Theory and Practices of Deep Learning

Chang Jun Qing 1002088

Week 8 Coding HW Report

1 Quick README

Code is split into 5 different files and are as follows
utils.py - Code provided on the tutorial written into functions
model.py - LSTM model used
train.py - train function
test.py - validation function as well as eval function for confusion matrix
run.py - main body of code that runs, edit hyperparameters here
Code tested on MacbookPro with Python 3.6.8 and PyTorch version 1.0.1.post2

2 Task 1

2.1 Results analysis

In this task of trying 3 different hidden layer size with 1 or 2 layers I have chosen to use the sizes 64, 128, 256. The other hyperparameters chosen are 100000 iterations, learning rate of 0.01, and the graph is plot every 1000 iterations. In a ideal scenario, multiple learning rates should be tested with the best rate used for testing. From the results we can see that all the loss have a similar shape which decreases. For 1 layer results, overall, the accuracy across the results for is around 66% This accuracy however can possibly improved as it can be seen from the training loss that we can train more to make it plateau further. For 2 layer results, the accuracy is about 55-60%. Similarly we can see that at 100000 iteration the 2 layer LSTM seems to not plateau. Thus, more training might have been better for it.
We should also note that while more training might make the 2 layer LSTM better, at the current settings of 100000 iterations, it already more time than that of a 1 layer LSTM. Thus, 2 layer LSTM might not be very cost efficient.

2.2 Graphs - 1 Layer

2.2.1 Size 64

Accuracy of Model: 6713/10000 (67.13%)

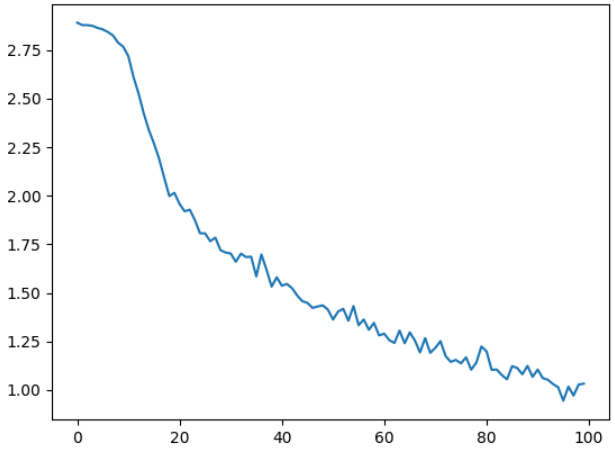

Figure 1: 1-64 Layer Loss

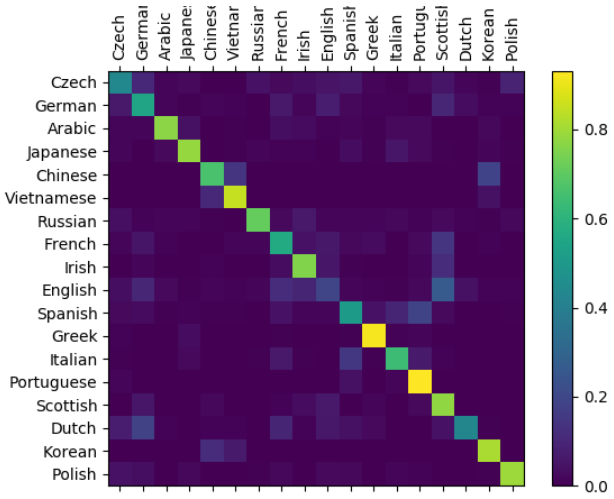

Figure 2: 1-64 Layer Confusion Matrix

2.2.2 Size 128

Accuracy of Model: 6620/10000 (66.2%)

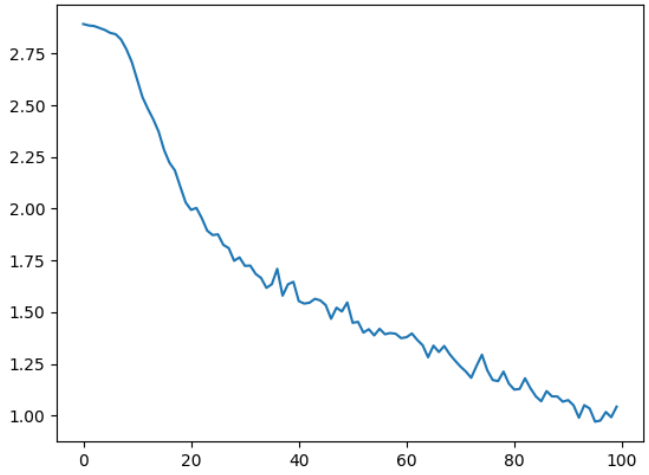

Figure 3: 1-128 Layer Loss

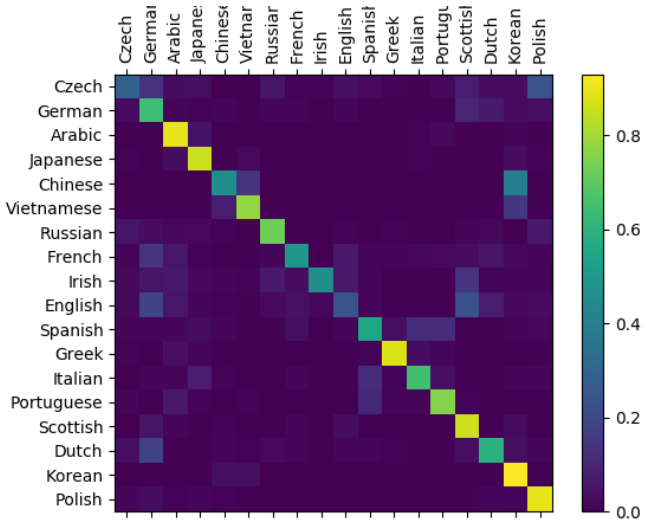

Figure 4: 1-128 Layer Confusion Matrix

2.2.3 Size 256

Accuracy of Model: 6612/10000 (66.12%)

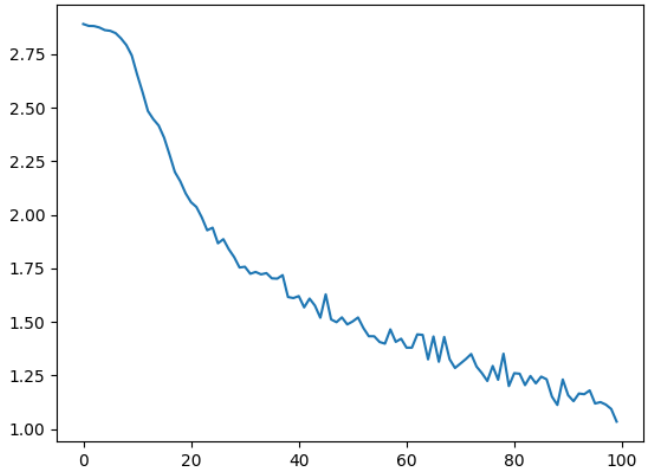

Figure 5: 1-256 Layer Loss

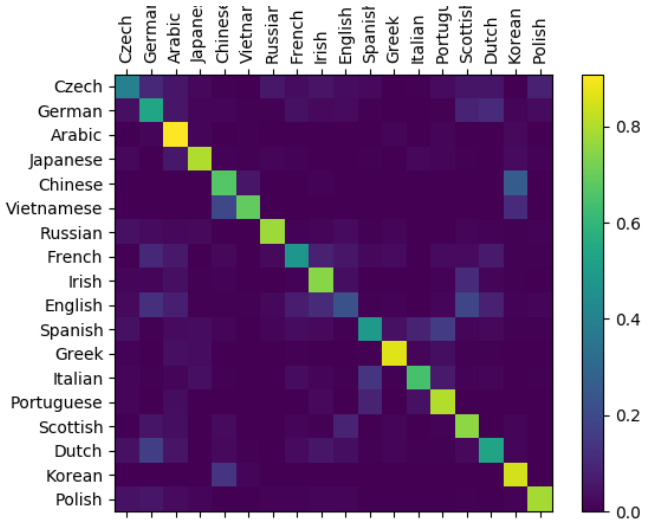

Figure 6: 1-256 Layer Confusion Matrix

2.3 Graphs - 2 Layer

2.3.1 Size 64

Accuracy of Model: 5879/10000 (58.79%)

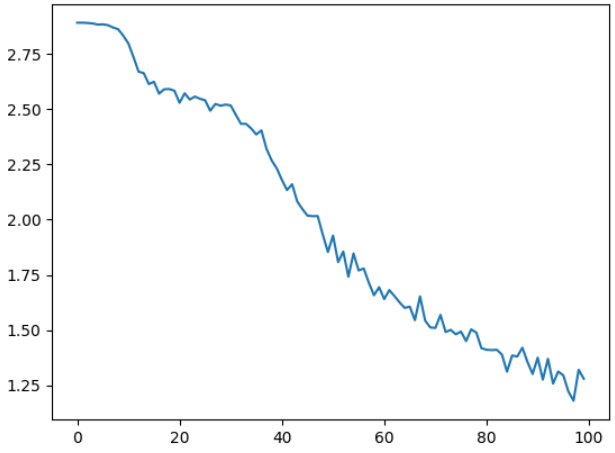

Figure 7: 2-64 Layer Loss

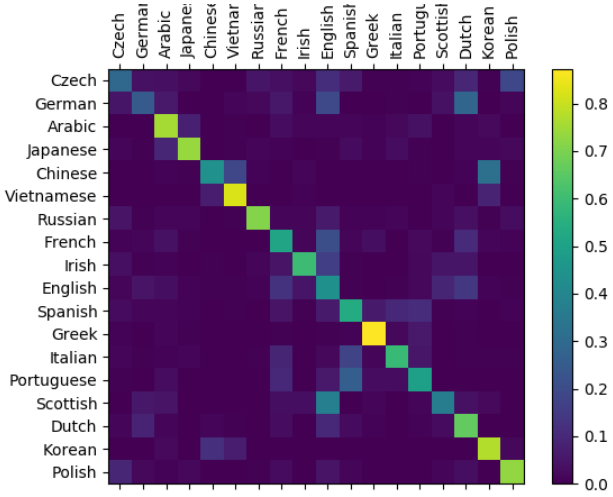

Figure 8: 2-64 Layer Confusion Matrix

2.3.2 Size 128

Accuracy of Model: 5519/10000 (55.19%)

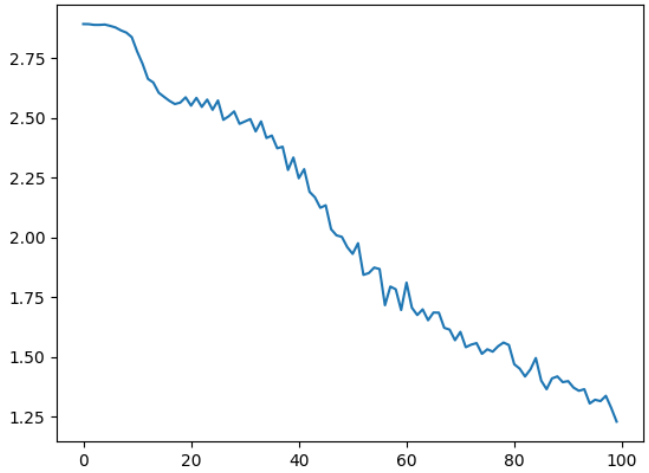

Figure 9: 2-128 Layer Loss

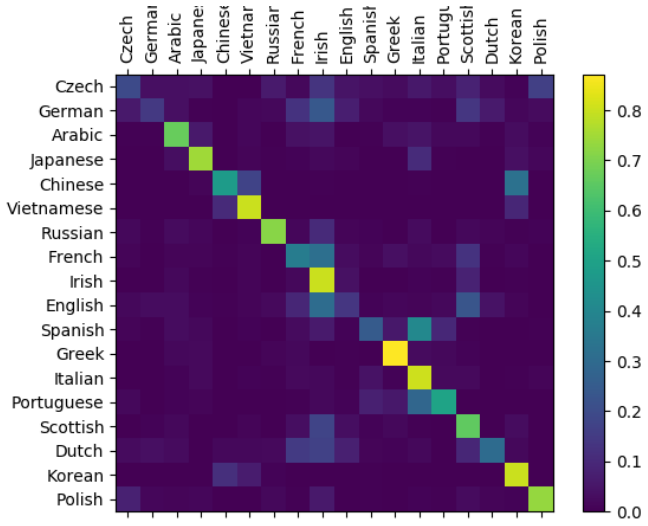

Figure 10: 2-128 Layer Confusion Matrix

2.3.3 Size 256

Accuracy of Model: 5556/10000 (55.56%)

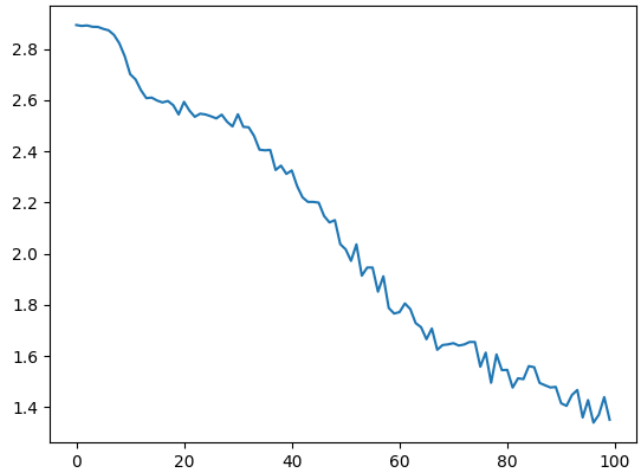

Figure 11: 2-256 Layer Loss

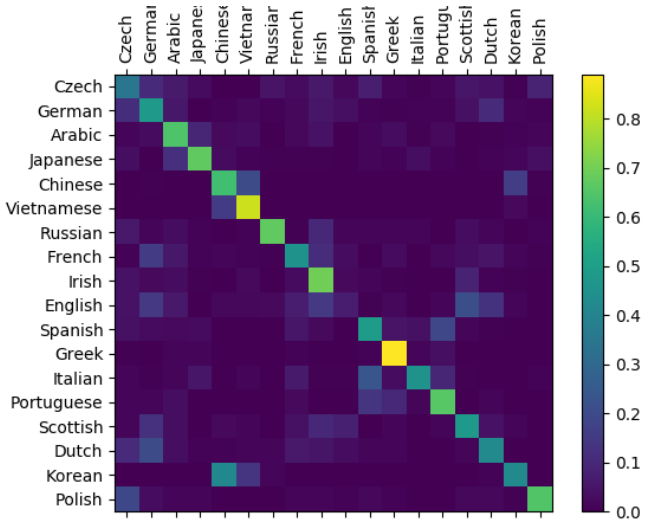

Figure 12: 2-256 Layer Confusion Matrix

3 Task 2

For this task, I chose the model with 1 layer and each layer of 64 size as amongst the models done in Task 1, it has the highest accuracy and is the most cost efficient. From the results, batch size 1 seems to work the best. for batch size 10 and 30, they both have the same issue where only the first item of the batch can be predicted correctly, and the rest of the batch is predicted as the same language even with 2 different implementations of selecting the batch. This causes the images that will be shown later, where there is a dim diagonal on the confusion matrix but most of the prediction is on one class.
It could likely be due to an implementation error that caused it as it seems like the weights are not updating properly. Searching online for many hours and trying different method did not work. However, I would be interested to learn how to implement the batch properly.

Figure 13: Challenge

Figure 14: Challenge

For batch size 1, it is done in Task 1. For batch size 10 and 30, the following are the results.

3.1 Batch Size 10

Accuracy of Model: 20148/100000 (20.148%)

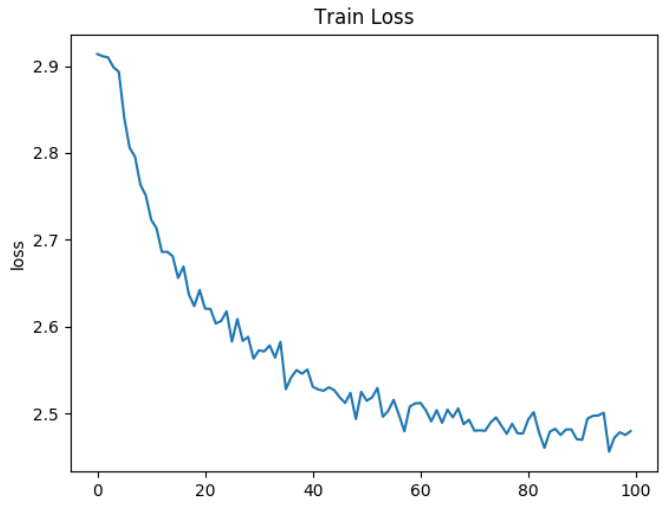

Figure 15: 1-64 Layer (Batch Size 10) Training Loss

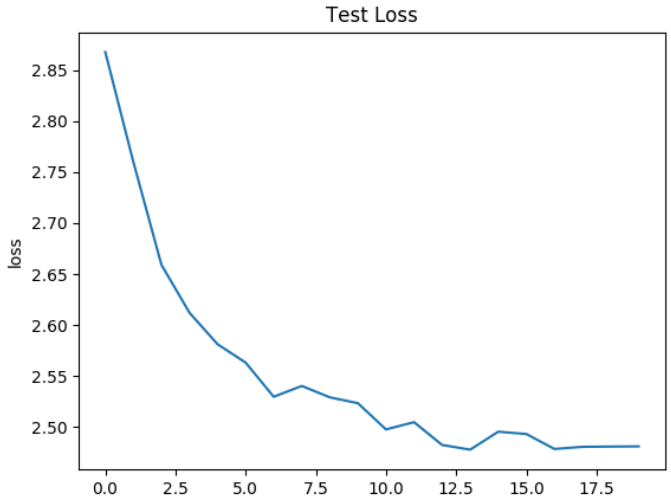

Figure 16: 1-64 Layer (Batch Size 10) Test Loss

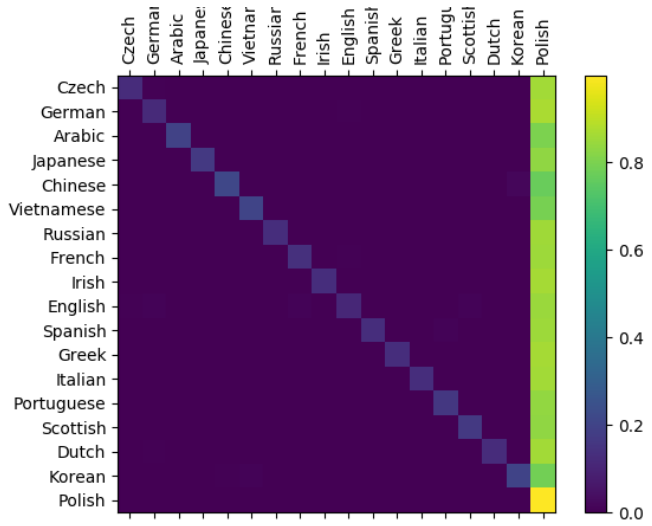

Figure 17: 1-64 Layer (Batch Size 10) Confusion Matrix

3.2 Batch Size 30

Accuracy of Model: 33133/300000 (11.044333333333332%)

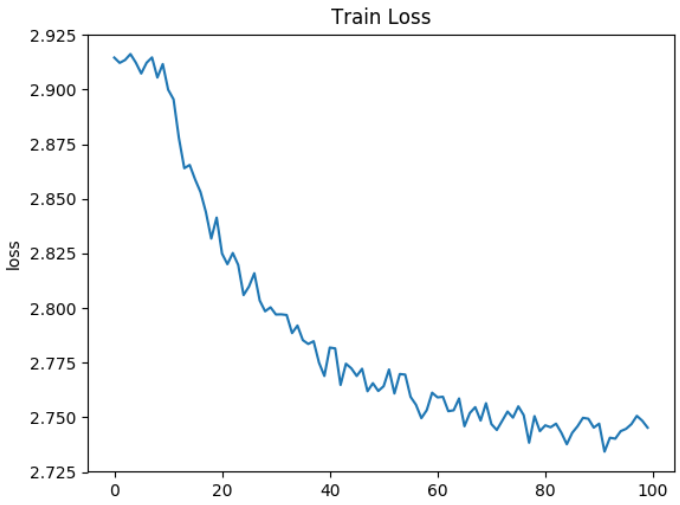

Figure 18: 1-64 Layer (Batch Size 30) Training Loss

Figure 19: 1-64 Layer (Batch Size 30) Test Loss

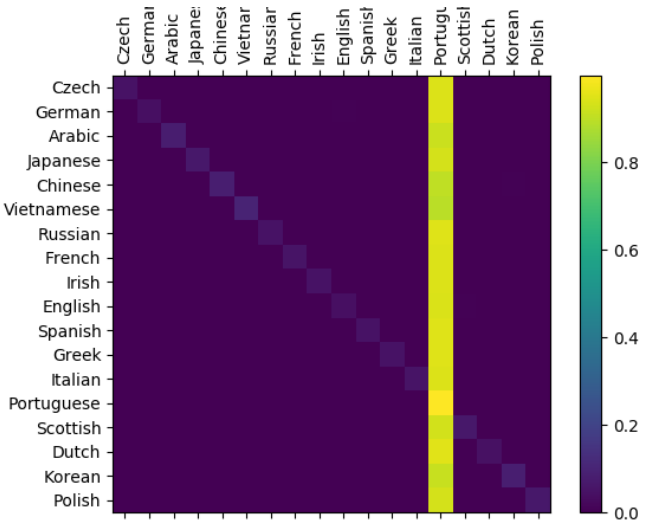

Figure 20: 1-64 Layer (Batch Size 30) Confusion Matrix