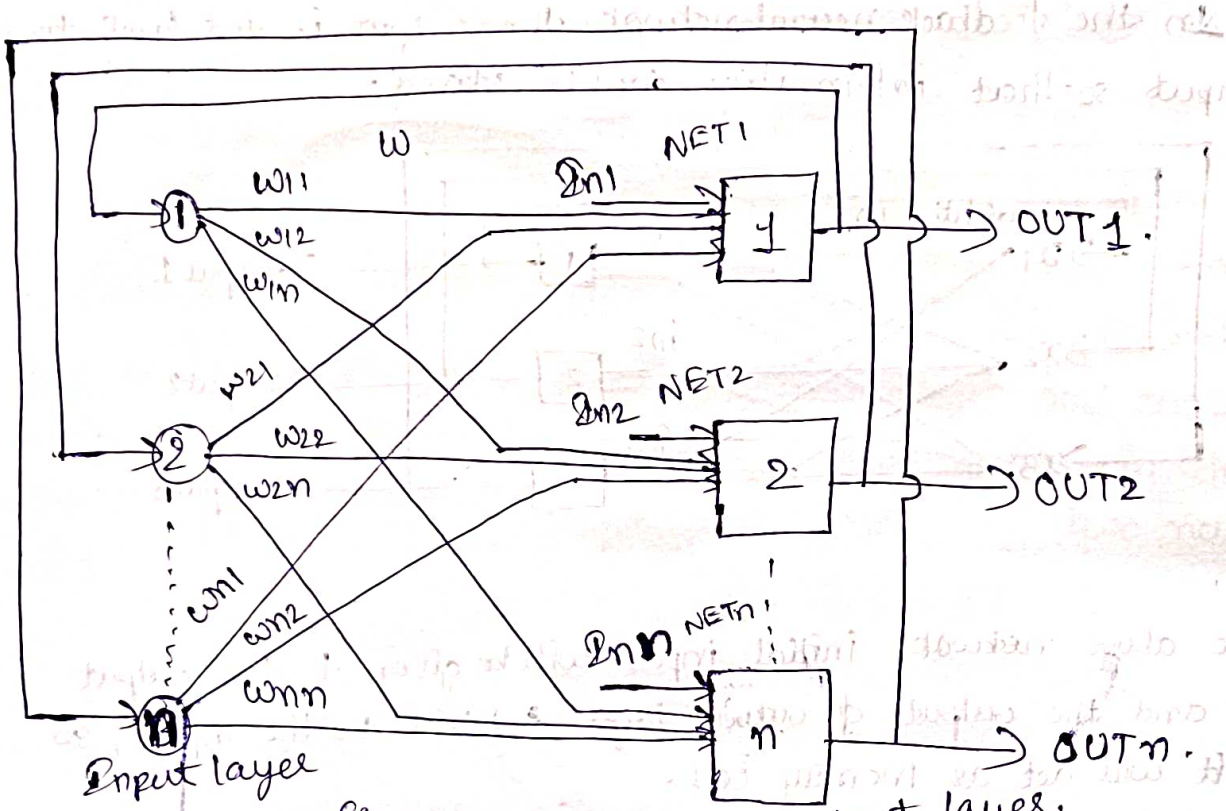


Fig: Feedback ANN.

Initial inputs In1, In2, ..., Inn will be applied to the neurons in the output layer to produce initial outputs OUT1, OUT2, ..., OUTn. The outputs are fed back to the neurons in the input layer. The input layer forwards to process the inputs to the output layer and output is recalculated. Again the output is fed back to the input layer. Like this again and again, the outputs fed back to the input until the output becomes constant.

A scientist called Hopfield extensively worked in the theory and architectures of Hopfield networks. So these feedback networks also called as "Hopfield ANN".

The activation values of each neuron in the output layer will be calculated as follows :-

$NET_1 = I_{n1} + w_{11} OUT_1 + w_{21} OUT_2 + \cdots + w_{n1} OUT_n$.

$NET_2 = I_{n2} + w_{12} OUT_1 + w_{22} OUT_2 + \cdots + w_{n2} OUT_n$.

$NET_n = I_{nn} + w_{1n} OUT_1 + w_{2n} OUT_2 + \cdots + w_{nn} OUT_n$.

-) Generalized formula

$$NET_j = I_{nj} + \sum_{i=1}^{n} w_{ij} OUT_i.$$

- The outputs of each neuron in the o/p layer is calculated by processing the activation value through the activation function. Based on the activation function used, there will be two types of Hopfield networks:

1) Discrete Hopfield Network.
2) Continuous Hopfield Network.

- In the Discrete Hopfield Network, a discrete activation function Binary activation function will be used.

$OUT_j = 1$ if $NET_j > 0$.

$OUT_j = 0$ if $NET_j \leq 0$.

- In the continuous Hopfield Network, continuous activation function such as Sigmoidal Activation function is used to find the output of a neuron. As the Sigmoidal activation function;

$$OUT_j = \frac{1}{1 + e^{-NET_j}}.$$

A scientist called Hopfield extensively worked in the theory and architectures of Hopfield networks. So these feedback networks also called as "Hopfield ANN".

The activation values of each neuron in the output layer will be calculated as follows :-

$NET_1 = In_1 + w_{11} OUT_1 + w_{21} OUT_2 + \cdots + w_{n1} OUT_n.$

$NET_2 = In_2 + w_{12} OUT_1 + w_{22} OUT_2 + \cdots + w_{n2} OUT_n.$

$\vdots$

$NET_n = In_n + w_{1n} OUT_1 + w_{2n} OUT_2 + \cdots + w_{nn} OUT_n.$

-> Generalized formula

$$NET_j = In_j + \sum_{i=1}^{n} w_{ij} OUT_i.$$

- The outputs of each neuron in the o/p layer is calculated by processing the activation value through the activation function. Based on the activation function used, there will be two types of Hopfield networks:

1) Discrete Hopfield Network.
2) Continuous Hopfield Network.

- In the Discrete Hopfield Network, a discrete activation function Binary activation function will be used.

$OUT_j = 1$ if $NET_j > 0.$

$OUT_j = 0$ if $NET_j \leq 0.$

- In the continuous Hopfield Network, continuous activation function such as Sigmoidal Activation function is used to find the output of a neuron. As the Sigmoidal activation function;

$$OUT_j = \frac{1}{1 + e^{-NET_j}}.$$

* Self Organising Maps (SOMs) :- (Or) Kohenen Maps :

Self Organising maps are developed by a scientist called Kohenen, so these networks are also called as Kohenen networks. These networks will make use of unsupervised learning for training. So these are called as Self-Organising maps. These soms & Kohenen maps are mainly applied to following networks (or) applications.

1.) For the classification applications (classification of images, items, classification of letters, etc...)

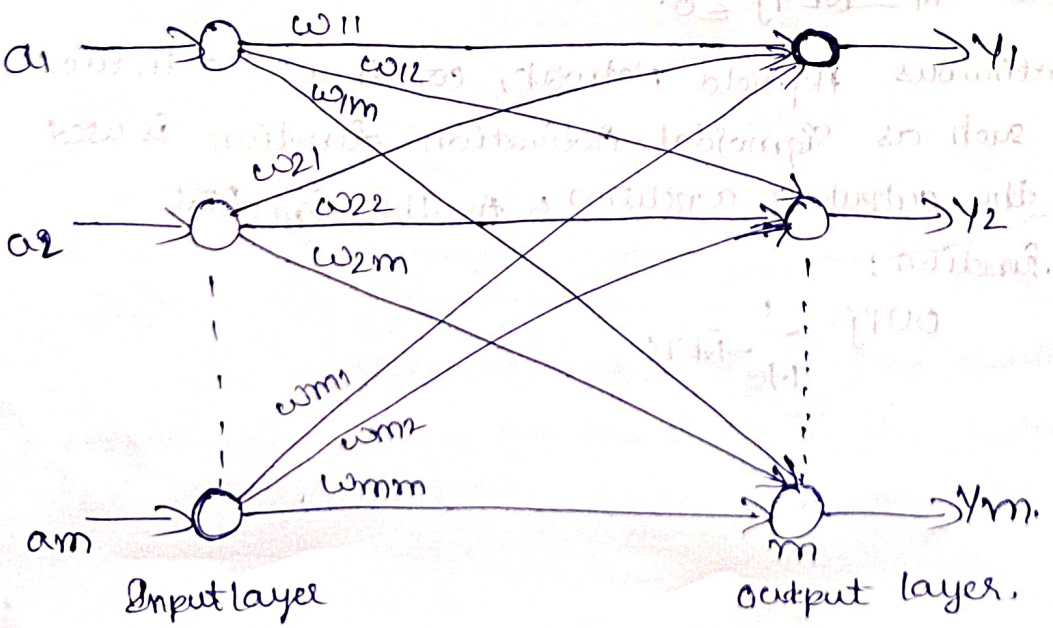
2.) To convert the high dimensional data into low dimensional data (i.e; from 4D, 3D, 2D to 1D).

3.) The Kohenen developed the SOMs to overcome the one of the major drawback of the time complexity.

→ Disadvantage in Back Propagation Algorithm :-

To train a given neural network, all the weights between input and output layers must be updated so to train the neural network using back propagation algorithm, it may take several hours & days & weeks sometimes. The above disadvantage will be overcome by Kohenen SOM using a logic "Winner's Take All".

As per let us consider the Kohenen map architecture as given below :



As per Kohenen, Euclediam distance from each neuron in the o/p layer is calculated the neuron having less Euclediam distance is called as winner so the weights corresponding to that winner neuron only updated such that time complexity of training will be decreased.

$$(a_1-w_{11})^2+(a_2-w_{21})^2+\cdots\cdots+(a_n-w_{n1})^2$$

$$(a_1-w_{12})^2+(a_2-w_{22})^2+\cdots\cdots+(a_n-w_{n2})^2$$

$$\vdots$$

$$(a_1-w_{1n})^2+(a_2-w_{2n})^2+\cdots\cdots+(a_n-w_{nn})^2$$

- Generalized Equation:

$$D_j=\sum_{j=1}^{m}\sum_{i=1}^{n}(a_i-w_{ij})^2$$

let us consider the above network, the second neuron in the output layer is winner. Then, it is sufficient to adjust the weights:

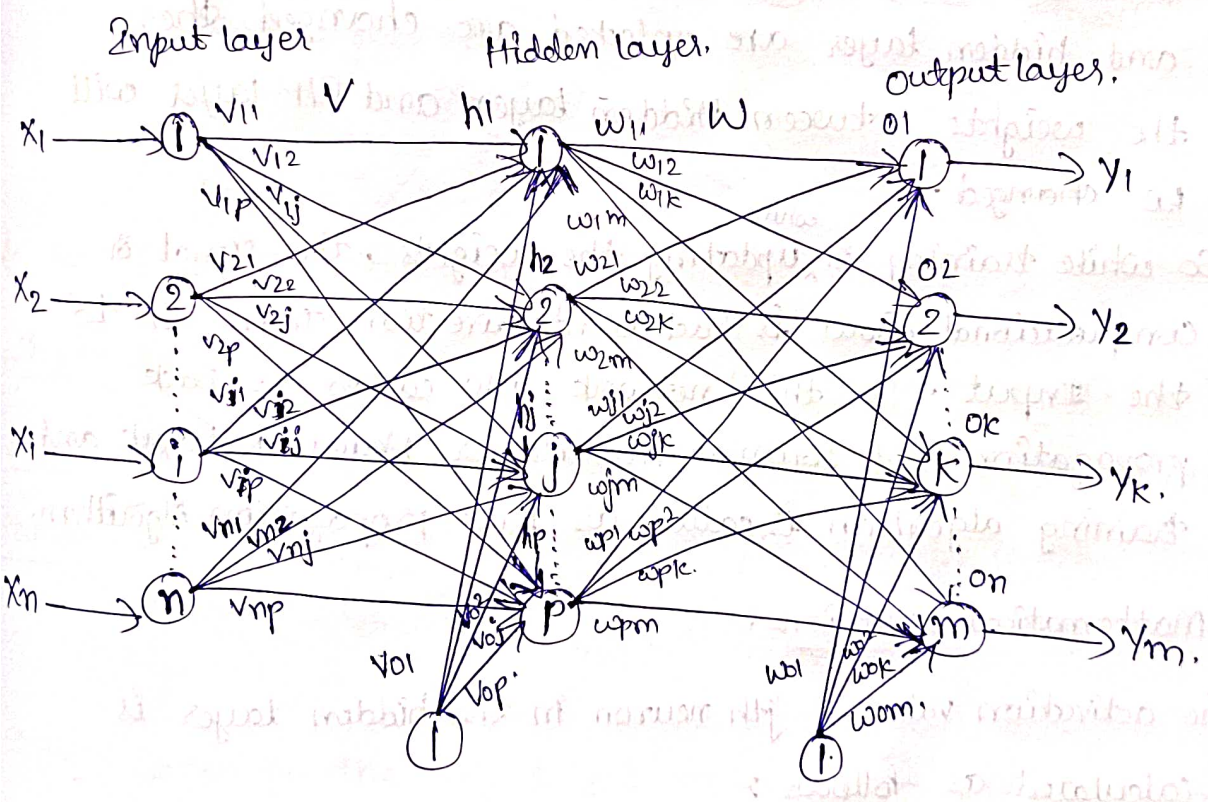
$w_{12}, w_{22}, \cdots\cdots, w_{n2}$ to train the NN.

14/12/23

UNIT-II

* Back Propagation Algorithm :-

- Back propagation Algorithm is a step by step procedure to train the multi-layer Neural network.
- It is also called as Generalised Delta learning rule.
- Back propagation uses Gradient descent method to minimize squarred error.
- It is a supervised learning Algorithm.
- The architecture of the Back propagation neural network is as shown in below figure:



- Back propagation neural network is having 3 layers :-
  1.) Input layer 2.) Hidden layer and 3.) Output layer.
- Since the computations are performed at more than one layer, it is called as Multi layer Neural Network.
- While calculating the output of the Neural network, the signal flows in feed forward direction only from the input to the output. So this network is also called as

feed forward Multi-layer Neural Network.

- Once the actual output of the Neural Network calculated, it is compared with the Target value and error is calculated. There are 2 ways of calculation of errors:-

1.) Absolute Error → $E_k = t_k - y_k$.

2.) Squarred Error. → $E_k = \frac{1}{2}(t_k - y_k)^2$.

* Most of the training algorithms make use of squared error, as it is drastically decreases iteration by iteration.

- To minimise the error, first the weights between o/p layer and hidden layer are updated are changed then the weights between Hidden layer and I/P layer will be changed.

- So while training & while updating the weights, the signal & computational flow is backward direction from o/p to the Input. So this network also called as Back propagation feed forward Multilayer Neural Network and training algorithm is called as Back Propagation Algorithm.

* Mathematical expressions:-

- The activation value of jth neuron in the hidden layer is calculated as follows:-

$$h_j = x_1 v_{1j} + x_2 v_{2j} + \cdots + x_i v_{ij} + \cdots + x_n v_{nj} + v_{oj}.$$

$$\boxed{h_j = v_{oj} + \sum_{i=1}^{n} x_i v_{ij}.}$$

- The output of the jth neuron is calculated by processing activation value h_j through sigmoidal activation function.

$$\boxed{H_j = \frac{1}{1+e^{-h_j}}.}$$

- The activation value of kth neuron in the output layer is calculated as follows:-

$$O_k = H_1 W_{1k} + H_2 W_{2k} + \cdots + H_j W_{jk} + \cdots + H_p W_{pk} + W_{ok}.$$

$$\boxed{O_k = W_{ok} + \sum_{j=1}^{p} H_j W_{jk}.}$$

- The o/p of the kth neuron in the o/p layer is calculated using;

$$\boxed{Y_k = \frac{1}{1+e^{-O_k}}.}$$

- Once the o/p is calculated, squarred error is calculated as follows:-

$$\boxed{E = \frac{1}{2}(Y_k - t_k)^2.}$$

- To minimise the error, weights are adjusted as follows:

$$W_{new} = W_{old} + \Delta W.$$
$$W_{onew} = W_{oold} + \Delta W_o.$$
$$V_{new} = V_{old} + \Delta V.$$
$$V_{onew} = V_{oold} + \Delta V_o.$$

Back Propagation Algorithm:-

Step1:- Initialise weights V, V_o, W, W_o to random values.

Step2:- Apply the inputs $X_1, X_2, \ldots, X_i, \ldots, X_n$ to the I/P layer.

Step3:- Calculate activation value of jth neuron in the hidden layer;

$$h_j = V_{oj} + \sum_{i=1}^{n} X_i V_{ij}.$$

Step4:- Calculate the o/p of jth neuron in the hidden layer.

$$H_j = \frac{1}{1+e^{-h_j}}.$$

Step5:- Determine the activation value of kth neuron in the o/p layer.

$$O_k = W_{ok} + \sum_{j=1}^{p} H_j W_{jk}.$$

Step6:- Calculate the o/p of kth neuron in the o/p layer.

$$Y_k = \frac{1}{1+e^{-O_k}}.$$

Step-7:- Calculate the error at the kth neuron in the o/p layer.

$$E = \frac{1}{2}(Y_k - t_k)^2.$$

Step-8:- The weights are continuously adjusted until the squared error is below Tolerance value (0.001).

$$W_{new} = W_{old} + \Delta W.$$

$$W_{onew} = W_{oold} + \Delta W_o.$$

$$V_{new} = V_{old} + \Delta V.$$

$$V_{onew} = V_{oold} + \Delta V_o.$$