

Numerical approximation of gradients

(DESCRIPTION)

Text, Setting up your optimization problem. Numerical approximation of gradients. Website, deep learning, dot, A.I.

(SPEECH)

When you implement back propagation you'll find that there's a test called creating checking that can really help you make sure that your implementation of back prop is correct.

Because sometimes you write all these equations and you're just not 100% sure if you've got all the details right and internal back propagation.

So in order to build up to gradient and checking, let's first talk about how to numerically approximate computations of gradients and in the next video, we'll talk about how you can implement gradient checking to make sure the implementation of backdrop is correct.

So

(DESCRIPTION)

New slide, Checking your derivative computation.

(SPEECH)

lets take the function f and replot it here and remember this is f of θ equals θ cubed, and let's again start off to some value of θ .

Let's say θ equals 1.

Now instead of just nudging θ to the right to get $\theta + \epsilon$, we're going to nudge it to the right and nudge it to the left to get $\theta - \epsilon$, as was $\theta + \epsilon$.

So this is 1, this is 1.01, this is 0.99 where, again, ϵ is same as before, it is 0.01.

It turns out that rather than taking this little triangle and computing the height over the width, you can get a much better estimate of the gradient if you take this point, f of $\theta - \epsilon$ and this point, and you instead compute the height over width of this bigger triangle.

So for technical reasons which I won't go into, the height over width of this bigger green triangle gives you a much better approximation to the derivative at θ .

And you saw it yourself, taking just this lower triangle in the upper right is as if you have two triangles, right?

This one on the upper right and this one on the lower left.

And you're kind of taking both of them into account by using this bigger green triangle.

So rather than a one sided difference, you're taking a two sided difference.

So let's work out the math.

This point here is F of $\theta + \epsilon$.

This point here is F of $\theta - \epsilon$.

So the height of this big green triangle is f of $\theta + \epsilon$ minus f of $\theta - \epsilon$.

And then the width, this is 1ϵ , this is 2ϵ .

So the width of this green triangle is 2ϵ .

So the height of the width is going to be first the height, so that's F of θ plus ϵ minus F of θ minus ϵ divided by the width.

So that was 2ϵ which we write that down here.

And this should hopefully be close to g of θ .

So plug in the values, remember f of θ is θ^3 .

So this is $\theta + \epsilon$ is 1.01.

So I take a cube of that minus 0.99 θ^3 of that divided by 2×0.01 .

Feel free to pause the video and practice in the calculator.

You should get that this is 3.0001.

Whereas from the previous slide, we saw that g of θ , this was $3\theta^2$ so when θ was 1, so these two values are actually very close to each other.

The approximation error is now 0.0001.

Whereas on the previous slide, we've taken the one sided of difference just $\theta + \epsilon$ we had gotten 3.0301 and so the approximation error was 0.03 rather than 0.0001.

So this two sided difference way of approximating the derivative you find that this is extremely close to 3.

And so this gives you a much greater confidence that g of θ is probably a correct implementation of the derivative of F .

When you use this method for grading, checking and back propagation, this turns out to run twice as slow as you were to use a one-sided difference.

It turns out that in practice I think it's worth it to use this other method because it's just much more accurate.

The little bit of optional theory for those of you that are a little bit more familiar of Calculus, it turns out that, and it's okay if you don't get what I'm about to say here.

But it turns out that the formal definition of a derivative is for very small values of ϵ is f of $\theta + \epsilon$ minus f of θ minus ϵ over 2ϵ .

And the formal definition of derivative is in the limits of exactly that formula on the right as ϵ goes to 0.

And the definition of limit is something that you learned if you took a Calculus class but I won't go into that here.

And it turns out that for a non zero value of ϵ , you can show that the error of this approximation is on the order of ϵ^2 , and remember ϵ is a very small number.

So if ϵ is 0.01 which it is here then ϵ^2 is 0.0001.

The big O notation means the error is actually some constant times this, but this is actually exactly our approximation error.

So the big O constant happens to be 1.

Whereas in contrast if we were to use this formula, the other one, then the error is on the order of ϵ .

And again, when ϵ is a number less than 1, then ϵ is actually much bigger than ϵ^2 which is why this formula here is actually much less accurate approximation than this formula on the left.

Which is why when doing gradient checking, we rather use this two-sided difference when you compute $f(\theta + \epsilon)$ minus $f(\theta - \epsilon)$ and then divide by 2ϵ rather than just one sided difference which is less accurate.

If you didn't understand my last two comments, all of these things are on here.

Don't worry about it.

That's really more for those of you that are a bit more familiar with Calculus, and with numerical approximations.

But the takeaway is that this two-sided difference formula is much more accurate.

And so that's what we're going to use when we do gradient checking in the next video.

So you've seen how by taking a two sided difference, you can numerically verify whether or not a function g , g of θ that someone else gives you is a correct implementation of the derivative of a function f .

Let's now see how we can use this to verify whether or not your back propagation implementation is correct or if there might be a bug in there that you need to go and tease out