Intros

Structure: Weekly meetings, more like office hours.
↪ also, show me what has been done in the code.

Go over rough idea:

To do:

▷ Watch 3b1b videos

▷ Find meeting time for next week
(July 3rd only day I am available)

▷ Code implementation
(probably best to use python. Google collab..?)

▷ Data processing:
- Use Mnist or USPS ?

▷ Implementation: no hidden layers first ?

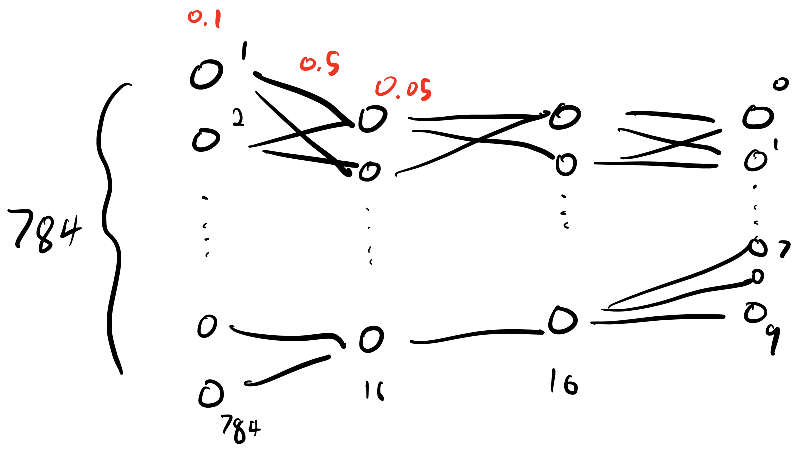
▷ Recommended: Students meet once per week for a couple hours to try to code the neural network.

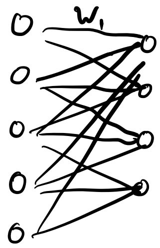
$28 \left\{ \boxed{7} \right.$ $\longleftrightarrow$ vector w/ $28 \cdot 28 = \underline{784}$ components.

$\underbrace{}_{28}$

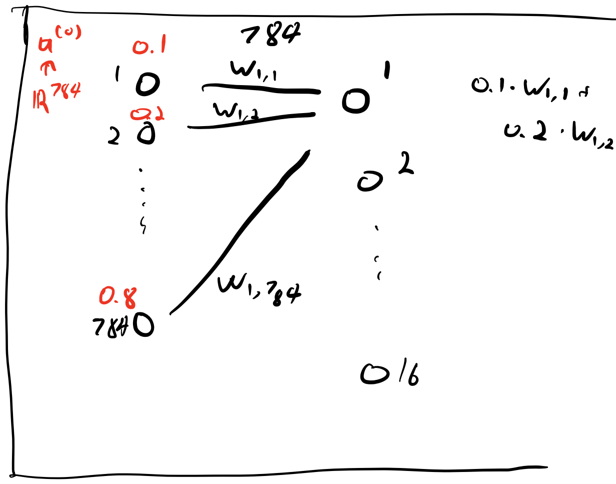
$\updownarrow$

$\mathbb{R}^{784}$





(multilayer perceptron)



$0.1 \cdot w_{1,1} + 0.2 \cdot w_{1,2} + \dots + 0.8 \cdot w_{1,784}$

First layer: 784 neurons.

input space: $[0,1]^{784}$

Final layer: 10 neurons

2 inner layers, 16 neurons each.

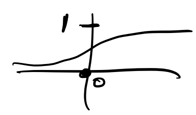
Layer 0   Layer 1

$$\mathbb{R}^{784} \longrightarrow \mathbb{R}^{16}$$

$$a^{(0)} \longmapsto \sigma\left(W^{(1)} a^{(0)} + b^{(1)}\right)$$

or ReLu

$$\underset{\text{Sigmoid.}}{\sigma}(x) = \frac{1}{1+e^{-x}}$$



$$W^{(1)} = \begin{pmatrix} w_{1,1} & \cdots & w_{1,784} \\ w_{2,1} & \cdots & w_{2,784} \\ & \vdots & \\ w_{16,1} & \cdots & w_{16,784} \end{pmatrix} \quad b^{(1)} = \begin{pmatrix} b_1 \\ \vdots \\ b_{16} \end{pmatrix} \quad a^{(0)} \begin{pmatrix} a_1^{(0)} \\ \vdots \\ a_{784}^{(0)} \end{pmatrix}$$

$W_1 \in M_{16 \times 784}(\mathbb{R})$ weight matrix.

$b_1 \in M_{16 \times 16}(\mathbb{R})$ bias matrix.

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

$$a_i^{(1)} = \sigma\left(\left(\sum_{j=0}^{784} w_{i,j} a_j^{(0)}\right) + b_i^{(1)}\right)$$

Layer 0   Layer 1   Layer 2   Layer 3

$$\begin{tikzcd}
\mathbb{R}^{784} \arrow[r, "A_1"] \arrow[rrr, bend left, "A"] & \mathbb{R}^{16} \arrow[r, "A_2"] & \mathbb{R}^{16} \arrow[r, "A_3"] & \mathbb{R}^{10}
\end{tikzcd}$$

$$\sigma(W^{(1)}\cdot + b^{(1)}) \qquad \sigma(W^{(2)}\cdot + b^{(2)}) \qquad \sigma(W^{(3)}\cdot + b^{(3)})$$

$$16\times 784 \quad 16\times 1 \qquad 16\times 16 \quad 16\times 1 \qquad 10\times 16 \quad 10\times 1$$

Overall activation

$$A = \sigma\left(W^{(3)}\cdot\sigma\left(W^{(2)}\cdot\sigma\left(W^{(1)}(\bullet) + b^{(1)}\right) + b^{(2)}\right) + b^{(3)}\right)$$

$$A_3 \circ A_2 \circ A_1(\bullet)$$

$\boxed{\text{activation function:}}$

$$A : \mathbb{R}^{784} \longrightarrow \mathbb{R}^{10} \qquad A = A_{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}, W^{(3)}, b^{(3)}}$$

(notation: $A_{\underline{W}, \underline{b}}$)

Choices	# parameters:	Choices	# parameters
$W^{(1)} \in M_{784\times 16}(\mathbb{R})$	$784\times 16 = 12{,}544$	$b^{(1)} \in \mathbb{R}^{16}$	16
$W^{(2)} \in M_{16\times 16}(\mathbb{R})$	$16\times 16 = 256$	$b^{(2)} \in \mathbb{R}^{16}$	16
$W^{(3)} \in M_{16\times 10}(\mathbb{R})$	$16\times 10 = 160$	$b^{(3)} \in \mathbb{R}^{10}$	10

total # parameters : 13,002.

Training:

Cost function built from a training set.

Collection of pairs

$$TS = \{ (I_1, L_1), (I_2, L_2), \ldots, (I_n, L_n) \}$$

where $I_k \in \mathbb{R}^{784}$ is an input, $L_i \in \mathbb{R}^{10}$ is the correct output.

$$C = C_{TS} : \mathbb{R}^{13{,}002} \longrightarrow \mathbb{R},$$

really

$$C : M_{784\times16} \times \mathbb{R}^{16} \times M_{16\times16} \times \mathbb{R}^{16} \times M_{16\times10} \times \mathbb{R}^{10} \longrightarrow \mathbb{R}$$

$$\left(W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}, W^{(3)}, b^{(3)}\right) \longmapsto \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\text{average over all images in the training data.}} \underbrace{\| A_{\underline{W},\underline{b}}(I_i) - L_i \|^2}_{\text{distance of the guess } (A_{\underline{W},\underline{b}}(I_i)) \text{ from the answer } L_i}$$

Intuition of Cost function :

$C(\underline{W}, \underline{b})$ is small $\iff$ the weights $\underline{W}, \underline{b}$ do a good job guessing the correct answer.

$\underline{W} = W^{(1)}, W^{(2)}, W^{(3)}$

$\underline{b} = b^{(1)}, b^{(2)}, b^{(3)}$

$\Rightarrow$ $\boxed{\text{Goal}}$ should be to finds weights $\underline{W}, \underline{b}$ minimizing C.

Gradient Descent

Idea: ① Start with Random inputs $(\underline{W}_0, \underline{b}_0)$ to C.

② Find the direction of $(\underline{W}_0, \underline{b}_0)$ that leads the the larges decrease in C.

③ Change $(\underline{W}_0, \underline{b}_0)$ to $(\underline{W}_1, \underline{b}_1)$ according to the direction from ②.

More precisely:

For 2, use $-\nabla C(\underline{w}, \underline{b})$.

$$C : \mathbb{R}^n \to \mathbb{R}, \qquad \nabla C : \mathbb{R}^n \to \mathbb{R}$$

$$\nabla C = \begin{pmatrix} \frac{\partial C}{\partial x_1} \\ \vdots \\ \frac{\partial C}{\partial x_n} \end{pmatrix}$$

$$\left(\text{e.g.} \quad \begin{aligned} C : \mathbb{R}^2 &\to \mathbb{R} \\ (x,y) &\mapsto x^2 y + y \end{aligned} , \quad \nabla C = \begin{pmatrix} \frac{\partial C}{\partial x} \\ \frac{\partial C}{\partial y} \end{pmatrix} = \begin{pmatrix} 2xy \\ x^2 + 1 \end{pmatrix} \right)$$

☆ To do: details for Gradient descent ☆

$$= \frac{1}{n}\sum_{i=1}^{n} \| \underbrace{A_3 A_2 A_1(a^{(0)})}_{a^L} - L$$

$$\begin{tikzcd}
0 & w^{(1)} \arrow[dr] & \overbrace{a^{(0)}}^{I_i} \arrow[d] & b^{(1)} \arrow[dl] \\
1 & w^{(2)} \arrow[dr] & a^{(1)} \arrow[d] & b^{(2)} \arrow[dl] \\
2 & w^{(3)} \arrow[dr] & a^{(2)} \arrow[d] & b^{(3)} \arrow[dl] \\
3 & L_i \arrow[dr] & a^{(3)} \arrow[d] & \\
\text{Cost} & & C &
\end{tikzcd}$$