00:33:31 sampling init, prefix tokens, tokenization

try to set up the identical thing on the left here that matches hug and face on the right so here we've sampled from the pipeline and we sampled five times up to 30 tokens with the prefix of hello I'm a language model and these are the completions that we achieved so we're going to try to replicate that on the left here so number turn sequences is five max length is 30 so the first thing we do of course is we initialize our model then we put it into evaluation mode now this is a good practice to put the model into eval when you're not going to be training it you're just going to be using it and I don't actually know if this is doing anything right now for the following reason our model up above here contains no modules or layers that actually have a different uh Behavior at training or evaluation time so for example Dropout batch norm and a bunch

of other layers have this kind of behavior but all of these layers that we've used here should be identical in both training and evaluation time um so so potentially model that eval does nothing but then I'm not actually sure if this is the case and maybe pytorch internals uh do some clever things depending on the evaluation mode uh inside here the next thing we're doing here is we are moving the entire model to Cuda so we're moving this all of the tensors to GPU so I'm sshed here to a cloud box and I have a bunch of gpus on this box and here I'm moving the entire model and all of its members and all of its tensors and everything like that everything gets shipped off to basically a whole separate computer that is sitting on the GPU and the GPU is connected to the uh CPU and they can communicate but it's basically a whole separate computer with its own computer architecture and it's really well catered to

parallel processing tasks like those of running neural networks so I'm doing this so that the model lives on the GPU a whole separate computer and it's just going to make our code a lot more efficient because all of this stuff runs a lot more efficiently on the gpus so that's the model itself now uh the next thing we want to do is we want to start with this as the prefix when we do the generation so let's actually create those prefix tokens so here's the code that I've written we're going to import the tich token library from open Ai and we're going to get the gpt2 encoding so that's the tokenizer for gpt2 and then we're going to encode this string and get a list of integers which are the tokens uh now these integers here should actually be fairly straightforward because we can just copy paste this string and we can sort of inspect what it is in tick tokenizer so just pasting that in these are the tokens that are going to come out so this list of

integers is what we expect tokens to become and as you recall if you saw my video of course all the tokens they're just little string chunks right so these are this is the chunc of this string into gpt2 tokens so once we have those tokens it's a list of integers we can create a torch tensor out of it in this case it's eight tokens and then we're going to replicate these eight tokens for five times to get five rows of eight tokens and that is our initial um input X as I call it here and it lives on the GPU as well so X now is this idx that we can put into forward to get our logits so that we know what comes as the sixth token uh sorry as the ninth token in every one of these five rows okay and we are now