Shubham Kumar

2015098

HW-3

A1= No, we cannot build a neural network of arbitary length to learn XOR.

XOR is ~~a linear~~ non-linearly seprable. Neural network using just linear activation function is a generalized linear model

$\therefore$ XOR can't be classified by it.

$$y = f(w_2(w_1 x + b_1) + b_2)$$

It's same as SVM with linear function.

$$f(x) = B(Ax + a) + b$$
$$= \underbrace{BA}_{\text{constant}} x + \underbrace{Ba + b}_{\text{constant}}$$

$$\therefore f(x) = C_1 x + C_2$$

A2 = Sigmoid

Reasons :-

Sigmoid Saturation , Killing Gradients

When sigmoid neuron's activation saturates at either 0 or 1, the gradient at these regions is almost 0.
∴ no signal flow through neuron.

Remedy : ① Initialize the weights by extra one.
② ~~s~~ Take small value of weights initially.

Relu : It will perform better than sigmoid.

ReLU also have problems like dead neuron. It can be fragile during ~~trai~~ training and die.
Large gradients results in ~~[illegible]~~ update weight that gradient flowing could be zero.
This kills neurons.

Remedy ⇒ Keep the learning rate low.

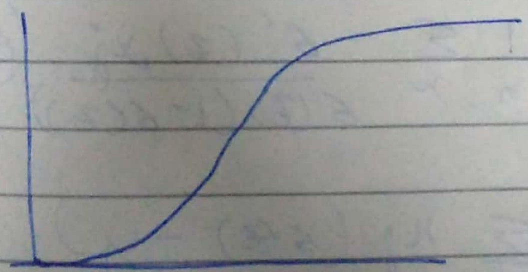
A3 = For quadratic, learning is slow when error is large.

$$C = \frac{(y-a)^2}{2}$$

$$a = \sigma(z)$$

$$\frac{\partial C}{\partial w} = (a-y)\sigma'(z)x = a\sigma'(z)$$

Illy $\frac{\partial C}{\partial b} = a\sigma'(z)$



z

∴ $\frac{\partial C}{\partial w}$, $\frac{\partial C}{\partial b}$ get very small

∴ slow learning.

Cross entropy cost,

$$C = -\frac{1}{n}\sum_x [y \ln a + (1-y)\ln(1-a)]$$

I. $\boxed{C > 0}$

II. If neuron's actual output is close to desired o/p, x, then cross entropy will be close to zero

3.

$$\frac{\partial C}{\partial w_j} = -\frac{1}{n}\sum\left(\frac{y}{\sigma(z)} - \frac{(1-y)}{1-\sigma z}\right)\sigma'(z)x_j$$

$$= \frac{1}{n}\sum_x \frac{\sigma'(z)x_j}{\sigma(z)(1-\sigma(z))}(\sigma(z)-y)$$

$$= \frac{1}{n}\sum_x x_j(\sigma(z)-y)$$

This tells rate at which weight learns is controlled by $\sigma(z) - y$

∴ Larger error ⇒ more it will learn.

Scanned by CamScanner