Theory and Practice of Deep Learning

Chang Jun Qing 1002088

Week 9 Coding HW

1 Quick README

Main code is hw.py where training occurs there
sample.py is a playground code where one can run it in the format

```
python sample.py <starting character(s)>
```

that would produce sentences with the respective starting characters
outputs.txt contains all the samples across epochs

2 Train and Test

2.1 Overview

Training was done on a Windows Desktop with a GTX1060 running python 3.6.6 and torch 1.0.0
Sample.py was tested on a Macbook pro with torch 1.0.1.post2 and python 3.6.8

Training was done in 2 phases. Initially, learning rate was set at 0.0001, however, after a few epochs, loss can be seen to be increasing over time instead after the 1st epoch. So training was stopped and the 1 epoch model was loaded and trained again. This time with a learning rate of 0.00001 as well as a learning rate decay of 0.1 times each epoch.
Some other hyperparameters used were:

- # of layers: 2
- # of hidden dim: 200
- # iterations before sampling: 3000
- Criterion: Negative Log Likelihood

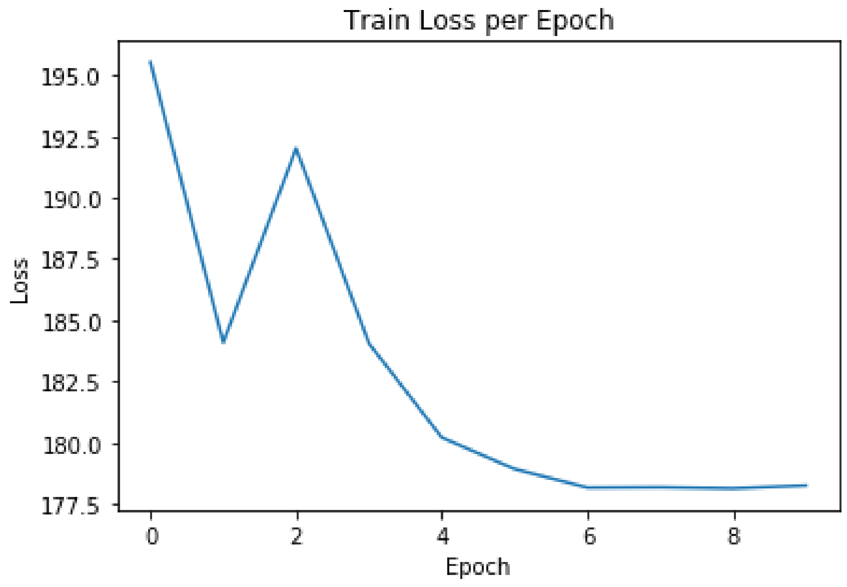

Figure 1: Train Loss

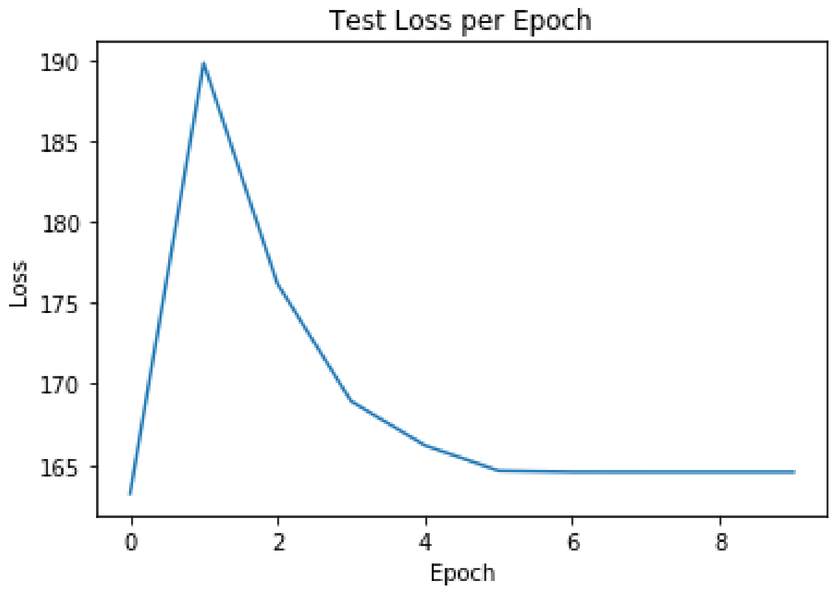

Figure 2: Test Loss

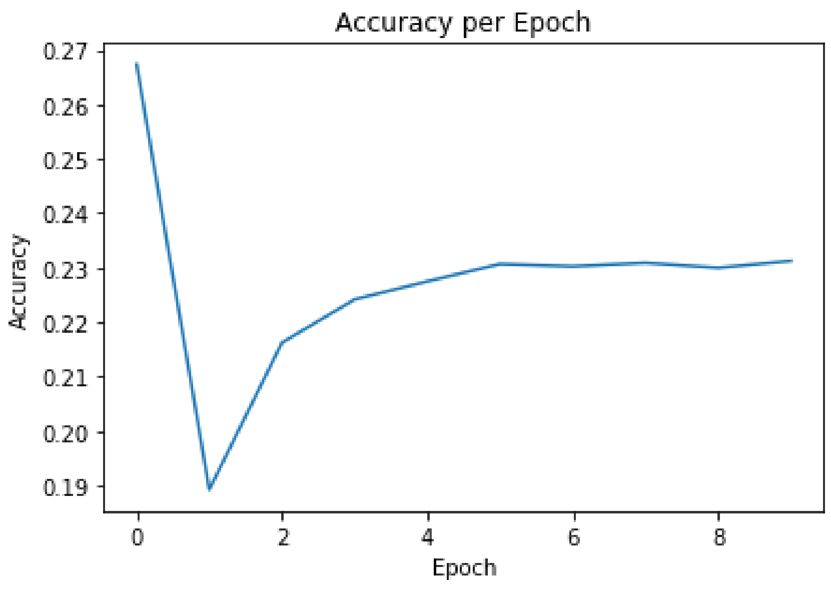

Figure 3: Accuracy

2.2 Results Discussion

As we can see there are some random spikes and falls in the first 2 epochs. This is likely due to the 2 phase training. For the accuracy, one reason why accuracy might be low/random is due to the fact that the comparison of next character that I did was not done using argmax. Instead, I sampled the next character similar to how sampling is done.
Overall, we can see that the loss plateau, which means the model converged.
Base on test loss, the best model is actually that of the first epoch, however, because it is the first epoch, sentence structuring might not be very proper, thus I have provided both the model from the first epoch, as well as that of the last epoch.

3 Some Samplings

My personal favourite is

```
OYIE: It's in.
```

which happened in the 2nd epoch as it is the first properly worded sentence that happen.
With the model at epoch 10, using the K or S at the starting has a very high chance of the model using KIRK and SPOCK as the person talking. So we can do this

```
python sample.py KSKSKS
```

which will give us a conversation
KIRK: Ete se to metom the and ar to mase s, and.
SPOCK: He to the cortn mel stre the beise lee, the an to mane a s you ges the un of dan and a fean al
KIRK: Au the we he sor of of ar that san and hee, jutand stun of be. I of of a se of fom alt alcher i
SPOCK: The cin al nakit. To of the I that wad andte the are con alt s of the that ne sle it the I a i
KIRK: Marase of he me be I a undt aral is the me of me ne an thowouvene ande beany on of is that thon
SPOCK: Womsaisaningondk penounche that aner womut of snit a is mor feat the wun the be a of you is al

As we can see, the sentences do not make much sense this is due to the fact that we are doing character level generation. A solution to this would be to do word by word prediction which will require word embedding