00:41:47 sample, auto-detect the device

random model is actually fairly straightforward because pytorch already initializes our model randomly and by default so when we create the GPT model and the Constructor this is all um all of these layers and modules have random initializers that are there by default so when these linear layers get created and so on there's default Constructors for example using the Javier initialization that we saw in the past uh to construct the weights of these layers and so creating a random model instead of a gpt2 model is actually fairly straightforward and we would just come here and instead we would create model equals GPT and then we want to use the default config GPT config and the default config uses the 124 M parameters so this is the random model initialization and we can run it and we should be able to get uh results now the results here of course are total garbage carbal and that's because this is random model and so we're just getting all these random token string pieces chunked up totally at random so that's what we have right now uh now one more thing I wanted to point out by the way is in case you do not have Cuda available because you don't have a GPU you can still follow along with uh with what we're doing here uh to some extent uh and probably not to the very end because by the end we're going to be using multiple gpus and actually doing a serious training run uh but for now you can actually follow along decently okay uh so one thing that I like to do in pytorch is I like to autod detect the device that is available to you so in particular you could do that like this so here we are trying to detect a device to run on that has the highest compute capability you can think about it that way so by default we start with CPU which of course is available everywhere because every single computer will have a CPU but then we can try to detect do you have a GPU you so use a Cuda and then if you don't have a Cuda uh do you at least have MPS MPS is the back end for Apple silicon so if you have a Macbook that is fairly new you probably have apple silicon on the inside and then that has a GPU that is actually fairly capable uh depending on which MacBook you have and so you can use MPS which will be potentially faster than CPU and so we can print the device here now once we have the device we can actually use it in place of Puda so we just swap it in and notice that here when we call model on X if this x here is on CPU instead of GPU then it will work fine because here in the forward which is where P to will come when we create a pose we were careful to use the device of idx to create this tensor as well and so there won't be any mismatch where one tensor is on CPU one is on GPU and uh that you can't combine those but here we are um carefully initializing on the correct device as indicated by the input to this model so this will autod detect device for me this will be of course GPU so using device Cuda uh but uh you can also run with um as I mentioned another device and it's not going to be too much slower so if I override device here oops if I override device equals CPU then we'll still print Cuda of course but now we're actually using CPU one 2 3 4 5 6 okay about 6 seconds and actually we're not using torch compile and stuff like that which will speed up everything a lot faster as well but you can follow even on a CPU I think to a decent extent um so that's note on that okay so I do want to loop around eventually into what it means to have different devices in pytorch and what it is exactly that pytorch does in the background for you when you do something like module. 2 device or where you take a torch tensor and do A2 device and what exactly happens and how that works but for now I'd like to get to training and I'd like