Unit - 1

* Neural Network :

→ A processing device, either an algorithm or an actual hardware
→ Inspired by brains.
→ Ability to learn by example
→ Flexible and powerful

⇒ Artificial Neural Networks :

→ An information - processing model inspired by the way biological nervous systems (brain) process information.
→ Learn by example as people
→ Specific application
→ Composed of large number of highly interconnected processing elements (Neurons) working in unison to solve specific problems.
→ Ex. applications - pattern recognition
data classification

• Advantages :

- Adaptive Learning
Ability to learn how to do tasks based on data given for training

- Self-organization
Creates its own organization/representation of info it receives during training

Scanned with CamScanner

- Real time operation:
  Computations may be carried out in Parall

- Fault tolerance via redundant info. Coding:
  - Partial destruction of NN leads to
  - degradation of performance.
  - Some capabilities may be retained.

⇒ Problems:
- Provide a reason behind computer generated answer - Why loan denied.
- How network learned, why it recommends a particular decision - Black box

⇒ Applications:
- Book

* Soft Computing:

→ Problem-Solving technologies:

- Hard Computing
  - Precise models
    - Symbolic logic reasoning
    - Traditional numerical modeling & search
- 1994 Soft Computing
  - Approximate models
    - Approximate Reasoning
    - Functional approx. & Randomized search

Scanned with CamScanner

→ Combination of GA, NN, FL
→ Hybrid technique:
- good learning capacity
- better learning time
- less sensitivity to the prob. of local extremes
- Generate fuzzy knowledge base (linguistic representation & low complexity)

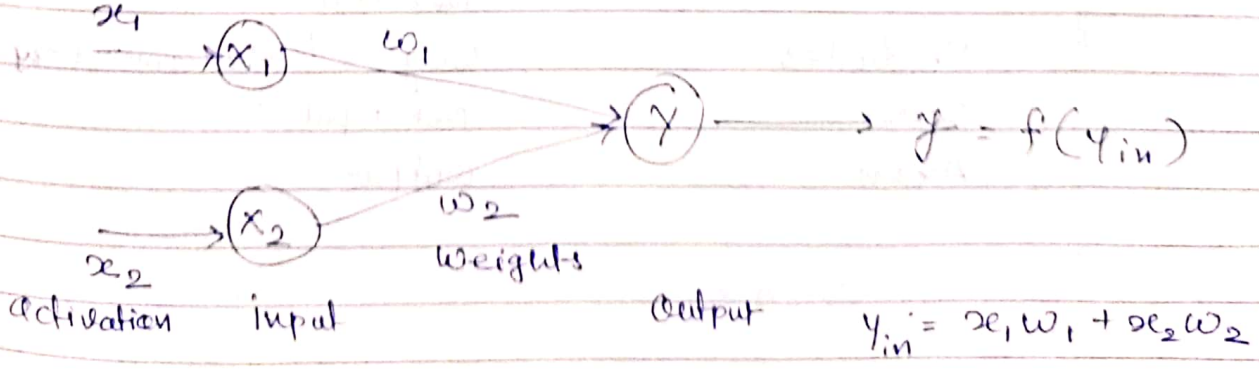
---

* ANN:

→ Neurons / Nodes / units:
- operate in parallel

→ Connection link:
- Each neuron is connected with other
- It has associated weights - contains info. about the input signal - used by NN to solve a problem.

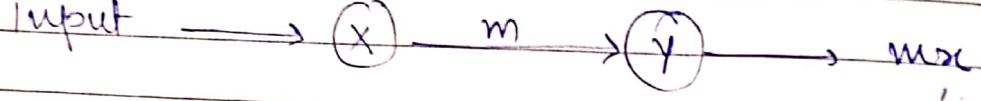
→ Activation / activity level:
- Each neuron has internal state
- function of the inputs the neuron receives
- transmits activation signal to other, only one signal at a time.



Scanned with CamScanner

→ The output y of output neuron Y can be obtained by applying activations over the net input (function of net input)

→ The function to be applied over the net input is called activation function.



$Y = max$

* Biological Neural Network :

→ Biological Neuron:
- 3 Parts - Soma / cell body
  - Dendrites - connected
  - Axon - carries impulses

→ $y_{in} = x_1 w_1 + \dots x_n w_n = \sum_{i=1}^{n} x_i w_i$

BNN	Vs	ANN
Cell		Neuron
Dendrites		Weights / interconnections
Soma		Net input
Axon		Output

Scanned with CamScanner

→ Comparison:

	Brain	Computer
• Speed	- Milliseconds	- nanoseconds
• Processing	- Parallel	- Parallel - Faster
• Size & complexity	- Higher	- depends on design
• Memory	- Store more Retrieving not always good	- Storage can be overload Retrieving is good More adaptability
• Tolerance	- Fault tolerant capability	- No fault tolerance
• Control	- No control unit	- Control unit
Mechanism	- Interconnections & strengths based on chemicals	- Simpler interconne. No chemical actions

* Evolution of NN:
In book - Self study

* Basic models of ANN:

1) Connections
→ Single layer feed forward
→ Multilayer "
→ Single node with its own feedback
→ Single layer recurrent
→ Multi layer "

2) Learning

i) Supervised -
- Input vector has corresponding target vector

Scanned with CamScanner

→ During training - input vector → output v
→ Actual output vector compare target vector
→ Difference - error signal
→ Adjustments of weights

ii) Unsupervised
→ Input vectors of similar type are grouped without using training data.

iii) Reinforcement learning
→ Only critic information is available not exact info.
→ Receives feedback from its environment

3) Activation Functions

→ Applied over the net input to calculate the output of an ANN.

→ Why non linear ?
- Multilayer network - output obtained remains same as single layer while using linear fun^n

i) Identity fun.
ii) Binary step fun
iii) Bipolar step fun
iv) Sigmoidal - Binary
- Bipolar
v) Ramp

Scanned with CamScanner

* Important Terminologies :

1) Weights :
   - → Contains information about input signal
   - → 'Info.' is used to solve a problem.
   - → Matrix form n - processing elements m no. of weights per neuron

2) Bias :
   - → Impact in calculating the net input
   - → Considered like another weight
   - → By adding a component $x_0 = 1 \rightarrow b_j$

3) Threshold : (θ)
   - → Set value based upon which the final output of the network may be calculated.

4) Learning Rate : (α)
   - → To control the amount of weight adjustment at each step of training. (0 to 1)

5) Momentum Factor :
   - → Back propagation network
   - → Convergence is faster if momentum added to weight updation process

6) Vigilance Parameter (ρ)
   - → ART network
   - → To control degree of similarity required for patterns to be assigned to same cluster unit.
   - → (0.7 to 1)

Scanned with CamScanner

* Example 1:

→ AND Function (1 - 1.9)

x_1	x_2	y	θ	y_{in}
0	0	0		
0	1	0		
1	0	0		
1	1	1		

$y_{in} \geq \theta \Rightarrow y = 1$

$y_{in} \leq \theta \Rightarrow y = 0$

Input 1 :

$x_1 = 0, \quad x_2 = 0, \quad y = 0$

$y_{in} = 0 \qquad x_1 w_1 + x_2 w_2$

$y = 0$

$y_{in} \leq \theta \Rightarrow y = 0 \qquad (0 \leq \theta)$

$\theta = [0 \ldots 1] \qquad \theta = 0.1 \ldots 0.9 \qquad \theta > 0$

take $\theta = 1$ ⇒ Checks for all conditions

Input 2 :

$x_1 = 0 \qquad x_2 = 1 \qquad y = 0$

$y_{in} = 1$

$y_{in} \leq \theta \Rightarrow y = 0$

$(1 \leq \theta) \Rightarrow \theta = 1$ True

$2 > \theta$

Input - 3

$x_1 = 1$, $x_2 = 0$ $y = 0$

$y_{in} = 1$

$y_{in} \leq \theta \Rightarrow y = 0$

for $\theta = 1 \Rightarrow 1 \leq \theta = 1 \leq 1$ ✓ True

Input - 4

$x_1 = 1$ $x_2 = 1$ $y = 1$

$y_{in} = 2$

$y_{in} > \theta \Rightarrow y = 1$

$2 > \theta \Rightarrow 2 > 1$ (1.9) True

$2 > 2^-$

So the range is (1 .. 1.9)

$\theta = [1 - 1.9]$

Example 2:

→ OR Function

x_1	x_2	$y \rightarrow \theta$	y_{in}
0	0	0	
0	1	1	
1	0	1	
1	1	1	

Scanned with CamScanner

AND

```
x1 = [0 0 1 1]
x2 = [1 0 1 0]

w = 1
t = 1
v = x1 * w + x2 * w

for i = 1:4
    if v(i) > t
        y(i) = 1
    else
        y(i) = 0
    end
end
disp(y)
```

out - 0 1 1 2
0 0 0 1

OR

```
t=0
if v(i) > t
    y(i) = 0
else
    y(i) = 1
```

Scanned with CamScanner

* Linear Separability:

→ ANN does not provide Exact solution for nonlinear problem.

→ The separation of the input space into regions is based on whether the network response is positive or negative

→ Decision line:

• Bipolar

$$b + \sum_{i=1}^{n} x_i w_i = 0$$

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

$$\Rightarrow \quad x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$

→ Find the values of w_1, w_2, b

• Threshold - θ

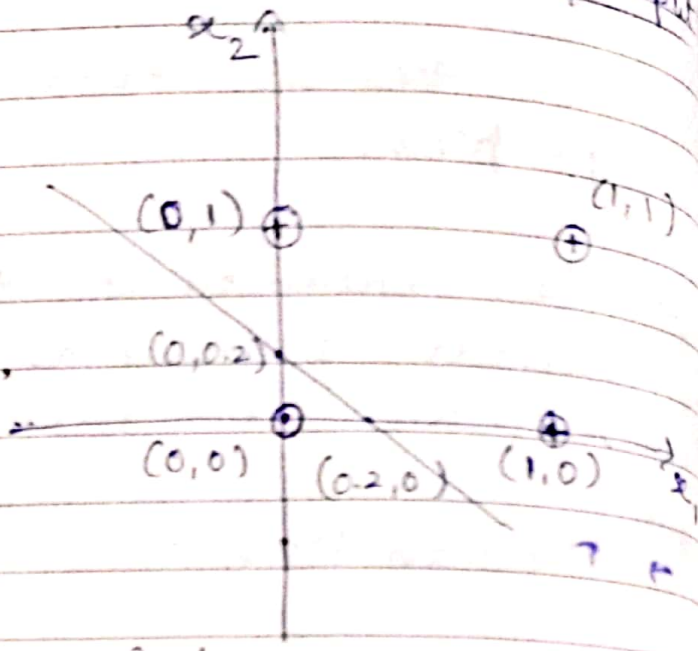
$$x_1 w_1 + x_2 w_2 = \theta$$

$$x_2 = -\frac{w_1}{w_2} x_1 + \frac{\theta}{w_2} \qquad (\theta = -b)$$

Scanned with CamScanner

* Example -1 : OR Function , binary input

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



⊙ → $y = 0$

⊕ → $y = 1$

(No. of lines = no. of inputs)

Activation Function :

$$y_{in} > \theta \Rightarrow y = 1$$

$$y_{in} \leq \theta \Rightarrow y = 0$$

Points $(0.2, 0)$ $(0, 0.2)$ — x_1, y_1 x_2, y_2

Line equation - $y = mx + c$

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{0.2 - 0}{0 - 0.2} = -1$$

$$\boxed{m = -1}$$

→ $y_1 = m x_1 + c$

$c = y_1 - m x_1$

$= 0 - (-1)(0.2)$

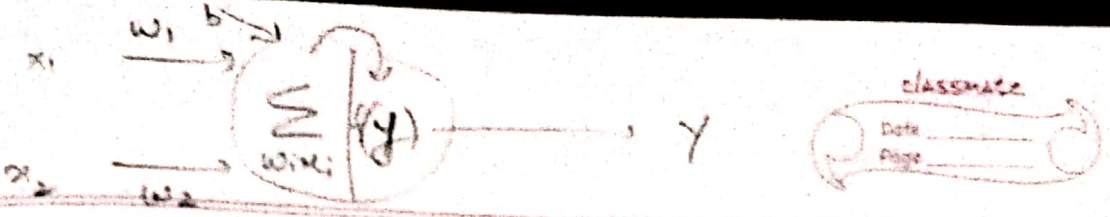
$\boxed{c = 0.2}$

$y = mx + c$

$= -x + 0.2$

$x_2 = -x_1 + 0.2$ ——— (1)

Scanned with CamScanner



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

Comparing equation (1) & (2)

$$\frac{w_1}{w_2} = 1 \qquad \frac{b}{w_2} = -0.2$$

$$w_1 = w_2 = 1 \quad , \quad b = -0.2$$

$$v_{(0,0)} = a_1 w_1 + a_2 w_2 + b$$
$$= 0 \times 1 + 0 \times 1 - 0.2 \leq 0 \Rightarrow y = 0$$

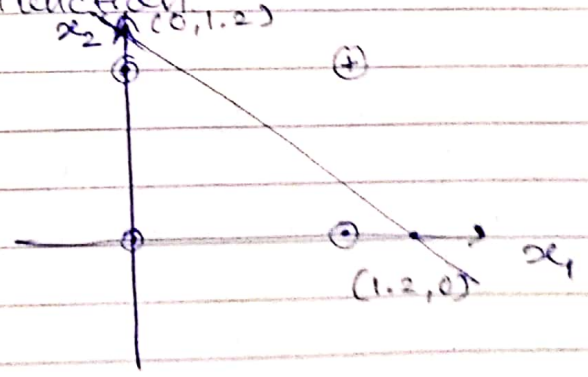
$$v_{(0,1)} = 0 \times 1 + 1 \times 1 - 0.2 > 0 \Rightarrow y = 1$$

$$v_{(1,0)} = 1 \times 1 + 0 \times 1 - 0.2 > 0 \Rightarrow y = 1$$

$$v_{(1,1)} = 1 \times 1 + 1 \times 1 - 0.2 > 0 \Rightarrow y = 1$$

* Example - 2 : AND Function

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



$m = -1$ $\quad w_1 = w_2 = 1 \quad , \quad b = -1.1$

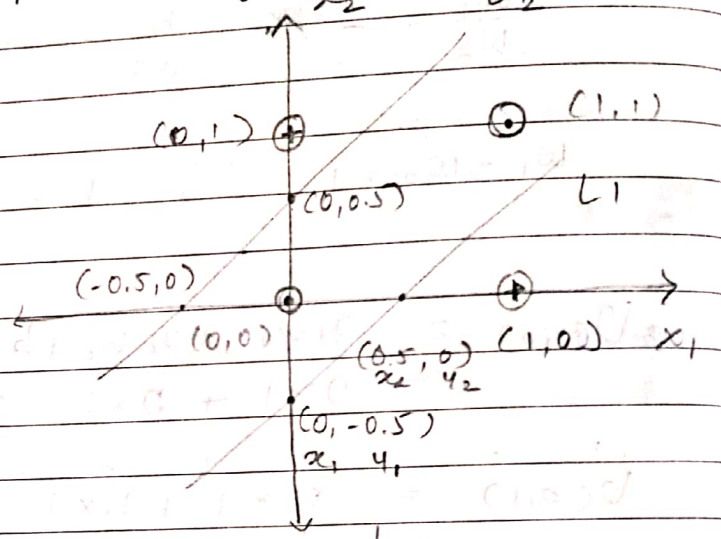
$c = 1.2$

Scanned with CamScanner

* Ex - 3 XOR function

→ XOR is nonlinear or linearly non separable problem as its patterns cannot be separated by a single line

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



⇒ For line 1 = Output - z_1 | $(x_1, y_1)(0, -0.5)$
$(x_2, y_2)(0.5, 0)$

$y = mx + c$

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{0 - (-0.5)}{0.5 - 0} = 1$$

$\boxed{m = 1}$

$c = y_1 - m x_1$
$= (-0.5) - 1 \times 0$
$\boxed{c = -0.5}$

$y = x - 0.5$
⇒ $\boxed{x_2 = x_1 - 0.5}$ — (1)

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

Scanned with CamScanner

$\frac{w_1}{w_2} = -1$ $\frac{b}{w_2} = -0.5$

$w_1 = 1$, $w_2 = -1$ $b = -0.5$

Activation function :

$y_{z_1} > 0 \Rightarrow z_1 = 1$

$y_{z_1} \leq 0 \Rightarrow z_1 = 0$

$(0,0) \rightarrow y_{z_1} = b + w_1 x_1 + w_2 x_2$

$= -0.5 + (+1) \times 0 + 1 \times 0 \quad < 0 \Rightarrow z_1 = 0$

$(0,1) \rightarrow -0.5 + (+1) \times 0 + -1 \times 1 \quad < 0 \Rightarrow z_1 = 0$

$(1,0) \rightarrow -0.5 + +1 \times 1 + (-1) \times 0 \quad > 0 \Rightarrow z_1 = 1$

$(1,1) \rightarrow -0.5 + +1 \times 1 + (-1) \times 1 \quad < 0 \Rightarrow z_1 = 0$

x_1	x_2	z_1
0	0	0
0	1	0
1	0	1
1	1	0

For Line 2 - Output z_2 - (x_1, y_1) $(-0.5, 0)$
(x_2, y_2) $(0, 0.5)$

$m = \frac{0.5 - 0}{0 - (-0.5)} = 1$

$C = 0.5$ $\rightarrow y = x + 0.5$

$x_2 = x_1 + 0.5$ — (1)

Scanned with CamScanner

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

$$\frac{w_1}{w_2} = -1 \qquad \frac{b}{w_2} = -0.5$$

$$w_1 = -1 \quad w_2 = 1 \quad b = -0.5$$

Activation $y_{z_2} > 0 \Rightarrow z_2 = 1$

$y_{z_2} \leq 0 \Rightarrow z_2 = 0$

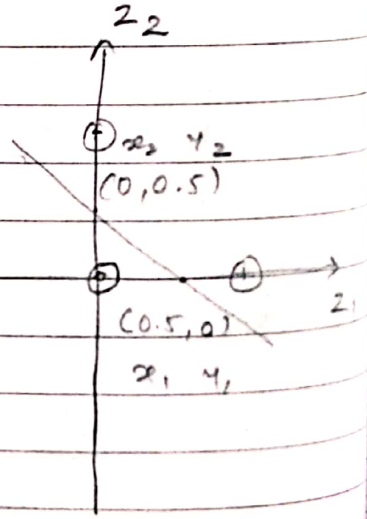
$(0,0) \rightarrow -0.5 + (-1)(0) + (1 \times 0) < 0 \Rightarrow z_2 = 0$

$(0,1) \rightarrow -0.5 + (-1)(0) + (1 \times 1) > 0 \Rightarrow z_2 = 1$

$(1,0) \rightarrow -0.5 + (-1) \times 1 + 1 \times 0 < 0 \Rightarrow z_2 = 0$

$(1,1) \rightarrow -0.5 + (-1 \times 1) + 1 \times 1 < 0 \Rightarrow z_2 = 0$

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



$$m = \frac{0.5 - 0}{0 - 0.5} = -1$$

$$c = -m x_1 - y_1$$

$$\boxed{c = 0.5}$$

Scanned with CamScanner

$$x_2 = -x_1 + 0.5$$

$$x_2 = -\frac{w_1}{w_2} \cdot x_1 - \frac{b}{w_2}$$

$$\frac{w_1}{w_2} = +1 \qquad \frac{b}{w_2} = -0.5$$

$$w_1 = w_2 = 1 \qquad b = -0.5$$

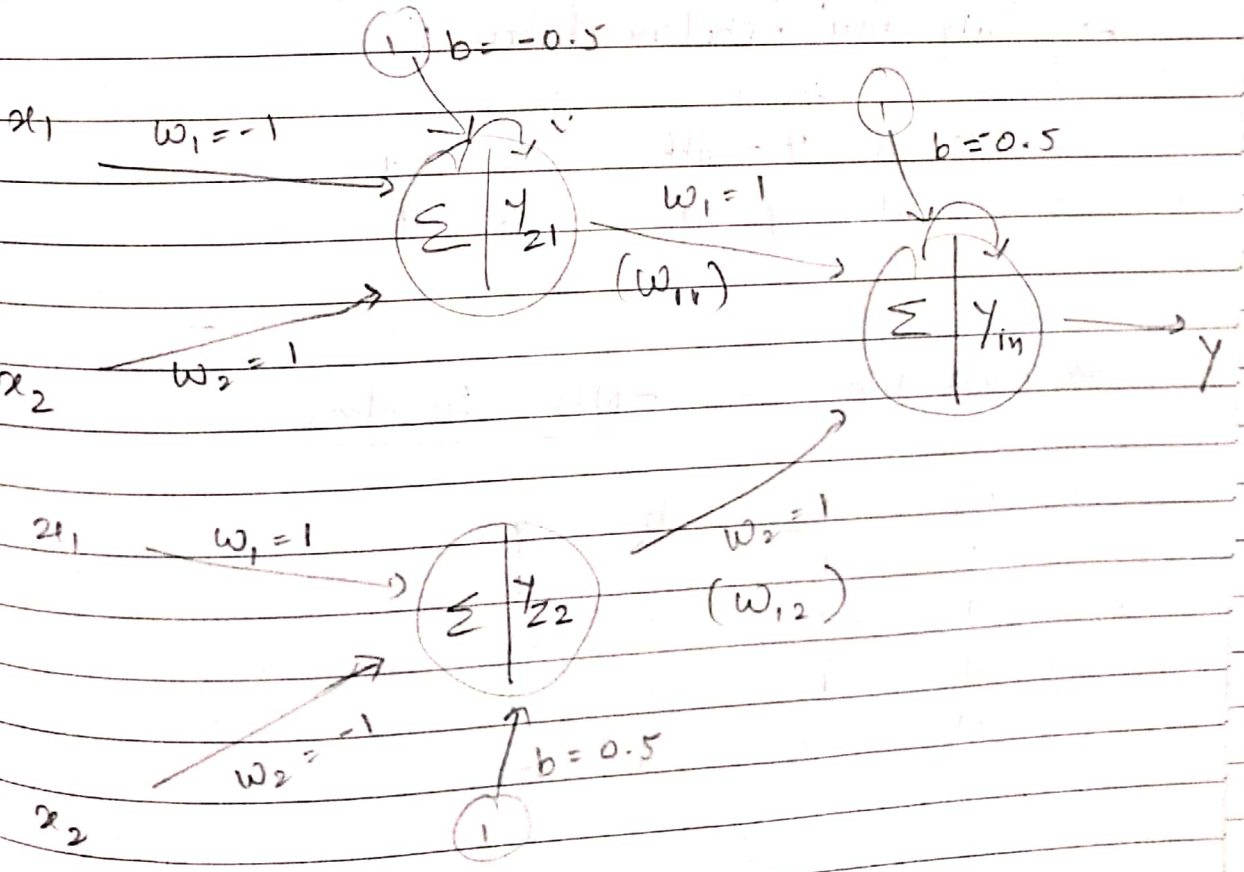
Activation function

$$y_{in} > 0 \Rightarrow y = 1$$
$$y_{in} \leq 0 \Rightarrow y = 0$$

$(0,0) \rightarrow -0.5 + 1\times0 + 1\times0 < 0 \Rightarrow y = 0$

$(0,1) \rightarrow -0.5 + 1\times0 + 1\times1 > 0 \Rightarrow y = 1$

$(1,0) \rightarrow -0.5 + 1\times1 + 1\times0 > 0 \Rightarrow y = 1$



Scanned with CamScanner

* Hebb Network

→ Synaptic Gap (brain)

→ Weight factor is found to increase proportionately to the product of the input and the learning signal.

→ If two interconnected neurons are 'on' simultaneously then weight associated with these neurons can be increased by the modification made in their synaptic gap.

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

→ Only for bipolar data :

	target	input
- if	y - off	x - on
- if	y - off	x - off

⇒ Example - AND Function

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

Scanned with CamScanner

Initially : $w_1 = w_2 = b = 0$

→ Input - 1 $(x_1\ x_2\ b)$ $(1\ 1\ 1)$ target (y) (1)

$\Delta b = y = 1$

$w_i(n) = w_i(0) + x_i y$ $\Delta w_1 = x_1 y = 1$

$\Delta w_2 = x_2 y = 1$

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

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

$b(n) = b(0) + y = 0 + 1 = 1$

→ Input - 2 $(x_1\ x_2\ b)$ $(1\ -1\ 1)$ $y = -1$ $\Delta b = -1$

$\Delta w_1 = -1$

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

$w_1(n) = 1 + 1 \times (-1) = 0$

$w_2(n) = 1 + (-1)(-1) = 2$

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

→ Input - 3 $(x_1\ x_2\ b)$ $(-1\ 1\ 1)$ $y = -1$ $\Delta b = -1$

$\Delta w_1 = 1$

$[w_1\ w_2\ b] = [0\ 2\ 0]$ $\Delta w_2 = -1$

$w_1(n) = w(0) + x_1(y)$

$w_1(n) = 0 + (-1)(-1) = 1$

$w_2(n) = 2 + (1)(-1) = 1$

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

→ Input - 4 $(-1\ -1\ 1)$ $y = -1$ $\Delta b = -1$

$\Delta w_1 = 1$

$[w_1\ w_2\ b]$ $[1\ 1\ -1]$ $\Delta w_2 = 1$

$w_1(n) = 1 + (-1)(-1) = 2$

$w_2(n) = 1 + (-1)(-1) = 2$

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

Scanned with CamScanner

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$

I4 → $w_1 = w_2 = 2 \qquad b = -2$

$x_2 = -x_1 + 1$ — (1)

I3 → $w_1 = 1 \quad w_2 = 1 \quad b = -1$

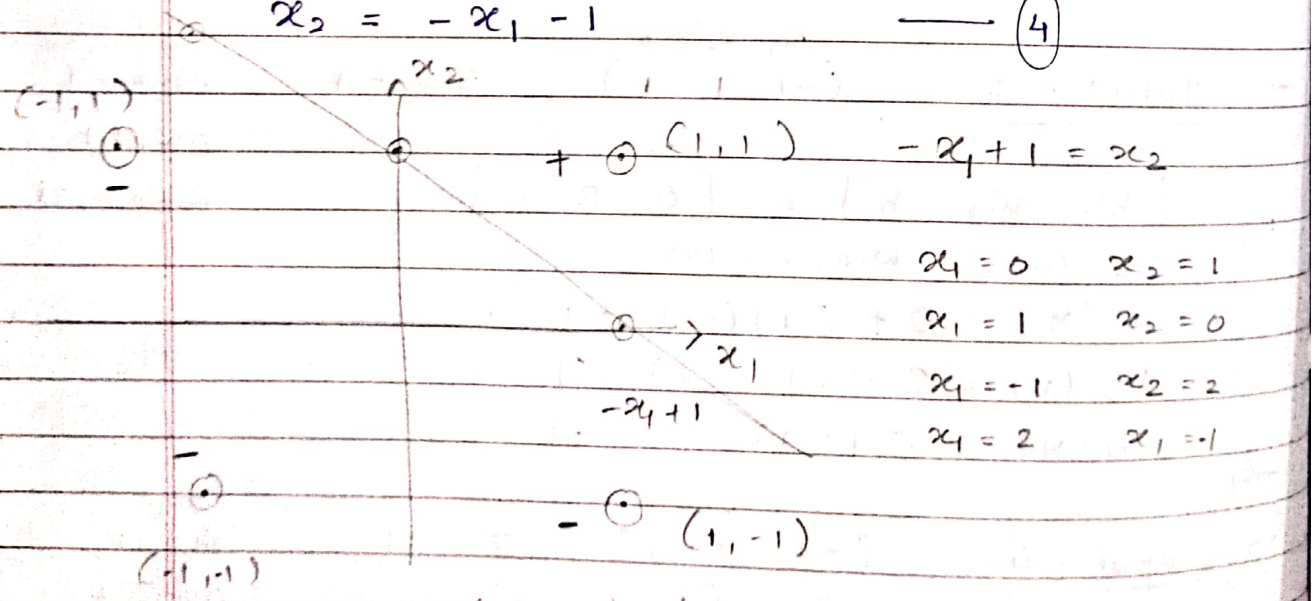
$x_2 = -x_1 + 1$ — (2)

I2 → $w_1 = 0 \quad w_2 = 2 \quad b = 0$

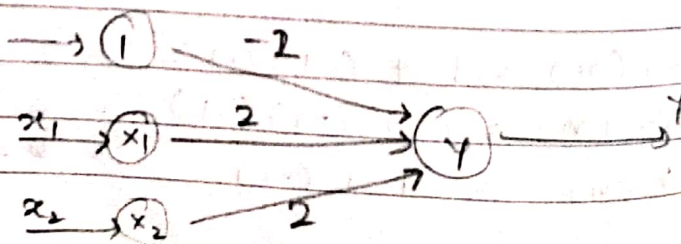
$x_2 = 0$ — (3)

I1 → $w_1 = 1 \quad w_2 = 1 \quad b = 1$

$x_2 = -x_1 - 1$ — (4)



Hebb Network



Scanned with CamScanner

2) OR Function (bipolar)

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

→ Initial weights $w_1 = w_2 = 0$, bias = 0

Input - 1 $[x_1\ x_2\ b] = [1\ 1\ 0]$ $\Delta b = 1$, $\Delta w_1 = \Delta w_2 = 1$

$w_1(n) = w_1(0) + x_1 y\ (\Delta w_1) = 1$
$w_2(n) = w_2(0) + x_2 y\ (\Delta w_2) = 1$
$b(n) = b(0) + y\ (\Delta b) = 1$

Input - 2 $[1\ -1\ 1]$, $w_1 = 1$, $w_2 = 1$, $b = 1$

$\Delta w_1 = 1 \times 1 = 1$ $\quad w_1(n) = w_1(0) + \Delta w_1 = 2$
$\Delta w_2 = -1 \times 1 = -1$ $\quad w_2(n) = w_2(0) + \Delta w_2 = 0$
$\Delta b = 1 = 1$ $\quad b(n) = b(0) + \Delta b = 2$

Input - 3 $[-1\ 1\ 1]$ $w_1 = 2$ $w_2 = 0$, $b = 2$

$\Delta w_1 = (-1) \times 1 = -1$ $\quad w_1(n) = w_1(0) + \Delta w_1 = 1$
$\Delta w_2 = 1 \times 1 = 1$ $\quad w_2(n) = w_2(0) + \Delta w_2 = 1$
$\Delta b = 1 = 1$ $\quad b(n) = b(0) + \Delta b = 3$

Input - 4 $[-1\ -1\ 1]$ $w_1 = 1$ $w_2 = 1$ $b = 3$ $y = -1$

$\Delta w_1 = -1 \times -1 = 1$ $\quad w_1(n) = 1 + 1 = 2$
$\Delta w_2 = -1 \times -1 = +1$ $\quad w_2(n) = 1 + (+1) = 2$
$\Delta b = -1$ $\quad b(n) = 3 + (-1) = 2$

Scanned with CamScanner

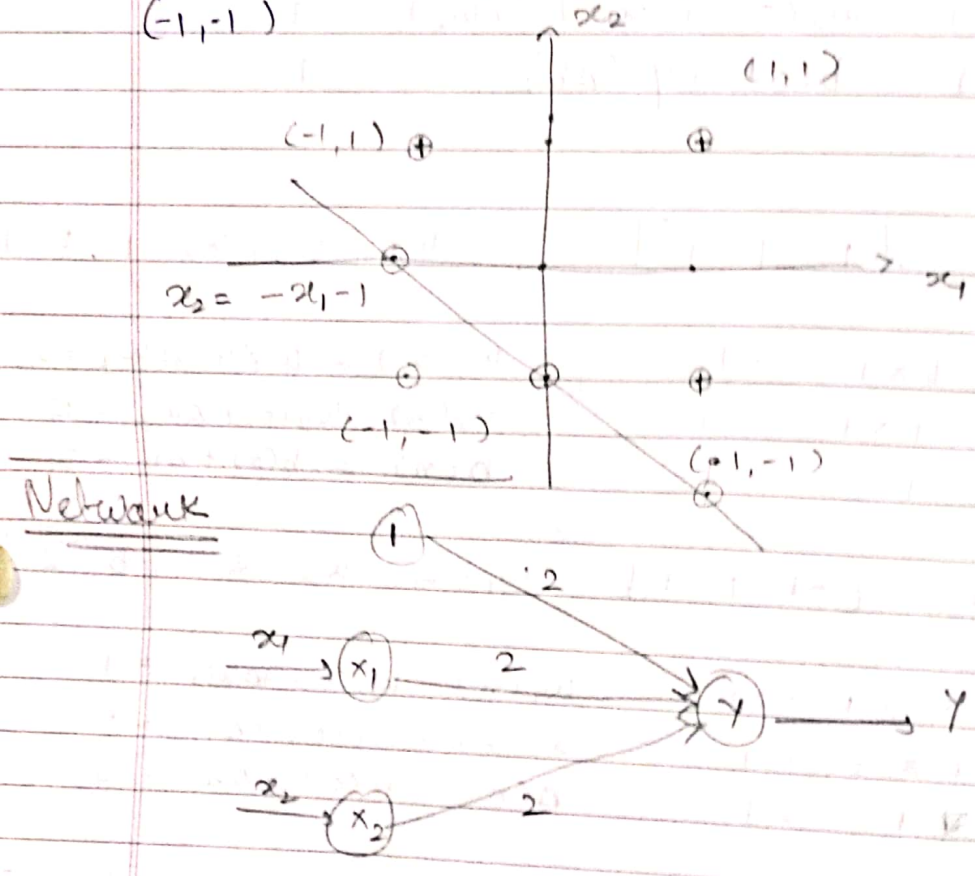
$$x_2 = -\frac{w_1}{w_2} x_1 + \frac{b}{w_2}$$

L1 : $x_2 = -x_1 - 1$
(1, 1)

L2 $x_2 = -\frac{2}{0} x_1 - \frac{2}{0}$
(1, -1)

L3 $x_2 = -x_1 - 3$
(-1, 1)

L4 $x_2 = -x_1 - 1$
(-1, -1)



Network

Scanned with CamScanner

3) Hebb rule for following pattern classification

'+' Symbol → Value 1
Empty → Value 0

+	+	+
	+	
+	+	+

'I'

+	+	+
+		+
+	+	+

'O'

→ I belongs to members of class

Pattern	Inputs									target
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	y
I	1	1	1	-1	1	-1	1	1	1	1
O	1	1	1	1	-1	1	1	1	1	-1

⇒ Single Layer Network, Nine input, One bias

Initial weights & bias = 0

Case-1 : Pattern 'I'

→ $\Delta w_i = x_i y$ $i = 1$ to 9

$\Delta w_1 = 1 \times 1 = 1$
$\Delta w_2 = 1 \times 1 = 1$
$\Delta w_3 = 1 \times 1 = 1$
$\Delta w_4 = -1 \times 1 = -1$
$\Delta w_5 = 1 \times 1 = 1$
$\Delta w_6 = -1 \times 1 = -1$
$\Delta w_7 = 1 \times 1 = 1$
$\Delta w_8 = 1 \times 1 = 1$
$\Delta w_9 = 1 \times 1 = 1$
$\Delta b = y = 1$

$w_i(n) = w_i(0) + \Delta w_i$

$w_1(n) = 0 + 1 = 1$
$w_2(n) = 0 + 1 = 1$
$w_3(n) = 0 + 1 = 1$
$w_4(n) = -1$
$w_5(n) = 1$
$w_6(n) = -1$
$w_7(n) = 1$
$w_8(n) = 1$
$w_9(n) = 1$
$b(n) = 1$

Scanned with CamScanner

Case - 2 : Pattern '0'

$W_1 = [1\ 1\ 1\ -1\ 1\ -1\ 1\ 1\ 1\ 1]$ $\quad y = -1$

$W_1(n) = W_1(0) + x_1 y$

$W_1(n) = 1 + 1 \times -1 = 0$ $\quad W_6(n) = -1 + 1 \times -1 = -2$

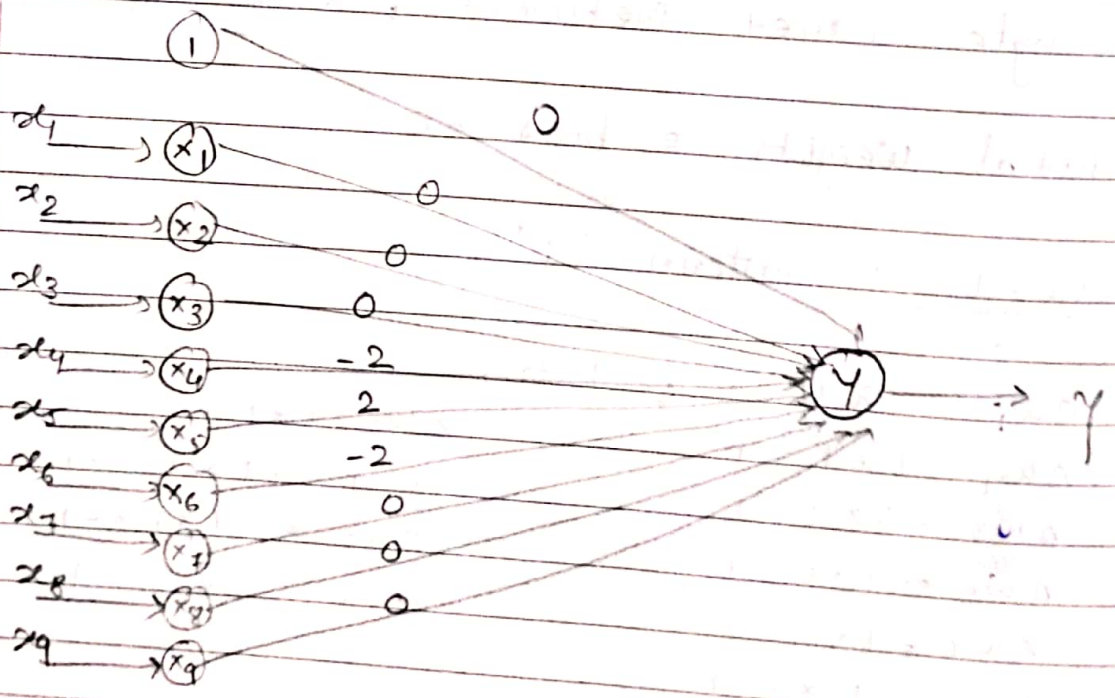
$W_2(n) = 1 + 1 \times -1 = 0$ $\quad W_7(n) = 1 + 1 \times -1 = 0$

$W_3(n) = 1 + 1 \times -1 = 0$ $\quad W_8(n) = 1 + 1 \times -1 = 0$

$W_4(n) = -1 + 1 \times -1 = -2$ $\quad W_9(n) = 1 + 1 \times -1 = 0$

$W_5(n) = 1 + -1 \times -1 = 2$ $\quad b(n) = 1 + 1 \times -1 = 0$

$W_n = [0\ 0\ 0\ -2\ 2\ -2\ 0\ 0\ 0\ 0]$



Scanned with CamScanner

* McCulloch - Pitts Neuron : (M-P Neuron)

→ Earliest neural network - 1943
→ Neurons connected by directed weighted paths
→ Activation is binary - Fire or not fire
→ Excitatory weights - positive - Same weights
→ Inhibitory weights - Negative
→ Threshold - If net input to the neuron is greater than the threshold then neuron fires.

Activation Function :

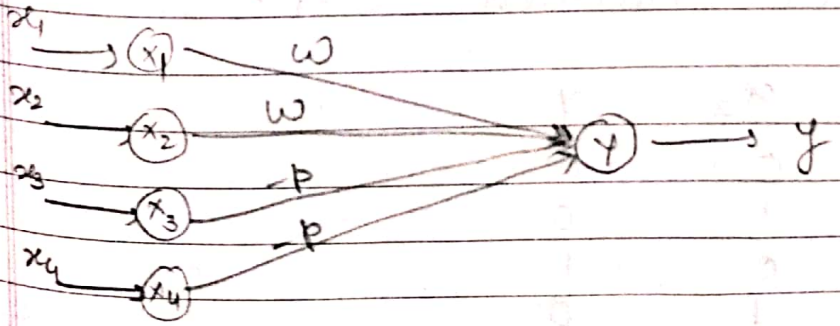
$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq \theta \\ 0 & \text{if } y_{in} < \theta \end{cases}$$

For inhibition to be absolute, threshold should

$$\theta \geq nw - p$$

n - no. of neurons
w - Magnitude of +ve weights
p - Magnitude of -ve weights

→ No training algorithms
→ Analysis - to determine values of weights and threshold.



Scanned with CamScanner

(1) Ex-1 AND Function (binary data

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

→ Output is high if both inputs are high

Let's take $w_1 = 1$ $w_2 = 1$

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

[0, 0] → $y_{in} = 0 + 0 = 0$
[0, 1] → $y_{in} = 0 + 1 = 1$
[1, 0] → $y_{in} = 1 + 0 = 1$
[1, 1] → $y_{in} = 1 + 1 = 2$ → $\theta =$

$\theta \geq nw - p$

$2 \times 1 - 0$

$\underline{\theta \geq 2}$

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

(2) AND NOT Function ($x_1 \bar{x}_2$)

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

Scanned with CamScanner

→ Case-1 $w_1 = 1$ $w_2 = 1$

[0,0] → $y_{in} = 0 + 0 = 0$
[0,1] → $= 0 + 1 = 1$
[1,0] → $= 1 + 0 = 1$
[1,1] → $= 1 + 1 = 2$

→ It is not possible to fire the neuron for input (1,0) Only. These weights are not Suitable.

→ Case-2 $w_1 = 1$ $w_2 = -1$

[0,0] → $y_{in} = 0 + 0 = 0$
[0,1] $= 0 + (-1) = -1$
[1,0] $= 1 + 0 = 1$ ← ✓
[1,1] $= 1 - 1 = 0$

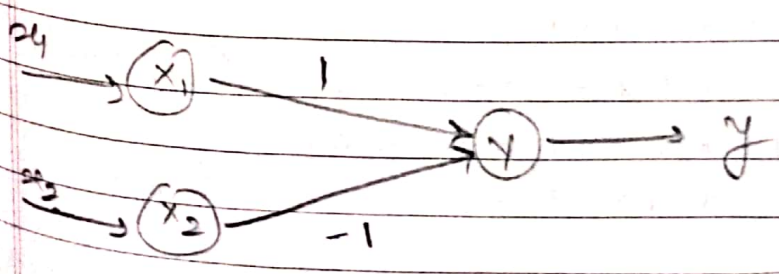
Output is high if first input is high and second input is low

$\theta \geq nw - p$

$2 \times 1 - 1$

$\theta \geq 1$

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



Scanned with CamScanner

(3) Ex-3 XOR Function

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

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

$\downarrow \qquad \downarrow$

$z_1 \qquad z_2$

$y = z_1 \text{ (OR) } z_2$

→ Single layer Net is not sufficient

(1) $z_1 = x_1 \bar{x}_2$ $\qquad w_{11} = w$

x_1	x_2	z_1
0	0	0
0	1	0
1	0	1
1	1	0

Case-1 → $w_{11} = w_{21} = 1$

(0,0) → $x_1 w_1 + x_2 w_2 = 0$
(0,1) → $0 \times 1 + 1 \times 1 = 1$
(1,0) → $1 \times 1 + 0 \times 1 = 1$
(1,1) → $1 \times 1 + 1 \times 1 = 2$

Case-2 → $w_{11} = 1 \quad w_{21} = -1$

(0,0) → $0 \times 1 + 0 \times -1 = 0$
(0,1) → $0 \times 1 + 1 \times -1 = -1$
(1,0) → $1 \times 1 + 0 \times -1 = 1$ ⟸
(1,1) → $1 \times 1 + 1 \times -1 = 0$

$\theta \geq nw - 1$

$2 \times 1 - 1$

$\theta \geq 1$

Scanned with CamScanner

(2) $Z_2 = \bar{x}_1 x_2$

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

Case-1 : $w_{12} = w_{22} = 1$

$(0,0) \rightarrow x_1 w_{12} + x_2 w_{22} = 0 \times 1 + 0 \times 1 = 1$
$(0,1) \rightarrow 1$
$(1,0) \rightarrow 1$
$(1,1) \rightarrow 2$

Case-2 : $w_{12} = -1 \quad w_{22} = 1$

$(0,0) \rightarrow 0$
$(0,1) \rightarrow 1 \quad \Leftarrow \quad \theta \geq 1$
$(1,0) \rightarrow -1$
$(1,1) \rightarrow 0$

(3) $Y = Z_1 \text{ OR } Z_2$

x_1	x_2	Y	z_1	z_2
0	0	0	0	0
0	1	1	0	1
1	0	1	1	0
1	1	0	0	0

Case-1 $v_1 = v_2 = 1$

$(0,0) \rightarrow 0 \times 1 + 0 \times 1 = 0$
$(0,1) \rightarrow 0 \times 1 + 1 \times 1 = 1$ } $\Leftarrow$
$(1,0) \rightarrow 1 \times 1 + 0 \times 1 = 1$
$(1,1) \rightarrow 0 \times 1 + 0 \times 1 = 0$

Scanned with CamScanner

Supervised Learning Networks

* Perceptron Networks:

→ Single layer feed-forward

→ Three units :
Sensory unit (Input)
Associator unit (Hidden) - Feature Predicates
Response unit (output) - Pattern Recognizers

→ Sensory - Associator - Fixed weights (1, 0, -1)
- Binary activation (0, 1)

→ Response unit ∴ Activation 0, 1, -1
→ Binary step - threshold θ

→ Associator - Response → Binary output signal

→ Output - $y = f(y_{in})$

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

→ Weight updation bet^n associator & Response

→ Error Calculation - comparison bet^n target & output

→ Weight updation - If $(y \neq t)$ then

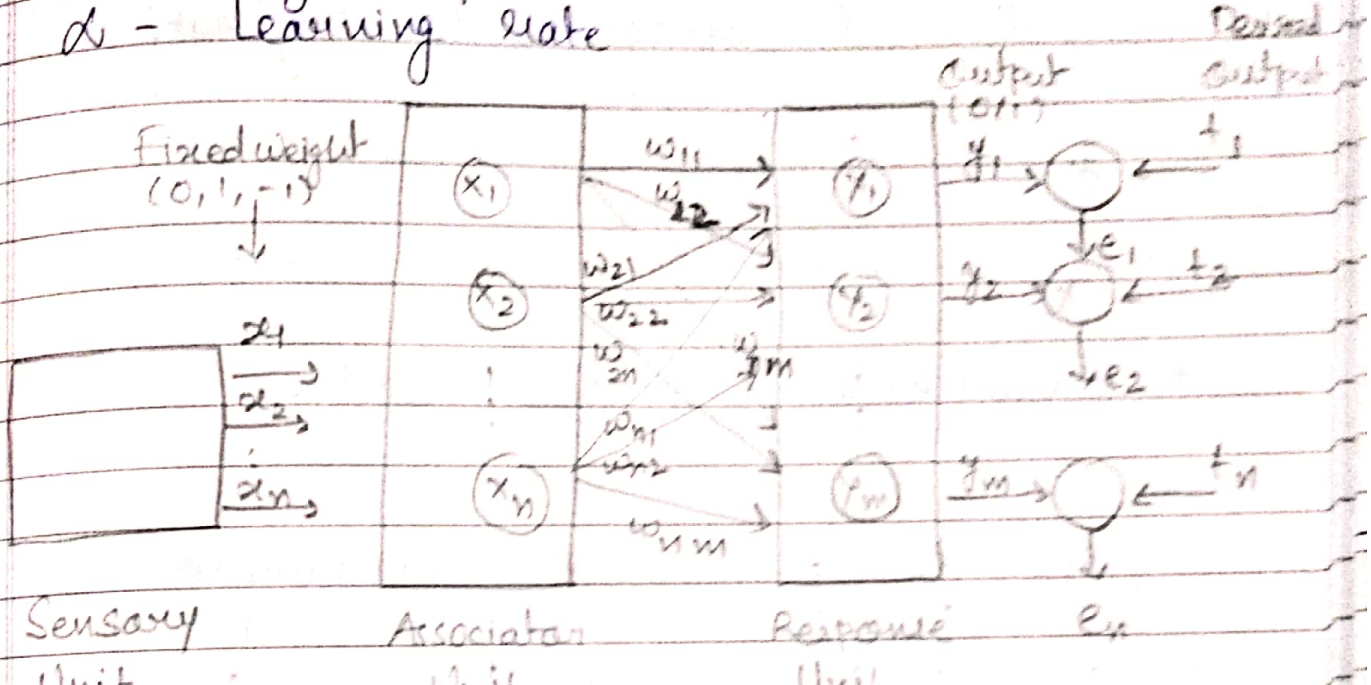
$w_i(new) = w_i(old) + \alpha t x_i$

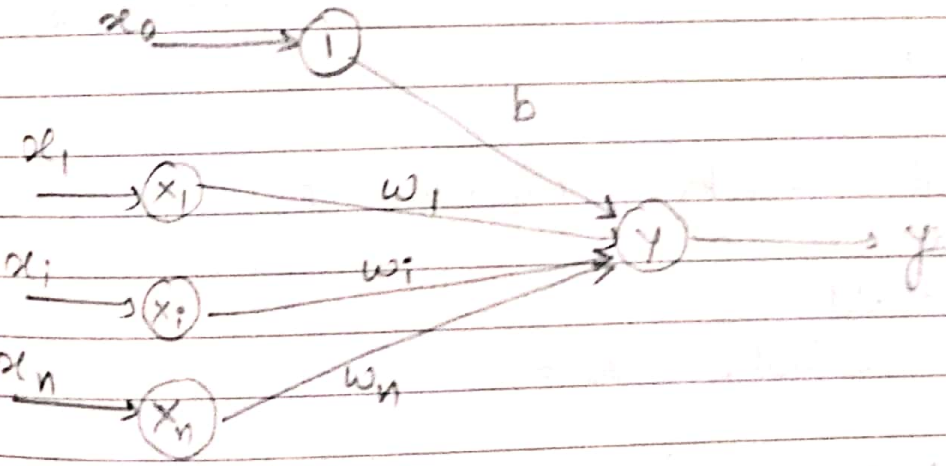
$b(new) = b(old) + \alpha t$

Scanned with CamScanner

t - target output (1, -1)

α - Learning rate





Goal : Classify the input pattern as member or not member to a particular class

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

Multiple output : $y_{inj} = b_j + \sum_{i=1}^{n} x_i w_{ij}$

output unit - j - 1 to m

⇒ Algorithms are not sensitive to initial values of weights & learning rate.

* Example - 1 AND Function

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

Activation Function

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

⇒ Initial Weights - $w_1 = w_2 = 0$
⇒ Bias $b = 0$
⇒ Threshold $= 0$
⇒ Learning Rate $\alpha = 1$

Epoch-1 | Input 1 : $x_1 = 1$, $x_2 = 1$, $t = 1$

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

$\because y = 0 \Rightarrow y_{in} = 0$

Check → $y = t$, $t = 1$ $y = 0 \Rightarrow y \neq t$

So Weight updation

Scanned with CamScanner

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

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

Here, $\Delta w_i = \alpha t x_i$

$\Delta b = \alpha t$

$\Delta w_1 = 1 \times 1 \times 1 = 1 \Rightarrow w_1(new) = 0 + 1 = 1$

$\Delta w_2 = 1 \times 1 \times 1 = 1 \Rightarrow w_2(new) = 0 + 1 = 1$

$\Delta b = 1 \times 1 = 1 \Rightarrow b(new) = 0 + 1 = 1$

$\Rightarrow w_1 = 1$, $w_2 = 1$, $b = 1$ - These are final weights after first input pattern is presented.

$\Rightarrow$ The same process is repeated for all the input patterns. Process is stopped when all the targets became equal to calculated output.

$\Rightarrow$ When $y = t$, weight changes Δw_i will not perform and $\Delta w_i = \Delta b = 0$.

Input 2 : $x_1 = 1$ $x_2 = -1$ $t = -1$

$w_1 = 1$ $w_2 = 1$ $b = 1$

$y_{in} = 1 + 1 + (-1) = 1$

$y = 1$ & $t = -1$ $\therefore y \neq t$

$\Delta w_1 = 1 \times (-1)(1) = -1$ $w_1 = 1 - 1 = 0$

$\Delta w_2 = 1 \times (-1)(-1) = 1$ $w_2 = 1 + 1 = 2$

$\Delta b = 1 \times (-1) = -1$ $b = 1 - 1 = 0$

Scanned with CamScanner

Input-3 $x_1 = -1$ $x_2 = 1$ $t = -1$

$w_1 = 0$ $w_2 = 2$ $b = 0$

$y_{in} = 0 + (-1)\times 0 + 1 \times 2 = 2 > 0$

$y = 1$ & $t = -1$ $\Rightarrow$ $y \neq t$

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

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

$\Delta b = 1 \times (-1) = -1$ $b = -1$

Input-4 $x_1 = -1$ $x_2 = -1$ $t = -1$

$w_1 = 1$ $w_2 = 1$ $b = -1$

$y_{in} = -1 + (-1)(1) + (-1)(1) = -3 < 0$

$y = -1$, $t = -1$ $\Rightarrow$ $y = t$

No weight updation $\Delta w_i = \Delta b = 0$

$w_1 = 1$ $w_2 = 1$ $b = -1$, Same weights

EPOCH-2 $w_1 = 1$ $w_2 = 1$ $b = -1$ $\theta = 0$ $\alpha = 1$

Input-1 $x_1 = 1$ $x_2 = 1$ $t = 1$

$y_{in} = -1 + 1 \times 1 + 1 \times 1 = 1 > 0$

$y = 1$, $t = 1$ $\Rightarrow$ $y = t$ $\Rightarrow$ No weight change

$\Rightarrow$ $w_1 = 1$ $w_2 = 1$ $b = -1$

Scanned with CamScanner

classmate
Date
Page

Input - 2 $x_1 = 1 \quad x_2 = -1 \quad t = -1$

$y_{in} = -1 + (1)(1) + (-1)(1) = -1 < 0 \Rightarrow y = -1$

$y = -1 \;, t = -1 \Rightarrow y = t$ no weight change

Input - 3 $x_1 = -1 \quad x_2 = 1 \quad t = -1$

$y_{in} = -1 + (-1)(1) + (1)(1) = -1 < 0 \Rightarrow y = -1$

$y = -1 \;, t = -1 \Rightarrow y = t$ No weight change

Input - 4 $x_1 = -1 \quad x_2 = -1 \quad t = -1$

$y_{in} = -1 + (-1) \times 1 + (1) \times 1 = -3 < 0 \Rightarrow y = -1$

$y = -1 \;, t = -1 \Rightarrow y = t$ No weight change

Final weights & Bias $w_1 = 1 \quad w_2 = 1 \quad b = -1$

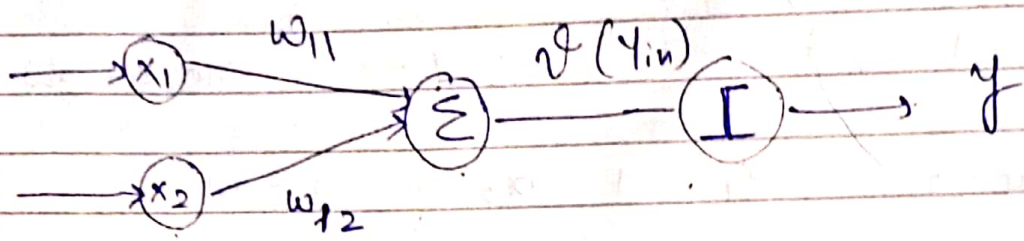
$$x_2 = \frac{-w_1}{w_2} x_1 - \frac{b}{w_2}$$

$$\boxed{x_2 = -x_1 + 1}$$

Same for OR, ANDNOT, NAND

Scanned with CamScanner

* Finding the equation of Δw_{ji} (Error) in neural network.



Take $y_{in} = v = \sum x_i w_{ji} \Rightarrow \frac{\partial v}{\partial w_{ji}} = x_i$

$y = f(v) \Rightarrow \frac{\partial y}{\partial v} = f'(v)$

$E = (t-y)^2 \Rightarrow \frac{\partial E}{\partial y} = -2(t-y)$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y} \times \frac{\partial y}{\partial v} \times \frac{\partial v}{\partial w_{ji}}$$

$$\Delta w_{ji} = -2(t-y) \times f'(v) \times x_i$$

So,

$$w_{ji}(\text{new}) = w_{ji}(\text{old}) + \Delta w_{ji}$$

Scanned with CamScanner

Insert Delete

classmate Date Page

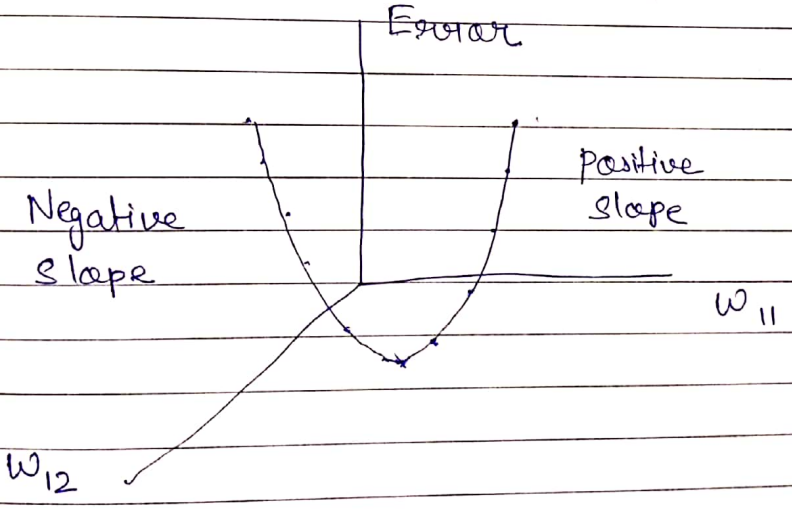
* Gradient Descent Method (Steepest Descent)

→ A gradient measures how much an output function changes if you change in input.

→ Gradient is partial derivative with respect to its learning

→ Higher the gradient, steeper the slope and faster the model.

→ But if slope is 0, model stops learning.



⇒ Widro - Hoff Rule / Delta Rule

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

If initial weights are taken as 0

$$y_{in} = 0 \Rightarrow y = 0$$

$$\Delta w_{ji} = \alpha (t - y) x_{ji}$$

Scanned with CamScanner

$$\therefore \boxed{\Delta w_{ji} = \alpha t x_{ji}}$$

For Single output $j = 1$

$$\boxed{\Delta w_i = \alpha t x_i}$$

Scanned with CamScanner

* Adaptive Linear Neuron (ADALINE)

→ The units with linear activation function are called linear units.

→ A network with Single linear unit is called an Adaline.

→ Uses bipolar activation, weights and bias are adjustable

→ It has only one output unit.

→ It works & trained using delta rule or Least Mean Square (LMS) rule or Widrow-Hoff rule.

→ Delta rule is derived from gradient descent method, its training continues forever converging only asymptotically to the solution.

→ Delta rule updates weights to minimize the difference between net input to the output unit and the target value & Minimize the error.

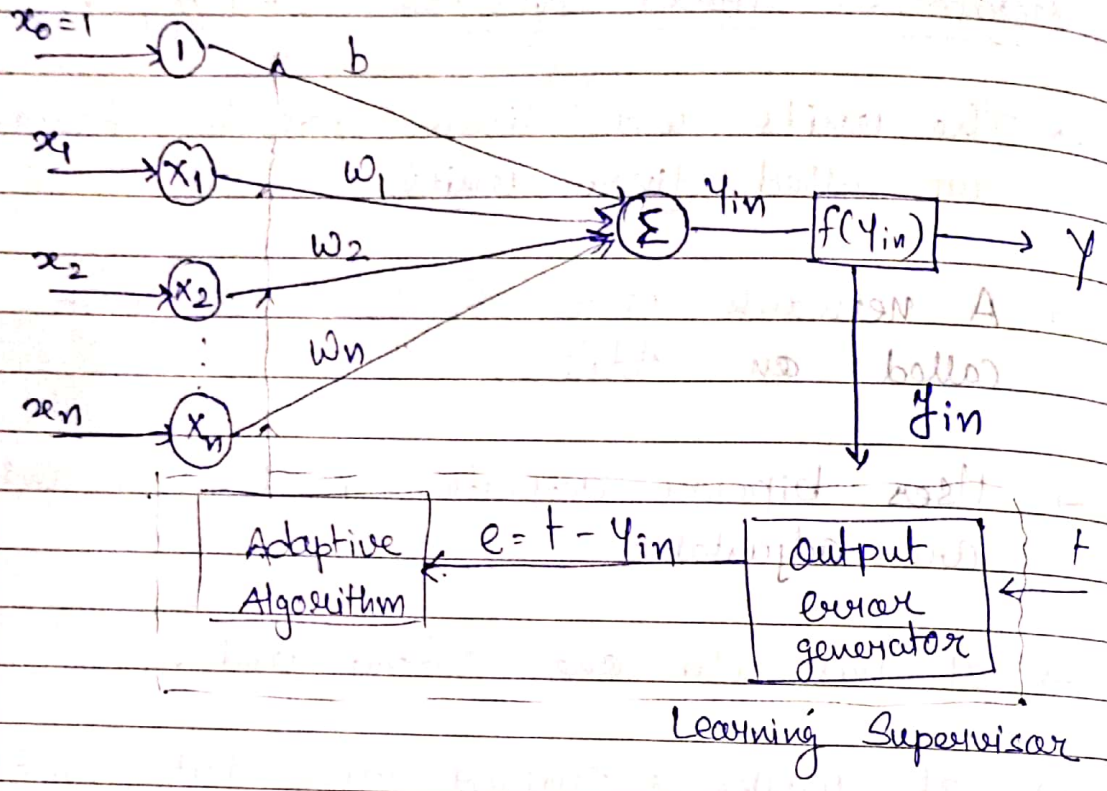
→ Delta rule for adjusting the weight of ith pattern

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

→ For several output units adjusting weight from ith input unit to jth output unit

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

Scanned with CamScanner



Learning Supervisor

* Example - 1 - OR Function :

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

Step-1 Initialize weights, bias & learning rate random values.

* Input - 1

i) $w_1 = w_2 = b = 0.1$, $\alpha = 0.1$

ii) Net input $y_{in} = b + \sum x_i w_i$

$= b + x_1 w_1 + x_2 w_2$

$= 0.1 + 1 \times 0.1 + 1 \times 0.1$

$y_{in} = 0.3$

Scanned with CamScanner

iii) Now Compute $(t - y_{in}) = 1 - 0.3 = 0.7$

Update the weights, $w_i(new) = w_i(old) + \alpha(t - y_{in})x_i$

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

$\Delta w_1 = \alpha (t - y_{in}) x_1 = 0.1 \times (0.7) \times 1 = 0.07$

$\Delta w_2 = \alpha (t - y_{in}) x_2 = 0.1 \times (0.7) \times 1 = 0.07$

$\Delta b = \alpha (t - y_{in}) = 0.1 (0.7) = 0.07$

v)

$w_1(new) = w_1(old) + \Delta w_1 = 0.1 + 0.07 = 0.17$

$w_2(new) = w_2(old) + \Delta w_2 = 0.1 + 0.07 = 0.17$

$b(new) = b(old) + \Delta b = 0.1 + 0.07 = 0.17$

vi) Error $E = (t - y_{in})^2 = (0.7)^2 = 0.49$

* Input-2

$w_1 = 0.17 \quad w_2 = 0.17 \quad b = 0.17 \quad t = 1 \quad \alpha = 0.1 \quad x_1 = 1 \quad x_2 = -1$

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

$= 0.17 + 1 \times 0.17 + (-1) \times 0.17$

$= 0.17$

$t - y_{in} = 1 - 0.17 = 0.83$

$\Delta w_1 = \alpha (t - y_{in}) x_1 = 0.1 \times 0.83 \times 1 = 0.083$

$\Delta w_2 = 0.1 \times 0.83 \times (-1) = -0.083$

$\Delta b = 0.1 \times 0.83 = 0.083$

$w_1(new) = 0.17 + 0.083 = 0.253$

$w_2(new) = 0.17 + (-0.083) = 0.087$

$b(new) = 0.17 + 0.083 = 0.253$

$E = (t - y_{in})^2 = 0.83^2 = 0.6889$

Scanned with CamScanner

* Multiple Adaptive Linear Neurons (MADALINE)

→ Many Adalines in parallel with single output unit whose value is based on certain selection rule

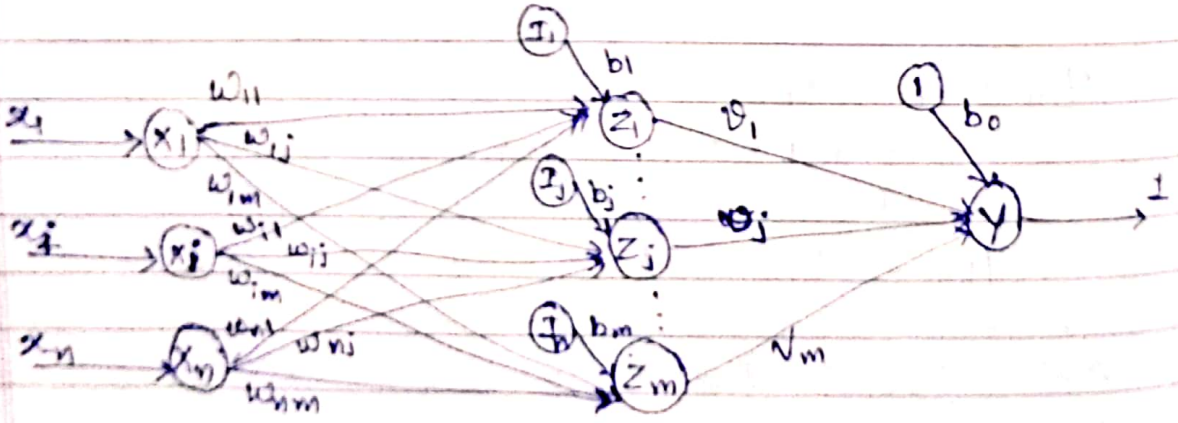
→ Majority Vote Rule - True or False
→ AND Rule - True iff both inputs True

→ Weights bet^n Adaline & Madaline layer are fixed, positive & same value (equal value)

→ Weights bet^n Input & Adaline are adjusted

• Architecture :

- n - input layer units
- m - Adaline layer units
- 1 - Madaline layer unit (output)



→ The weights $v_1, v_2 \ldots v_m$ & bias b_0 that enter into output unit Y are determined so that Response of unit Y is 1

$$v_1 = v_2 = \ldots v_m = \frac{1}{2} \quad \& \quad b_0 = \frac{1}{2}$$

Scanned with CamScanner

→ Activation for Adaline & Madaline

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

→ Steps:

1) Initialize the weights w_{ij} and v_i. Here weights bet^n hidden & output layer are fixed (v_i).

2) Calculate net input to hidden units

$$z_{inj} = b_j + \sum_{i=1}^{n} x_i w_{ij} \quad , \; j = 1 \text{ to } m$$

3) Calculate output of each hidden unit

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

4) Calculate the net input entering into output unit

$$y_{in} = b_0 + \sum_{J=1}^{m} z_j v_j$$

5) Calculate the output y by applying activation function over net input y_{in}

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

Scanned with CamScanner

6) Calculate error and update weights

i) → If $t = y$ - no updation

ii) If $t \neq y$ and $t = +1$,

- Updates weights on Z_j, whose net input is close to 0 ($\approx 0.xxx$)

$$b_j(new) = b_j(old) + \alpha(1 - z_{inj})$$

$$w_{ij}(new) = w_{ij}(old) + \alpha(1 - z_{inj})\, x_i$$

iii) If $t \neq y$ and $t = -1$,

Update weights on Z_k, whose net input is +ve

$$w_{ik}(new) = w_{ik}(old) + \alpha(-1 - z_{ink})\, x_i$$

$$b_k(new) = b_k(old) + \alpha(-1 - z_{ink})$$

7) Stopping Condition - Weight Converges

Scanned with CamScanner

Example - XOR Function

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

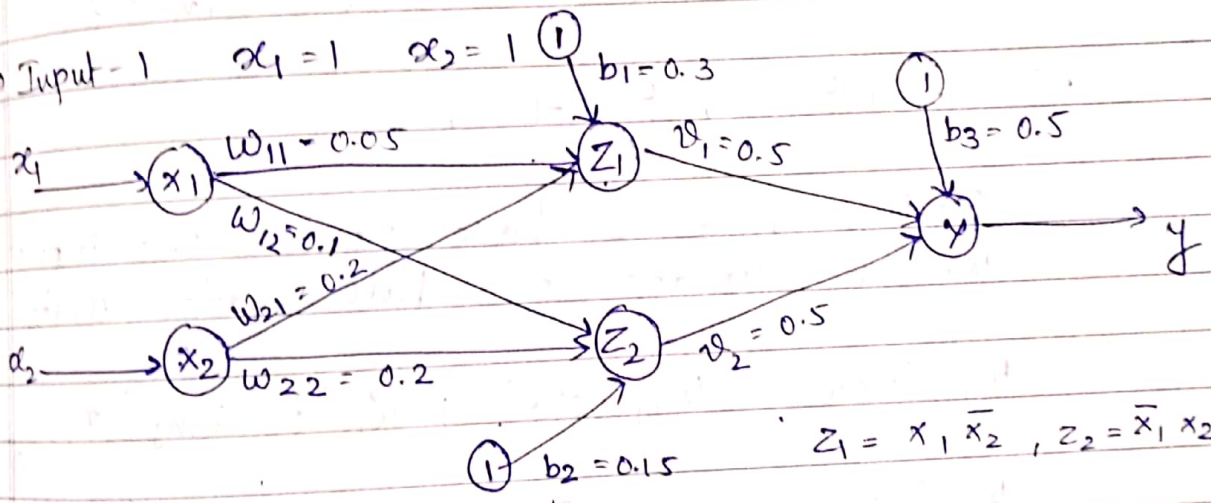
Initial values: $\alpha = 0.5$

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

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

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

→ Input - 1 $x_1 = 1$ $x_2 = 1$



$Z_1 = x_1 \bar{x}_2$, $Z_2 = \bar{x}_1 x_2$

i) Net input Calculation for Adaline layer

$$Z_{in1} = b_1 + x_1 w_{11} + x_2 w_{21} = 0.55$$

$$Z_{in2} = b_2 + x_1 w_{21} + x_2 w_{22} = 0.45$$

ii) Output of Adaline layer Neurons - Activation Fun"

$$Z_1 = f(Z_{in1}) = f(0.55) = 1 \quad (\because Z_{in1} > 0)$$

$$Z_2 = f(Z_{in2}) = f(0.45) = 1 \quad (\because Z_{in2} > 0)$$

Scanned with CamScanner

iii) Output goes into Madaline, net input

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

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

$= 1.5$

iv) Output of Madaline $y = f(y_{in})$

$y = f(1.5) = 1 \quad (\because y_{in} > 0)$

v) $y \neq t \quad , t = -1 \quad ,$

Both z_1 & z_2 have positive net input, weight updates on both hidden units

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

$w_{11}(new) = w_{11}(old) + \alpha (t - z_{in_1}) x_1 = -0.725$

$w_{12}(new) = w_{12}(old) + \alpha (t - z_{in_2}) x_1 = -0.625$

$w_{21}(new) = w_{21}(old) + \alpha (t - z_{in_1}) x_2 = -0.575$

$w_{22}(new) = w_{22}(old) + \alpha (t - z_{in_2}) x_2 = 0.525$

$b_1(new) = b_1(old) + \alpha (t - z_{in_1}) = -0.475$

$b_2(new) = b_2(old) + \alpha (t - z_{in_2}) = -0.575$

Scanned with CamScanner

Input-2 $x_1 = 1$ $x_2 = -1$ $t = 1$

$z_{in1} = b_1 + x_1 w_{11} + x_2 w_{21} = -0.625$

$z_{in2} = b_2 + x_1 w_{12} + x_2 w_{22} = -0.675$

$z_1 = f(-0.625) = -1$

$z_2 = f(-0.675) = -1$

$y_{in} = b_3 + z_1 v_1 + z_2 v_2 = 0.5 + (-0.5) + (-0.5) = -0.5$

$y = f(y_{in}) = -1$, $t = 1$

$\Rightarrow y \neq t$ & $t = 1$, Values closest to 0 should be updated, therefore w_{ij} both will be updated.

Scanned with CamScanner

* Back - Propagation Network :

→ Multilayer feed-forward networks
→ Processing elements with continuous differentiable activation functions.
→ Weight update algo - gradient - descent
→ Error is propagated back to the hidden unit

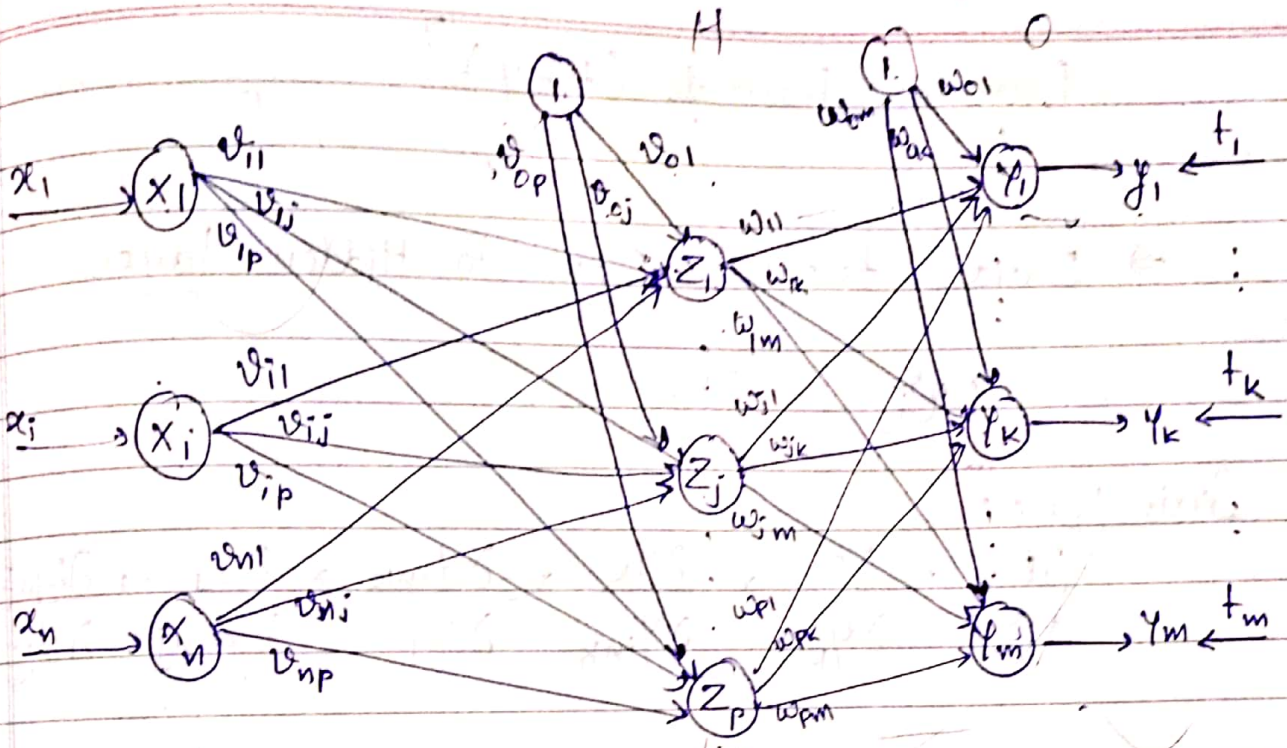
→ Aim : Train the NN to achieve a balance bet^{n} net's ability to respond (Memorization) and Ability to give reasonable responses to the input that is similar but not identical to the ones used in training (generalization)

→ Weights are calculated during training
→ To update weights, error must be calculated
→ At the hidden layer - no direct info. of error
→ Minimize the output error
→ Three stages :
  - Feed-forward of input training pattern
  - Calculation & back-propagation of error
  - Updation of weights.

Architecture :

→ Input, Hidden, Output layer
→ Biases at hidden & output layer acts as weights
→ During back propagation - Signals are sent in reverse direction
→ Output - binary or bipolar
→ Activation - Any

Scanned with CamScanner



x - Input training vector $(x_1 \dots x_i \dots x_n)$

t - target output vector $(t_1 \dots t_k \dots t_m)$

x_i - Input unit i

v_{oj} - Bias on j^{th} hidden unit

w_{ok} - Bias on k^{th} output unit

z_j - Hidden unit j

Net input to z_j $\quad z_{inj} = v_{oj} + \sum_{i=1}^{n} x_i v_{ij}$

$$z_j = f(z_{inj})$$

Net input to y_k $\quad y_{ink} = w_{ok} + \sum_{j=1}^{p} z_j \cdot w_{jk}$

y_k = Output unit k

$$y_k = f(y_{ink})$$

δ_k = error correction weight adjustment for w_{jk} (output to hidden) - error of y_k

δ_j = error correction weight adjustment for v_{ij} error of hidden unit z_j

Scanned with CamScanner

Error $E = \frac{1}{2}(t-y)^2$

⇒ Error from Input to Hidden layer

$$\Delta v_{ij} = \frac{\partial E}{\partial v_{ij}}$$

Chain Rule:

$$\frac{\partial E}{\partial v_{ij}} = \frac{\partial E}{\partial y_k} \times \frac{\partial y_k}{\partial y_{ink}} \times \frac{\partial y_{ink}}{\partial z_j} \times \frac{\partial z_j}{\partial z_{inj}} \times \frac{\partial z_{inj}}{\partial v_{ij}}$$

Now, $\frac{\partial E}{\partial y_k} = -(t_k - y_k)$

$\frac{\partial y_k}{\partial y_{ink}} = f'(y_{ink})$

$\frac{\partial y_{ink}}{\partial z_j} = w_{jk}$

$\frac{\partial z_j}{\partial z_{inj}} = f'(z_{inj})$

$\frac{\partial z_{inj}}{\partial v_{ij}} = x_i$

So, $\frac{\partial E}{\partial v_{ij}} = -(t_k - y_k) \times w_{jk} \times f'(y_{ink}) \times f'(z_{inj}) \times x_i$

Scanned with CamScanner

→ Error from Hidden to output layer :

$$\Delta w_{jk} = \frac{\partial E}{\partial w_{jk}}$$

$$= \frac{\partial E}{\partial y_k} \times \frac{\partial y_k}{\partial y_{ink}} \times \frac{\partial y_{ink}}{\partial w_{jk}}$$

$$= -(t_k - y_k) \times f'(y_{ink}) \times z_j$$

→ Error correction term for output unit y_k ($k = 1$ to m)

$$\Rightarrow \delta_k = (t_k - y_k) \cdot f'(y_{ink})$$

Updates in weight & bias

$$\Delta w_{jk} = \alpha \delta_k z_j \quad ; \quad \Delta w_{ok} = \alpha \delta_k$$

→ Error correction term for hidden unit z_j ($j = 1$ to p)

$$\delta_{inj} = \sum_{k=1}^{m} \delta_k \cdot w_{jk}$$

$$\delta_j = \delta_{ij} \cdot f'(z_{inj})$$

Updates in weight & bias

$$\Delta v_{ij} = \alpha \delta_j x_i \quad ; \quad \Delta v_{oj} = \alpha \delta_j$$

Scanned with CamScanner

⇒ Activation function -

Binary Sigmoidal (0 to 1)

Bipolar Sigmoidal (-1 to 1)

⇒ Features:

- Continuity
- Differentiability
- Non decreasing monotony

Scanned with CamScanner