Page No.
Date

Q1] What is soft computing? List and explain its application

Ans Soft computing is a computing model develope to solve the non linear problems which involve uncertain, imprecise and approximate solutions of a problem
These types of problems are considered as real-life problems where the human like intelligence is required to solve it.
The soft computing term is define by Dr. Lotfi Zadeh, according to him, soft computing is an approach which imitates the human mind to reason and learns it in an environment of uncertainty and impression.
It is aeated through two elements adaptivity and knowledge and has a set of tools such as fuzzy logic, neural networks, genetic algorithm etc.
The soft computing model is distinct from its model known as hard computing model because it does not work on the mathematical model of problem soling.

* Application of Soft Computing
1] Agricultural Production Engineering
2] Medicine and Biology application
3] Construction and Design Engineering
4] Computer Engineering

Teacher's Sign.: _____

5] Computational Process
6] Natural Environmental Engineering
7] Fault - Tolerance.
8] Machine learning
9] Signal processing
10] Mechanical engineering
11] Materials Engineering
12] Disease diagnosis
13] Nano ~~tec~~ technology
14] Pattern Recognition.

Page No.
Date

Q2) Give comparison between Soft and Hard computing

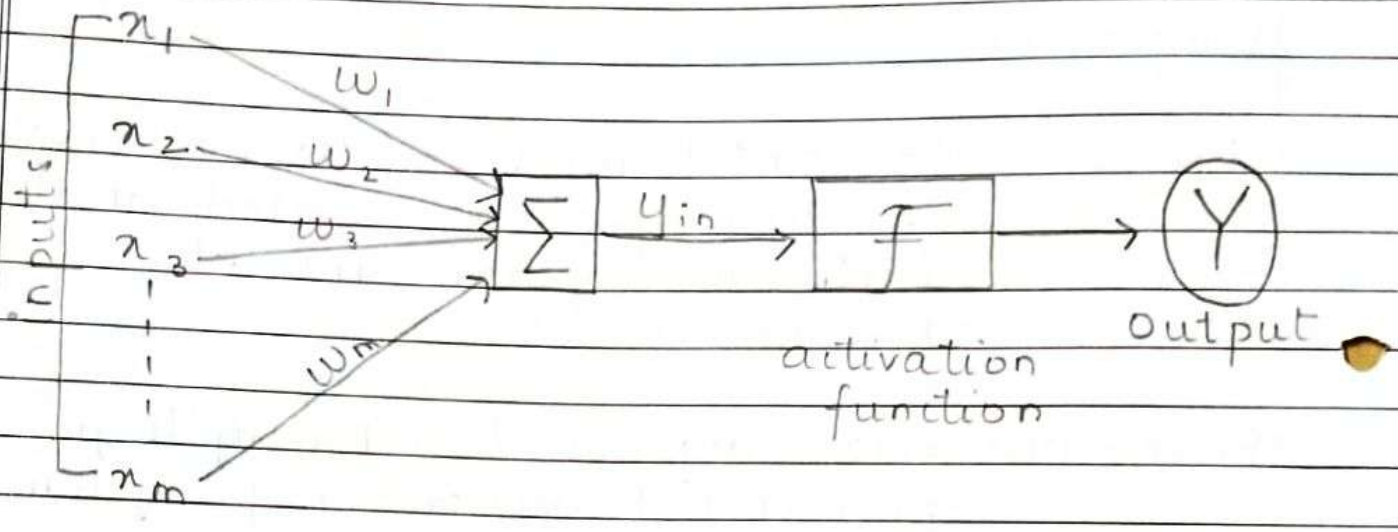
Basis for comparison	Soft Computing	Hard computing
Basic	Tolerant to imprecision uncertainity, partial truth and approximation	Uses precisely stated analytical model.
Based on	Fuzzy logic and probabilistic reasoning	Binary logic and crisp system
Features	Approximation and dispositionality	Precision and categoricity
Nature	stochastic	Deterministic
Works on	Ambiguous and noisy data	Exact input data.
Computation	Can perform parallel computation	Sequential.
Result	Approximate	Produces precise outcome

Teacher's Sign.: _____

Page No.
Date

Q3] Explain the structure of neural network with suitable diagram.

Ans.



1] An artificial neuron is a mathematical function conceived as a model of biological neurons, a neural network.

2] Artificial neurons are elementary units in an artificial neural network.

3] The artifical neuron receives one or more inputs and sums them to produce an output

4] Neuron are the basic units of a neural network.

5] In natue, neurons have a number of dendrites (input) a cell nucleus (processor) and an axon (output) ...

6] As you can see they have several inputs, for each input there is a weight (the weight of that specific condition.

Page No.
Date

Q4] what is probabilistic reasoning? explain with example

Ans Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainity in knowledge.
In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty
We use probability in probabilistic reasoning because it provides away to handle the uncertainity that is the result of someone's laziness and ignorance
In the real world there are lots of scenarios where the certainity of something is not confirmed such as "It will rain today." "behavior of someone for some situations." "A match between two teams or two players".
There are probable sentences for which we can assume that it will happen but not sure about it, so here we use probabilistic reasoning

When there are unpredictable outcomes
when specification or possibilities of predicates becomes too large to handle
when an unknown error occurs during an experiment

Teacher's Sign: _____

Page No.
Date

25] list and explain different types of activation function

Ans There are several types of activation function.

1] Identity function
It is a linear function and can be defined as:

$$f(x) = x \text{ for all } x$$

The output here remains the same as input. The input layer uses the identity activation function.

2] Binary step function
This function can be defined as

$$f(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ 0 & \text{if } x < \theta \end{cases}$$

where θ represents the threshold value. This function is more widely used in single layer nets to convert the net input to an output that is binary (1 or 0).

3] Bipolar step function
This function can be defined as:

$$f(x) \begin{cases} 1 & \text{if } x \geq \theta \\ -1 & \text{if } x < \theta \end{cases}$$

where θ represents the threshold value. This function is also used in single layer nets

to convert the net input to an output that is bipolar (+1, -1)

4] Sigmodial function - The sigmodial function are widely used in back propogation net because of the relationship between the value of the functions at a point and the value of the derivatives at that point which reduces the computational burden during training.

a] Binary sigmoid function - It is also termed as logistic sigmoid function or unipolar sigmoid function

It can be defined as

$$f(x) = \frac{1}{1 + e^{-\lambda x}}$$

where λ is the steepness parameter. The derivative of the function is

$$f'(x) = \lambda f(x)[1 - f(x)].$$

It ranges the sigmoid function is from 0 to 1.

b] Bipolar sigmoid function.

This function can be defined as.

$$f(x) = \frac{2}{1 + e^{-\lambda x}} - 1 = \frac{1 - e^{-\lambda x}}{1 + e^{-\lambda x}}$$

where λ is the steepness function, ranges is between -1 and +1. The derivative function

$$f(x) = \frac{\lambda}{2}[1 + f(x)][1 - f(x)]$$

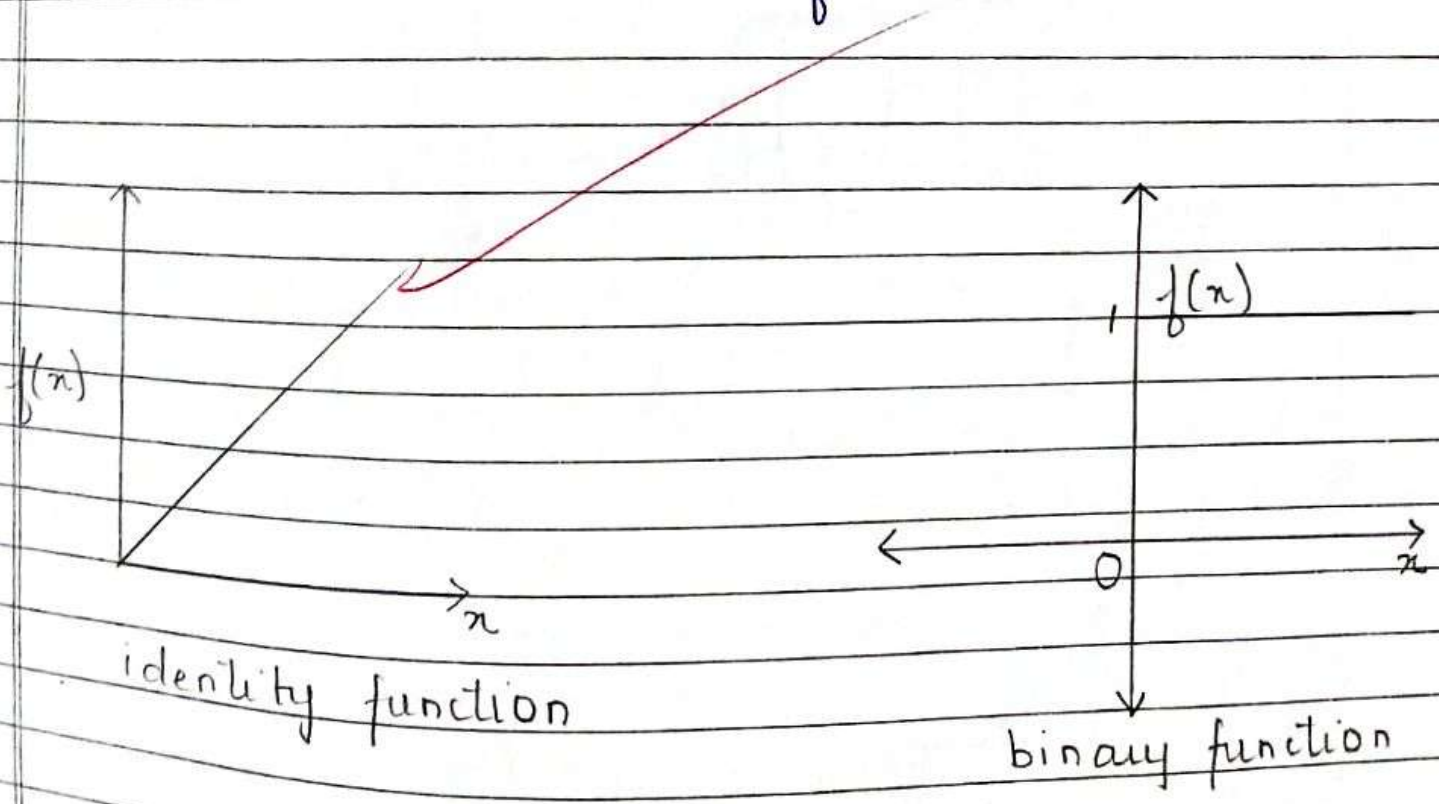
The bipolar sigmoid function is closely related to hyperbolic tangent function.

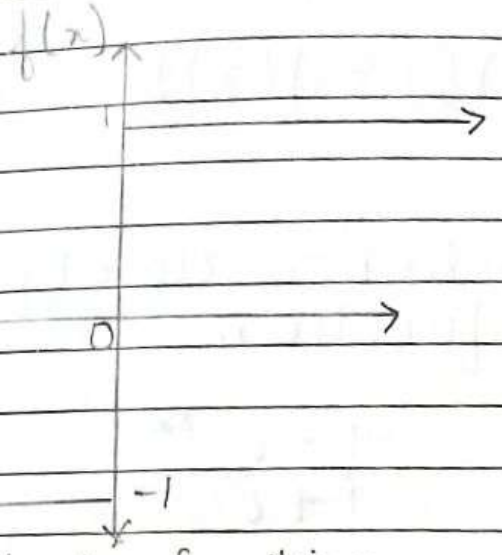
$$b(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

5] Ramp function

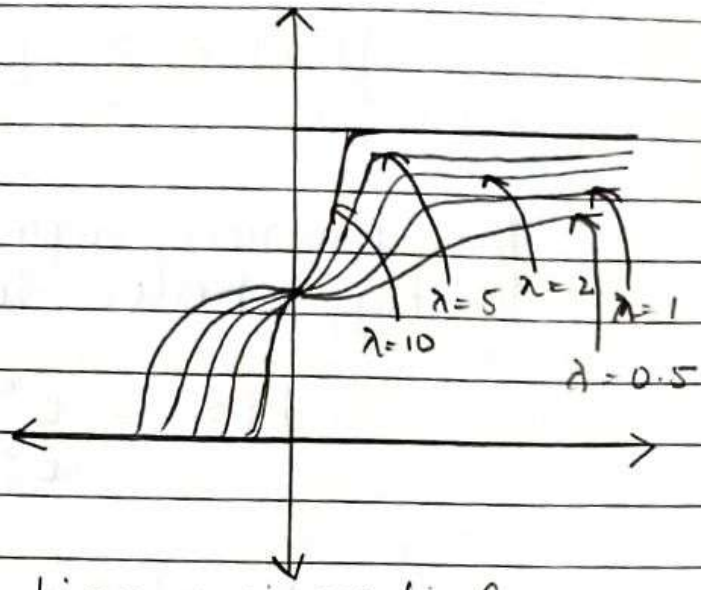
The ramp function can be defined as

$$f(x) = \begin{cases} 1 & \text{if } x > 1 \\ x & \text{if } 0 \leq x \leq 1 \\ 0 & \text{if } x < 0 \end{cases}$$

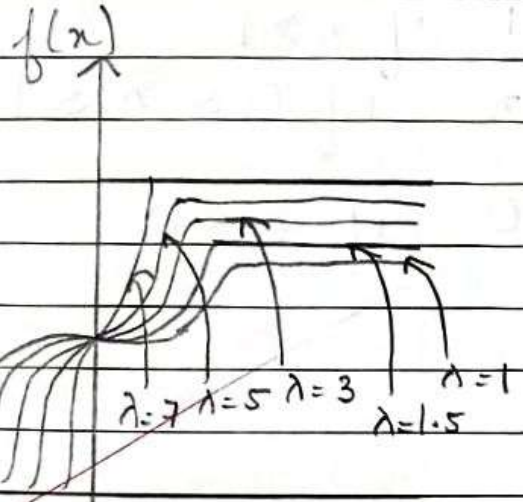




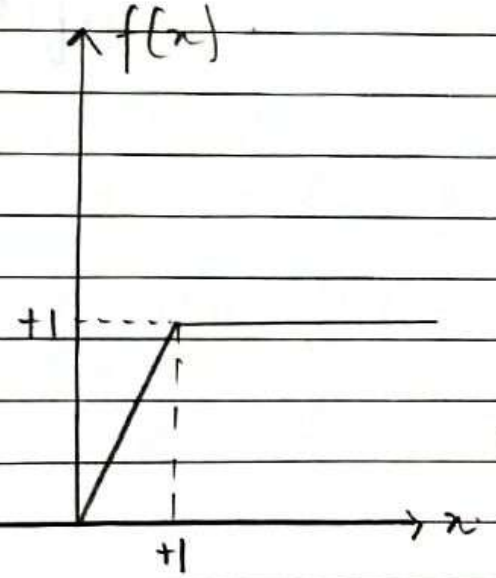
Bipolar function



binary sigmodial function.



bipolar sigmodial function.

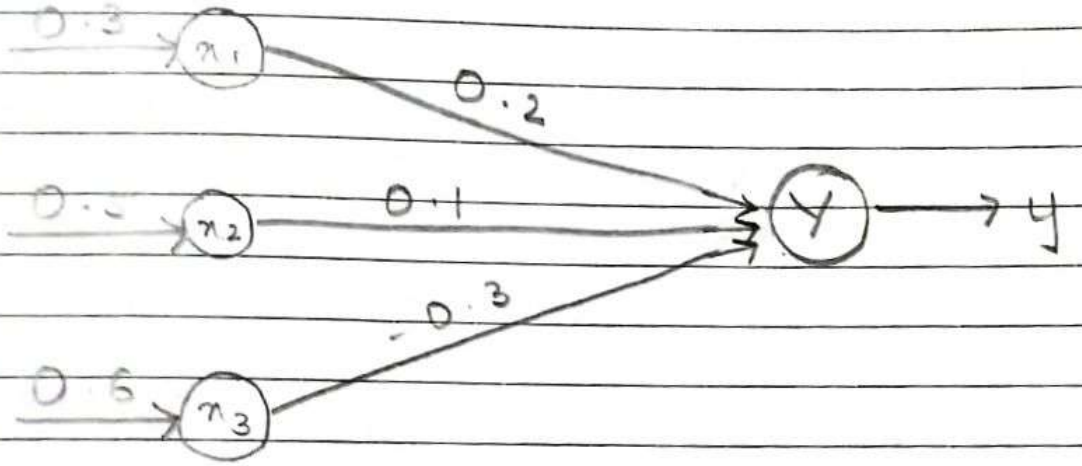


Ramp function

Teacher's Sign.: _____

Page No.
Date

Q6] For the given network calculate the output y neuron, for given inputs and weight $[x_1\ x_2\ x_3] = [0.3, 0.5, 0.6]$ $[w_1, w_2, w_3] = [0.2, 0.1, -0.3]$.



$[x_1\ x_2\ x_3] = [0.3, 0.5, 0.6]$
$[w_1\ w_2\ w_3] = [0.2, 0.1, -0.3]$

The net input can be calculated as.

$$y_{in} = x_1w_1 + x_2w_2 + x_3w_3$$
$$= 0.3 \times 0.2 + 0.5 \times 0.1 + 0.6 \times (-0.3)$$
$$= 0.06 + 0.05 - 0.18$$
$$y_{in} = -0.07.$$

Teacher's Sign.:

Q1] Design a hebb network to implement AND function (use bipolar input and target).

Ans

Inputs			Target
x_1	x_2	b	y
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Initially the weights and bias are set to zero

$$w_1 = w_2 = b = 0$$

Fist input $[x_1\ x_2\ b] = [1\ 1\ 1]$ target = 1
setting the initial weights as old weight and applying Hebb rule

$$w_i(new) = w_i(old) + x_i y$$

$$w_1(new) = w_1(old) + x_1 y = 0 + 1 \times 1 = 1$$

$$w_2(new) = w_2(old) + x_2 y = 0 + 1 \times 1 = 1$$

$$b(new) = b(old) + y = 0 + 1 = 1.$$

Second input $[x_1\ x_2\ b] = [1\ -1\ 1] = -1$
setting the old weight $[w_1\ w_2\ b] = [1\ 1\ 1]$

The weight changes here is

$$\Delta w_1 = x_1 y = 1 \times -1 = -1$$

$$\Delta w_2 = x_2 y = -1 \times -1 = 1$$

$$\Delta b = y = -1$$

The new weight here are

$w_1(new) = w_1(old) + \Delta w_1 = 1 - 1 = 0$

$w_2(new) = w_2(old) + \Delta w_2 = 1 + 1 = 2$

$b(new) = b(old) + \Delta b = 1 - 1 = 0$.

Third input = $[x_1 \; x_2 \; b] = [-1 \; 1 \; 1] = -1$

setting old weight $[w_1 \; w_2 \; b] = [0 \; 2 \; 0]$.

The weight changes here.

$\Delta w_1 = x_1 y = (-1)(-1) = 1$

$\Delta w_2 = x_2 y = 1 \times -1 = -1$

$\Delta b = y = 1 \times -1 = -1$

The new weight here are.

$w_1(new) = w_1(old) + \Delta w_1 = 0 + 1 = 1$

$w_2(new) = w_2(old) + \Delta w_2 = 2 + (-1) = 1$

$b(new) = b(old) + \Delta b = 0 + (-1) = -1$

Fourth input $[x_1 \; x_2 \; b] = [-1 \; -1 \; 1] = -1$

setting old weight $[w_1 \; w_2 \; b] = [1 \; 1 \; -1]$

$\Delta w_1 = x_1 y = -1 \times -1 = 1$

$\Delta w_2 = x_2 y = -1 \times -1 = 1$

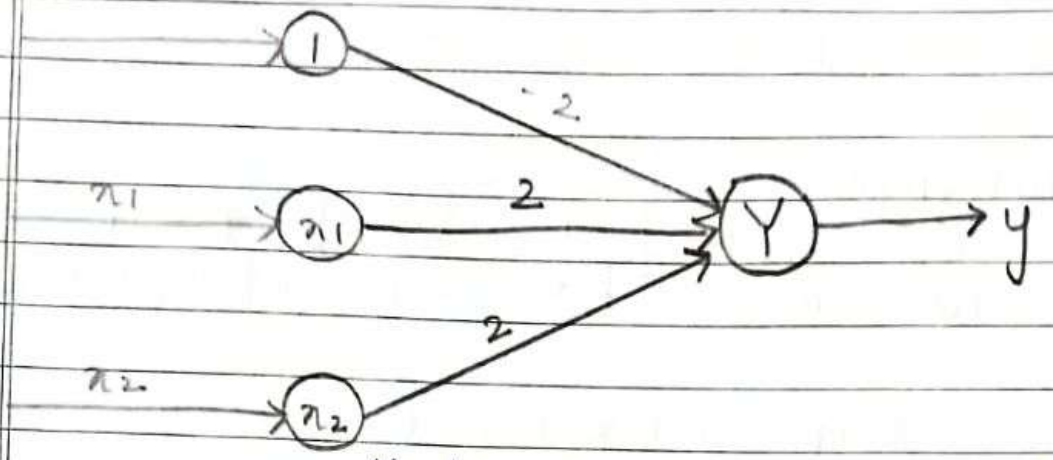
$\Delta b = y = 1 \times -1 = -1$

The weight changes here.

$w_1(new) = w_1(old) + \Delta w_1 = 1 + 1 = 2$

$w_2(new) = w_2(old) + \Delta w_2 = 1 + 1 = 2$

$b(new) = b(old) + \Delta b = -1 + (-1) = -2$



Hebb ~~functi~~ network for AND function.

Q2] Design a hebb network to logical AND NOT function (bipolar input and target).

Truth Table

x_1	x_2	y
1	1	-1
1	-1	1
-1	1	1
-1	-1	1

Initially the weights and bias both are set to zero

$$w_1 = w_2 = b = 0$$

First input $[x_1\ x_2\ b] - [1\ 1\ 1]$ $t = -1$

$w_1(new) = w_1(old) + x_1 y = 0 + 1(-1) = -1$

$w_2(new) = w_2(old) + x_2 y = 0 + 1(-1) = -1$

$b = 0 + (-1) = -1$

$\Delta w_1 = 1 \times -1 = -1$

$\Delta w_2 = 1 \times -1 = -1$

$\Delta b = -1$.

Second input

$[x_1 \quad x_2 \quad b] \quad [1 \quad -1 \quad 1] \quad t = 1$

$[w_1 \quad w_2 \quad b] = [-1 \quad -1 \quad -1]$ (old weight)

$\Delta w_1 = x_1 y = 1 \times 1 = 1$

$\Delta w_2 = x_2 y = -1 \times 1 = -1$

$\Delta b = y = 1$

The weights changes are here.

$w_{1(new)} = w_{1(old)} + \Delta w_1 = -1 + 1 = 0$

$w_{2(new)} = w_{2(old)} + \Delta w_2 = -1 + (-1) = -2$

$b = b_{(old)} + \Delta b = +1 + -1 = 0$

third input

$[x_1 \quad x_2 \quad b] = [-1 \quad 1 \quad 1] \quad t = 1.$

$[w_1 \quad w_2 \quad b] = [0 \quad -2 \quad 0]$ (old weight)

$\Delta w_1 = x_1 y = -1(1) = -1$

$\Delta w_2 = x_2 y = 1(1) = 1$

$\Delta b = y = 1$

The weight changes are here.

$w_{1(new)} = w_{1(old)} + \Delta w_1 = 0 + (-1) = -1$

$w_{2(new)} = w_{2(old)} + \Delta w_2 = -2 + 1 = -1.$

$b_{(new)} = b_{(old)} + y = 0 + 1 = 1.$

Fourth input =

$[x_1 \quad x_2 \quad b] = [-1 \quad -1 \quad 1] \quad t = 1$

$[w_1 \quad w_2 \quad b] = [-1 \quad -1 \quad 1]$ (old weight).

$\Delta w_1 = x_1 y = -1(1) = -1$

$\Delta w_2 = x_2 y = 1(1) = -1$

$\Delta b = y = 1 = 1$

The new weights here are

$w_1 = w_{(old)} + \Delta w_1$

$= -1 + -1 = -2$

$w_2 = w_{(old)} + \Delta w_2$

$= -1 + (-1) = -2$

$b = 1 + 1 = 2$

Q3] Draw and explain the perception neuron with its architecture.

Ans: Perceptron networks comes under single layer feed forward network and are also called simple perceptions. various types of perceptions are designed by Rosenblatt (1962). A single perceptions networks was discovered by Block in 1962

The perception network consists of three units namely, sensory unit, associator unit and response unit

The sensory units are connected to associator units with fixed weights having values 1, 0 or -1 which are assigned at random.

The binary activation function is used in sensory unit and associator unit

The response unit has an activation of 1, 0 or -1. The binary step with fixed threshold θ is used as activation for associator. The output signals that are sent from the associator unit to the response unit are only binary.

The output of the perception network.

$$y = f(y_{in}).$$

where $f(y_{in})$ is activation and is defined as:

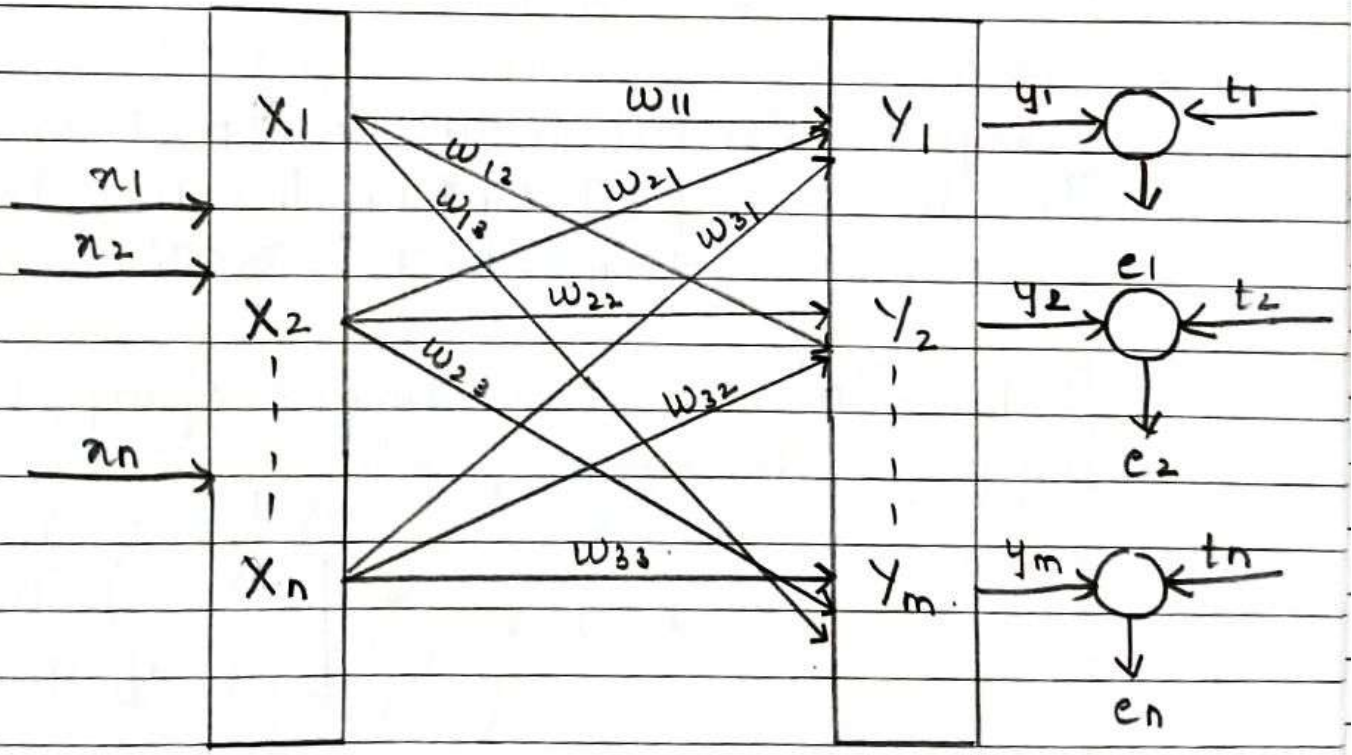
$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \leq y_{in} \leq \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

The perception learning rule is used in the weight updation between the associator unit and the response unit. for each training input the net will calculate the response and it will determine whether or not an error has occured. The error calculation is based on the comparision of the values of targets with those of the calculated output

The weight on the connection from the units the nonzero signal will get adjusted suitably. The weight will be adjusted on the basis of the learning rule if an error has occured for pattern.

$$w_{i(new)} = w_{i(old)} + \alpha t x_i$$

$$b_{(new)} = b_{(old)} + \alpha t$$



Q4) Explain perception training algorithm for single output class

Ans step 0 - Initialize the weight and the bias. Also initialize the learning rate $\alpha (0 < \alpha \leq 1)$ For simplicity α is set to 1.

step 1 - Perform steps 2-6 until the final stopping condition is false.

step 2 - Perform step 3-5 for each training pair indicated by s:t

step 3 - The input layer containing input units is applied with identity function

$$x_i = s_i$$

step 4 - Calculate the output of the network. To do so, first obtain the net input

$$y_{in} = b + \sum_{i=1}^{n} x_i w_i$$

where 'n' is the number of input neuron then apply activation function

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \leq y_{in} \leq \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

step 5 - Weight and bias adjustment compare the value of the actual output and

output

If $y \neq t$, then

$$w_i(new) = w_i(old) + \alpha t x_i$$

$$b(new) = b(old) + \alpha t$$

else we have

$$w_i(new) = w_i(old)$$

$$b(new) = b(old).$$

step 6 - Train the network until there is no weight change. This is the stopping condition for the network.

If this condition is not met, then start again from step 2

05] Implement logical OR function with binary input and target using perceptron training algorithm upto 1 epoch.

The truth table for OR function with binary input

x_1	x_2	t
1	1	1
1	0	1
0	1	1
0	0	0

The perception network which uses perception learning rule, is used to train the OR function. The network architecture. The intial value of the weights and bias are taken as zero

$$w_1 = w_2 = b = 0.$$

$$y_{in} = b + x_1 w_1 + x_2 w_2$$
$$= 0 + 1(0) + (1)(0) = 0$$
$$y \neq t \qquad y = 0.$$

$$w_1 = w_{(new)} + \alpha t x_1 = 0 + 1(1)(1) = 1$$
$$w_2 = w_{(new)} + \alpha t x_2 = 0 + 1(1)(1) = 1$$
$$b = b_{(new)} + \alpha t = 0 + 1(1) = 1.$$
$$\Delta w_1 = \alpha t x_1 = 1(1)(1) = 1$$
$$\Delta w_2 = \alpha t x_2 = 1(1)(1) = 1$$
$$\Delta b = \alpha t = 1(1) = 1.$$

Second input

$$[w_1 \; w_2 \; b] = [1 \; 1 \; 1]$$
$$[x_1 \; x_2 \; t] = [1 \; 0 \; 1]$$

$$y_{in} = b + x_1 w_1 + x_2 w_2$$
$$= 1 + 1(1) + 0(1) = 2 \quad \cancel{y \neq t}.$$
$$y = 1 \qquad y = t$$

~~weight~~ no weight changes take place

$$w_{(new)} = w_{(old)} =$$
$$1 = 1$$

$$w_2(new) = w_2(old) = 1 = 1$$

$b(new) = b(old) = 1 = 1$

third input

$[w_1 \; w_2 \; b] = [1 \; 1 \; 1] \; [x_1 \; x_2 \; b] = [0 \; 1 \; 1]$

$y_{in} = b + x_1 w_1 + x_2 w_2$

$= 1 + 0(1) + 1(1) = 2$

$= y = 1 \qquad y = t$

no weight changes took place.

$w(new) = w_1(old)$

$1 = 1.$

$w_2(new) = w_2(old) = 1 = 1$

$b = b(old)$

Fourth input

$[w_1 \; w_2 \; b] = [1 \; 1]$

$[x_1 \; x_2 \; b] = [0 \; 0 \; 0].$

$y_{in} = b + x_1 w_1 + x_2 w_2$

$= 0 + 0(0) + 0(0)$

$= 0.$

$y = 0.$

No weights changes take place

Q1 Define learning, differentiate between supervised learning and unsupervised learning

The main property of an ANN is its capability to learn. learning or training is a process by means of which a neural network adapts itself to a stimulus by making proper parameter adjustment resulting in the production of desired response ~~Broadly there~~.

	Supervised Learning	Unsupervised Learning
Input Data	Uses known and labeled Data as input	Uses Unknown Data as input
Computational Complexity	Very complex	less computation Complexity
Real Time	Uses off-line analysis	Uses Real-Time analysis of Data.
Number of Classes	Number of Classes are known	Number of Classes are not known.
Accuracy of Result	Accurate and Reliable Results	Moderate Accurate and Reliable Results.

7] state the training algorithm used for the Hebb network.

Ans The training algorithm

Step 0 - First initialize the weights. Basically in this network they may be set to zero i.e $w_i = 0$ for $i = 1$ to n where 'n' may be the total number of input neurons

step 1 - steps 2 - 4 have to be performed for each input training vector and target output pair s : t

~~The above five steps complete the algorithm process. In~~

Step 2 - Input unit activations are set. Generally the activation function of input layer is identity function: $x_i = s_i$ for $i = 1$ to n.

step 3 - Output unit activation are set $y_i = t_i$

step 4 - weight adjustment and bias adjustment are performed

$$w_i(new) = w_i(old) + x_i y$$

$$b(new) = b(old) + y$$

$$w(new) = w(old) + xy.$$

Here the change in weight can be expressed as

$$\Delta w = xy$$

As a result

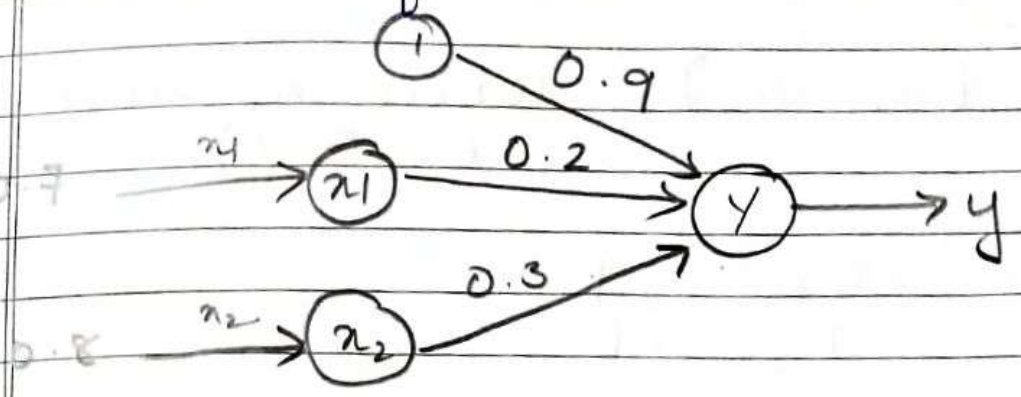
$$w(new) = w(old) + \Delta w$$

8] Write a short note on McCulloch Pits neuron model.

The McCulloch Pits neuron was the earliest neural network discovered in 1943. It is usually called as M-P-neuron. The M-P neurons are connected by directed weighted paths. It should be noted that the activation of a M-P neuron is binary, that is at any time step the neuron may fire or may not fire. The weights associated with the communication links may be excitatory (weight is positive) or inhibitory (weight is negative). All the excitatory connected weights entering into a particular neuron will have same weights. The threshold plays a major role in M-P-neuron.

There is a fixed threshold for each neuron and if the net input to the neuron is greater than the threshold then the neuron fires. Also it should be noted that any nonzero ~~inhi~~ inhibitory input would prevent the neuron ~~fires~~ from firing. The M-P neuron are most widely used in the case of logic function.

Q9] Calculate the output of neuron Y for the net shown in figure. Use binary and bipolar activation function.



$$y_{in} = b + x_1 w_1 + x_2 w_2$$
$$= 0.9 + 0.2 \times 0.7 + 0.3 \times 0.8$$
$$= 0.9 + 0.14 + 0.24$$
$$y_{in} = 1.28$$

Using Binary Sigmoidal function:

$$y_{in} = f(y_{in}) = \frac{1}{1 + e^{-y_{in}}}$$
$$= \frac{1}{1 + e^{-1.28}}$$
$$y_{in} = 1.27$$

Bipolar Sigmoidal function:

$$y_{in} = f(y_{in}) = \frac{2}{1 + e^{-y_{in}}} - 1$$
$$= \frac{2}{1 + e^{-1.28}} - 1 = \frac{1 - e^{-1.28}}{1 + e^{-1.28}}$$
$$= 1$$

Q10] Design neural network with only one MP neuron that implements the basic logic operation

NAND $(x_1 \quad x_2)$ where x_1 and $x_2 \in (0, 1)$.

x_1	x_2	y
0	1	1

consider $w_1 = 1$, $w_2 = 1$

$$y_{in} = x_1 w_1 + x_2 w_2$$
$$= 0(1) + 1(1)$$
$$= 0 + 1 = 1.$$

θ - threshold

$$\theta \geq nw - p$$
$$= 1 \times 0 - 1$$
$$\theta = -1.$$

$$y = f(y_{in}) = \begin{cases} 1 & y_{in} \geq 1 \\ 0 & y_{in} < 1 \end{cases}$$

consider $w_1 = 1 \quad w_2 = -1.$

$$y_{in} = x_1 w_1 + x_2 w_2$$
$$= 0(1) + 1(-1)$$
$$= 0 - 1 = -1.$$

θ = threshold

$\theta \geq nw - p$.

$= 1 \times 1 - 1$

$\theta = 0$

$$y = f(y_{in}) = \begin{cases} 1 & y_{in} \geq 1 \\ 0 & y_{in} < 1 \end{cases}$$