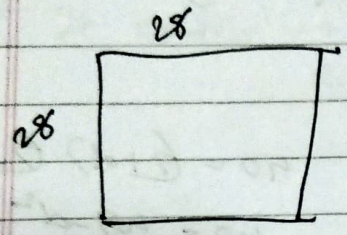
Neural Networks

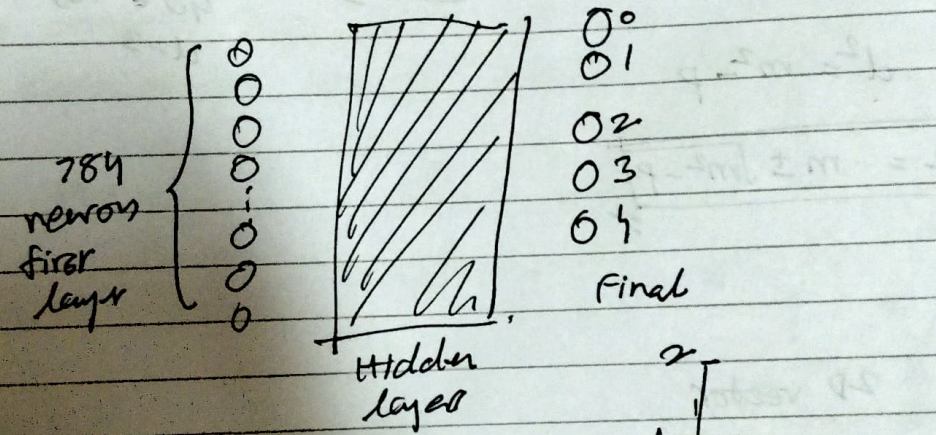
convolutional NN → image recognition

Neuron → Thing that holds a number



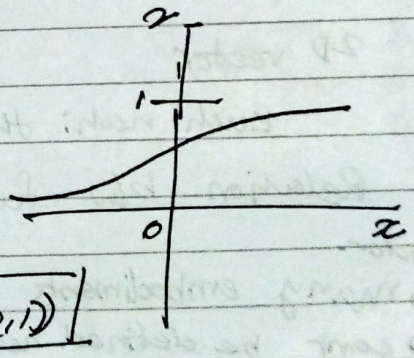
$28 \times 28 = 784$

(1.0) Activation



Sigmoid

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



sigmoid [To convert in range (0,1)]

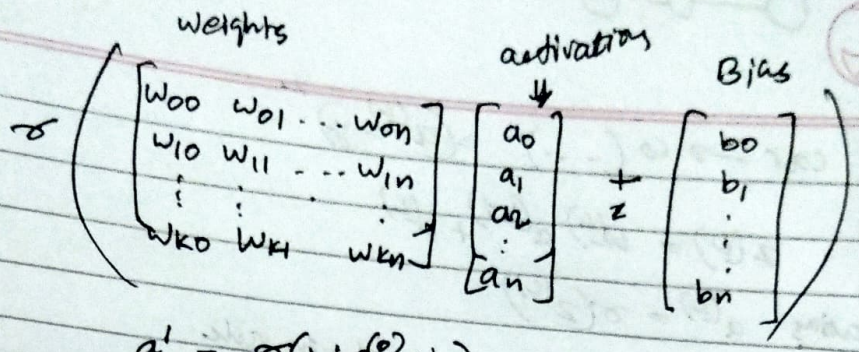
$\sigma(w_1 a_1 + w_2 a_2 + w_3 a_3 + \dots - 10)$

weighted sum

"bias"

weights
biases

Findings correct weights & biases



$$a^1 = \sigma(W a^{(0)} + b)$$

Neuron → Network → Function

Cost → Adding square of differences

Cost

High ↑ Not accurate

Small ↓ accuracy

Gradient descent

Neural Network $f(x)$

784 input

10 output

13 002 Parameters

⇒

cost function

Input: 13002 wts/bs

output: cost

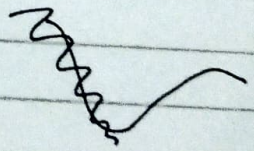
Parameters:

Gradient

↳ what nudges cause fastest change

Backpropogation

Schocastic training descent

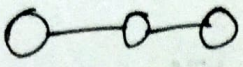


$0.2 = \sigma(W_0 a_0 + w_1 a_1 + \dots w_{n-1} a_{n-1} + b)$

Increase b

Increase w_i (in proportion to a_i)

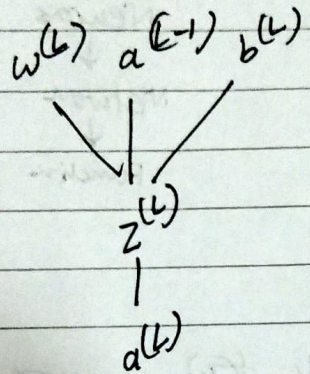
change a_i (in proportion to w_i)



cost $\longrightarrow$ $C_0(\ldots) = (a^{(L)} - y)^2$

Weighted sum $z^{(L)} = w^{(L)} a^{(L-1)} + b^{(L)}$

Activations $a^{(L)} = \sigma(z^{(L)})$



chain rule

$$\boxed{\frac{dC_0}{\partial w^{(L)}} = \frac{dz^{(L)}}{dw^{(L)}} \cdot \frac{da^{(L)}}{dz^{(L)}} \frac{\partial C_0}{\partial a^{(L)}}}$$

use here

$$\frac{\partial C_0}{\partial a^{(L)}} = 2(a^{(L)} - y)$$

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'(z^{(L)})$$

$$\frac{\partial z^{(L)}}{\partial w^{(L)}} = a^{(L-1)}$$

Similarly for

$\frac{dC_0}{\partial a^{(L-1)}}$ & $\frac{\partial C_0}{\partial b^{(L)}}$