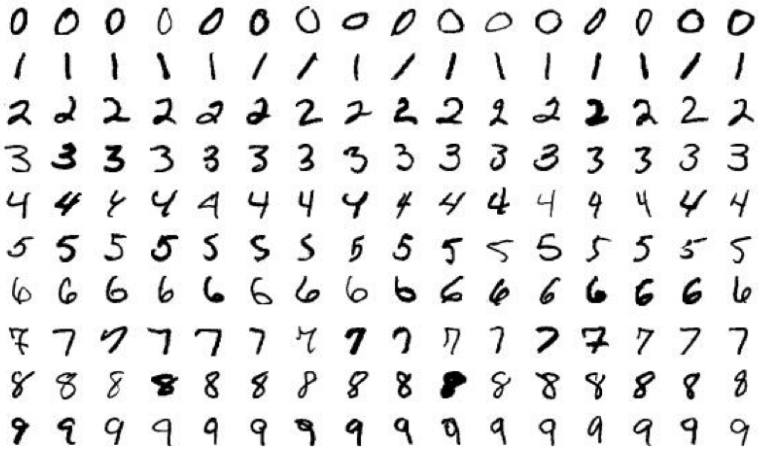
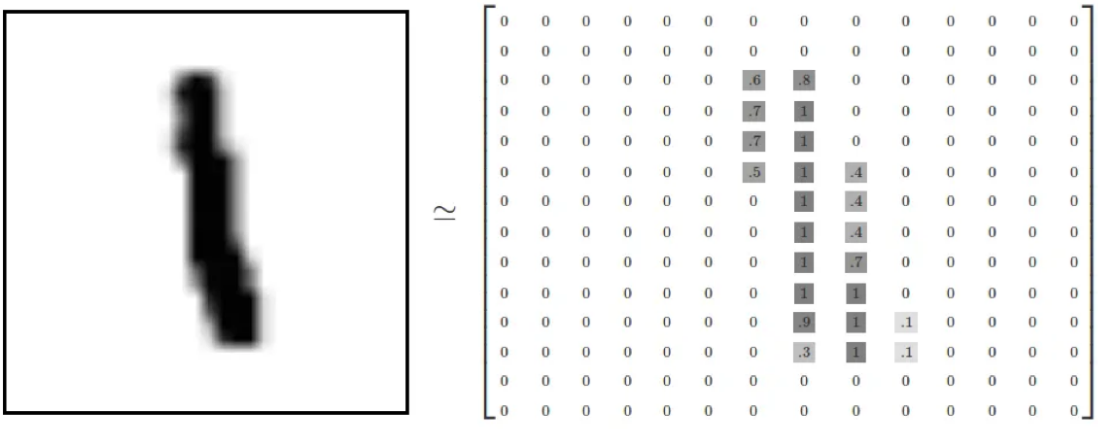
BUILDING A NEURAL NETWORK FROM SCRATCH

WHAT IS MNIST DATASET ?



- IMAGES $28 \times 28 \times 1$ → GRAYSCALE IMAGES



(0, 1, 2, ... , 9)

10 CLASSES

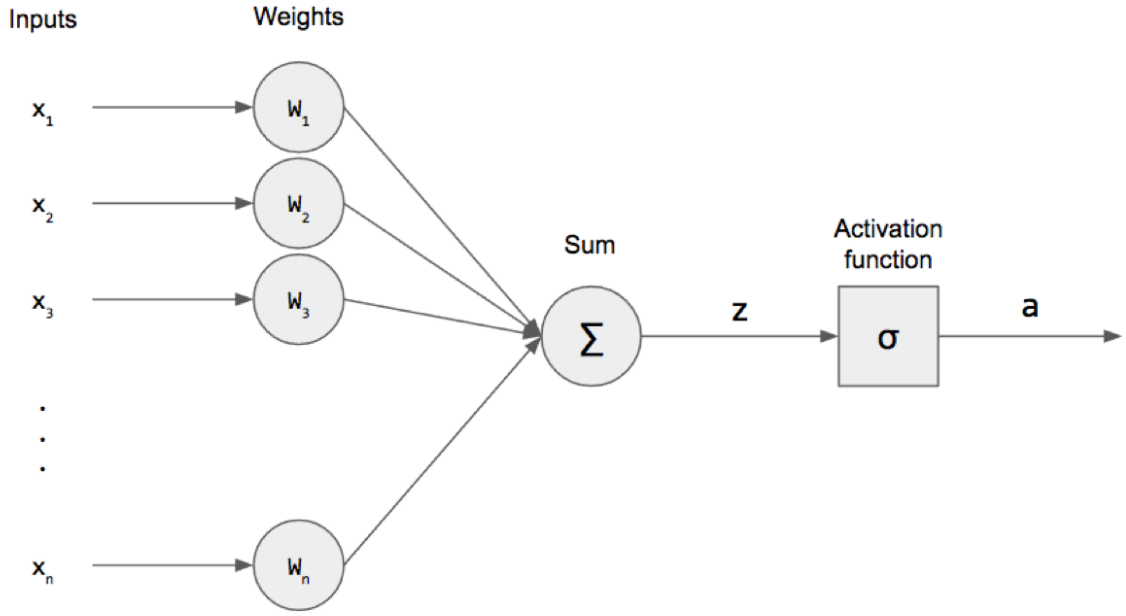
784 PIXELS

EACH PIXEL

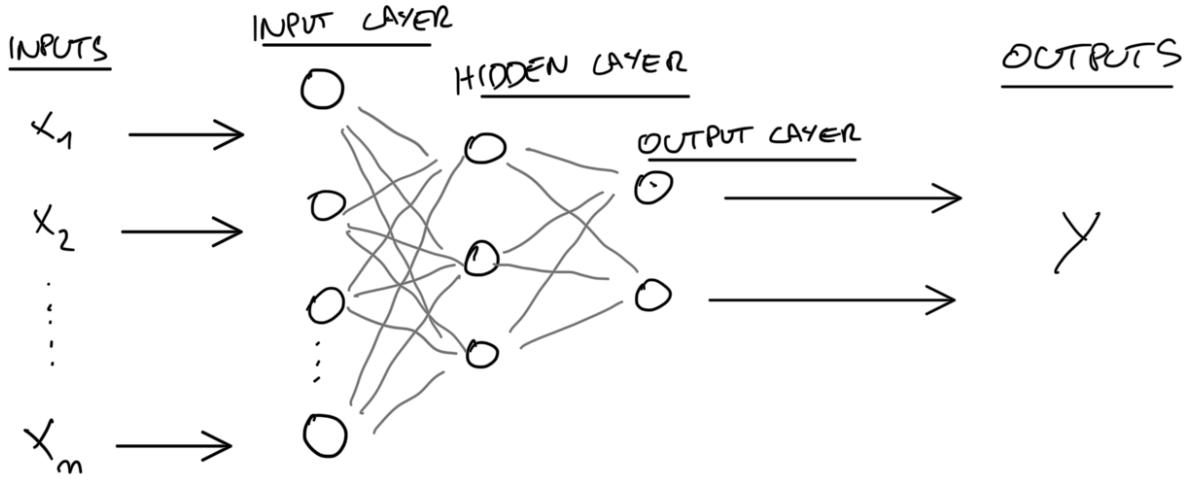
VALUE IS BETWEEN
0 - 255

WHAT IS A NN ?

- PERCEPTRON



- MULTI LAYER PERCEPTRON



HOW TO FEED AN IMAGE IN THE INPUT LAYER?

$X = \begin{bmatrix} \text{---} & x^{(1)} & \text{---} \\ \text{---} & x^{(2)} & \text{---} \\ & \vdots & \\ \text{---} & x^{(m)} & \text{---} \end{bmatrix}^T =$

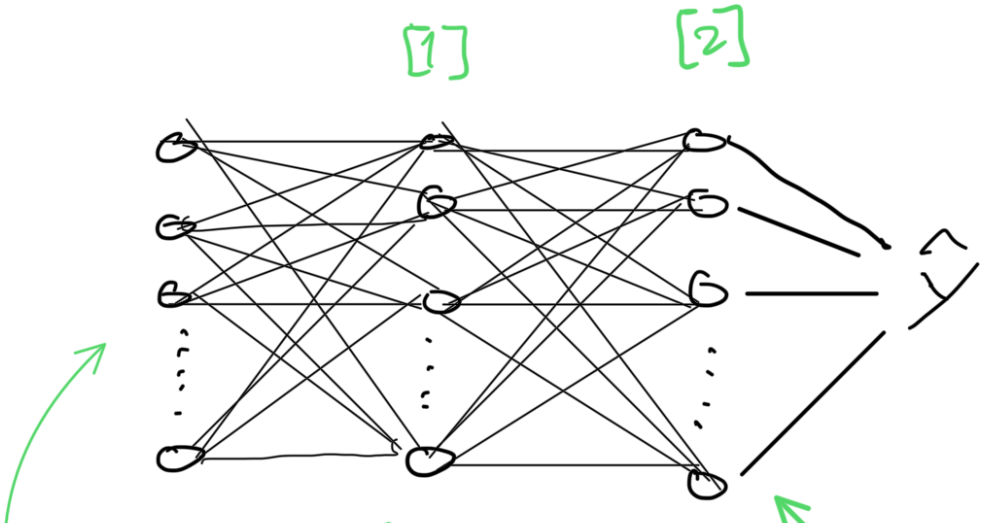
EACH ROW IS A SAMPLE

$= \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix}$

m = TRAINING SAMPLES

→ EACH COLUMN IS A SAMPLE

(784 ROWS × m COLUMNS)



784 — INPUT LAYER

10

10 — OUTPUT LAYER

WHY 10 NODES IN THE OUTPUT LAYER?

LAYERS COMPOSITION

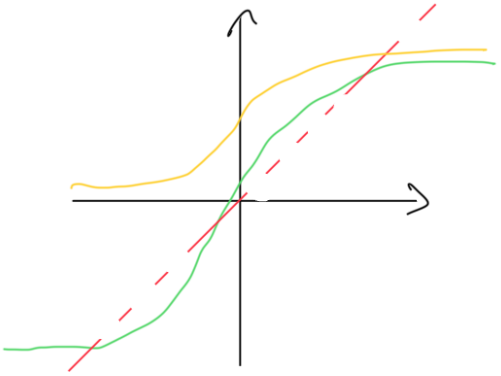
$$A^{[0]} = X \quad (784 \times m)$$

$$Z^{[1]} = W^{[1]} A^{[0]} + b^{[1]}$$

$(10 \times m)$ (10×784) $(784 \times m)$ $10 \times 1 \rightarrow 10 \times m$

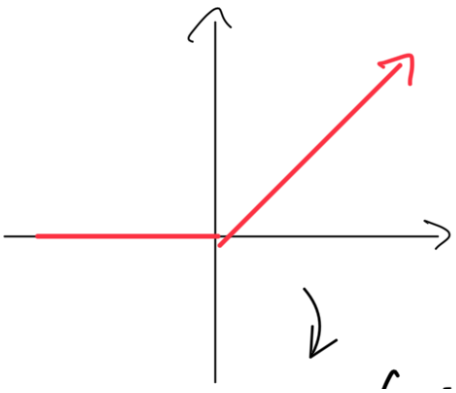
WITHOUT THE ACTIVATION FUNCTION THE NEW LAYER $A^{[1]}$ ITS JUST A LINEAR COMBINATION OF THE PREVIOUS LAYER

$$A^{[1]} = g(Z^{[1]}) = ReLU(Z^{[1]})$$



— tanh
— sigmoid

— ReLU



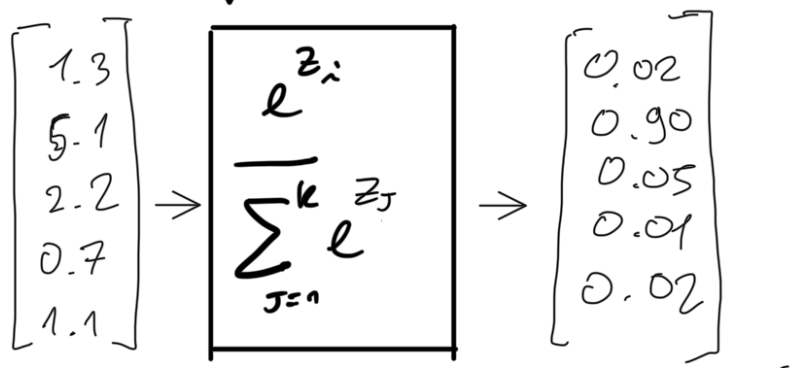
$$Z^{[2]} = W^{[2]} A^{[1]} + b^{[2]}$$

$10 \times m$ 10×10 $10 \times m$ $10 \times 1 \Rightarrow 10 \times m$

$A^{[2]} = g^{[2]}(Z^{[2]})$

$$A^{[L]} = softmax\left(Z^{[L]}\right)$$

$$ReLU(x) = \begin{cases} x \text{ if } x > 0 \\ 0 \text{ if } x \leq 0 \end{cases}$$



OUTPUT LAYER

PROBABICITIES

FORWARD PROPAGATION

WE NEED GOOD WEIGHTS AND BIASES

HOW DO WE LEARN THEM?

BACKPROPAGATION PHASE

$$dZ^{[2]} = A^{[2]} - Y$$

$dZ^{[2]}$: $10 \times m$ — REPRESENTS A SORT OF "ERROR" OF THE SECOND LAYER

$A^{[2]}$:

$$\begin{bmatrix} 0.1 \\ 0.2 \\ 0.05 \\ 0.03 \\ 0.90 \\ \vdots \end{bmatrix}$$

Y: ONE-HOT ENCODING

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

i.e. $y = 4$

$$dW^{[2]}_{10 \times 10} = \frac{1}{m} \underbrace{dZ^{[2]}}_{10 \times m} \underbrace{A^{[1]T}}_{m \times 10}$$

$$db^{[2]}_{10 \times 1} = \frac{1}{m} \sum dZ^{[2]}_{10 \times 1}$$

COMPUTE HOW MUCH THE WEIGHTS AND BIASES CONTRIBUTES TO THE ERROR

$dW^{[2]}$ IS THE DERIVATIVE OF THE LOSS FUNCTION W.R.T. TO THE WEIGHTS IN THE SECOND LAYER

$db^{[2]}$ IT'S THE AVERAGE OF THE ABSOLUTE ERROR

NOW WE HAVE TO PROCEED WITH THE BACKPROPAGATION PHASE

↓

BY DOING THE FORWARD PROPAGATION IN REVERSE

TAKE THE ERROR FROM THE PREVIOUS LAYER AND WE APPLY THE WEIGHTS AND BIASES IN REVERSE TO GET THE ERROR FOR THE FIRST LAYER

$g^{[1]\prime}$ IS THE DERIVATIVE OF THE ACTIVATION

$$dZ^{[1]} = W^{[2]T} dZ^{[2]} .* g'(Z^{[1]})$$

$10 \times m \quad 10 \times 10 \quad 10 \times m \quad 10 \times m$

FUNCTION

↓

WE HAVE TO UNDO THE ACTIVATION FUNCTION

$$dW^{[1]} = \frac{1}{m} dZ^{[1]} X^T$$

$10 \times 784 \quad 10 \times m \quad m \times 784$

$$db^{[1]} = \frac{1}{m} \sum dZ^{[1]}$$

$10 \times 1 \quad 10 \times 1$

UPDATING PHASE

α = LEARNING RATE

↓

IT'S AN HYPERPARAMETER

$$\begin{bmatrix} W^{[1]} = W^{[1]} - \alpha\, dW^{[1]} \\ b^{[1]} = b^{[1]} - \alpha\, db^{[1]} \\ W^{[2]} = W^{[2]} - \alpha\, dW^{[2]} \\ b^{[2]} = b^{[2]} - \alpha\, db^{[2]} \end{bmatrix}$$

↓

WE ARE DONE NOW?

↓

NO

↓

WE HAVE TO REPEAT THIS CYCLE FOR MULTIPLES TIME IN ORDER

TO LEARN GOOD WEIGHTS AND BIASES