For this project I used PyTorch. I start off by defining a batch size as 1 and my program can only handle that since I was never able to understand how to adapt my output and input value to the loss function correctly. I also send my tensors to the GPU if it is available to quicken the process. Then I created a Function called one\_hot to create a tensor of my input sequences. Next I went through and created my RNN model. I initialized hit with a hidden\_size of 300 and hidden\_layer of 2. I then initialize my RNN and run through the forward process and initialize a tensor. Next I have the predict function which allows me to have the RNN model predict the next character and the sample function uses that prediction to create a replication of Shakespeare.

In my programs main code we read in the text with 100 characters at a time, this allows the program to also recognize “\n” to skip lines. I create a set with these sentences so my RNN model knows the options of answers. Then I create dictionaries to get from int to chars and vice versa. Then I create input and target sequences by removing the first and last characters of the 100 character separations they have. Then I use the dictionaries to assign the characters to numbers. I use adam as my optimizer but changed the learning rate to .5 for a little more accuracy. I also use CrossEntropyLoss for my loss function. I only print off the loss of my function every 100 Loops to decrease the amount of outputs and increase speed. There is an extra output at the end of my program that simply tells you how long the program took, which can be useful to know if the program I taking too long and what your epoch amount should be.

Here is my output and what the model generates when allowed 50 characters with the starting word “MENE” (I was curious if it could finish the name correctly). One last important note is that I used three epochs here.

Text

Description automatically generated