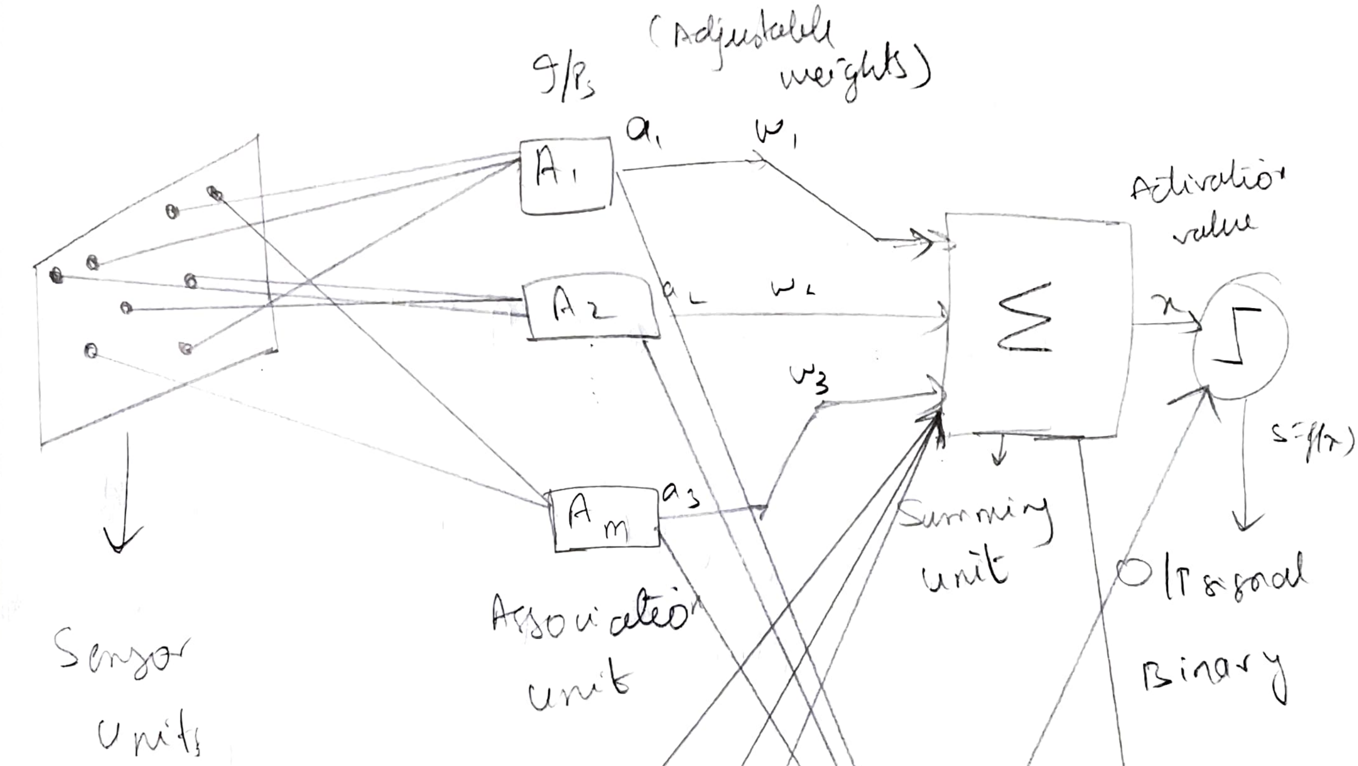
39110373

CA 8E-1 Deep learning

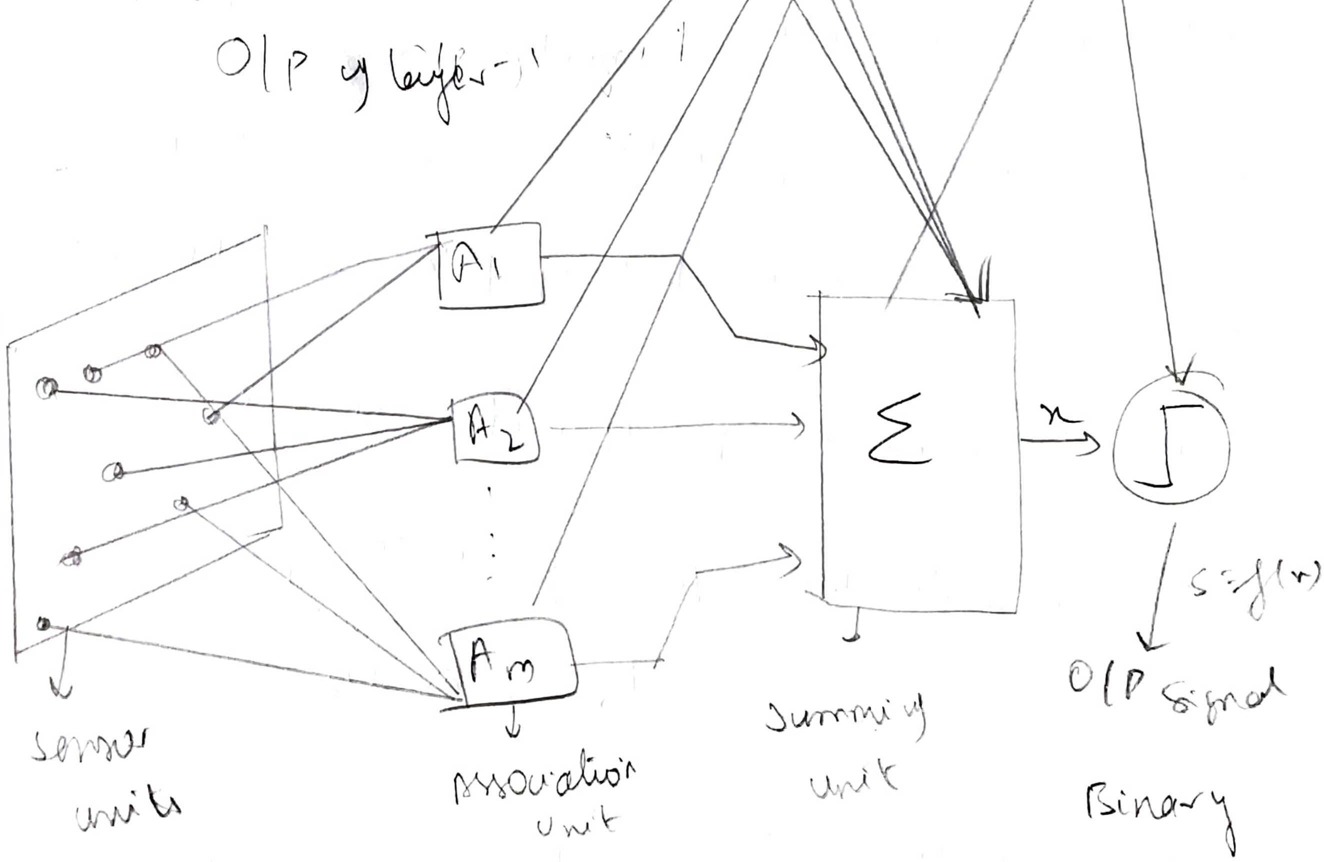
HARIHARAN

PART-B

(7)



O/P of layer-1



O/P of layer-2

39110373

Why

Activation function $x = \sum_{i=1}^{m} w_i a_i - \theta$

O/P signal $s = f(x)$

Error $\Rightarrow \delta = b - s$

weight change = $\Delta w = \eta \delta a_i$

Algorithm.

① Initialize the weights (w_i) & bias (B_0) to small random ~~value~~ values almost near to ~~z~~ zero.

② Set learning rate in the range of 0-1.

③ check for stop condn. If not do 3 to 7

④ for every ~~trainig~~ training pair do 4 to 7.

⑤ set activation of O/P units : $x_i = S_i$ for $i = 1$ to M

⑥ Calculate the O/P Response using

$$Y_m = b_0 + \sum x_i w_i$$

⑦ Activation funcn.

$$y = \begin{cases} 1, & \text{if } y_m = \theta \\ 0, & \text{if } y_m <= \theta \\ -1 & \text{if } y_m < -\theta \end{cases}$$

(8) If target is $\neq$ actual O/P then update weights as:

$$w_{i(new)} = w_{i(old)} + \text{change in weights vector}$$

where,

change in weight vector = $\eta t_i x_i$

$t_i \rightarrow$ target O/P of i^{th} iteration

$x_i \rightarrow$ I/P of i^{th} vector.

$$b_{0(new)} = b_{0(old)} + \text{change in bias}$$

change in Bias = ηt_i

Else

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

$$b_{0(new)} = b_{0(old)}$$

(9) Test for Stop condn.

(8) This ~~trainig~~ training algorithm can be divided as:-

(i) Initialization of Bias, weights.

(ii) Feed forward

(iii) Back propagation for Error

(iv) Updating of ~~all~~ weights and biases

39110773

Algorithm

(i) Initialization of weights

Step1: Initialize the weights to small random values near zero

Step2: While stop cond^n false do 3 to 10

Step3: For each pair do 4 to 9.

(ii) Feed forward

Step4: Each I/P xi received and forwarded to higher layers

Step5: ~~Each~~ Hidden unit sums its ~~weigl~~ weighted I/P as follows

$$z_{inj} = w_{oj} + \sum x_i w_{ij}$$

Applying activation func^n.

$$z_j = f(z_{inj})$$ → This is passed to O/P layer.

Step6: O/P unit sums it's weighted I/P$_s$

$$y_{ink} = v_{oj} + \sum z_j v_{jk}$$

~~Agg~~ Again Apply activation func^n.

$$\cancel{Y_k = f(Y_j}$$

$$Y_k = f(Y_{ink})$$

(III) Backpropagation for Errors:

Step 7: $\delta_k = (t_k - Y_k) f(Y_{ink})$

Step 8: $\delta_{inj} = \Sigma \delta_i V_{jk}$

(IV) Updating of weight and bias

Step 8: weight Correction.

$$\Delta w_{ij} = \alpha \delta_k Z_j$$

bias Correction

$$\Delta w_{oj} = \alpha \delta_k$$

Step 9:

New weight $w_{ij(new)} = w_{ij(old)} + \Delta w_{ij}$

$$v_{jk(new)} = v_{jk(old)} + \Delta V_{jk}$$

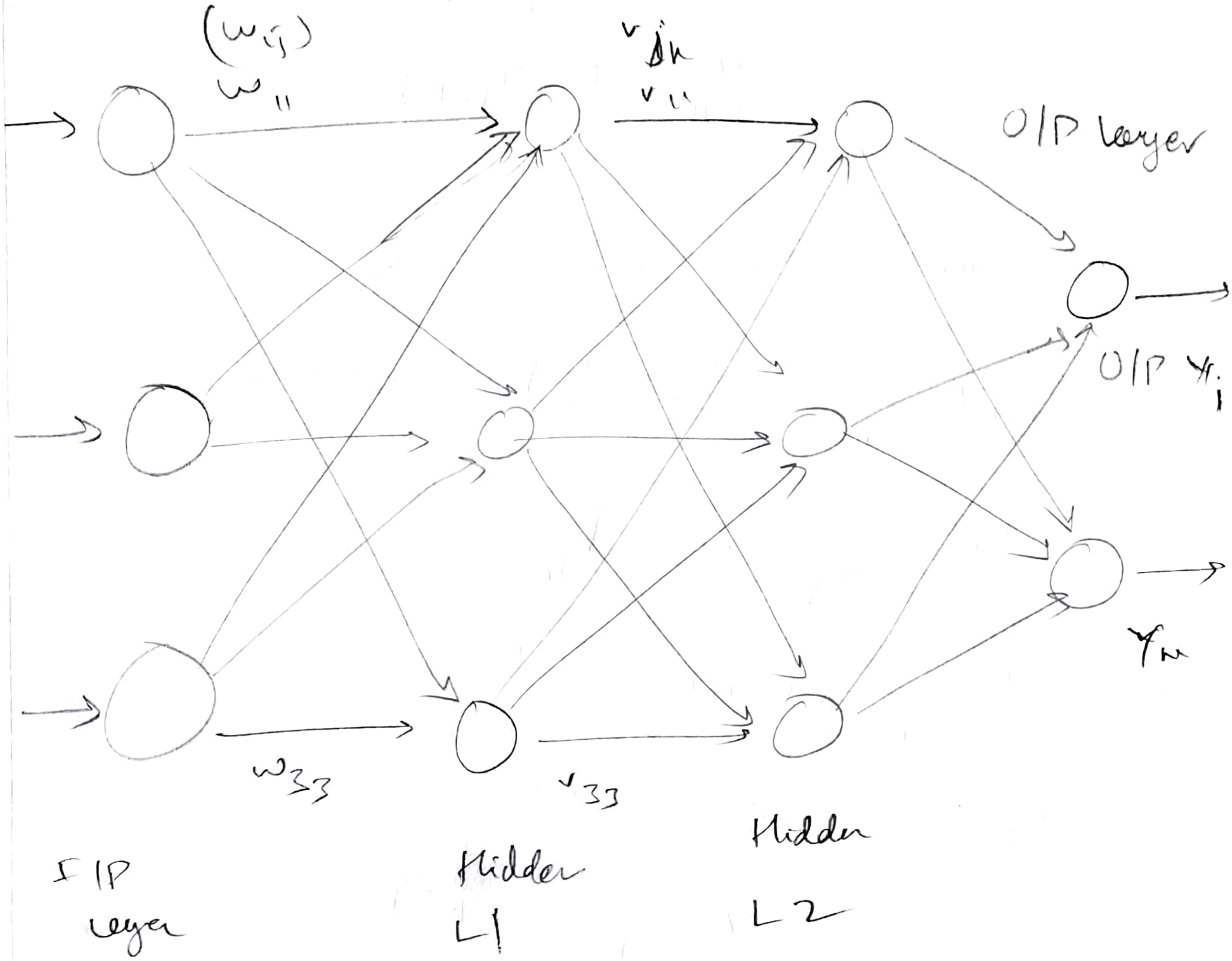
New bias

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

$$v_{ok(new)} \quad v_{ok(old)} + \Delta v_{ok}$$

39110373
~~3446~~

Step 10: Test for stop cond n



PART-A

(1) Size: The no. of neurons AI AN is much less than that of ~~compared~~ compared to biological neural.

-> They are made of oscillators - thus this gives them ability to filter I/P and to resonate with noise.

3911037 (4)

(2) ~~The~~ ~~thi~~ Universal Approximation theorem ~~to~~ tells that, Neural Network has a kind of universtality i.e. no matter what f(x) is ~~the~~ there is a network that can approx ~~a~~ approach the and get the work done.

(3) Stochastic - Gradient - Descent Algorithm can be ~~the~~ used ~~for~~ ~~This~~ ~~part~~ to avoid getting trapped

(4) ~~A~~ Neural Networks procedure can be used to standardize the real world data.

(5)

	Deep	Shallow
(i)	Express highly complex funct over I/P space	Express in one hidden layer and same no of neurons
(ii)	Can decode highly curved manifolds, in I/P space into flat manifolds	Shallow networks cannot ~~dea~~ decode complex manifolds