Building a Neural Network

Lets take MNIST Dataset. →

- Contains (28 × 28) px images
- 10 classes (0 - 9)
- No. of pixels per image = 784

So the input would be of form →

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

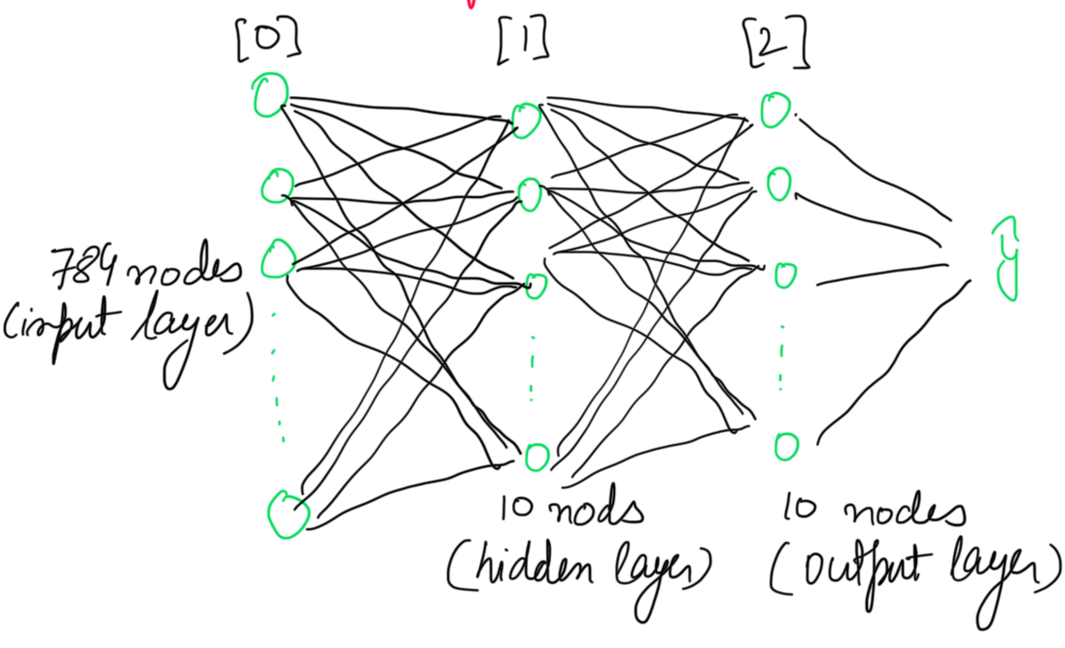
→ each row is 784 colums each representing a pixel

$$= \begin{bmatrix} | & | & | & & | \\ x(1) & x(2) & x(3) & \cdots & x(m) \\ | & | & | & & | \end{bmatrix}$$

→ instead of each row now each column is an example

Goal →
- Take in image
- Predict which class it represents.

Visual of our NN



Traing the NN →

Traing the Neural Network is 3 stage process

Stage 1 → Forward propagation | Forward Pass

Take image and pass it through the NN to compute the output.

Stage 2 → Back Propagation

Take the predicted output and compare to actual label, move towards layers and see how weights and biases contributed to the error.

Stage 3 → Update parameters

Based on calculated error update the weights

Stage 1 → Forward propagation

→ $A_{[0]}$ → input layer $= X \ (784 \times m)$

→ $Z_{[1]}$ → hidden layer $= W_{[1]} \times A_{[0]} + b_{[1]}$

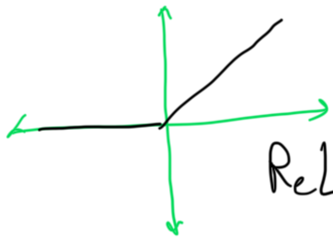
$W_{[1]}$ → weight matrix (10×784) dim for the hidden layer connections

$b_{[1]}$ → bias (10×1) dim

→ Apply activation function to the hidden layer.

$$A_{[1]} = g(Z_{[1]})$$

we will use $g \rightarrow \text{ReLU}(x) \begin{cases} x & \text{if } x>0 \\ 0 & \text{if } x\le 0 \end{cases}$



ReLU graph

→ $Z_{[2]} = W_{[2]} A_{[1]} + b_{[2]}$

$W_{[2]} \rightarrow$ weight matrix (10 x 10) dim for output layer connections

$b_{[2]} \rightarrow$ bias (10 x 1) dim

→ Apply softmax to get probability over all classes

$$A_{[2]} = \text{softmax}(Z_{[2]})$$

$$\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Note → sum of all probabilities = 1 after softmax

Stage 2 – Backward Propagation

if label = 4 then $Y = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

$$\underset{10 \times m}{dZ_{[2]}} = \underset{10 \times m}{A_{[2]}} - \underset{10 \times m}{Y}$$

$$\underset{10 \times 10}{dW_{[2]}} = \frac{1}{m} \, \underset{10 \times m}{dZ_{[2]}} \, \underset{m \times 10}{A_{[1]}^{T}}$$

$$\underset{10 \times 1}{db_{[2]}} = \frac{1}{m} \sum \underset{10 \times 1}{dZ_{[2]}}$$

$$\underset{10 \times m}{dZ_{[1]}} = \overbrace{\underset{10 \times 10}{W_{[2]}^{T}} \, \underset{10 \times m}{dZ^{[2]}}}^{\text{Losses from second layer}} \; .* \; \overbrace{g'_{[1]}(\underset{10 \times m}{Z_{[1]}})}^{\text{To compensate for ReLU}}$$

$$\underset{10 \times 784}{dW_{[1]}} = \frac{1}{m} \, \underset{10 \times m}{dZ_{[1]}} \, \underset{m \times 784}{A_{[0]}^{T}}$$

$$\underset{10 \times 1}{db_{[1]}} = \frac{1}{m} \, \underset{10 \times 1}{dZ_{[1]}}$$

Stage 3 - Update parameters

$$\rightarrow W_{[1]} = W_{[1]} - \alpha \, dW_{[1]}$$
$$\rightarrow b_{[1]} = b_{[1]} - \alpha \, db_{[1]}$$
$$\rightarrow W_{[2]} = W_{[2]} - \alpha \, dW_{[2]}$$
$$\rightarrow b_{[2]} = b_{[2]} - \alpha \, db_{[2]}$$

$\rightarrow$ α = learning rate (LR)

→ These 3 stages are repeated for a set number of times → epochs

Forward pass → Back Propagation → Update parameters → Forward pass