18-Aug-2022

NEURAL NETS (karpathy)

A class of mathematical expressions

NN uses Backprop

BACKPROP

more general concept.
doesn't care about NN.

DERIVATIVE

of a fn.

how does the fn respond,
when we slightly 'bump up' the x?
with what sensitivity?
goes up? goes down? stays same?
↓ ↓ ↓
+ve slope | -ve slope | 0 slope

slope → raise/run → a/b

The func. (operation) inside neural nets can be very trivial like +, − or complex like 'tanh'

↳ As long as we can know DERIVATIVE of a func, we can BACKPROP through it

* ... A → B → C ... gradient @ A = $\frac{d}{dA}(B) \times \text{grad}(C)$

multiply derivative with gradient till then
i.e., neighbour downstream

this is the 'chain Rule' use

grad(C) × $\frac{dL}{dc}$ where L is final loss func.

CHAIN RULE

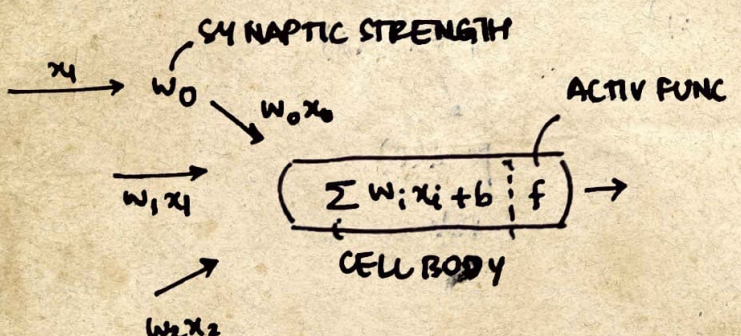
"if z depends on y
& y depends on x → $\frac{dz}{dx} = dy \cdot \frac{dz}{dy} \times \frac{dy}{dx}$
then z depends on x as well"

'product of rates of changes'

* when a single node gets back gradient from 2 or more nodes, we ACCUMULATE the gradient. Add them.

C → A, C → B

$\text{grad}(C) = \frac{dA}{dc} \times \frac{dL}{dA} + \frac{dB}{dc} \times \frac{dL}{dB}$



WEIGHTS

Synaptic strengths of each input

BIAS

Innate 'trigger happiness'
i.e., regardless of input, this makes it a bit more or bitless happy.

LOSS

A single number to evaluate performance of the whole neural network.