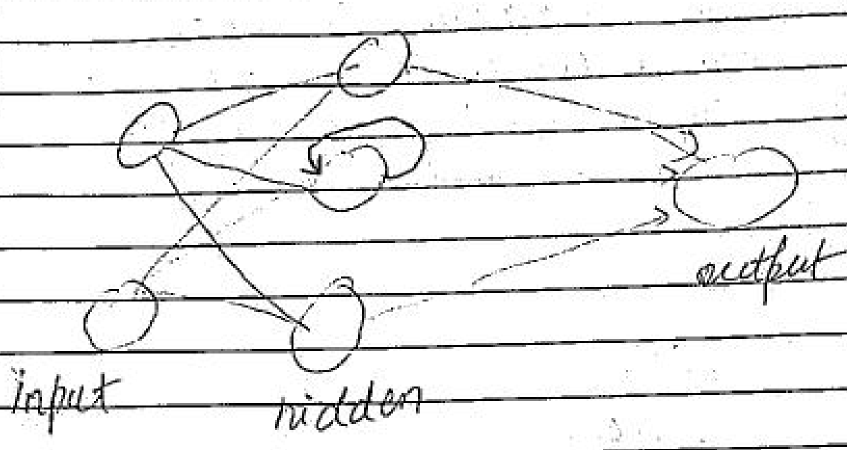
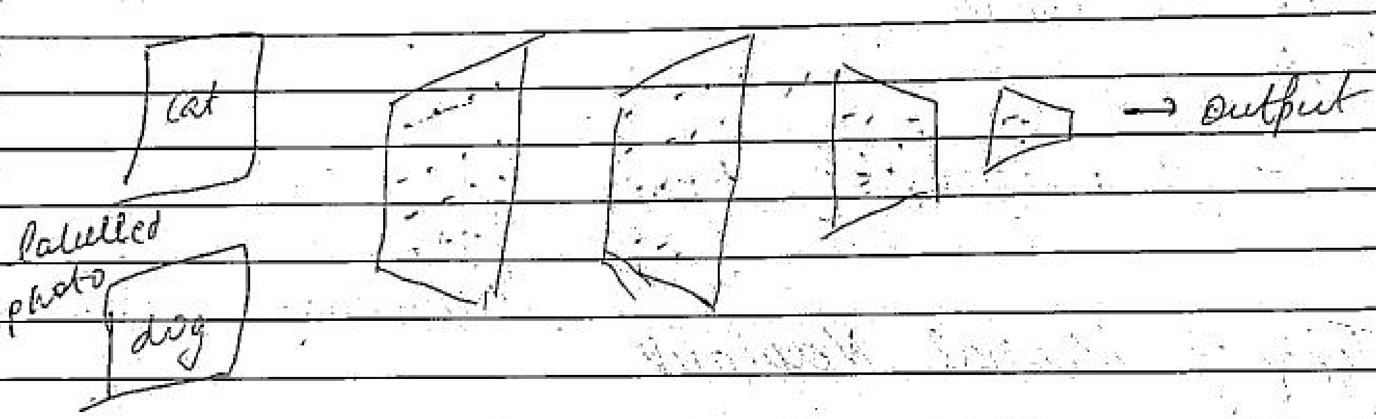
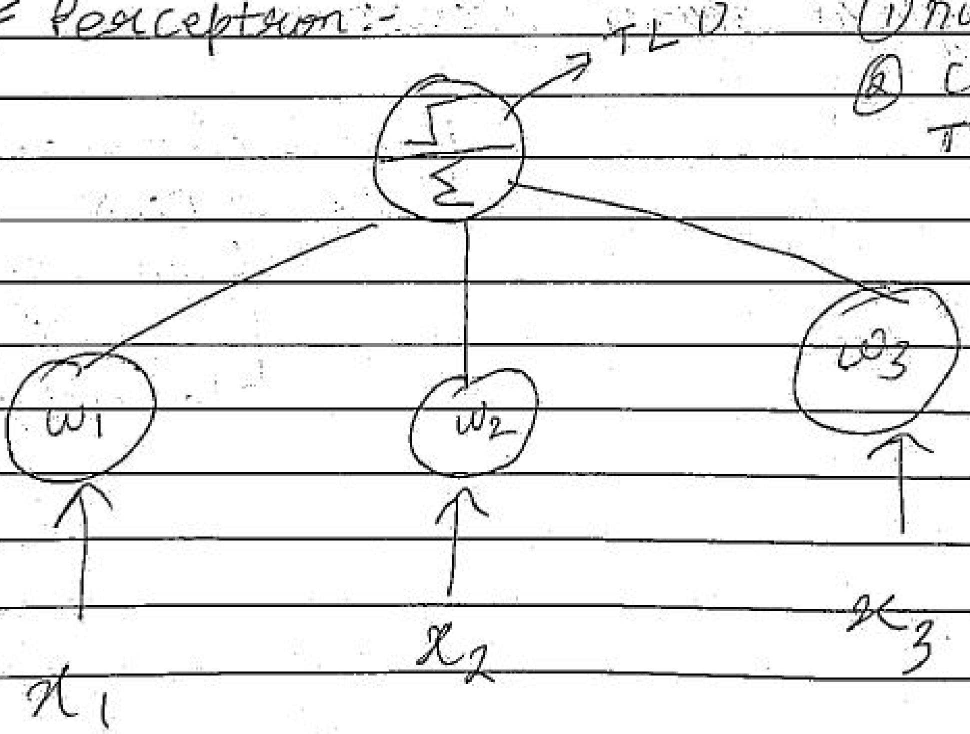
Recurrent Neural Network



Convolutional Neural Network (CNN)



Perceptron :-

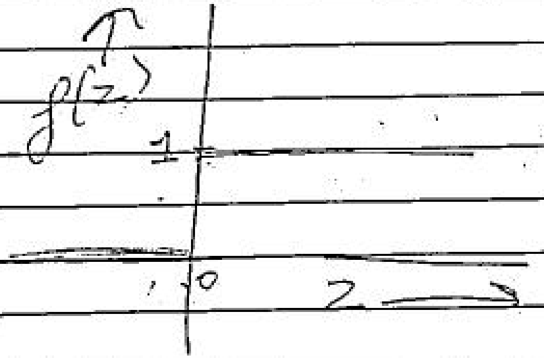


① numerical v|es
② Classification
TLU - Threshold logical unit

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

① Activation function ⇒ step function

Graph - step function $= \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}$



MULTI - 3 class = 3 TLU
4 class = 4 TLU

Training of Perceptron :-
⇒ Reinforcement Learning

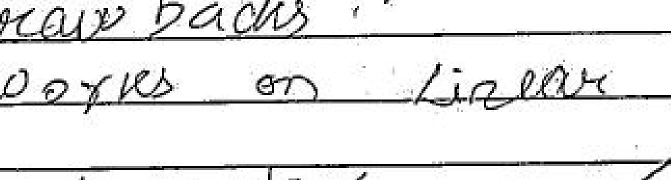
Learning Rule of Perceptron :-

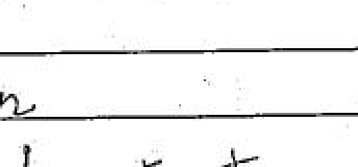
$w_{ij\,(new)} = w_{ij}\,(old) + \underset{lr = 0.05\ app}{\eta}\,(y_{act} - y_{prd})\,x_i$

since x_i is fixed, as it is the input. we can bring changes in weights

Drawbacks :-

① Works on Linear Activation



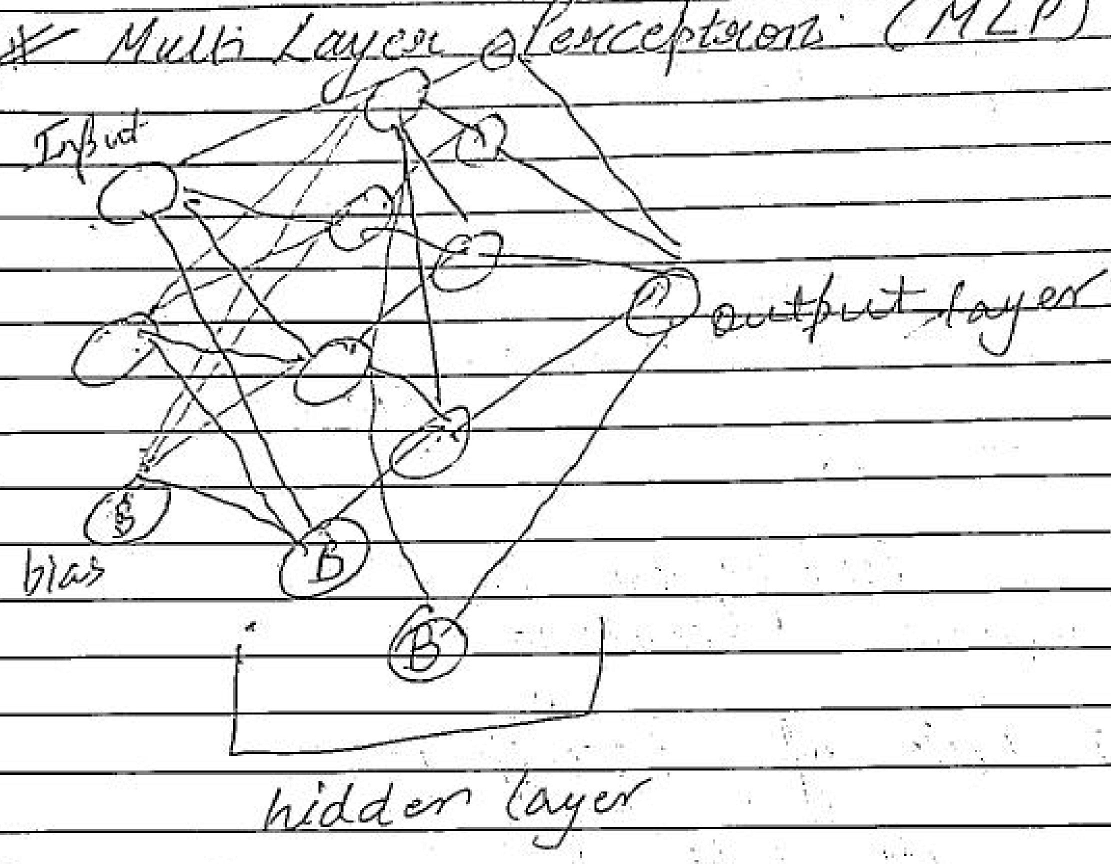


Stochastic Gradient descent

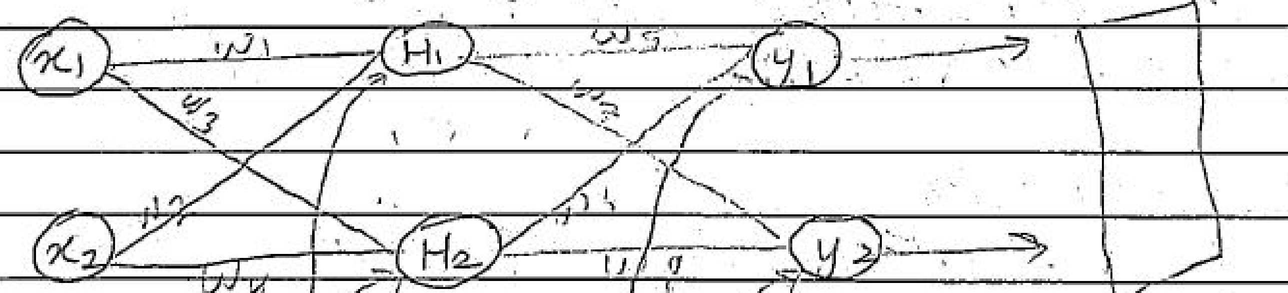
② cannot process complicated data

Overcome Drawbacks
→ We introduce Multilayer Perceptron

Multi Layer Perceptron (MLP)



ANN → many layers → DNN



$$H_1 = x_1 * w_1 + x_2 * w_2 + b_1$$

$$\sigma = \frac{1}{1+e^{-x}} \qquad \text{out } H_1 = \frac{1}{1+e^{-H_1}}$$

Exampel

$x_1 = 0.05$ $\quad w_1 = 0.15$ $\quad w_5 = 0.40$

$\quad w_2 = 0.20$ $\quad w_6 = 0.45$ $\quad T_1 = 0.01$

$x_2 = 0.1$ $\quad w_3 = 0.25$ $\quad w_7 = 0.50$ $\quad T_1 = 0.99$

$\quad w_4 = 0.30$ $\quad w_8 = 0.55$

$b_1 = 0.35$

$b_2 = 0.6$

Forward Pass [Feed Forward]

$H_1 = x_1 w_1 + x_2 w_2 + b_1$

$= (0.05)(0.15) + (0.10)(0.20) + 0.35$

$= 0.3775$

$\text{out } H_1 = \frac{1}{1+e^{-H_1}} = \frac{1}{1+e^{-0.3775}} = 0.593$

$H_2 = x_1 w_3 + x_2 w_4 + b_2$

$= (0.05)(0.25) + (0.1)(0.30) + 0.6$

$= 0.03$

$\text{Out } H_2 = \frac{1}{1+e^{-H_2}} = \frac{1}{1+e^{-0.03}}$

$= 0.596$

Now calculating

$y_1 = \text{output } H_1 * w_5 + \text{output } H_2 * w_6 + b_2$

$= 0.593 * 0.4 + 0.596 * 0.45 + 0.6$

$= 1.105$

$$\text{out } y_1 = \frac{1}{1+e^{-1.105}} = 0.751$$

similarly

$$\text{out } y_2 = 0.772$$

Calculating Total Error (MSE)

$$E_{total} = \sum \frac{1}{2}(\text{target} - \text{output})^2$$

$$= \frac{1}{2}(T_1 - \text{out } y_1)^2 + \frac{1}{2}(T_2 - \text{out } y_2)^2$$

$$E_{total} = 0.298$$

Backward Pass (To update weights)

$$w_5 = \text{new} = \text{old val} - \underset{LR}{\underset{\downarrow}{\alpha}}\left(\frac{\partial E}{\partial \text{val}}\right)$$

$$\text{error of } w5 = \frac{\partial E_{total}}{\partial w_5}$$

$$= \frac{\partial E_{total}}{\partial \text{out } y_1} \times \frac{\partial \text{out } y_1}{\partial y_1} \times \frac{\partial y_1}{\partial w_5}$$

$$\Downarrow$$

$$\frac{1}{2}(T_1 - \text{out } y_1)^2 + \frac{1}{2}(T_2 - \text{out } y_2)^2$$

$$\frac{1}{2} \times 2(T_1 - \text{out } y_1) \times (-1) = 0.741$$

$$\frac{\partial \, out\, y_1}{\partial y_1} = (out\, y_1)(1 - out\, y_1) \qquad \sigma'(x) = \sigma(x)(1-\sigma(x))$$

$$= 0.751(1-0.751)$$

$$= 0.186$$

$$\frac{\partial y_1}{\partial w_5} = \frac{\partial \, out\, H_1 * w_5 + out\, H_2 * w_6 + b_2}{\partial w_5}$$

$$1 * out\, H_1 *$$

$$\frac{\partial y_1}{\partial w_5} = out\, H_1 = 0.593$$

$$\frac{\partial E_{total}}{\partial w_5} = 0.741 * 0.186 * 0.593$$
$$= 0.0821$$

$$\underset{new}{w_5} = old\ w_5 - \eta \frac{\partial E_{total}}{\partial w_5}$$

$$= 0.4 - 0.5 * 0.0821$$
$$= 0.358$$

$w_6 = 0.408$ $w_7 = 0.511$ $w_8 = 0.561$

new updated weights

Date :

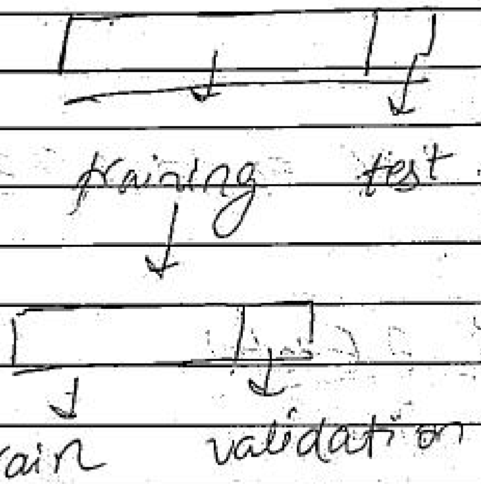
Now at hidden layer, updating w_1, w_2, w_3, w_4

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial output H_1} * \frac{\partial out H_1}{\partial H_1} * \frac{\partial H_1}{\partial w_1}$$

$w_1 = 0.149$ $w_2 = 0.199$ $w_3 = 0.249$
$w_4 = 0.299$

Type of datasets :-

(1) train data (2) test data (3) validation set



validation = find best values of hyperparamete

what we actually do ?

find out score on validation set → Tweak Model according to result
↓
Confirm result on test set. ← pick model that works best on validation set