Training computer to recognize handwritten digits.

- Using MINST data base of 28×28 pixel digits

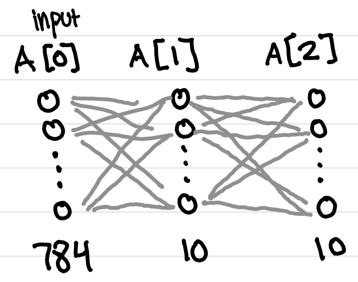
$28\ \boxed{784}\ 28$ } training image dimensions | each pixel has 0...255 weighting for intensity

We have m training images, put into a row matrix (csv):

$$X = \begin{bmatrix} \text{———} x^{[1]} \text{———} \\ \text{———} x^{[2]} \text{———} \\ \vdots \\ \text{———} x^{m} \text{———} \end{bmatrix}^T \Rightarrow \begin{bmatrix} | & | & & | \\ x^{[1]} & x^{[2]} & \cdots & x^{[m]} \\ | & | & & | \end{bmatrix}$$

Goal: Take the 28×28 or 784 pixel image and narrow to 10 digit options (0, 1, ..., 9)

Two layer neural network: [1]: hidden layer, [2]: output layer



Training the Network: ★ For this Neural net m = 1

1. Forward Propagation: running through our network [0] → [2]

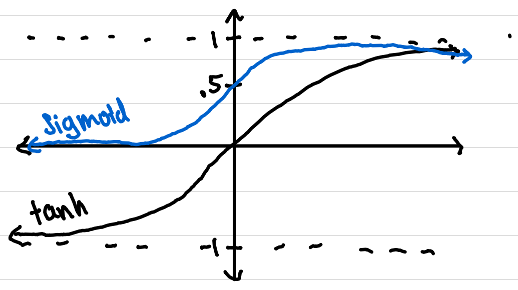
$$A^{[0]} = X \quad [784 \times m]$$
* input layer

$$\underset{[10\times m]}{Z^{[1]}} = \underset{[10\times 784]}{W^{[1]}}\ \underset{[784\times m]}{A^{[0]}} + \underset{[10\times m]}{b^{[1]}}$$
** unactivated A[1] layer

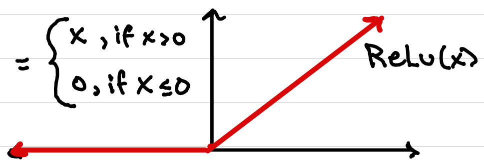
$W^{[1]}$ corresponds to a weight for each connection between the neurons in our $A^{[0]}$ to $A^{[1]}$ layer.

$$A^{[1]} = \underset{\text{activation function}}{g}(Z^{[1]}) = \text{ReLu}(Z^{[1]})$$
*** Normalized layer using activation funct.

Different Activation Functions:



$$= \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \le 0 \end{cases}$$



- Using ReLu
- σ and tanh are more complex

Going from layer 1 to layer 2:

$$\underset{10\times m}{Z^{[2]}} = \underset{10\times 10}{W^{[2]}}\ \underset{10\times m}{A^{[1]}} + \underset{10\times m}{b^{[2]}}$$

$$A^{[2]} = \text{Softmax}(Z^{[2]})$$

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

- We use this to derive a column of values that add to 1

2. Backwards Propagation: Optimization of weightings/biases

Second Layer

$$\underset{10\times m}{dZ^{[2]}} = \underset{10\times m}{A^{[2]}} - \underset{10\times m}{Y}$$

- taking prediction and subtracting the one hot encoding
- One hot Y puts a 1 at the index which is encoding
- shows the error of the second layer

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

$$db^{[2]} = \frac{1}{m} \sum \underset{10\times 1}{dZ^{[2]}}$$
- Average of absolute error

First Layer

$$\underset{10\times m}{dZ^{[1]}} = \underset{10\times 10}{W^{[2]T}}\ \underset{10\times m}{dZ^{[2]}} \;.*\; g'(\underset{10\times m}{Z^{[1]}})$$
- derivative of activation

$$\underset{10\times 784}{dW^{[1]}} = \frac{1}{m}\ \underset{10\times m}{dZ^{[1]}}\ \underset{m\times 784}{X^T}$$

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

3. Updating Parameters — α: learning rate - independent of model

$$W^{[1]} := W^{[1]} - \alpha\, dW^{[1]}$$

$$b^{[1]} := b^{[1]} - \alpha\, db^{[1]}$$

$$W^{[2]} := W^{[2]} - \alpha\, dW^{[2]}$$

$$b^{[2]} := b^{[2]} - \alpha\, db^{[2]}$$