CS231n- Lecture 3

February 21, 2017

1 Optimization

1.1 Computational Graph

We need to see computational graph. It's huge in Convolutional Neural Networks and Neural Turing Machine.
$f(x, y, z) = (x + y)z$
e.g x=-2, y=5, z=-4
$q = x + y$
$\frac{\partial q}{\partial x} = 1$
$\frac{\partial q}{\partial y} = 1$ $f = qz$
$\frac{\partial f}{\partial q} = z$
$\frac{\partial f}{\partial z} = q$

We made a forward pass, now we'll make a backward one $\frac{\partial f}{\partial f} = 1$
$\frac{\partial f}{\partial z} = x + y = 3$
The influence of z on f is three times in positive magnitude
$\frac{\partial f}{\partial q} = z = -4$
if q increases by h, then f decreases by 4 times that magnitude $\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial y} = -4 * 1 = -4$
Similarly, $\frac{\partial f}{\partial x} = -4$
Example: $f(w, x) = \frac{1}{1+e^{(-(w_0x_0+w_1x_1+w_2))}}$

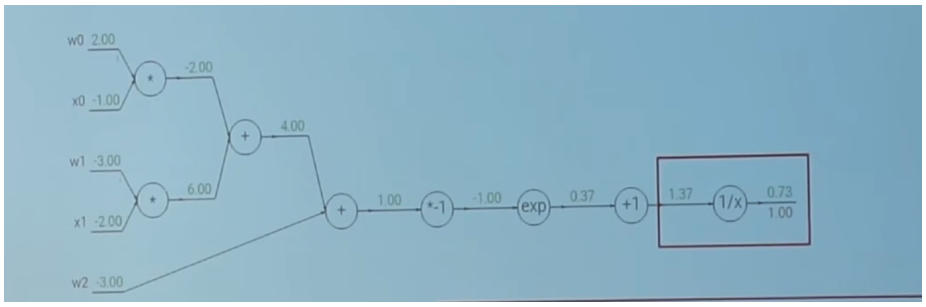

Figure 1: The Computational Graph

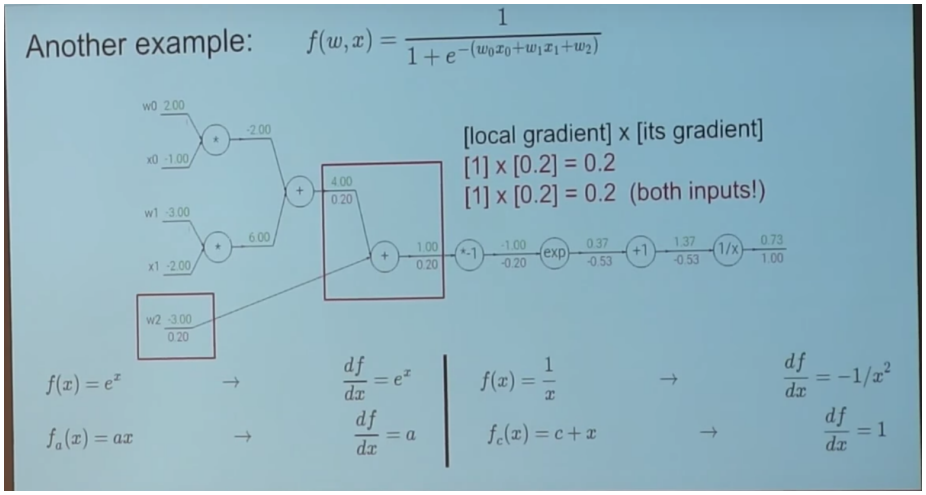

Figure 2: The Computational Graph after a few updates(AdditionGate)

$f(x) = e^x -> \frac{\partial f}{\partial x} = \frac{1}{x}$
$f(x) = \frac{1}{x} -> \frac{\partial f}{\partial x} = \frac{-1}{x^2}$
So we get gradient at this position to be $\frac{-1}{1.37^2} * 1.00 = -0.53$
Next level we get $1 * (-0.53) = -0.53$
This goes on and on, we multiply local gradient with the gates gradient

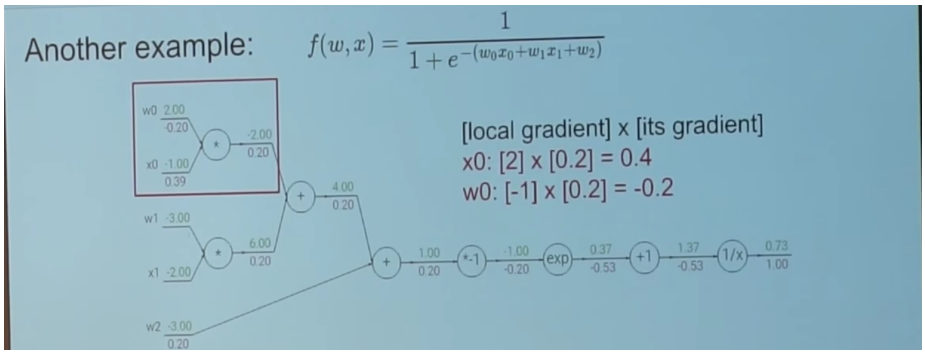

Figure 3: The Computational Graph after a few updates(MultGate)

We can collapse into one sigmoid function
From now on rely on slides from "http://cs231n.stanford.edu/slides" only extra notes here.
Understanding backward flow's intuition is very important.
add gate is a gradient distributor. distributes equally
max gate is a gradient router, larger one gets all the smaller gets 0. in backprop we are routing to the max value since only it contributes to the final thing. If equal, we just pick one but odds of it happening are rare.
mul gate: gradient switcher
there is never any loop in this. they are always DAGs
Caching helps a lot sometimes
We need to do forward and backward for every gate. Forward computes loss, backwards analytical gradient and then updates.