Hebbian Learning Algorithm

Hebb network was stated by Donald hebb in 1949. According to Hebbs rule, the weights are found to increase proportionately to the product of input and output. It means that in a Hebb network, if two neuron are interconnected then the weights associated with neurons can be increased by changes in the synaptic gap.

This network is suitable for bipolar data. The hebbian learning rule is generally applied to logic gates. The weights are updated as.

$$w_{new} = w_{old} + x \cdot y.$$

* Implement logical AND. function with bipolar inputs using hebbian learning.

$w_1 = w_2 = 0$, $b = 0$, $\alpha =$

Input		Bias	target
x_1	x_2	b	y
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

First input vector
[1, 1], target is 1
$w_{new} = w$, $y = sign(wix_i) = -1$,
w_{old} = The new weights will be
$w_{new} = w_{old} + x * y$
$w_{1new} = w_{1old} + x_1 * y$
$= 0 + 1 * 1$
$= 1$ $\therefore \Delta w_1 = 1$

$w_{2new} = w_{2old} + x_2 * y$
$= 0 + 1 \times 1$
$= 1$ $\therefore \Delta w_2 = 1$

$b_{new} = b_{old} + y$
$= 0 + 1$
$= 1.$ $\therefore \Delta b = 1$

Second input
(1, −1), target is −1, $y = sign(wix_i)$
$w_{1new} = w_{1old} + x_1 * y$
$= 1 + 1 * -1$
$= 0$ $\therefore \Delta w_1 = -1$

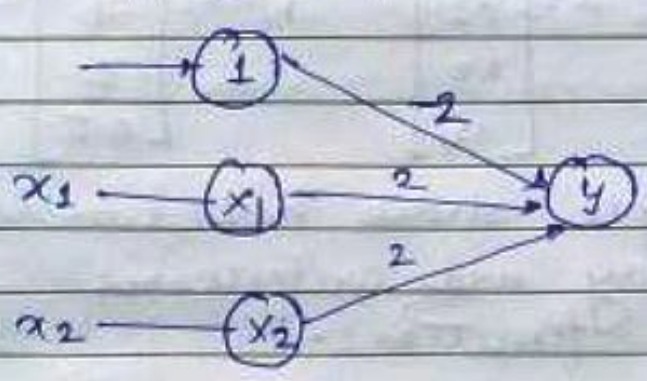
$w_{2new} = w_{2old} + x_2 * y$
$= 1 + -1 * -1$
$= 2$ $\therefore \Delta w_2 = 1$

$b_{new} = b_{old} + y$
$= 1 + -1$
$= 0$ $\therefore \Delta b = -1$

Similarly the other weights are calculated for all inputs.

Input		bias	target	Δw_1	Δw_2	Δb	w_1	w_2	b
x_1	x_2	b	y						
1	1	1	1	1	1	1	1	1	1
1	-1	1	-1	-1	1	-1	0	2	0
-1	1	1	-1	1	-1	-1	1	1	-1
-1	-1	1	-1	1	1	-1	2	2	-2

Hebb network for AND function



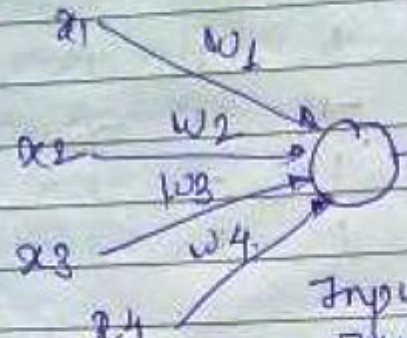
Hebbian learning is purely feed forward, unsupervised learning and can work for all activation functions:

If cross product of output and input is positive, this result in an increase of weights, otherwise the weight decreases

$$\Delta w_i = c r x$$
$$= c f(w_i^t x) x$$
$$\Delta w_{ij} = c f(w_i^t x) x_j$$

(Zuroda)

Hebbian learning



Input

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Initial weights

$$w^t = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

This example shows hebbian learning with binary and continuous activation functions.

$c = 1$

Since the initial weights are of nonzero value the network has been trained beforehand.

1) Assume first that bipolar binary neurons are used, and thus $f(net) = sgn(net)$

Step 1:

Input X_1 applied to network results in activation of net as below

$$net^1 = w^{1t} x_1 = \begin{bmatrix} 1 & -1 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = 3$$

The updated weights are

$$w^2 = w^1 + sgn(net^1) x_1$$

$$= w^1 + x_1$$

$$w^2 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ -3 \\ 1.5 \\ 0.5 \end{bmatrix}$$

Step 2:

$$net^2 = w^{2t} x_2 = \begin{bmatrix} 2 & -3 & 1.5 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ 1.5 \end{bmatrix} = -0.25$$

The updated weights are

$$w^3 = w^2 + sgn(net^2) x_2 = w^2 - x_2 = \begin{bmatrix} 1 \\ -2.5 \\ 3.5 \\ 2 \end{bmatrix}$$

Step 3:

$$net^3 = w^{3t} x_3 = \begin{bmatrix} 1 & -2.5 & 3.5 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix} = -3$$

The updated weights are

$$w^4 = w^3 + sgn(net^3) x_3 = w^3 - x_3 = \begin{bmatrix} 1 \\ -3.5 \\ 4.5 \\ 0.5 \end{bmatrix}$$

It can be seen that learning with discrete $f(net)$ and $c = 1$ results in adding or subtracting the entire input pattern vectors to and from the weight vector respectively.

In the case of a continuous f(net). the weight corresponden incrementing / decrementing vector is scaled down to a fractional value of the input pattern.

2) Continuous bipolar activation function.

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

Solution :- Set $x = x_1$, assume $\lambda = 1$

$$x = \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}$$

$$net_1 = w_1^t x = \begin{bmatrix} 1 & -1 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = 3$$

$$o_1 = f(net_1) = \frac{2}{1+\exp^{(-3)}} - 1 = \frac{2}{1+0.049} - 1 = 0.906$$

$$\Delta w_1 = c\, o_1\, x_1$$

$$= (1) \times 0.906 \times \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.906 \\ -1.812 \\ 1.359 \\ 0 \end{bmatrix}$$

So the updated w_2 is calculated as

$$w_2 = w_1 + \Delta w_1$$

$$= \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 0.906 \\ -1.812 \\ 1.359 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.906 \\ -2.812 \\ 1.359 \\ 0.5 \end{bmatrix}$$

step 2: $X = X_2$

$$X = \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}$$

$$net_2 = W_2^t X$$

$$= \begin{bmatrix} 1.906 & -2.812 & 1.359 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}$$

$$= [1.906 + 1.406 - 2.718 - 0.75]$$

$$= -0.156$$

$$O_2 = f(net_2) = \frac{2}{1 + e^{0.156}} - 1 = -0.0774$$

$$\Delta W_2 = -C O_2 X$$

$$= -(1) \times (-0.0774) \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix} = \begin{bmatrix} -0.0774 \\ 0.0387 \\ 0.1548 \\ 0.1161 \end{bmatrix}$$

So, updated weight vector W_3 is computed as

$$W_3 = W_2 + \Delta W_2$$

$$= \begin{bmatrix} 1.906 \\ -2.812 \\ 1.359 \\ 0.5 \end{bmatrix} + \begin{bmatrix} -0.0774 \\ 0.0387 \\ 0.1548 \\ 0.1161 \end{bmatrix} = \begin{bmatrix} 1.828 \\ -2.773 \\ 1.513 \\ 0.616 \end{bmatrix}$$

Step 3: Set $X = X_3$

$$X = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

$$net_3 = W_3^t X$$

$$= \begin{bmatrix} 1.828 & -2.773 & 1.513 & 0.616 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix} = -3.362$$

$$O_3 = f(net_3) = \frac{2}{1 + e^{3.362}} - 1 = -0.932$$

$$\Delta W_3 = C O_3 X$$

$$= (1) \times (-0.932) \times \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.932 \\ 0.932 \\ -1.398 \end{bmatrix}$$

The updated weight vector W_4 is computed as

$$W_4 = W_3 + \Delta W_3$$

$$= \begin{bmatrix} 1.828 \\ -2.773 \\ 1.513 \\ 0.616 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.932 \\ 0.932 \\ -1.398 \end{bmatrix} = \begin{bmatrix} 1.828 \\ -3.705 \\ 2.445 \\ -0.782 \end{bmatrix}$$

★ Given the following input vectors and initial weight vector determine final weights for hebbian learning of a single-neuron network.

$w_1 = [1, -1]^t$ $c = 1$

$x_1 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$ $x_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ $x_3 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ $x_4 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$

Solve for:

1) Bipolar, binary activation
2) Bipolar continuous activation ($\lambda = 1$)

Solution :=

1) Bipolar binary activation

$f(net) = sgn(net)$

step 1:

$net^1 = w_i^t \cdot x_1 = [1 \ -1] \cdot \begin{bmatrix} 1 \\ -2 \end{bmatrix} = 1 + 2 = 3$

$w_2^2 = w_1 + c\,sgn(net^1) \cdot x_1$

$= \begin{bmatrix} 1 \\ -1 \end{bmatrix} + 1 \times \begin{bmatrix} 1 \\ -2 \end{bmatrix} = \begin{bmatrix} 2 \\ -3 \end{bmatrix}$

step 2:

$net^2 = w_2^t \cdot x_2 = [2 \ -3] \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} = -3$

$w_3 = w_2 + c\,sgn(net^2) \cdot x_2$

$= \begin{bmatrix} 2 \\ -3 \end{bmatrix} + (-1) \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} =$

$= \begin{bmatrix} 2 \\ -3 \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ -4 \end{bmatrix}$

step 3:

$net_3 = w_3^T \cdot x_3 = \begin{bmatrix} 2 & -4 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = 4 - 12 = -8$

$w_4 = w_3 + c \cdot sgn(net_3) \cdot x_3$

$= \begin{bmatrix} 2 \\ -4 \end{bmatrix} + (-1) \times \begin{bmatrix} 2 \\ 3 \end{bmatrix}$

$= \begin{bmatrix} 2 \\ -4 \end{bmatrix} + \begin{bmatrix} -2 \\ -3 \end{bmatrix} = \begin{bmatrix} 0 \\ -7 \end{bmatrix}$

step 4:

$net_4 = w_4^T \cdot x_4 = \begin{bmatrix} 0 & -7 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \end{bmatrix} = 7$

$w_5 = w_4 + c \cdot sgn(net_4) \cdot x_4$

$= \begin{bmatrix} 0 \\ -7 \end{bmatrix} + 1 \times \begin{bmatrix} 0 \\ -1 \end{bmatrix}$

$= \begin{bmatrix} 0 \\ -7 \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ -8 \end{bmatrix}$

2) Bipolar continuous (i.e sigmoidal) activation function

$net_i = w_i^T \cdot x_i$

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

$\Delta w_i = c \cdot O_i \cdot X$

learning parameter = 1

step 1:

Set $x = x_1$

$net_1 = w_1^T \cdot x_1 = \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \end{bmatrix} = 3$

$w_2 = w_1 + c$

$o_1 = \frac{2}{1+e^{-3}} - 1 = \frac{2}{1.0497} - 1 = 0.9053$

$w_2 = w_1 + c \cdot o_1 \cdot x_1$

$= \begin{bmatrix} 1 \\ -1 \end{bmatrix} + 1 \times 0.9053 \times \begin{bmatrix} 1 \\ -2 \end{bmatrix}$

$= \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} 0.9053 \\ -1.810 \end{bmatrix} = \begin{bmatrix} 1.9053 \\ -2.81 \end{bmatrix}$

step 2:

$net_2 = w_2^T \cdot x_2 = \begin{bmatrix} 1.9053 & -2.81 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = -2.81$

$o_2 = \frac{2}{1+e^{+2.81}} - 1 = -0.886$

$w_2 = w_2 + c \cdot o_2 \cdot x_2$

$= \begin{bmatrix} 1.9053 \\ -2.81 \end{bmatrix} + 1 \times -0.886 \times \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$= \begin{bmatrix} 1.9053 \\ -2.81 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.886 \end{bmatrix}$

$= \begin{bmatrix} 1.905 \\ -3.696 \end{bmatrix}$

step 3:

$$net_3 = w_3^t x_3 = \begin{bmatrix} 1.905 & -3.696 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = -7.278$$

$$O_3 = \frac{2}{1+e^{7.278}} - 1 = -0.9986$$

$$w_4 = w_3 + \alpha \cdot O_3 \cdot x_3$$

$$= \begin{bmatrix} 1.905 \\ -3.696 \end{bmatrix} + 1 \times (-0.9986) \cdot \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

$$= \begin{bmatrix} 1.905 \\ -3.696 \end{bmatrix} + \begin{bmatrix} -1.9972 \\ -2.9958 \end{bmatrix}$$

$$= \begin{bmatrix} -0.092 \\ -6.691 \end{bmatrix}$$

step 4:

$$net_4 = w_4^t x_4 = \begin{bmatrix} -0.092 & -6.691 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

$$= 6.691$$

$$O_4 = \frac{2}{1+e^{-6.691}} - 1 = 0.9975$$

$$w_5 = w_4 + \alpha \, O_4 \cdot x_3$$

$$= \begin{bmatrix} -0.092 \\ -6.691 \end{bmatrix} + 1 \times (0.9975) \times \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} -0.092 \\ -5.6925 \end{bmatrix} = \begin{bmatrix} -0.092 \\ -7.68 \end{bmatrix}$$

* Use perceptron learning rule to ~~obtain~~ train the network. The set of input training vectors are as follows:

$$x_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} \quad x_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix} \quad x_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$$

and initial weight vector is $w_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$

Learning constant = 1.
Desired responses for x_1, x_2 and x_3 are $d_1 = -1$, $d_2 = -1$, $d_3 = 1$ respectively. Calculate the weights after one complete cycle.

Solution:—

For perceptron learning rule

$$net_i = w_i^t x$$
$$o_i = sgn(net_i)$$
$$\Delta w_i = c \cdot (d - o_i) \cdot x_i$$

Step 1:

First training pair:

$$x = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} \quad d = -1$$

Compute $net_i = w_i^t x$

$$= \begin{bmatrix} 1 & -1 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = 2.5$$

$$o_i = sgn(net_i) = 1$$

$$w_2 = w_1 + c(d_i - o_i) \cdot x_i$$

$$= \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + (0.1) \times (-1 - 1) \cdot \begin{bmatrix} 1 \\ -2 \\ 0 \\ +1 \end{bmatrix} =$$

$$= \begin{bmatrix} -0.2 \\ 0.4 \\ 0 \\ 0.2 \end{bmatrix} + \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix}$$

step2:

$$net_2 = w_2^T \cdot x_2 =$$

$$= [0.8 \quad -0.6 \quad 0 \quad 0.7] \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix} = -1.2$$

$$o_2 = sgn(net_2) = -1$$

not required

$$w_3 = w_2 + c(d_2 - o_2) \cdot x_2$$

$$= \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix} + (0.1) \times (-1 + 1) \times \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix}$$

weights are not updated because $d_2 = o_2$

step3:

$net_3 = W_3^t \cdot x_3$

$= [0.8 \quad -0.6 \quad 0 \quad \text{---}] \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix} = -2.1$

$O_3 = sgn(net_3) = -1$

$W_4 = W_3 + c \cdot (d_3 - O_3) \cdot x_3$

$= \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix} + (1) \times (1+1) \times \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$

$= \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.3 \end{bmatrix} + \begin{bmatrix} -0.2 \\ 0.2 \\ 0.1 \\ 0.2 \end{bmatrix}$

$= \begin{bmatrix} 0.6 \\ -0.4 \\ 0.1 \\ 0.5 \end{bmatrix}$

Consider the following set of input vectors

$$x_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} \quad x_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix} \quad x_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$$

$d_1 = -1$ $d_2 = -1$, $d_3 = 1$ are the desired responses after x_1, x_2, x_3 respectively.

Initial weight vector $w_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$

randomly chosen learning constant $c = 0.1$

$\lambda = 1$ for fnet.

Use delta learning rule to calculate final weights. Use bipolar continuous activation function.

$$net_1 = w_1^t x_1$$

$$= \begin{bmatrix} 1 & -1 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = 2.5$$

$$o^1 = f(net^1) = \frac{2}{1 + \exp^{-1 \times 2.5}} - 1, = 0.848$$

$$f'(net^1) = \frac{1}{2}[1 - (o^1)^2] =$$

$$= \frac{1}{2}[1 - 0.848^2]$$

$$= 0.140$$

$$w^2 = c \cdot (d_1 - o^1) f'(net^1) x_1 + w^1$$

$$= (0.1) \times (-1 - 0.848) \times 0.140 \times \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} + \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

$$= \begin{bmatrix} 0.974 \\ -0.948 \\ 0 \\ 0.526 \end{bmatrix}$$

Step 2:

Input is vector x_2, weight vector is w^2

$$x_2 \cdot net^2 = w_2^{2t} x_2$$

$$= \begin{bmatrix} 0.974 & -0.948 & 0 & 0.526 \end{bmatrix} \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix}$$

$$= -1.948$$

$$o^2 = f(net^2) = \frac{2}{1+\exp^{-\lambda net^2}} - 1 = -0.075$$

$$f'(net^2) = \frac{1}{2}[1-(o^2)^2]$$

$$= \frac{1}{2}[1-(-0.075)^2]$$

$$= 0.218$$

$$w^3 = c\cdot(d_2 - o^2)\cdot f'(net^2)\times x_2 + w^2$$

$$(0.1)\times(-1+0.75)\times 0.218 \times \begin{bmatrix}0\\1.5\\-0.5\\-1\end{bmatrix} + \begin{bmatrix}0.974\\-0.948\\0\\0.526\end{bmatrix}$$

$$= \begin{bmatrix}0.974\\-0.956\\0.002\\0.531\end{bmatrix}$$

Step 3:

Input is x_3, weight vector is w^3

$$net^3 = w^{3t}\cdot x_3$$

$$= [0.974, -0.956, 0.002, 0.531] \begin{bmatrix}-1\\1\\0.5\\-1\end{bmatrix}$$

$$= -2.46$$

$$o^3 = f(net^3) = \frac{2}{1+\exp^{-\lambda net^3}} - 1 = -0.842$$

$$f'(net^3) = \frac{1}{2}\left[1 - (o^3)^2\right]$$

$$= \frac{1}{2}\left[1 - (-0.842)^2\right]$$

$$= 0.145$$

$$f'(net^3) =$$

$$w^4 = c\,(d_3 - o^3) \cdot f'(net^3) \cdot x_3 + w^3 \cdot$$

$$= (0.1) \times (1 - (-0.842)) \times 0.145 \times \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix} + \begin{bmatrix} 0.974 \\ -0.956 \\ 0.002 \\ 0.531 \end{bmatrix}$$

$$= \begin{bmatrix} 0.947 \\ -0.929 \\ 0.016 \\ 0.505 \end{bmatrix}$$

Perform two training steps of the network using data learning rule for $\lambda = 1$, $c = 0.25$. Train the network using the following data pairs.

$$\left(x_1 = \begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix} \; d_1 = -1\right) \quad \left(x_2 = \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix} \; d_2 = +1\right)$$

The initial weights are $w = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

Solution :=

Step 1 : $net^1 = w_1^t \cdot x$

$$= \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix}$$

$$= 1$$

$$o^1 = \frac{2}{1+\exp^{-\lambda \cdot net}} - 1 = \frac{2}{1+\exp^{-1}} - 1 = 0.463$$

$$f'(net^1) = \frac{1}{2}\left[1 - o^2\right] = \frac{1}{2}\left[1 - 0.463^2\right] = 0.393$$

$$w_2 = w_1 + \Delta w_1 = w_1 + c \cdot (d_1 - o^1) f'(net^1) \cdot x$$

$$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + (0.25)(-1 - 0.463)(0.393) \cdot \begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -0.287 \\ 0 \\ 0.1437 \end{bmatrix} = \begin{bmatrix} 0.713 \\ 0 \\ 1.1437 \end{bmatrix}$$

step2:

$$net^2 = w_2 \pm x$$

$$= \begin{bmatrix} 0.713 & 0 & 0.1437 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} -0.4307 \end{bmatrix}$$

$$o^2 = f(net^2) = \frac{2}{1+\exp^{-\lambda net}} - 1 = \frac{2}{1+\exp^{0.4307}} - 1$$

$$= -0.2119$$

$$f'(net^2) = \frac{1}{2}\left[1 - o^2\right]$$

$$= \frac{1}{2}\left[1 - (-0.2119)^2\right]$$

$$= 0.477$$

$$\Delta w_2 = c \cdot (d_2 - o^2) \cdot f'(net^2) \cdot x_2$$

$$= (0.25) \cdot (1 - (-0.2119)) \cdot 0.477 \cdot \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.145 \\ -0.29 \\ -0.145 \end{bmatrix}$$

$$w_3 = w_2 + \Delta w_2$$

$$= \begin{bmatrix} 0.713 \\ 0 \\ 1.1437 \end{bmatrix} + \begin{bmatrix} 0.145 \\ -0.29 \\ -0.145 \end{bmatrix} = \begin{bmatrix} 0.858 \\ -0.29 \\ 0.998 \end{bmatrix}$$

1 Given below are three input vectors and three initial weight vectors for a three neuron competitive neurons. Calculate the resulting weights from training the competitive layer with winner take all algorithm. Consider $\alpha = 0.5$. The applied input pattern is $P_1\ P_2\ P_3\ P_1\ P_2\ P_3$.

Solution:-

$$P_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \quad P_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad P_3 = \begin{bmatrix} 0.709 \\ 0.709 \end{bmatrix}$$

$$w_1 = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w_2 = \begin{bmatrix} -0.894 \\ 0.447 \end{bmatrix} \quad w_3 = \begin{bmatrix} -0.447 \\ 0.894 \end{bmatrix}$$

Here weight matrix $w = \begin{matrix} w_1 & w_2 & w_3 \end{matrix}$

$$\begin{bmatrix} 0 & -0.894 & -0.447 \\ -1 & 0.447 & 0.894 \end{bmatrix}$$

step 1:

$$net^1 = w^t \cdot P_1$$

$$= \begin{bmatrix} 0 & -1 \\ -0.894 & 0.447 \\ -0.447 & 0.894 \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0.894 \\ 0.447 \end{bmatrix} \rightarrow \text{Max}$$

Since the second neuron's output 0.894 is maximum, second neuron wins. So update only w_2

$$w_{2new} = w_{2old} + \alpha (p_1 - w_{2old})$$

$$= \begin{bmatrix} -0.894 \\ 0.447 \end{bmatrix} + (0.5)\left(\begin{bmatrix} -1 \\ 0 \end{bmatrix} - \begin{bmatrix} -0.894 \\ 0.447 \end{bmatrix} \right)$$

$$= \begin{bmatrix} -0.894 \\ 0.447 \end{bmatrix} + 0.5 \begin{bmatrix} -0.106 \\ 0.447 \end{bmatrix}$$

$$= \begin{bmatrix} -0.947 \\ 0.223 \end{bmatrix}$$

So after the first input pattern is presented new weight vector is

$$W = \begin{bmatrix} 0 & -0.947 & -0.447 \\ -1 & 0.223 & 0.894 \end{bmatrix}$$

Step 2:

$$net^2 = w^t \cdot p_2$$

$$= \begin{bmatrix} 0 & -1 \\ -0.947 & 0.223 \\ -0.447 & 0.894 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 \\ 0.223 \\ 0.894 \end{bmatrix} \rightarrow max$$

Since the third neuron's output 0.894 is the maximum, third neuron wins. Hence update only third neuron's weight.

$$w_{3new} = w_{3old} + \alpha(P_2 - w_{3old})$$

$$= \begin{bmatrix} -0.447 \\ 0.894 \end{bmatrix} + (0.5)\left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} -0.447 \\ 0.894 \end{bmatrix}\right)$$

$$= \begin{bmatrix} -0.223 \\ 0.947 \end{bmatrix}$$

Hence after pattern P_2 has been presented. The updated weight matrix W is

$$W = \begin{bmatrix} 0 & -0.947 & -0.223 \\ -1 & 0.223 & 0.947 \end{bmatrix}$$

Step 3:

$$net^3 = W^t P_3$$

$$= \begin{bmatrix} 0 & -1 \\ -0.947 & 0.223 \\ -0.223 & 0.947 \end{bmatrix} \begin{bmatrix} 0.709 \\ 0.709 \end{bmatrix}$$

$$= \begin{bmatrix} -0.709 \\ -0.513 \\ 0.512 \end{bmatrix} \rightarrow \text{max}$$

Hence winner is third neuron.

So update w_3

$$w_{3new} = w_{3old} + \alpha(P_3 - w_{3old})$$

$$= \begin{bmatrix} -0.223 \\ 0.947 \end{bmatrix} + (0.5)\left(\begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix} - \begin{bmatrix} -0.223 \\ 0.947 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 0.242 \\ 0.827 \end{bmatrix}$$

$$W = \begin{bmatrix} 0 & -0.947 & 0.242 \\ -1 & 0.223 & 0.827 \end{bmatrix}$$

step 4:

$$net = w^t \cdot p_1$$

$$= \begin{bmatrix} 0 & -1 \\ -0.947 & 0.223 \\ 0.242 & 0.827 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0.947 \\ 0.242 \end{bmatrix} \rightarrow max$$

$$w_{2new} = w_{2old} + \alpha (p_1 - w_{2old})$$

$$= \begin{bmatrix} -0.947 \\ 0.223 \end{bmatrix} + (0.5) \left(\begin{bmatrix} -1 \\ 0 \end{bmatrix} - \begin{bmatrix} -0.947 \\ 0.223 \end{bmatrix} \right)$$

$$= \begin{bmatrix} -0.9735 \\ 0.112 \end{bmatrix}$$

$$\therefore w = \begin{bmatrix} 0 & -0.9735 & 0.242 \\ -1 & 0.112 & 0.827 \end{bmatrix}$$

step 5:

$$net = w^t p_2$$

$$\begin{bmatrix} 0 & -1 \\ -0.9735 & 0.112 \\ 0.242 & 0.827 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 \\ 0.112 \\ 0.827 \end{bmatrix} \rightarrow max$$

Update w_3

$$W_{3new} = W_{3old} + \alpha(P_2 - W_{3old})$$

$$= \begin{bmatrix} 2.415 \\ 0.827 \end{bmatrix} + 0.5\left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.2415 \\ 0.827 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 0.1207 \\ 0.9135 \end{bmatrix}$$

$$\therefore W = \begin{bmatrix} 0 & -0.9785 & 0.1207 \\ -1 & 0.112 & 0.9135 \end{bmatrix}$$

Step 6:

$$net_6 = W^t \cdot P_3$$

$$\begin{bmatrix} 0 & -1 \\ -0.9785 & 0.1207 \\ 0.112 & 0.9135 \end{bmatrix} \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix}$$

$$= \begin{bmatrix} -0.707 \\ -0.609 \\ 0.731 \end{bmatrix} \rightarrow \text{max}$$

Perform two training steps using widrow-hoff learning rule.

Solution :=

$$x_1 = \begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix} \quad d_1 = -1 \qquad x_2 = \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix} \quad d_2 = 1$$

$$w = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \qquad \lambda = 1, \quad c = 0.25$$

Solution :=

$$o_j = w_i^t x$$

$$\Delta w_i = c(d_i - o_i) \cdot x$$

step 1:

$$o^1 = w_i^t x_1$$

$$= \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix} = 1$$

$$\Delta w_1 = c(d_1 - o^1) \cdot x_1$$

$$= (0.25)(-1 - 1) \begin{bmatrix} 2 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0.5 \end{bmatrix}$$

$$w_2 = w_1 + \Delta w_1$$

$$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 \\ 0 \\ 0.5 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0 \\ 1.5 \end{bmatrix}$$

step 2:

$$O_2 = w_2^t \times 2$$

$$= \begin{bmatrix} 0 & 0 & 1.5 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix} = -1.5$$

$$\Delta w_2 = c(d_2 - O_2) \times 2$$

$$= (0.25) \cdot (2.5) \cdot \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix}$$

$$= 0.625 \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix}$$

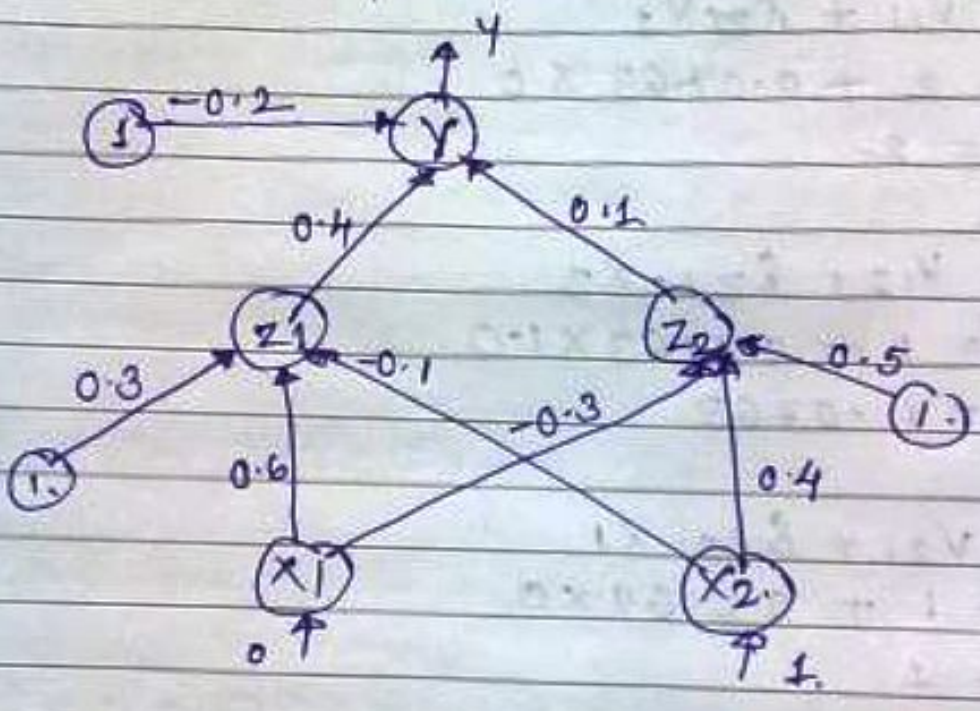
$$= \begin{bmatrix} 0.625 \\ -1.25 \\ -0.625 \end{bmatrix}$$

$$\Delta w_3 = w_2 + \Delta w_2$$

$$= \begin{bmatrix} 0 \\ 0 \\ 1.5 \end{bmatrix} + \begin{bmatrix} 0.625 \\ -1.25 \\ -0.625 \end{bmatrix}$$

$$= \begin{bmatrix} 0.625 \\ -1.25 \\ 0.875 \end{bmatrix}$$

* Using back propagation network, find the new weights for the net shown in figure. It is presented with the input pattern [0, 1] and the target output is 1. Use learning rate $\alpha = 0.25$ and binary sigmoidal activation function.



Solution :-

$[V_{11}, V_{12}, V_{01}] = [0.6, -0.1, 0.3]$

$[V_{21}, V_{22}, V_{02}] = [-0.3, 0.4, 0.5]$

$[W_1, W_2, W_0] = [0.4, 0.1, 0.2]$

$\alpha = 0.25$, $t = 1$, $\lambda = 1$

Activation function = binary sigmoidal.

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

Calculate the input net for z_1

$$Z_{in1} = x_1 \times V_{11} + x_2 \times V_{12} + 0.3$$
$$= 0 \times 0.6 + 1 \times -0.1 + 0.3$$
$$= 0.2$$

$$Z_{in2} = x_1 \times V_{21} + x_2 \times V_{22} + 0.5$$
$$= 0 \times -0.3 + 1 \times 0.4 + 0.5$$
$$= 0.9$$

$$z_1 = f(Z_{in1}) = \frac{1}{1+e^{-0.2}} = 0.5498$$

$$z_2 = f(Z_{in2}) = \frac{1}{1+e^{-0.9}} = 0.7109$$

$$y_{in} = z_1 \times w_1 + z_2 \times w_2 + w_0$$
$$= 0.5498 \times 0.4 + 0.7109 \times 0.1 + (-0.2)$$
$$= 0.09101$$

$$y = f(y_{in}) = \frac{1}{1+e^{-0.09101}} = 0.5227$$

Compute error term

$$\delta_K = (t_K - y_K) \cdot f'(y_{ink})$$
$$f'(y_{ink}) = f(y_{in})[1 - f(y_{in})]$$
$$= 0.5227 \cdot [1 - 0.5227]$$
$$= 0.2495$$
$$\delta_0 = (1 - 0.5227) \times 0.2495$$
$$= 0.1191$$

Find the changes in weights between hidden and output layers.

$W_1 new = W_1 old + \eta \delta_0 z_1$

$= 0.4 + 0.25 \times 0.1191 \times 0.5498$

$= 0.4 + 0.0164$

$= 0.4164$

$W_2 new = W_2 old + \eta \delta_0 z_2$

$= 0.1 + 0.25 \times 0.1191 \times 0.7109$

$= 0.12117$

Compute error signal terms of hidden layer

$\delta y_j = y_j(1 - y_j) \cdot \sum_{k=1}^{K} \delta_{0k} \cdot w_{kj}$

Two hidden nodes in hidden layer, z_1, z_2

$\delta z_1 = [0.5498 \times (1 - 0.5498)] \times 0.1191 \times 0.4$

$= 0.2475 \times 0.1191 \times 0.4$

$= 0.2475 \times 0.04764$

$= 0.0118$

$\delta z_2 = [0.7109 \times (1 - 0.7109)] \times 0.1191 \times 0.1$

$= 0.2055 \times 0.1191 \times 0.1$

$= 0.00245$

Adjust weights of hidden layer

$V \leftarrow V + \eta \delta y \cdot z$

$v_{11} = v_{11} + \eta \cdot \delta z_1 \cdot x_1$

$= 0.6 + 0.25 \times 0.0118 \times 0$

$= 0.6$

$v_{12} = v_{12} + \eta \times \delta z_1 \times x_2$

$= -0.1 + 0.25 \times 0.0118 \times 1$

$= -0.1 + 0.00295$

$= -0.09705$

$v_{21} = v_{21} + \eta \times \delta z_2 \times x_1$

$= -0.3 + 0.25 \times 0.00245 \times 0$

$= -0.3$

$v_{22} = v_{22} + \eta \times \delta z_2 \times x_2$

$= 0.4 + 0.25 \times 0.00245 \times 1$

$= 0.400\,6125$

Update bias

$v_{01} = v_{01} + \eta \cdot \delta z_1$

$= 0.3 + 0.25 \times 0.0118$

$= 0.3 + 0.00295$

$= 0.30295$

$v_{02} = v_{02} + \eta \cdot \delta z_2$

$= 0.5 + 0.25 \times 0.00245$

$= 0.500\,6125$

$w_0 = w_0 + \eta \cdot \delta o$

$= -0.2 + 0.25 \times 0.1191$

$= -0.17022$

Unipolar continuous

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

$$\delta o_k = (d_k - o_k)(1-o_k)o_k \quad \text{for } k = 1 \cdots K$$

$$\delta y_j = y_j(1-y_j) \cdot \sum_{k=1}^{K} \delta o_k w_{kj} \quad \text{for } j = 1 \cdots J$$

Bipolar continuous

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

$$\delta o_k = \frac{1}{2}(d_k - o_k)(1-o_k^2) \quad \text{for } k = 1 \cdots K$$

$$\delta y_j = \frac{1}{2}(1-y_j^2) \sum_{k=1}^{K} \delta o_k \cdot w_{kj} \quad \text{for } j = 1 \cdots j$$

Using the madaline network, emplement XOR function with bipolar inputs and targets. Assume the required parameters

Solution :-

x_1	x_2	t
1	1	-1
1	-1	1
-1	1	1
-1	-1	-1

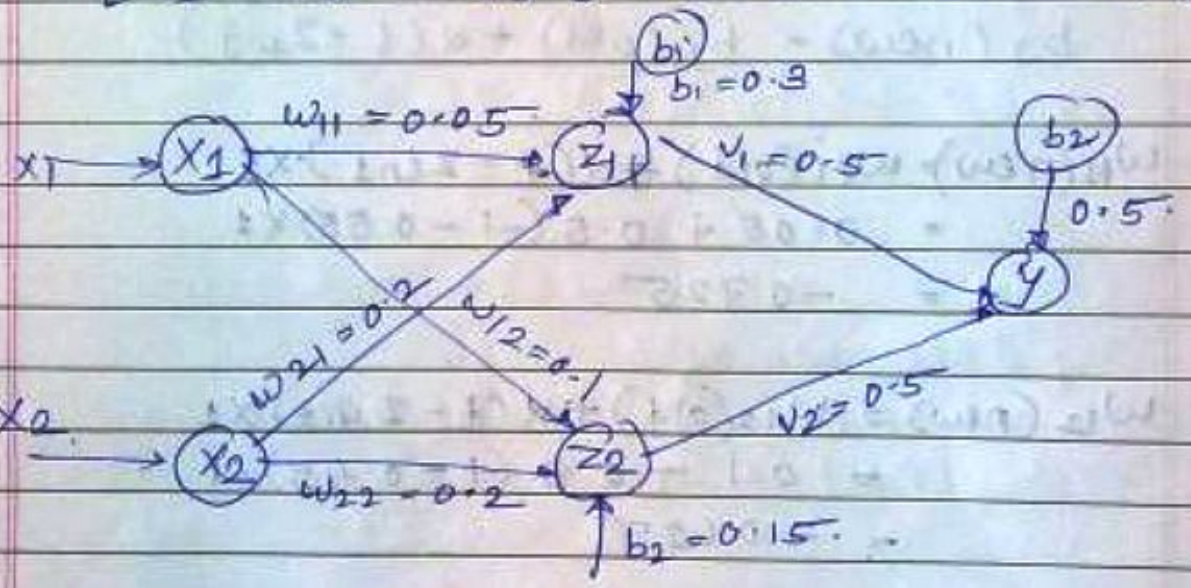
$\alpha = 0.5$

Initial weights and bias ~~$\alpha = 0.5$~~

$[w_{11} \quad w_{21} \quad b_1] = [0.05 \quad 0.2 \quad 0.3]$

$[w_{12} \quad w_{22} \quad b_2] = [0.1 \quad 0.2 \quad 0.15]$

$[v_1 \quad v_2 \quad b_3] = [0.5 \quad 0.5 \quad 0.5]$



Step 1 :-

$x_1 = 1, \quad x_2 = 1 \quad t = -1 \quad \alpha = 0.5$

$Z_{in1} = b_1 + w_{11} * x_1 + w_{21} \cdot x_2$

$= 0.3 + 0.05 \times 1 + 0.2 \times 1$

$= 0.55$

$Z_{in2} = b_2 + w_{12} x_1 + w_{22} \cdot x_2$

$= 0.15 + 0.1 \times 1 + 0.2 \times 1$

$= 0.45$

Apply activation function

$z_1 = f(z_{in1}) = f(0.55) = 1$

$z_2 = f(z_{in2}) = f(0.45) = 1$

$y_{in} = b_3 + v_1 z_1 + v_2 z_2$

$= 0.5 + 0.5 \times 1 + 0.5 \times 1$

$= 1.5$

$y = f(y_{in})$

$= f(1.5)$

$= 1.$

Since $y = 1$, it is not equal to $(t = -1)$

∴ update weights and bias

$w_{ij}(new) = w_{ij}(old) + \alpha(t - z_{inj}) \cdot x_i$

$b_j(new) = b_j(old) + \alpha(t - z_{inj})$

$w_{11}(new) = w_{11}(old) + \alpha(t - z_{in1}) \cdot x_1$

$= 0.05 + 0.5(-1 - 0.55) \times 1$

$= -0.725$

$w_{12}(new) = w_{12}(old) + \alpha(t - z_{in2}) x_1$

$= 0.1 + 0.5(-1 - 0.45) \times 1$

$= -0.625$

$b_1(new) = b_1(old) + \alpha(t - z_{in1})$

$= 0.3 + 0.5(-1 - 0.55)$

$= -0.475$

$w_{21}(new) = w_{21}(old) + \alpha(t - z_{in1}) \cdot x_2$

$= 0.2 + 0.5(-1 - 0.55) \times 1$

$= -0.575$

$$w_{22}(new) = w_{22}(old) + \alpha(t - z_{in2}) \cdot x_2$$
$$= 0.2 + 0.5 \times (-1 - 0.45) \times 1$$
$$= -0.525$$

$$b_2(new) = b_2(old) + \alpha(t - z_{in2})$$
$$= 0.15 + 0.5(-1 - 0.45)$$
$$= -0.575$$

So after 1 epoch weights are

$$\begin{bmatrix} w_{11} & w_{21} & b_1 \\ w_{12} & w_{22} & b_2 \end{bmatrix} = \begin{bmatrix} -0.725 & -0.575 & -0.475 \\ -0.625 & -0.525 & -0.575 \end{bmatrix}$$

* Use adaline network to train ANDNOT function with bipolar inputs and targets perform 2 epochs of training

Solution :=

x_1	x_2	t
1	1	-1
1	-1	1
-1	1	-1
-1	-1	-1

Let $w_1, w_2, b = 0.2$, $\alpha = 0.2$

Epoch 1.

Input (1, 1), target = -1

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

$= 0.2 + 0.2 \times 1 + 0.2 \times 1$

$= 0.6$

$(t - y_{in}) = (-1 - 0.6) = -1.6$

update weights

$w_i(new) = w_i(old) + \alpha (t - y_{in}) x_1$

$w_1(new) = w_1(old) + \alpha (t - y_{in}) x_1$

$= 0.2 + 0.2 (-1 - 0.6) \times 1$

$= -0.12$

$\Delta w_1 = -0.32$

$w_2(new) = w_2(old) + \alpha \cdot (t - y_{in}) \cdot x_2$

$= 0.2 + 0.2 (-1 - 0.6) \times 1$

$= -0.12$

$\Delta w_2 = -0.32$

$b(new) = b(old) + \alpha (t - y_{in})$

$= 0.2 + 0.2 \times (-1 - 0.6)$

$= -0.12$

$\Delta b_2 = -0.32$

Compute error

$E = (t - y_{in})^2 = (-1.6)^2 = 2.56$

Input $(1, -1)$, target $= 1$

$y_{in_2} = b + w_1 x_1 + w_2 x_2$

$= -0.12 + (-0.12) \times 1 + (-0.12) \times (-1)$

$= -0.12$

$(t - y) = (1 + 0.12) = (1 + 0.12) = 1.12$

Update weights

$w_1(new) = w_1(old) + \alpha(t - y_{in}) x_1$

$= -0.12 + 0.2(1 - (-0.12) \times 1$

$= -0.12 + 0.2 \times 1.12 \times 1$

$= 0.10$ $\therefore \Delta w_1 = 0.22$

$w_2(new) = w_2(old) + \alpha(t - y_{in}) x_2$

$= -0.12 + 0.2(1 - (-0.12) \times (-1)$

$= -0.12 + 0.2 \times 1.12 \times -1$

$= -0.34$ $\therefore \Delta w_2 = -0.22$

$b(new) = b(old) + \alpha(t - y_{in})$

$= -0.12 + 0.2 \times 1.12$

$= 0.10$ $\Delta b_1 = 0.22$

$E = (t - y_{in})^2 = (1.12)^2 = 1.25$

Input $(-1, 1)$, target $= -1$

$y_{in3} = b + w_1 x_1 + w_2 x_2$

$= 0.10 + 0.10 \times -1 + (-0.34) \times 1$

$= -0.34$

$(t - y_{in}) = (-1 - (-0.34))$

$= -1 + 0.34$

$= -0.66$

update weights.

$w_1(new) = w_1(old) + \alpha(t - y_{in})x_1$

$= 0.10 + 0.2(-1 - (-0.34)) \times -1$

$= 0.10 + 0.2 \times -0.66 \times -1$

$= 0.24$

$\Delta w_1 = 0.13$

$w_2(new) = w_2(old) + \alpha(t - y_{in})x_2$

$= -0.34 + 0.2(-1 - (-0.34)) \times 1$

$= -0.34 + 0.2 \times -0.66 \times 1$

$= -0.48$

$\Delta w_2 = -0.13$

$b(new) = b(old) + \alpha(t - y_{in})$

$= 0.10 + 0.2(-1 - (-0.34))$

$= 0.10 + 0.2 \times -0.66$

$= -0.08$

$\Delta b = -0.18$

$EE\ (t - y_{in})^2 = (-0.66)^2 = 0.43$

Input $(-1, -1)$, target $= -1$

$y_{in} = b + w_1x_1 + w_2x_2$

$= -0.13 + 0.24 \times -1 + -0.48 \times -1$

$= -0.13 - 0.24 + 0.48$

$= 0.21$

$(t - y_{in}) = (-1 - 0.21) = -1.2$

update weights

$w_1(new) = w_1(old) + \alpha(t - y_{in})x_1$

$= 0.24 + 0.2(-1 - 0.21)x - 1$

$= 0.24 + 0.2x - 1.2x - 1$

$= 0.48$

$w_2(new) = w_2(old) + \alpha(t - y_{in})x_2$

$= -0.48 + 0.2x - 1.2x - 1$

$= -0.24$

$b(new) = b(old) + \alpha(t - y_{in})$

$= -0.13 + 0.2x - 1.2$

$= -0.27$

$Error = (t - y_{in})^2 = (-1.2)^2 = 1.47$

Epoch 1 completed.

Total mean square error

Epoch 1 = $2.56 + 1.25 + 0.43 + 1.47 = 5.71$

Epoch 2 = $0.15 + 0.57 + 0.106 + 0.8 = 2.426$

EBPTA

Apply back propagation algorithm to find the final weights for the following net

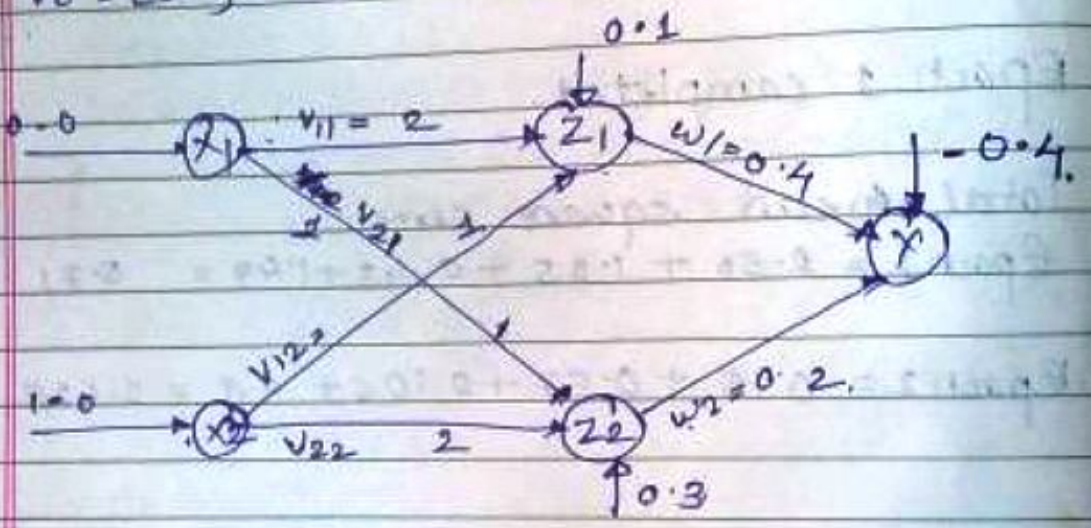
Inputs: $x = [0.0, 1.0]$

weights between hidden and output layers: $w = [0.4, 0.2]$

Bias on the output node is: $W_0 = [-0.4]$

weights between input and hidden layer $V = [2, 1, 1, 2]$

Bias on hidden nodes are $V_0 = [0.1, 0.3]$. Desired output: $d = 1.0$



Solution:- forward pass [step 2]

$$Z_{1net} = V_{11}X_1 + V_{12}X_2 + V_{01}$$
$$= 2 \times 0 + 1 \times 0.0 + 0.1$$
$$= 0 + 1 + 0.1$$
$$= 1.1$$

$$Z_1 = f(Z_{1net}) = \frac{2}{1 + e^{-\lambda net}} - 1 \quad \text{[bipolar continuous]}$$
$$= \frac{2}{1 + e^{-1.1}} - 1 = 0.5005$$

$z_{2net} = V_{21} \times X_1 + V_{22} \cdot X_2 + V_{02}$

$= 1 \times 0.0 + 2 \times 1.0 + 0.3$

$= 2.3$

$z_2 = f(z_{2net}) = \frac{2}{1 + e^{-\lambda net}} - 1$

$= \frac{2}{1 + e^{-2.3}} - 1$

$= 0.8178$

$y_{net} = W_1 z_1 + W_2 z_2 + W_{01}$

$= 0.4 \times 0.5005 + 0.2 \times 0.8178 + (-0.4)$

$= -0.03624$

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

$= \frac{2}{1 + e^{0.03624}} - 1$

$= -0.0181$

Step 3:

$E = E + \frac{1}{2}(d - y)^2$

$= \frac{1}{2}(1 - (-0.0181)^2)$

$= 0.5183$

Backward pass: (step 4)

Calculate errors δo, δy

$\delta o = \frac{1}{2}[(d_K - o_K) \cdot (1 - o_K^2)]$ for $K = 1 \cdots 10$

$\delta y = \frac{1}{2}(d - y)(1 - y^2)$

$= \frac{1}{2}(1 - (-0.0181))(1 - (-0.0181)^2)$

$= 0.5089$

Computing error signal terms of the hidden layer.

$$\delta_y = w_1^b \delta_0 \cdot f'_y$$

$$f'_y = \frac{1}{2}[1 - y_i^2]$$

$$\therefore \delta_{z1} = \frac{1}{2}[1 - z_1]^2 \cdot \delta_y \cdot w_1$$

$$= \frac{1}{2}(1 - 0.5005^2) \cdot (0.5089) \cdot (0.4)$$

$$= 0.0763$$

$$\delta_{z2} = \frac{1}{2}[1 - z_2]^2 \cdot \delta_y \cdot w_2$$

$$= \frac{1}{2}(1 - 0.8178)^2 (0.5089) \times 0.2$$

$$= 0.0169$$

Adjust weights of the output layer (w_1, w_2) (step 5)

$w \leftarrow w + \eta \delta_0 \cdot y'$ (assume $\eta = 1$)

$$\therefore w_{1new} = w_{1old} + \delta_y \cdot z_1$$

$$= 0.4 + 0.5089 \times 0.5005$$

$$= 0.6547$$

$$w_{2new} = w_{2old} + \delta_y \cdot z_2$$

$$= 0.2 + 0.5086 \times 0.8178$$

$$= 0.6162$$

$w_{01} = w_{01} \cdot \delta y(1)$

$= -0.4 + 0.5089$

$w_{01} = 0.1089$

Adjust weights of hidden layer (V).

(Step 6)

$V \leftarrow V + \eta \delta y \cdot z$

$V_{11} = V_{11} + \delta z_1 x_1$

$= 2 + 0.0763 \times 0$

$= 2$

$V_{12} = V_{12} + \delta z_1 \cdot x_2$

$= 1 + 0.0763 \times 1.0$

$= 1.0763$ 1.025

$V_{21} = V_{21} + \delta z_2 \cdot x_1$

$= 1 + 0.0169 \times 0$

$= 1$

$V_{22} = V_{22} + \delta z_2 \cdot x_2$

$= 2 + 0.0169 \times 1$

$= 2.0169$ 2.0016

$V_{01} = V_{01} + \delta z_1(1)$

$= 0.1 + 0.0763 \times 1$

$= 0.1763$ 0.125

$V_{02} = V_{02} + \delta z_2(1)$

$= 0.3 + 0.0169 \times 1$

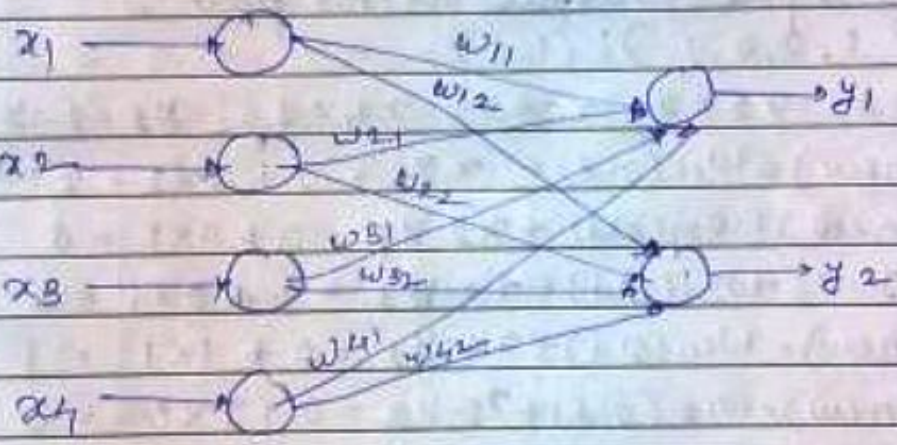
$= 0.3169$ 0.3016

Module 3

Train a heteroassociative memory network using Hebb rule to store input row vector. $s = (s_1, s_2, s_3, s_4, 65)$ to the output row vector $t = (t_1, t_2)$. The vector pairs are given in table.

	Inputs				Output	
	s_1	s_2	s_3	s_4	t_1	t_2
1	1	0	1	0	1	0
2	1	0	0	1	1	0
3	1	1	0	0	0	1
4	0	0	1	1	0	1

Solution: =



Weights are initialized to zero

For first pair (1, 0, 1, 0) : (1, 0)

Set the activation of the input units

$x_1 = 1, \quad x_2 = 0, \quad x_3 = 1, \quad x_4 = 0.$

Set the activation of the output unit

$y_1 = 1, \quad y_2 = 0$

Update the weights.

$w_{ij}(new) = w_{ij}(old) + x_i y_j$

$w_{11}(new) = w_{11}(old) + x_1 y_1$

$= 0 + 1 \times 1 = 1$

$w_{21}(new) = w_{21}(old) + x_2 y_1 = 0 + 0 \times 1 = 0$

$w_{31}(new) = w_{31}(old) + x_3 \cdot y_1 = 0 + 1 \times 1 = 1$

$w_{41}(new) = w_{41}(old) + x_4 \cdot y_1 = 0 + 0 \times 1 = 0$

$w_{12}(new) = w_{12}(old) + x_1 y_2 = 0 + 1 \times 0 = 0$

$w_{22}(new) = w_{22}(old) + x_2 y_2 = 0 + 0 \times 0 = 0$

$w_{32}(new) = w_{32}(old) + x_3 \cdot y_2 = 0 + 1 \times 0 = 0$

$w_{42}(new) = w_{42}(old) + x_4 \cdot y_2 = 0 + 0 \times 0 = 0$

for 2nd input layer.

The input-output vector pair is

(1, 0, 0, 1) : (1, 0)

$x_1 = 1, \quad x_2 = 0, \quad x_3 = 0, \quad x_4 = 1, \quad y_1 = 1, \quad y_2 = 0$

$w_{11}(new) = w_{11}(old) + x_1 y_1 = 1 + 1 \times 1 = 2$

$w_{21}(new) = w_{21}(old) + x_2 \cdot y_1 = 0 + 0 \times 1 = 0$

$w_{31}(new) = w_{31}(old) + x_3 \cdot y_1 = 1 + 0 \times 1 = 1$

$w_{41}(new) = w_{41}(old) + x_4 \cdot y_1 = 0 + 1 \times 1 = 1$

$w_{12}(new) = w_{12}(old) + x_1 \cdot y_2 = 0 + 1 \times 0 = 0$

$w_{22}(new) = w_{22}(old) + x_2 y_2 = 0 + 0 \times 0 = 0$

$w_{32}(new) = 0$

$w_{42}(new) = 0$

for 3rd input layer.

The input-output vector pair is

(1, 1, 0, 0) : (0, 1)

$x_1 = 1, \quad x_2 = 1, \quad x_3 = 0, \quad x_4 = 0, \quad y_1 = 0, \quad y_2 = 1$

Since $y_1 = 0$, the weights of y_1 are going to be the same $\therefore w_{11} = 2, \; w_{21} = 0, \; w_{31} = 1, \; w_{41} = 1$

Computing weights of y_2

$w_{12} = w_{12} + x_1 \times y_2 = 0 + 1 \times 1 = 1$

$w_{22} = w_{22} + x_2 \times y_2 = 0 + 1 \times 1 = 1$

$w_{32} = w_{32} + x_3 \times y_3 = 0 + 0 \times 1 = 0$

$w_{42} = w_{42} + x_4 \times y_4 = 0 + 0 \times 1 = 0$

For the ~~4~~ 4th input layer

the input-output pair is

$(0, 0, 1, 1) : (0, 1)$

$x_1 = 0, \quad x_2 = 0, \quad x_3 = 1, \quad x_4 = 1, \quad y_1 = 0, \quad y_2 = 1$

Since $y_1 = 0$, the weights of y_1 are going to be the same.

$w_{11} = 2, \quad w_{21} = 0, \quad w_{31} = 1, \quad w_{41} = 1$

Computing weights of y_2

$w_{12} = w_{12} + x_1 \times y_2 = 1 + 0 \times 1 = 1$

$w_{22} = w_{22} + x_2 \times y_2 = 1 + 0 \times 1 = 1$

$w_{32} = w_{32} + x_3 \times y_2 = 0 + 1 \times 1 = 1$

$w_{42} = w_{42} + x_4 \times y_2 = 0 + 1 \times 1 = 1$

∴ The final weight matrix is

$$W = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \\ w_{41} & w_{42} \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Train the heteroassociative memory network using outer products rule to store input row vectors $s = \{s_1, s_2, s_3, s_4\}$ to the output row vectors $t = \{t_1, t_2\}$. Use the vector pairs as shown below

	s_1	s_2	s_3	s_4	t_1	t_2
1	1	0	1	0	1	0
2	1	0	0	1	1	0
3	1	1	0	0	0	1
4	0	0	1	1	0	1

Solution :=

Use outer products rule to determine the weight matrix

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

First pair $s = [1, 0, 1, 0]$ $t = [1, 0]$. $p = 1$

$$s^T(1)\, t(1) = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$$

For second pair, $s = [1, 0, 0, 1]$ $t = [1, 0]$ $p = 2$

$$s^T(2)\, t(2) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}$$

For 3rd pair, $s = [1, 1, 0, 0]$ $t = [0, 1]$ $p = 3$

$$s^T(3) \cdot t(3) = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \cdot [0 \;\; 1] = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

For 4th pair. $s = [0, 0, 1, 1]$ $t = [0, 1]$ $p = 4$

$$s^T(4) \cdot t(4) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \cdot [0, 1] = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

The final weight matrix is the summation of all the individual weight matrices obtained for each pair.

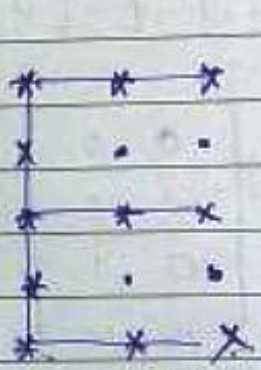
$$w(i) = \sum_{p=1}^{P} s^T(p) \cdot t(p)$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Bidirectional associative memory (BAM)

* Construct and test a BAM network to associate letters E & F with simple bipolar input- output vectors. Target output for E is (-1, 1) and for F is (1, 1). Display matrix size is 5×3



"E"

"F"

Solution :=

Input pattern.	Inputs	Targets	Weights
E	[1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 1]	[-1 1]	W_1
F	[1 1 1 1 1 1 1 -1 -1 1 -1 -1 1 -1 -1]	[1, 1]	W_2

X vectors as input

$$W = \{w_{ij}\} = \sum_{p=1}^{P} s_i(p) \cdot t_j(p)$$

W_1 =

$W_1 =$

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \end{bmatrix}_{15\times 1} \begin{bmatrix} -1 & 1 \end{bmatrix}_{1\times 2} = \begin{bmatrix} -1 & 1 \\ -1 & 1 \\ -1 & 1 \\ -1 & 1 \\ 1 & -1 \\ 1 & -1 \\ -1 & 1 \\ -1 & 1 \\ -1 & 1 \\ -1 & 1 \\ 1 & -1 \\ 1 & -1 \\ -1 & 1 \\ -1 & 1 \\ -1 & 1 \end{bmatrix}_{15\times 2}$$

$W_2 =$

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \end{bmatrix}$$

$W = W_1 + W_2 =$

$$\begin{bmatrix} 0 & 2 \\ 0 & 2 \\ 0 & 2 \\ 0 & 2 \\ 2 & 0 \\ 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \\ 0 & 2 \\ 0 & -2 \\ 0 & -2 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \end{bmatrix}_{15\times 2}$$

Testing the network with test vectors E & F

For pattern E compare the net input

$$y_{in} = \begin{bmatrix} 1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 0 & 2 \\ 0 & 2 \\ 0 & 2 \\ 2 & 0 \\ 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \\ 0 & 2 \\ 0 & -2 \\ 0 & -2 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \end{bmatrix} = \begin{bmatrix} -12 & 18 \end{bmatrix}$$

$y = f(y_{in}) = [-1 \quad 1]$ correct response

For test pattern F:

$$y_{in} = [1\ 1\ 1\ 1\ 1\ 1\ 1\ -1\ -1\ 1\ -1\ -1\ 1\ -1\ -1] \begin{bmatrix} 0 & 2 \\ 0 & 2 \\ 0 & 2 \\ 0 & 2 \\ 2 & 0 \\ 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \\ 0 & 2 \\ 0 & -2 \\ 0 & -2 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \end{bmatrix} = [12 \quad 18]$$

$y = f(y_{in}) = [1, 1]$ correct response.

Y vectors as input : weight matrix when Y vectors are used as input is obtained as the transpose of weight matrix when X's is presented as ~~output~~ input

Testing the network

1) for E, now input is $[-1\ 1]$ computing net input

$$y_{in} = X \cdot W^T$$

$$= \underset{1\times 2}{[-1\ 1]} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & 2 & 2 & 0 & -2 & -2 & 0 & 0 & 0 & 0 & -2 & -2 \\ 2 & 2 & 2 & 2 & 0 & 0 & 2 & 0 & 0 & 2 & -2 & -2 & 2 & 0 & 0 \end{bmatrix}_{2\times 15}$$

$$y_{in} = [2\ 2\ 2\ 2\ 2\ -2\ 2\ 2\ 2\ -2\ -2\ 2\ 2\ 2]$$

$y = f(y_{in})$

$= [1\ 1\ 1\ 1\ -1\ -1\ 1\ 1\ 1\ 1\ -1\ -1\ 1\ 1\ 1]$

Correct response

For F,

$y_{in} = X \cdot W^T$

$= [1\ 1] \begin{bmatrix} 0 & 0 & 0 & 0 & 2 & 2 & 0 & -2 & -2 & 0 & 0 & 0 & 0 & -2 & -2 \\ 2 & 2 & 2 & 2 & 0 & 0 & 2 & 0 & 0 & 2 & -2 & -2 & 2 & 0 & 0 \end{bmatrix}$

$y_{in} = [2\ 2\ 2\ 2\ 2\ 2\ 2\ -2\ -2\ 2\ -2\ -2\ 2\ -2\ -2]$

$y = f(y_{in}) = [1\ 1\ 1\ 1\ 1\ 1\ 1\ -1\ -1\ 1\ -1\ -1\ 1\ -1\ -1]$

Correct response.

Thus BAM network has been constructed and tested in both directions from X to Y and Y to X.

* Construct an autoassociative discrete Hopfield network with input vector [1 1 1 -1]. Test the discrete hopefield network with missing entries in first and second components of the stored vector.

Solution :=

The input vector is $x = [1\ 1\ 1\ -1]$.

The weight matrix is given by

$$W = \sum s^T(p)\,t(p) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \end{bmatrix} [1\ 1\ 1\ -1]$$

$$= \begin{bmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{bmatrix}$$

The weight matrix with no self-connection is

$$\therefore W = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

The binary representation of the input vector is $[1\ 1\ 1\ 0]$

We carry out asynchronous updation of weights here. Let it be $Y_1\ Y_4\ Y_3\ Y_2$

for the test input vector with two missing entries in first and second components of the stored vector.

Iteration 1:

step0: weights are initialized to store patterns:

$$W = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

step1: The input vector is $x = [0 \quad 0 \quad 1 \quad 0]$

step2: for this vector $y = [0 \quad 0 \quad 1 \quad 0]$

step3: choose y_i for updating of its activations.

$$y_{in_i} = x_i + \sum_{j=1}^{4} y_j w_{ji}$$

$$= 0 + [0 \quad 0 \quad 1 \quad 0] \begin{bmatrix} 0 \\ 1 \\ 1 \\ -1 \end{bmatrix}$$

$$= 0 + 1$$

$$= 1$$

Applying activations we get $y_{in_1} > 0 \Rightarrow y_1 = 1$.

Broadcasting y_1 to all other units we get

$y = [1 \quad 0 \quad 1 \quad 0]$ → No convergence.

step4: choose y_4 for updating its activations

$$y_{in4} = x_4 + \sum_{j=1}^{4} y_j w_{j4}$$

$$= 0 + \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \\ 0 \end{bmatrix}$$

$$= 0 - 1 - 1$$

$$= -2$$

Applying activations we get $y_{in4} < 0 \Rightarrow y_4 = 0$

Therefore $y = [1\ 0\ 1\ 0]$ → no convergence.

step5: choose y_3 for updating its activation

$$y_{in3} = x_3 + \sum_{j=1}^{4} y_j w_{j3}$$

$$= 1 + \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ -1 \end{bmatrix}$$

$$= 1 + 1 = 2$$

Applying activation we get $y_{in3} > 0 \Rightarrow y_3 = 1$

$\therefore y = [1\ 0\ 1\ 0]$ → no convergence

step6: choose y_2 for updating its activations

$$y_{in2} = x_2 + \sum_{j=1}^{4} y_j w_{j2}$$

$$= 0 + \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ -1 \end{bmatrix}$$

$$= 0 + 2$$

$$= 2$$

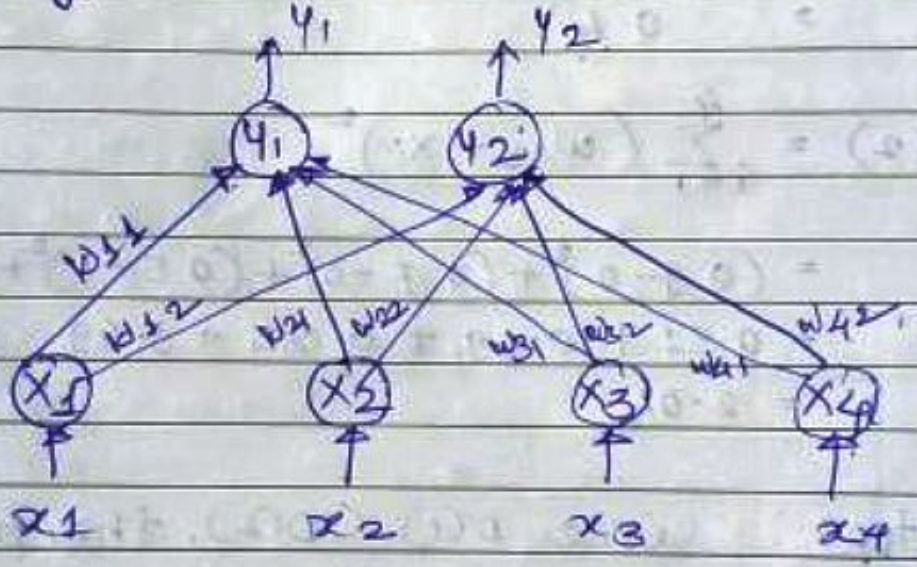
Applying activation we get $y_{in} \geq 0 \Rightarrow$
$y_2 = 1$
$\therefore y = [\ 1\ 1\ 1\ 0\ ]$ → converges with x.

Thus the output y has converged with vector x. in this iteration itself. But one more iteration can be done to check whether further activations are there or not.

* Construct a KSOFM to cluster the four input vectors [0 0 1 1] [1 0 0 0], [0 1 1 0] and [0 0 0 1]. The number of clusters to be formed is two. Assume an initial learning rate of 0.5

Solution :=

The number of input vectors is four and number of clusters to be formed is two. Thus $n = 4$, $m = 2$. The architecture of the KSOFM is given by



Step 0: Initialize the weights randomly between 0 and 1.

$$w_{ij} = \begin{bmatrix} 0.2 & 0.9 \\ 0.4 & 0.7 \\ 0.6 & 0.5 \\ 0.8 & 0.3 \end{bmatrix}$$

$R = 0$, $\alpha(0) = 0.5$

First input vector:

step 1: for $x = [0 \ 0 \ 1 \ 1]$

perform steps 2-4.

step 2: Calculate the euclidean.

$$D(j) = \sum (w_{ij} - x_i)^2$$

$$D(1) = \sum_{i=1}^{4} (w_{i1} - x_i)^2$$

$$D(1) = (0.2-0)^2 + (0.4-0)^2 + (0.6-1)^2 + (0.8-1)^2$$
$$= 0.04 + 0.16 + 0.16 + 0.04$$
$$= 0.4$$

$$D(2) = \sum_{i=1}^{4} (w_{i2} - x_i)^2$$

$$= (0.9-0)^2 + (0.7-0)^2 + (0.5-1)^2 + (0.3-1)^2$$
$$= 0.81 + 0.49 + 0.25 + 0.49$$
$$= 2.04$$

Step 3: Since $D(1) < D(2)$. therfore $D(1)$ is minimum. Hence the winning clucter unit is Y_1 ; i.e $J = 1$

step 4: Update the weights on the winning clucter unit $J = 1$.

$$w_{ij}(new) = w_{ij}(old) + \alpha [x_i - w_{ij}(old)]$$
$$w_{i1}(new) = w_{i1}(old) + 0.5 [x_i - w_{i1}(old)]$$

$$w_{11}(new) = w_{11}(old) + 0.5 [x_1 - w_{11}(old)]$$
$$= 0.2 + 0.5 [0 - 0.2]$$
$$= 0.1$$

$w_{21}(new) = w_{21}(old) + \alpha [x_2 - w_{21}(old)]$

$= 0.4 + 0.5 [0 - 0.4]$

$= 0.2$

$w_{31}(new) = w_{31}(old) + \alpha [x_3 - w_{31}(old)]$

$= 0.6 + 0.5 [1 - 0.6]$

$= 0.8$

$w_{41}(new) = w_{41}(old) + \alpha [x_4 - w_{41}(old)]$

$= 0.8 + 0.5 [1 - 0.8]$

$= 0.9$

The updated weight matrix after presentation of first input pattern is

$$w_{ij} = \begin{bmatrix} 0.1 & 0.9 \\ 0.2 & 0.7 \\ 0.8 & 0.5 \\ 0.9 & 0.3 \end{bmatrix}$$

Second input vector.

Step 1: for $x = [1\ 0\ 0\ 0]$ perform steps 2 - 4

step 2: Calculate the euclidean distance

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

$$D(1) = \sum_{i=1}^{4} (w_{i1} - x_i)^2$$

$D(1) = (0.1 - 1)^2 + (0.2 - 0)^2 + (0.8 - 0)^2 + (0.9 - 0)^2$

$= 0.81 + 0.04 + 0.16 + 0.81$

$= 2.3$

$D(2) = \sum_{i=1}^{4} (w_{i2} - x_i)^2$

$= (0.9 - 1)^2 + (0.7 - 0)^2 + (0.5 - 0)^2 + (0.3 - 0)^2$

$= 0.01 + 0.49 + 0.25 + 0.09$

$= 0.84$

step3: since $D(2) < D(1)$
Therefore $D(2)$ is minimum. Hence the winning cluster unit is y_2, i.e $J = 2$

step4: Update the weights on the winning cluster unit $J = 2$

$$w_{ij}(new) = w_{ij}(old) + \alpha [x_i - w_{ij}(old)]$$

$$w_{12}(new) = w_{12}(old) + 0.5[x_1 - w_{12}(old)]$$
$$= 0.9 + 0.5[1 - 0.9]$$
$$= 0.95$$

$$w_{22}(new) = w_{22}(old) + 0.5[x_2 - w_{22}(old)]$$
$$= 0.7 + 0.5[0 - 0.7]$$
$$= 0.35$$

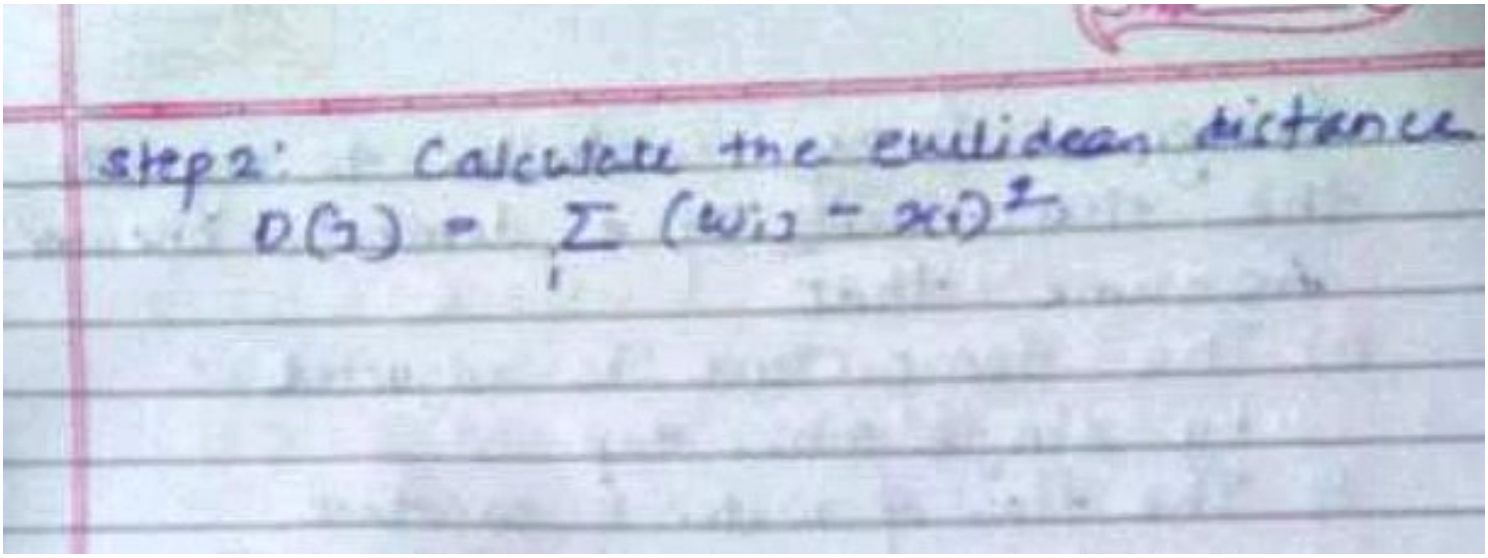
$$w_{32}(new) = w_{32}(old) + 0.5[x_3 - w_{32}(old)]$$
$$= 0.5 + 0.5[0 - 0.5]$$
$$= 0.25$$

$$w_{42}(new) = w_{42}(old) + 0.5[x_4 - w_{42}(old)]$$
$$= 0.3 + 0.5[0 - 0.3]$$
$$= 0.15$$

The updated weight matrix after presentation of second input pattern is

$$w_{ij} = \begin{bmatrix} 0.1 & 0.95 \\ 0.2 & 0.35 \\ 0.8 & 0.25 \\ 0.9 & 0.15 \end{bmatrix}$$

Third input vector
step1: For $x = [0\ 1\ 1\ 0]$ perform steps 2-4


step 2: Calculate the euclidean distance
$D(j) = \sum_i (w_{ij} - x_i)^2$

Design a fuzzy controller to regulate the tempreture of a domestic shower. Assume that

a) The tempreture is adjusted by single mixer tap.
b) The flow of water is constant.
c) control variable is the ratio of the hot to the cold water input.

The design should clearly mention the descriptors used for fuzzy sets and control variables, set of rules to generate control action and defuzzification.

Solution :- Step1:

- Here input is the position of the mixer tap. Assume that the position of mixer tap is measured in degrees (0° to 180°) It represents opening of the mixer tap in degrees. 0° indicates tap is closed and 180° indicates tap is fully opened.
- Output is tempreture of water according to the position of mixer tap. It is measured in °C.

Descriptors for input variable (position of mixer tap) are given below.

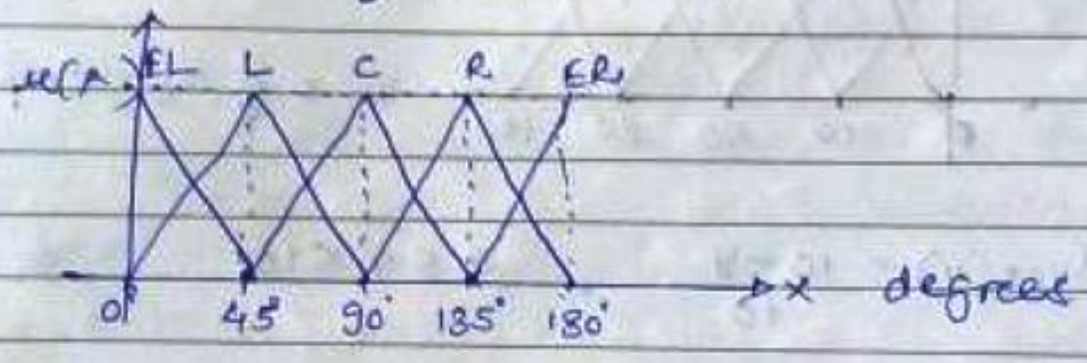
EL = Extreme left
L = left
C = centre.
R = right
ER = extreme right

descriptors for output variable (temperature of water) are given below

VCT : very cold temperature
CT : cold temperature
WT - warm temperature
HT - hot temperature
VHT - very hot temperature

Step 2: Define membership function for input and output variables

1. Membership function for input variable position of mixer tap



$$\mu_{EL}(x) = \frac{45-x}{45}, \quad 0 \le x \le 45$$

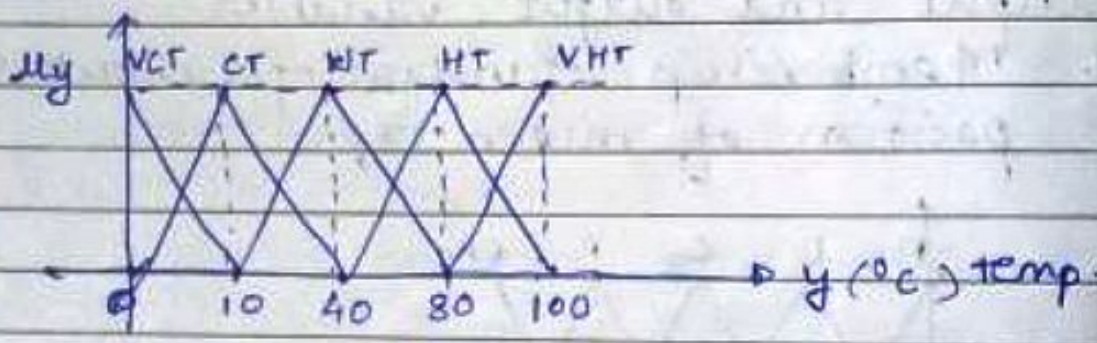
$$\mu_{L}(x) = \begin{cases} \frac{x}{45} & 0 \le x \le 45 \\ \frac{90-x}{45} & 45 < x \le 90 \end{cases}$$

$$\mu_{C}(x) = \begin{cases} \frac{x-45}{45} & 45 \le x \le 90 \\ \frac{135-x}{45} & 90 < x \le 135 \end{cases}$$

$$\mu_R(x) = \begin{cases} \frac{x-90}{45} & 90 \le x \le 135 \\ \frac{180-x}{45} & 135 < x \le 180 \end{cases}$$

$$\mu_{ER}(x) = \frac{x-185}{45}, \quad 135 \le x \le 180$$

Membership function for output variable- tempreture of water



$$\mu_{VCT}(y) = \frac{10-y}{10}, \quad 0 \le y \le 10$$

$$\mu_{CT}(y) = \begin{cases} \frac{y}{10} & 0 \le y \le 10 \\ \frac{40-y}{30} & 10 < y \le 40 \end{cases}$$

$$\mu_{WT}(y) = \begin{cases} \frac{y-10}{30} & 10 \le y \le 40 \\ \frac{80-y}{40} & 40 < y \le 80 \end{cases}$$

$$\mu_{HT}(y) = \begin{cases} \frac{y-40}{40} & 40 \le y \le 80 \\ \frac{100-y}{20} & 80 < y \le 100 \end{cases}$$

$\mu_{VHT}(y) = \frac{y-80}{20}, \quad 80 \le y \le 100$

step 3:
Rule base:

	Input (mixer tap position)	Output (temperature of water)
1)	EL	VCT
2)	L	CT
3)	C	MT
4)	R	HT
5)	ER	VHT

Step 4: Rule evaluation
Assume that mixer tap position is 75^0
This value $x = 75^0$ maps to following two MFs of rule 2 and 3

Rule 2: $\mu_L(x) = \frac{90-x}{45}$

Rule 3: $\mu_C(x) = \frac{x-45}{45}$

Substitute the value of $x = 75$ in above equations:

$$\mu_L(75) = \frac{90-75}{45} = \frac{1}{3}$$

$$\mu_C(x) = \frac{75-45}{45} = \frac{2}{3}$$

step 5: Defuzzification
We find the rule with the maximum strength = max (strength of rule 1, strength of Rule 2)

$= \max\left(\frac{1}{3}, \frac{2}{3}\right)$

$= \frac{2}{3}$

Thus rule 3 has the maximum strength. According to rule 3, if mixer tap position is C. then water tempreture is warm. so we use output MFs of warm water tempreture for defuzzification

$$\mu_{WT}(y) = \frac{y-10}{20}$$

$$\text{and } \mu_{WT}(y) = \frac{80-y}{40}$$

Since the strength of rule is 3 is $\frac{2}{3}$ substitute $\mu_{WT}(y) = \frac{2}{3}$ in the above two equations

$$\frac{y-10}{80} = \frac{2}{3} \Rightarrow y = 50$$

$$\frac{80-y}{40} = \frac{2}{3} \Rightarrow y = 53$$

Now take the average (mean) of these values

$$y^* = \frac{50+53}{2}$$

$$= 41.5^\circ C$$

Design a controller to determine wash time of a domestic washing machine. Assume that input is dirt and greece on cloths. Use three descriptors for input and five descriptors for output variables. Derive set of rules for controller action and defuzzification. The design should be supported by figures whereever possible. Show that if the cloths are soiled to a large degree the wash time will be more and vice-versa.

Solution :=

Step1 :-

Here inputs are dirt and greece. Assume that they are measured in percentage %. Output is wash time measured in minutes.

Descriptors for dirt are as follows.

SD - small dirt
MD - medium dirt
LD - large dirt

Descriptors for grease are [NG, MG, LG]

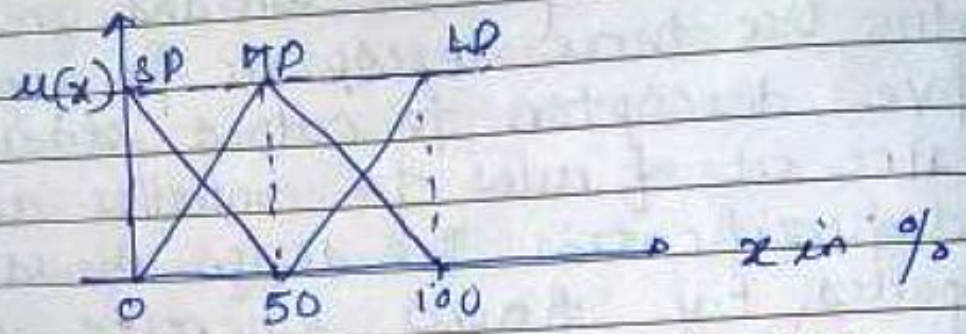
NG = No grease
MG = medium grease
LG = large grease

We use five descriptors for output variable

VS
S
M
L
VL

Step 2:

1) Membership function for dirt.



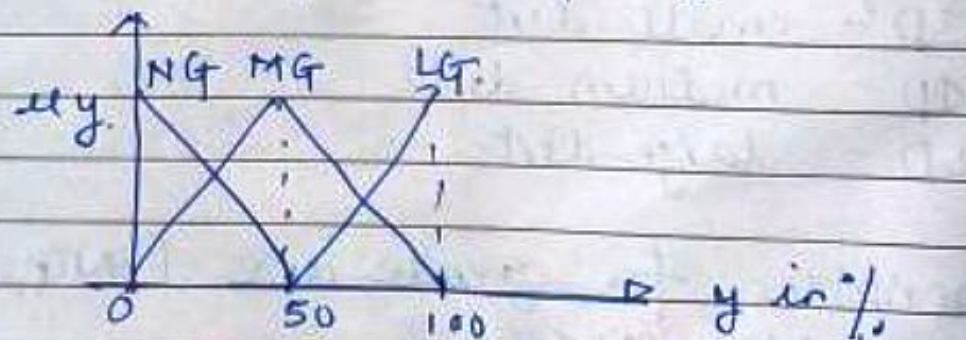
$$\mu_{SD}(x) = \frac{50 - x}{50} \qquad 0 \le x \le 50$$

$$\mu_{MD}(x) = \frac{x}{50} \qquad 0 \le x \le 50$$

$$\frac{100 - x}{50} \qquad 50 < x \le 100.$$

$$\mu_{LD}(x) = \frac{x - 50}{50} \qquad 50 \le x \le 100$$

Membership function for grease.



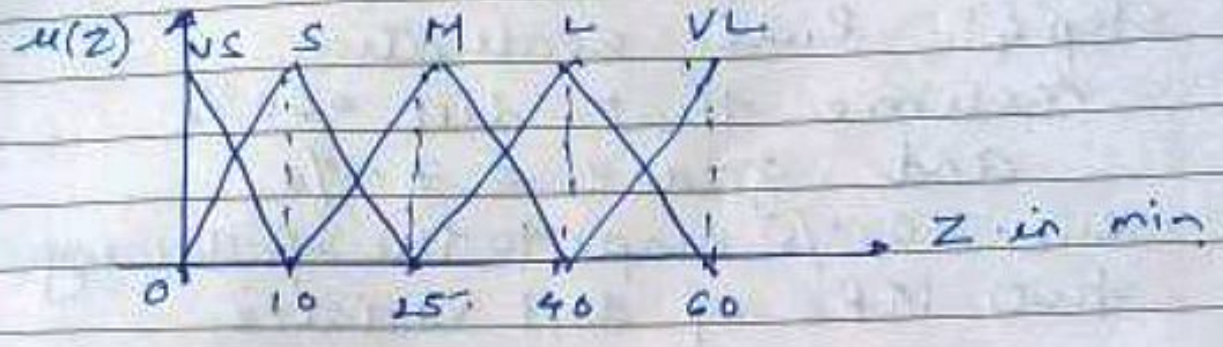
$$\mu_{NG}(y) = \frac{50 - y}{50} \qquad 0 \le y \le 50$$

$$\mu_{MG}(y) = \frac{y}{50} \qquad 0 \le y \le 50$$

$$= \frac{100 - y}{50} \qquad 50 < y \le 100$$

$$\mu_{LG}(4) = \frac{4-50}{50} \qquad 50 \le 4 \le 100$$

Membership function for waste time.



$$\mu_{VS}(Z) = \frac{10-Z}{10} \qquad 0 \le Z \le 10$$

$$\mu_S(Z) = \frac{Z}{10} \qquad 0 \le Z \le 10$$

$$\frac{25-Z}{15} \qquad 10 \le Z \le 25$$

$$\mu_M(Z) = \frac{Z-10}{15} \qquad 10 \le Z \le 25$$

$$= \frac{40-Z}{15} \qquad 25 \le Z \le 40$$

$$\mu_L(Z) = \frac{Z-25}{15} \qquad 25 \le Z \le 40$$

$$= \frac{60-Z}{20} \qquad 40 \le Z \le 60$$

$$\mu_{VL}(Z) = \frac{Z-40}{20} \qquad 40 \le Z \le 60$$

Step 3: Rule Base

x \ y	NG	MG	LG
SD	VS	M	L
MD	S	M	L
LD	M	L	VL

Step 4: Rule evaluation

Assume that dirt = 60%
and grease = 70%

dirt = 60% maps to the following two MFs of dirt variable.

$$\mu_{MD}(x) = \frac{100 - x}{50} \qquad \mu_{LD}(x) = \frac{x - 50}{50}$$

Similarly grease = 70% maps to the following two MF. of grease variable.

$$\mu_{MG}(y) = \frac{100 - y}{50} \qquad \mu_{LG}(y) = \frac{y - 50}{50}$$

∴ Evaluate

$$\therefore \mu_{MD}(x) = \frac{100 - 60}{50} = \frac{4}{5}$$

$$\mu_{LD}(x) = \frac{60 - 50}{50} = \frac{1}{5}$$

$$\mu_{MG}(y) = \frac{100 - 70}{50} = \frac{3}{5}$$

$$\mu_{LG}(y) = \frac{70 - 50}{50} = \frac{2}{5}$$

The above four equations lead to the following four rules that we are suppose to evaluate

1) dirt is medium and grease is medium
2) dirt is medium and grease is large
3) dirt is large and grease is medium
4) dirt is large and grease is medium

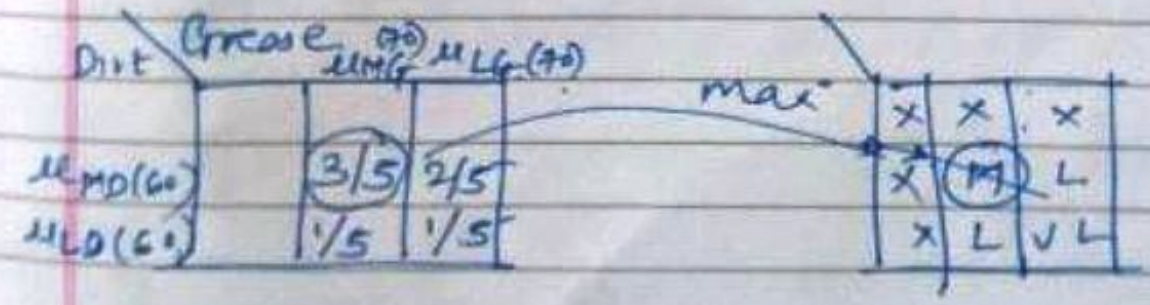
Strength of rule 1 = min ($\mu_{MD}(60)$ $\mu_{MG}(70)$
= min (4/5, 3/5)
= 3/5

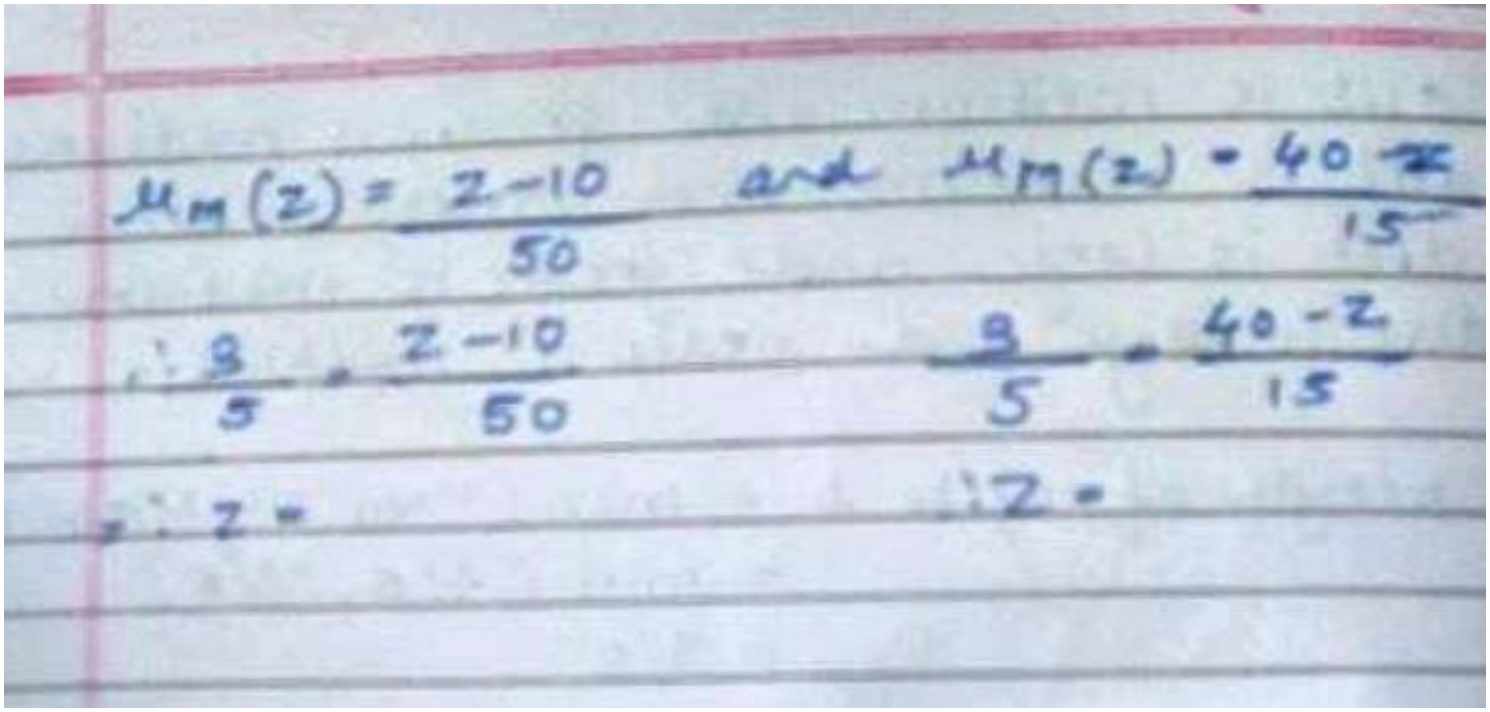
Strength of rule 2 = min ($\mu_{MD}(60)$, $\mu_{LG}(70)$)
= min (4/5, 2/5)
= 2/5

Strength of rule 3 = min ($\mu_{MD}(60)$, $\mu_{MG}(70)$)
= min (1/5, 3/5)
= 1/5

Strength of rule 4 = min ($\mu_{LD}(60)$, $\mu_{LG}(70)$)
= min (1/5, 2/5)
= 1/5

Rule strength mapping.

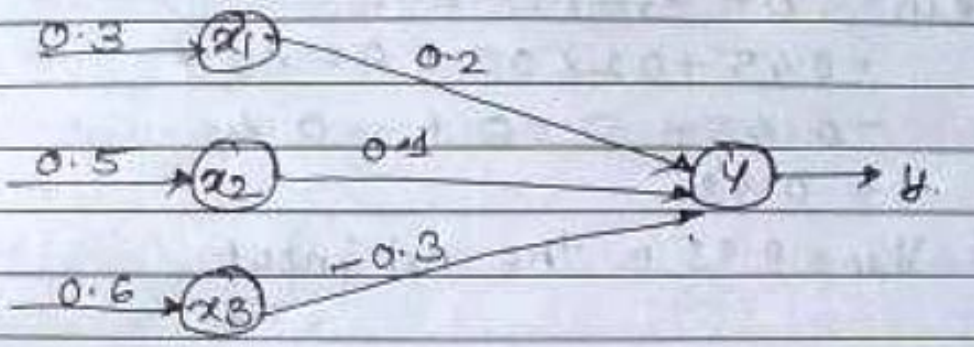



$\mu_M(z) = \frac{z-10}{50}$ and $\mu_M(z) = \frac{40-z}{15}$
$\therefore \frac{9}{5} = \frac{z-10}{50}$
$\frac{9}{5} = \frac{40-z}{15}$
$\therefore z =$
$\therefore z =$

Module 1

Artificial neural network.

1) for the network shown in figure. Calculate the net input to the output neuron.



Solution:-

$[x_1, x_2, x_3] = [0.3, 0.5, 0.6]$

$[w_1, w_2, w_3] = [0.2, 0.1, -0.3]$

The net input can be calculated as

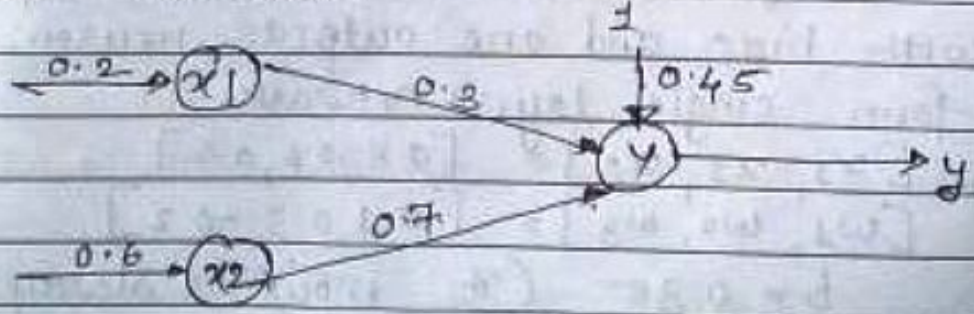
$y_{in} = x_1w_1 + x_2w_2 + x_3w_3$

$= 0.3 \times 0.2 + 0.5 \times 0.1 + 0.6 \times -0.3$

$= 0.06 + 0.05 - 0.18$

$= -0.07$

2) Calculate the net input for the network shown in figure with bias included in the network



Solution :-

$Y_{in} = [x_1, x_2] = [0.2, 0.6]$

$[w_1, w_2] = [0.3, 0.7]$

bias $b = 0.45$, bias input $x_0 = 1$

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

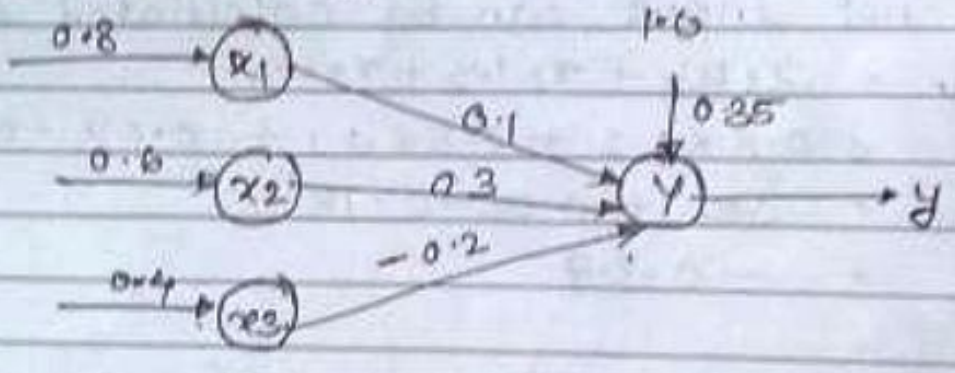
$= 0.45 + 0.2 \times 0.3 + 0.6 \times 0.7$

$= 0.45 + 0.06 + 0.42$

$= 0.93$

$\therefore Y_{in} = 0.93$ is the net input.

3) Obtain the output of neuron Y for the network shown in figure using activation functions. as i) binary sigmoidal ii) bipolar sigmoidal.



Solution :-

The given network has three input neurons with bias and one output neuron. These form single layer network.

$[x_1, x_2, x_3] = [0.8, 0.6, 0.4]$

$[w_1, w_2, w_3] = [0.1, 0.3, -0.2]$

$b = 0.35$ (its input is always 1)

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

$$= b + x_1 w_1 + x_2 w_2 + x_3 w_3$$

$$= 0.35 + 0.8 \times 0.1 + 0.6 \times 0.3 + 0.4 \times -0.2$$

$$= 0.35 + 0.08 + 0.18 - 0.08$$

$$= 0.53$$

i) for binary sigmoidal activation function

$$y = f(y_{in}) = \frac{1}{1 + e^{-y_{in}}} = \frac{1}{1 + e^{-0.53}} = 0.625$$

ii) For bipolar sigmoidal activation function

$$y = f(y_{in}) = \frac{2}{1 + e^{-y_{in}}} - 1 = \frac{2}{1 + e^{-0.53}} - 1 = 0.259$$

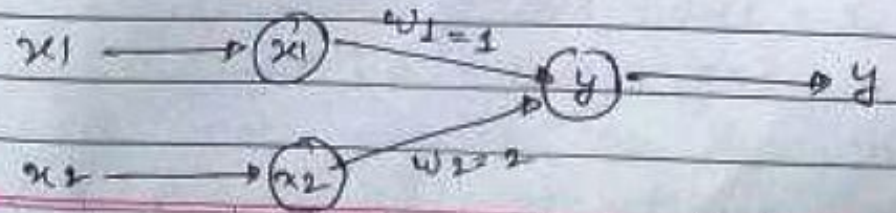
4) Implement AND function using McCulloch-Pitts neuron (take binary data)

Solution:-

Truth table for AND

x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

Assume the weights be $w_i = 1$



for an AND function, the output is high if bothe the inputs are high.

$\therefore x_1 = 1 \quad x_2 = 1$

for this case,

net input $y_{in} = x_1 w_1 + x_2 w_2$

$= 1 \times 1 + 1 \times 1$

$= 2$

Based on this input set, the threshold is calculated as 2 (θ). So if the ~~thresh~~ netput net input value is greater than or equal to 2, the neuron fires., else it doesnot fire.

This can also be calculted as.

$$\theta \geq nw - p$$

Here $n = 2$, $w = 1$, (excitatory input) and $p = 0$ (no inhibitory weights).

Substituting these values in above equation

$$\theta \geq 2 \times 1 - 0 \implies \theta \geq 2$$

Thus the output of neuron. Y can be written as

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 2 \\ 0 & \text{if } y_{in} < 2 \end{cases}$$

where "2" represents the threshold value.

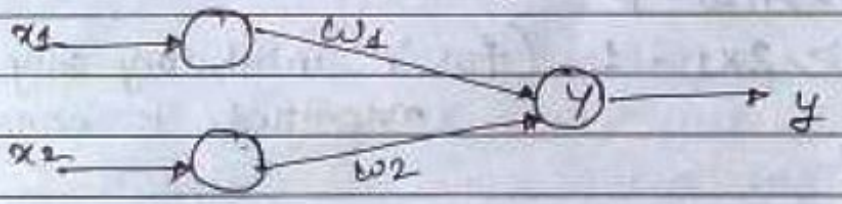
5 Implement ANDNOT function using Mc-Culloch-Pitts neuron. (use binary data)

Solution :-

In case of ANDNOT function, the response is true if the first input is true and the second input is false. for all other input variations, the output is false. The truth table for ANDNOT is

x_1	x_2	y
0	0	0
0	1	0
1	0	1
1	1	0

The given function gives an output only when $x_1 = 1$ and $x_2 = 0$. The weights have to be decided only after the analysis.



Case 1 :- Assume that both weights ~~are~~ w_1, w_2 are excitatory i.e

$$w_1 = w_2 = 1$$

for input

(1, 1) - $y_{in} = 1 \times 1 + 1 \times 1 = 2$

(1, 0) - $y_{in} = 1 \times 1 + 0 \times 1 = 1$

(0, 1) - $y_{in} = 0 \times 1 + 1 \times 1 = 1$

(0, 0) - $y_{in} = 0 \times 1 + 0 \times 1 = 0$

It is not possible to fire the neuron for input (1, 0) only. Hence the weights are not suitable.

Case 2:

Assume one weight as excitatory and the other as inhibitory i.e

$w_1 = 1, \quad w_2 = -1$

for the inputs calculate net input.

(1, 1)	$y_{in} = 1 \times 1 + 1 \times -1$	$= 0$
(1, 0)	$y_{in} = 1 \times 1 + 0 \times -1$	$= 1$
(0, 1)	$y_{in} = 0 \times 1 + 1 \times -1$	$= -1$
(0, 0)	$y_{in} = 0 \times 1 + 0 \times -1$	$= 0$

from the calculated net inputs, now it is possible to fire the neuron for input (1, 0) only by fixing a threshold of 1.

i.e $\theta \geq 1$ for Y unit. Thus

$w_1 = 1, \quad w_2 = -1, \quad \theta \geq 1$

The value of θ is calculated using the following:

$\theta \geq nw - p$

$\theta \geq 2 \times 1 - 1$ (for 'p' inhibitory only magnitude is considered)

Thus the output of neuron Y can be written as

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

6) Implement XOR function using Mc-Culloch Pitts neuron

Solution:-

The truth table for XOR function is

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

In this case the output is "ON" for only odd number of 1's. for the rest it is 'OFF'. XOR function cannot be represented by simple and single logic function; it is represented as

$$y = x_1\bar{x}_2 + \bar{x}_1 x_2$$

$$y = z_1 + z_2$$

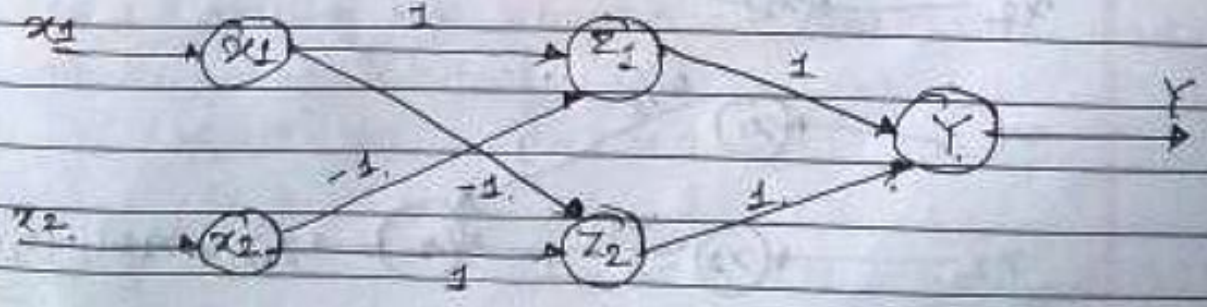
where

$z_1 = x_1\bar{x}_2$ (function 1)

$z_2 = \bar{x}_1 x_2$ (function 2)

$y = z_1$ (OR) z_2 (function 3)

A single-layer net is not sufficient to represent the function. An intermediate layer is necessary.



- First function ($Z_1 = x_1 \bar{x}_2$)...

The truth table for function Z_1 is shown in figure.

x_1	x_2	Z_1
0	0	0
0	1	0
1	0	1
1	1	0

The net representation is given as.

Case 1: Assume both weights are excitatory i.e

$w_{11} = w_{21} = 1$

calculate the net inputs for inputs

(0,0) $Z_{1in} = 0\times1 + 0\times1 = 0$

(0,1) $Z_{1in} = 0\times1 + 1\times1 = 1$

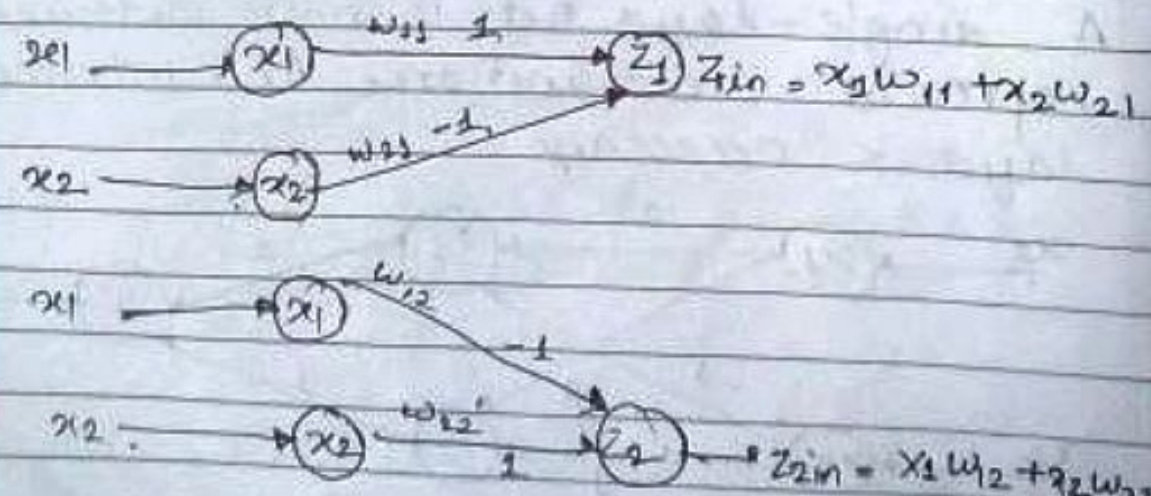
(1,0) $Z_{1in} = 1\times1 + 0\times1 = 1$

(1,1) $Z_{1in} = 1\times1 + 1\times1 = 2$

Hence, it is not possible to obtain function Z_1 using these weights.

Case 2: Assume one weight as excitatory and the other as inhibitory i.e

$w_{11} = 1 \quad w_{21} = -1$



Calculate the net inputs

(0,0) $z_{1in} = 0 \times 1 + 0 \times -1 = 0$

(0,1) $z_{1in} = 0 \times 1 + 1 \times -1 = -1$

(1,0) $z_{1in} = 1 \times 1 + 0 \times -1 = 1$

(1,1) $z_{1in} = 1 \times 1 + 1 \times -1 = 0$

On the basis of the calculated net input, it is possible to get the required output

$\therefore$ $w_{11} = +1$ $w_{21} = -1$, $\theta \geq 1$ for Z_1 neuron

Second function:- $(x_2 = \bar{x}_1 x_2)$.

The truth table for function Z_2 is shown below

x_1	x_2	Z_2
0	0	0
0	1	1
1	0	0
1	1	0

Case 1: Assume both weights are excitatory

i.e $w_{12} = w_{22} = 1$

Calculate the net input

(0,0) $= Z_{2in} = 0 \times 1 + 0 \times 1 = 0$

(0,1) $= Z_{2in} = 0 \times 1 + 1 \times 1 = 1$

(1,0) $= Z_{2in} = 1 \times 0 + 1 \times 1 = 1$

(1,1) $= Z_{2in} = 1 \times 1 + 1 \times 1 = 1$

Hence it is not possible to obtain function Z_2 using these weights

Case 2: Assume one weight as excitatory and other as inhibitory i.e

$w_{12} = -1$ $w_{22} = 1$.

Now calculate the net inputs

(0,0) $z_{2in} = 0 \times -1 + 0 \times 1 = 0$

(0,1) $z_{2in} = 0 \times -1 + 1 \times 1 = 1$

(1,0) $z_{2in} = 1 \times -1 + 0 \times 1 = -1$

(1,1) $z_{2in} = 1 \times -1 + 1 \times 1 = 0$

Thus based on this calculated net input, it is possible to get the required output

$w_{12} = -1$, $w_{22} = 2$, $\theta \geq 1$.

Third function ($y = z_1$ OR z_2)

x_1	x_2	z_1	z_2	y
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

Here the net input is calculated as

$y_{in} = z_1 v_1 + z_2 v_2$

Case 1: Assume both weights as excitatory ie

$v_1 = v_2 = 1$

Now calculate net input

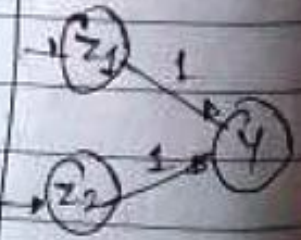
(0,0) $y_{in} = 0 \times 1 + 0 \times 1 = 0$

(0,1) $y_{in} = 0 \times 1 + 1 \times 1 = 1$

(1,0) $y_{in} = 1 \times 1 + 0 \times 1 = 1$

(1,1) $y_{in} = 1 \times 1 + 1 \times 1 = 0$

$\therefore \theta = 1$



The analysis is made for XOR function using M-P neuron. Thus for XOR function the weights are

$w_{11} = w_{22} = 1$

$w_{12} = w_{21} = -1$

$v_1 = v_2 = 1$

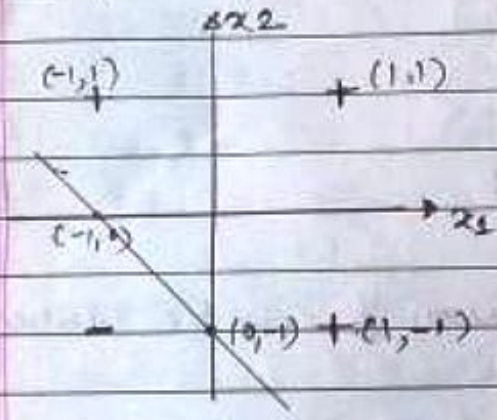
7) Using the linear seperability concept, obtain the response for OR function. (take bipolar inputs and bipolar outputs)

Solution:-

Truth table for "OR" using bipolar input

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

If output is '1' it is denoted as '+1' else "-1".



Assuming the coordinates as (-1,0) and (0,-1) as (x_1, y_1), (x_2, y_2). The slope, m can be calculated as

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{-1-0}{0-(-1)} = \frac{-1}{1} = -1$$

We now calculate c

$$y = mx + c$$
$$c = y - mx$$
$$c = y_1 - mx_1$$
$$= 0 - (-1)(-1)$$
$$= -1$$

Using this value

$$y = mx + c$$
$$= -1x - 1$$
$$= -x - 1$$

Here the quadrants are not x and y, but x_1 and x_2 so,

$$x_2 = -x_1 - 1.$$

(1)

This can be written as

$$x_2 = \frac{-w_1}{w_2} x_1 - \frac{b}{w_2} \quad (2)$$

Comparing equation 1 and 2, we get

$$\frac{w_1}{w_2} = \frac{1}{1} \qquad \frac{b}{w_2} = \frac{1}{1}$$

$\therefore w_1 = 1, \quad w_2 = 1, \quad b = 1$

Calculating the net input and output of OR using these weights and bias

x_1	x_2	b	$y_{in} = b + x_1 w_1 + x_2 w_2$	y
1	1	1	3	1
1	-1	1	1	1
-1	1	1	1	1
-1	-1	1	-1	-1

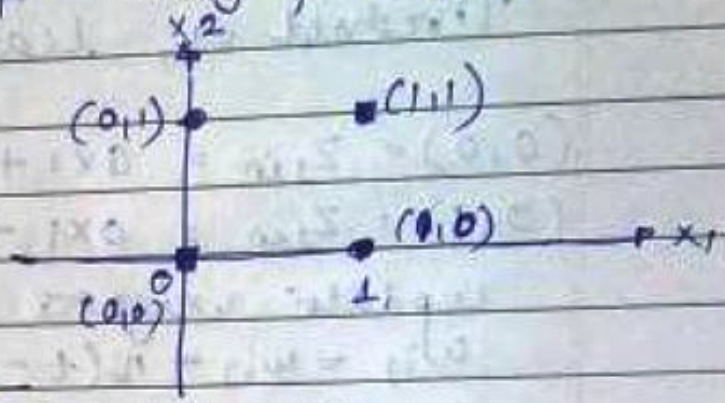
Thus the output of neuron y can be written as

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

where $\theta = 1$

XOR solved example using perceptron.

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



Data is not linearly seperable

$y = x_1 \bar{x_2} + \bar{x_1} x_2$

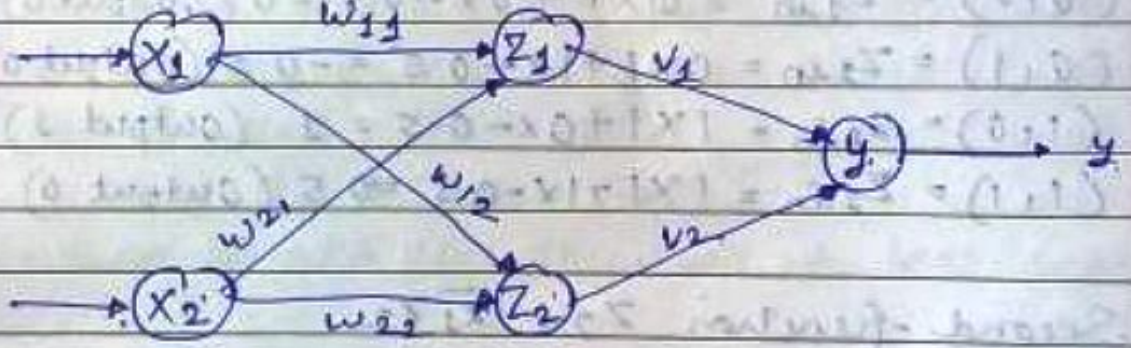
$y = Z_1 + Z_2$

where

$Z_1 = x_1 \bar{x_2}$ (function 1)

$Z_2 = \bar{x_1} x_2$ (function 2)

$y = Z_1$ OR Z_2 (function 3)



First function $Z_1 = x_1 \bar{x_2}$

(t = taget)

x_1	x_2	Z_1
0	0	0
0	1	0
1	0	1
1	1	0

Activation function

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

unipolar discrete

Assume the initial inputs $w_{11} = w_{21} = 1$
Threshold = 1, Learning rate = 1.5

$(0,0) - z_{1in} = 0\times1 + 0\times1 = 0$ (output = 0) = t
$(0,1) - z_{1in} = 0\times1 + 1\times1 = 1$ (output 1) ≠ t

update weights

$w_{ij} = w_{ij} + \eta(t-o)x_i$

$w_{11} = 1 + 1.5(0-1)\times0$
$= 0$

$w_{21} = 1 + 1.5(0-1)\times1$
$= -0.5$

~~$\Delta w_{21} = 1.5(0-1)\times1$~~
$=$

$w_{11} = 1 \quad w_{21} = -0.5$

$(0,0) - z_{1in} = 0\times1 + 0\times-0.5 = 0$ (output 0) = t
$(0,1) - z_{1in} = 0\times1 + 1\times-0.5 = -0.5$ (output 0) = t
$(1,0) - z_{1in} = 1\times1 + 0\times-0.5 = 1$ (output 1) = t
$(1,1) - z_{1in} = 1\times1 + 1\times-0.5 = 0.5$ (output 0) = t

Second function $z_2 = \bar{x}_1 x_2$

x_1	x_2	z_2 (target t)
0	0	0
0	1	1
1	0	0
1	1	0

activation function

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

Assume weights are $w_{21} = w_{22} = 1$
Threshold = 1
Learning rate = 1.5

$(0,0) \rightarrow z_{2in} = 0\times1 + 0\times1 = 0$ (output 0) $= t$

$(0,1) \rightarrow z_{2in} = 0\times1 + 1\times1 = 1$ (output 1) $= t$

$(1,0) = z_{2in} = 1\times1 + 0\times1 = 1$ (output 1) $\neq t$.

update weights.

$w_{ij} = w_{ij} + \eta (t - o) \cdot x_i$

$w_{12} = 1 + 1.5 (0-1) * 1$

$= -0.5$

$w_{22} = 1 + 1.5 (0-1) * 0$

$= 1$.

$\therefore w_{12} = -0.5$, $w_{22} = 1$

$(0,0) = z_{2in} = 0\times-0.5 + 0\times1 = 0$ (output 0) $= t$

$(0,1) - z_{2in} = 0\times-0.5 + 1\times1 = 1$ (output 1) $= t$

$(1,0) = z_{2in} = 1\times-0.5 + 0\times1 = -0.5$ (output 0) $= t$

$(1,1) = z_{2in} = 1\times-0.5 + 1\times1 = 0.5$ (output 0) $= t$

Third function $y = z_1$ OR z_2

$y_{in} = z_1 v_1 + z_2 v_2$

Assume $v_1 = v_2 = 1$, threshold = 1, learning rate = 1.5

$(0,0) = y_{in} = 0\times1 + 0\times1 = 0$ (output = 0) $= t$

$(0,1) = y_{in} = 0\times1 + 1\times1 = 1$ (output = 1) $= t$

$(1,0) = y_{in} = 1\times1 + 0\times1 = 1$ (output 1) $= t$

$(0,0) = y_{in} = 0\times1 + 0\times1 = 0$ (output 0) $= t$

x_1	x_2	z_1	z_2	y
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

$w_{11} = 1$, $w_{21} = -0.5$

$w_{12} = -0.5$, $w_{22} = 1$

$v_1 = v_2 = 1$

Implement AND function using perceptron network for bipolar inputs and targets.

Solution =

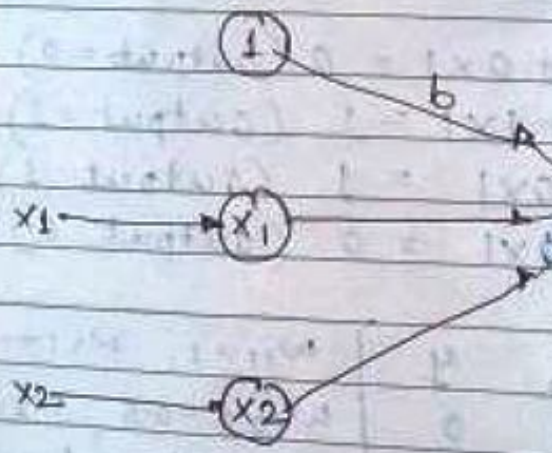
Table shows the truth table for AND function using bipolar inputs and targets

x_1	x_2	t
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

The perceptron network, which uses perceptron learning rule, is used to train ~~obtain~~ the AND function.

The input patterns are presented to the network one by one. When all the four input patterns are presented, then one epoch is said to be completed.

The initial weights and threshold are set to zero. i.e, $w_1 = w_2 = b = 0$, $\theta = 0$, $\alpha = 1$

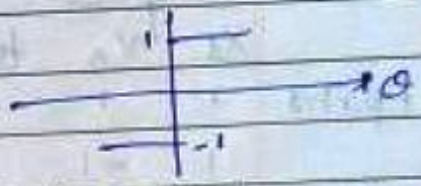


For the first input pattern $x_1 = 1$, $x_2 = 1$, $t = 1$
with weights and bias, $w_1 = 0$, $w_2 = 0$, $b = 0$
Calculate net input

$$y_{in} = b + x_1 w_1 + x_2 w_2$$
$$= 0 + 1 \times 0 + 1 \times 0 = 0$$

The output y is computed by applying activation over the net input calculated.

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



bipolar continuous

Here $\theta = 0$ $\therefore y_{in} = 0$, $\therefore y = 0$
check whether $t = y$, Here $t = 1$, $y = 0$, $t \neq y$
hence weight updation takes place

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

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

$w_2(new) = w_2(old) + \alpha(t - 0) \times 1$ $w_2 old + \alpha t \times 2$
$= 0 + 1(1 - 0) \times 1 = 0 + 1 \times 1 + 1 \times 1$
$= 1$ $= 1$, $\Delta w_2 = 1$

$b(new) = b(old) + \alpha t$
$= 0 + 1 \times 1$
$= 1.$

The weights $w_1 = 1$, $w_2 = 1$, $b = 1$ are the final weights after the first input pattern is presented.
The same process repeated for all the input patterns.

The process can be stopped when all the targets become equal to the calculated output or when a separating line can be obtained using the final weights for separating the positive responses from negative responses.

Input (0 0 0)

	x_1	x_2	b_{in}	target	y_{in}	y	Δw_1	Δw_2	Δb	w_1	w_2	b
Epoch 1	1	1	1	1	0	0	1	1	1	1	1	1
	1	-1	1	-1	1	1	-1	1	-1	0	2	0
	-1	1	1	-1	2	1	1	-1	-1	1	1	-1
	-1	-1	1	-1	-3	-1	0	0	0	1	1	-1
Epoch 2	1	1	1	1	1	1	0	0	0	1	1	-1
	1	-1	1	-1	-1	-1	0	0	0	1	1	-1
	-1	1	1	-1	-1	-1	0	0	0	1	1	-1
	-1	-1	1	-1	-3	-1	0	0	0	1	1	-1

epoch 1 : second input

(1, -1)

$y_{in} = b + w_i x_i$

$= 1 + 1 \times 1 + -1 \times 1$

$= 1$

$y = 1$

$w_1 = w_1(old) + \alpha t \times (-1 - 1) \times 1$

$= 1 + (-1)$

$= 0$

$w_1 new = w_1 old + \alpha t \times 1$

$= 1 + 1 \times -1 \times 1$

$= 0$

$\Delta w_1 = -1$

$w_2 new = w_2 old + \alpha t x_2$

$= 1 + 1 \times -1 \times -1$

$= 2$

$\Delta w_2 = 1$

$b new = b old + \alpha t$

$= 1 + 1 \times -1$

$= 0$

$\Delta b = -1$

epoch 1 : third input

(-1, 1)

$y_{in} = b + w_i x_i$

$= 0 + 0 \times -1 + 2 \times 1$

$= 2$

$y = 1$

$w_1 new = w_1 old + \alpha t x_1$

$= 0 + 1 \times -1 \times -1$

$= 0 + 1$

$= 1$

$\Delta w_1 = 1$

$w_2 new = w_2 old + \alpha t x_2$

$= 2 + 1 \times -1 \times 1$

$= 1$

$\Delta w_2 = -1$

$b = b old + \alpha t$

$= 0 + 1 \times -1$

$= -1$

$\Delta b = -1$

epoch 1 : fourth input
(-1, -1)

$y_{in} = b + \sum w_i x_i$
$= -1 + 1 \times -1 + 1 \times -1$
$= -3$
$y = -1$

$w_{1new} = w_{1old} + \alpha t x_1$
$= 1 + 1 \times -1 \times -1$
$= 1 + 1$
$= 2$
$\Delta w_1 = 1$

$w_{2new} = w_{2old} + \alpha t x_2$
$= 1 + 1 \times -1 \times -1$
$= 1 + 1$
$= 2$
$\Delta w_2 = 1$

$b_{new} = b_{old} + \alpha t$
$= -1 + 1 \times -1$
$= -2$
$\Delta b = -1$

target is same as output y.
∴ do not update weights