

Chapter 3: Backpropagation, Intuitively

Recap: Gradient Descent.

1. Initialize randomly

↳ random weights & biases.

2. Find "cost"

↳ output the network gives vs output you need.



add up the squares of differences

↳ Repeat for all of the training data.

3. Average the cost = total cost of the network.

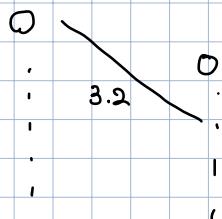
4. Find the negative gradient to change all of weights & biases.
to efficiently decrease the cost.

Backpropagation: algorithm for computing complicated gradient

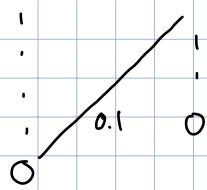
$$-\nabla C(\dots) = \begin{bmatrix} 0.16 \\ 0.72 \\ -0.93 \\ \vdots \\ 0.04 \\ 1.63 \end{bmatrix}$$


weights &
biases

How sensitive the cost function is
to each weights & biases.

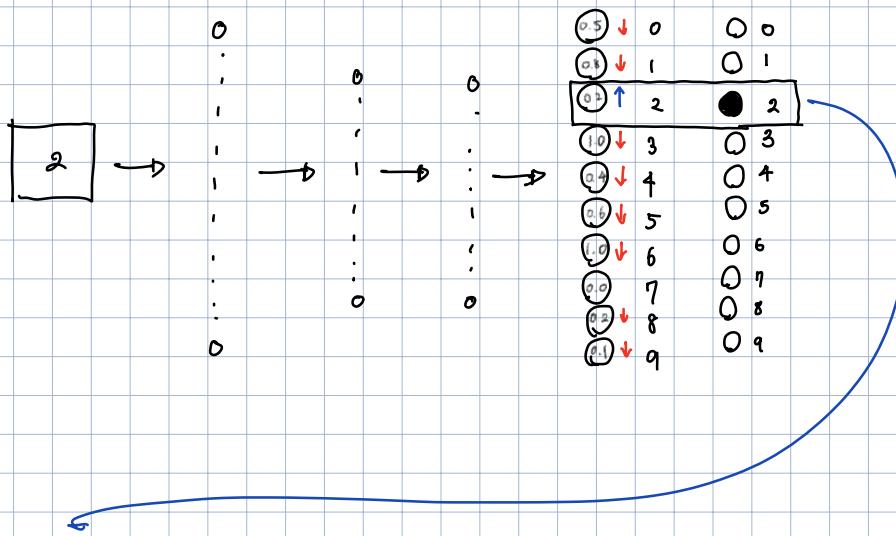


Cost of the function is 32 times
more sensitive to change this one
than 0.1 weight.



ex.) image of 2.

At first, initialized.



$$0.2 = \sigma(w_0a_0 + w_1a_1 + \dots + w_{n-1}a_{n-1} + b)$$

Our choice to change (nudge) 0.2 to be closer to 1?

- Increase b
- Increase w_i (in proportion to a_i)
- Change a_i

Hebbian theory

