02:21:06 learning rate scheduler: warmup + cosine decay

rate schedule that's used here in gpt3 is what's called a cosine Decay learning schedule with warmup and the way this looks is that the learning rate is basically starts right at around zero linearly rank s up over some amount of time and then comes down with this cosine sort of form and comes down to some kind of a minimum learning rate that's up to you so here the minimum learning rate is zero but uh here in the paper they said that they use cosine Decay for learning rate down to 10% of its value over the first 260 billion tokens and then training continues 10% after and there's a linear warmup over the first 375 million tokens so that's about the learn R so let's now implement this uh so I already implemented it here and the way this works is let me scroll down first here I changed our training Loop a little bit so this was a 4i in

Max steps I just change it to step now so that we have the notion of a step is a single optimization step in the in the for Loop and then here I get the LR for this step of the optimization using a new function I call get LR and then in pytorch to set the learning rate I think this is is the way to set the learning rate it's a little bit gnarly um because you have to basically there's a notion of different par parameter groups that could exist in the optimizer and so you actually have to iterate over them even though we currently have a single param group only um and you have to set the LR in this for Loop kind of style is is my impression right now so we have this look of LR we set the learning rate and then on the bottom I'm also printing it uh so that's all the changes I made to this Loop and then of course the get LR is my scheduler now it's worth pointing out that pytorch actually has learning rate schedulers and you can use them and I believe there's a cosine learning rate schedule in pytorch I just don't really love using that code because honestly it's like five lines of code and I fully understand what's happening inside these lines so I don't love to use abstractions where they're kind of in screwable and then I don't know what they're doing so personal style so the max learning rate here is let's say 3 E4 but we're going to see that in gpt3 here they have a table of what the maximum learning rate is for every model size so um for for this one basically 12 12 layer 768 gpt3 so the gpt3 small is roughly like a GPT 2124m we see that here they use a learning rate of 6 E4 so we could actually go higher um in fact we may want to try to follow that and just set the max LR here at six uh then the that's the maximum learning rate the minum learning rate is uh 10% of that per description in the paper some number of steps that we're going to warm up over and then the maximum steps of the optimization which I now use also in the for Loop down here and then you can go over this code if you like it's not U it's not terribly inside Flor interesting I'm just uh modulating based on the iteration number which learning rate uh there should be so this is the warm-up region um this is the region after the optimization and then this is the region sort of in between and this is where I calculate the cosine learning rate schedule and you can step through this in detail if you'd like uh but this is basically implementing this curve and I ran this already and this is what that looks like um so when we now run we start at um some very low number now note that we don't start exactly at zero because that would be not useful to update with a learning rate of zero that's why there's an it+ one so that on the zeroth iteration we are not using exactly zero we're using something very very low then we linearly warm up to maximum learning rate which in this case was 34 when I ran it but now would be 6 E4 and then it starts to decay all the way down to um 3 E5 which was at the time 10% of the original learning rate now one thing we are not following exactly is that they mentioned that um let me see if I can find it again we're not exactly following what they did because uh they mentioned that their training Horizon is 300 billion tokens and they come down to 10% of the initial learning rate of at 260 billion and then they train after 260 with 10% so basically their Decay time is less than the max steps time whereas for us they're exactly equal so it's not exactly faithful but it's um it's an okay um this is okay for us and for our purposes right now and um we're just going to use this ourselves I don't think it makes too too big of a difference honestly I should point out that what learning rate schedule you use is totally up to you there's many different types um coign learning rate has been popularized a lot by gpt2 and gpt3 but people have come up with all kinds of uh other learning rate schedules um and this is kind of like an active area of uh research as to which one is the most effective at train these networks okay next up the paper talks