Q2 - RNN

Same as the previous part I've started out from the architecture which was given to us in the tutorial.

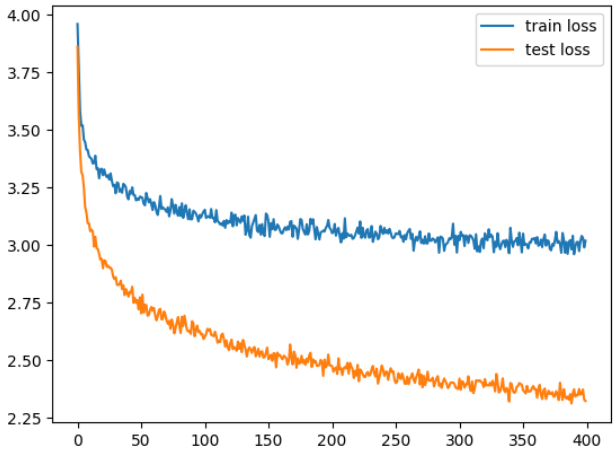
Because were generating a letter by using the letters the came before and the country of origin, instead of only combining the hidden state and the letter, I've also added the country of origin(as a one hot tensor) to the first linear layer. In addition I've added another step in which I **combine the hidden state with the output** and then apply another linear layer , after that I use a **dropout** function.

In this program I represented every word as a tensor of letters in which every letter is a one hot encoder vector for all possible letters(Small, Capital and . , ; ' ").

Each iteration(epoch) trains over a single name and every training iteration goes over every letter of the word and tries to predict the next one, taking into consideration the hidden state and the country in which the name came from. At the beginning I've use 10K iterations, but the names generated were too far off from real names, so I doubled the amount to **20K**.

furthermore, I've doubled the **hidden state size to 256**

Here is the loss graph(without the accuracy graph as mentioned in the assignment's forum)



And here are 5 examples of generated names:

Letter	Country	Name
g	French	Gare
L	Czech	Louch
L	Polish	Loawk
D	Irish	Deana
J	Russian	Jakin

[1]Maniatopoulos, Andreas-Antonios & Mitianoudis, Nikolaos. (2021). Learnable Leaky ReLU (LeLeLU): An Alternative Accuracy-Optimized Activation Function. Information. 12. 10.3390/info12120513.