Adam optimization algorithm

(SPEECH)
During

(DESCRIPTION)
Text, Optimization Algorithms. Adam optimization algorithm. Website, deep learning, dot, A.I.

(SPEECH)
the history of deep learning, many researchers including some very well-known researchers, sometimes proposed optimization algorithms and showed that they worked well in a few problems.

But those optimization algorithms subsequently were shown not to really generalize that well to the wide range of neural networks you might want to train.

So over time, I think the deep learning community actually developed some amount of skepticism about new optimization algorithms.

And a lot of people felt that gradient descent with momentum really works well, was difficult to propose things that work much better.

So, rms prop and the Adam optimization algorithm, which we'll talk about in this video, is one of those rare algorithms that has really stood up, and has been shown to work well across a wide range of deep learning architectures So, this is one of the algorithms that I wouldn't hesitate to recommend you try because many people have tried it and seen it work well on many problems.

And the Adam optimization algorithm is basically taking momentum and rms prop and putting them together.

So, let's see how that works.

(DESCRIPTION)
New slide, Adam optimization algorithm.

(SPEECH)
To implement Adam you would initialize: Vdw=0, Sdw=0, and similarly Vdb, Sdb=0.

And then on iteration T, you would compute the derivatives: compute dw, db using current mini-batch.

So usually, you do this with mini-batch gradient descent.

And then you do the momentum exponentially weighted average. So Vdw = ß.

But now I'm going to this ß1 to distinguish it from the hyper parameter ß2 we'll use for the rms prop proportion of this.

So, this is exactly what we had when we're implementing momentum except it now called hyper parameter ß1 instead of ß.

And similarly, you have VDB as follows: 1 - ß1 x db.

And then you do the rms prop update as well.

So now, you have a different hyperparemeter ß2 plus one minus ß2 dw^2.

And again, the squaring there is element y squaring of your derivatives dw.

And then sdb is equal to this plus one minus ß2 times db.

So this is the momentum like update with hyper parameter ß1 and this is the rms prop like update with hyper parameter ß2.

In the typical implementation of Adam, you do implement bias correction.

So you're going to have v corrected.

Corrected means after bias correction.

Dw = vdw divided by 1 minus ß1 to the power of T if you've done T iterations.

And similarly, vdb corrected equals vdb divided by 1 minus ß1 to the power of T.

And then similarly, you implement this bias correction on S as well.

So, that's sdw divided by one minus ß2 to the T and sdb corrected equals sdb divided by 1 minus ß2 to the T.

Finally, you perform the update.

So W gets updated as W minus alpha times.

So if you're just implementing momentum you'd use vdw, vw or maybe vdw corrected.

But now, we add in the rms prop portion of this.

So we're also going to divide by square roots of sdw corrected plus epsilon.

And similarly, B gets updated as a similar formula, vdb corrected, divided by square root S, corrected, db, plus epsilon.

And so, this algorithm combines the effect of gradient descent with momentum together with gradient descent with rms prop.

And this is a commonly used learning algorithm that is proven to be very effective for many different neural networks of a very wide variety of architectures.

So, this algorithm has a number of hyper parameters.

(DESCRIPTION)
New slide, Hyperparameters choice.

(SPEECH)
The learning with hyper parameter alpha is still important and usually needs to be tuned.

So you just have to try a range of values and see what works.

A common choice really the default choice for ß1 is 0.9.

So this is a moving average, weighted average of dw right this is the momentum light term.

The hyper parameter for ß2, the authors of the Adam paper, inventors of the Adam algorithm recommend 0.999.

Again this is computing the moving weighted average of dw2 as well as db squares.

And then Epsilon, the choice of epsilon doesn't matter very much.

But the authors of the Adam paper recommended it 10 to the minus 8.

But this parameter you really don't need to set it and it doesn't affect performance much at all.

But when implementing Adam, what people usually do is just use the default value.

So, ß1 and ß2 as well as epsilon.

I don't think anyone ever really tunes Epsilon.

And then, try a range of values of Alpha to see what works best.

You could also tune ß1 and ß2 but it's not done that often among the practitioners I know.

So, where does the term 'Adam' come from?

Adam stands for Adaptive Moment Estimation.

So ß1 is computing the mean of the derivatives.

This is called the first moment.

And ß2 is used to compute exponentially weighted average of the ²s and that's called the second moment.

So that gives rise to the name adaptive moment estimation.

But everyone just calls it the Adam authorization algorithm.

And, by the way, one of my long term friends and collaborators is call Adam Coates.

As far as I know, this algorithm doesn't have anything to do with him, except for the fact that I think he uses it sometimes.

But sometimes I get asked that question, so just in case you're wondering.

So, that's it for the Adam optimization algorithm.

With it, I think you will be able to train your neural networks much more quickly.

But before we wrap up for this week, let's keep talking about hyper parameter tuning, as well as gain some more intuitions about what the optimization problem for neural networks looks like.

In the next video, we'll talk about learning rate decay.