HW 1 Report

Kaggle Profile: https://www.kaggle.com/chittaranjan19

Part 1: Numpy

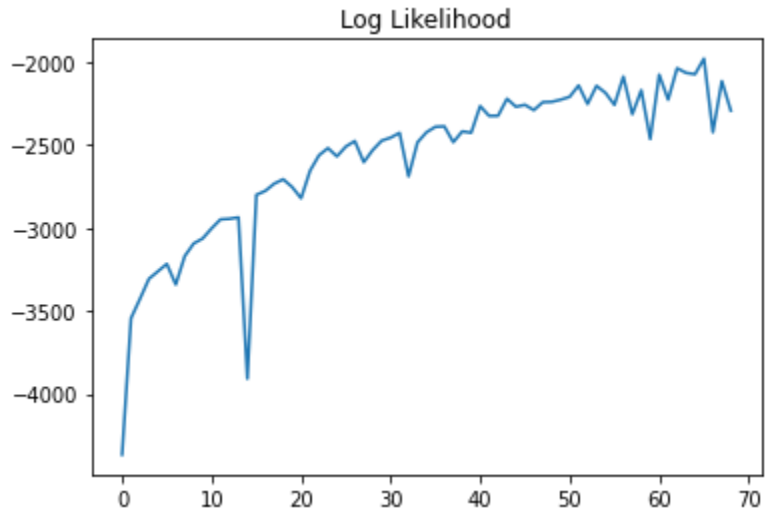


Figure 1: Plot of Log Likelihood for every 50 iterations

Part 2: PyTorch

Experiment 1:

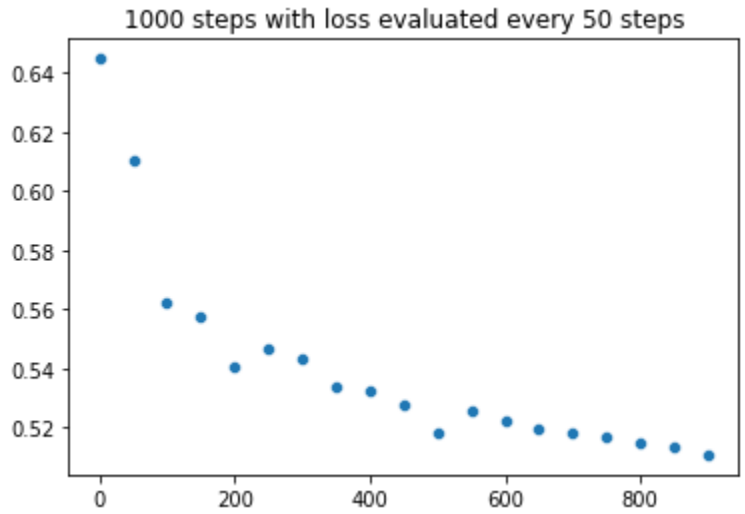
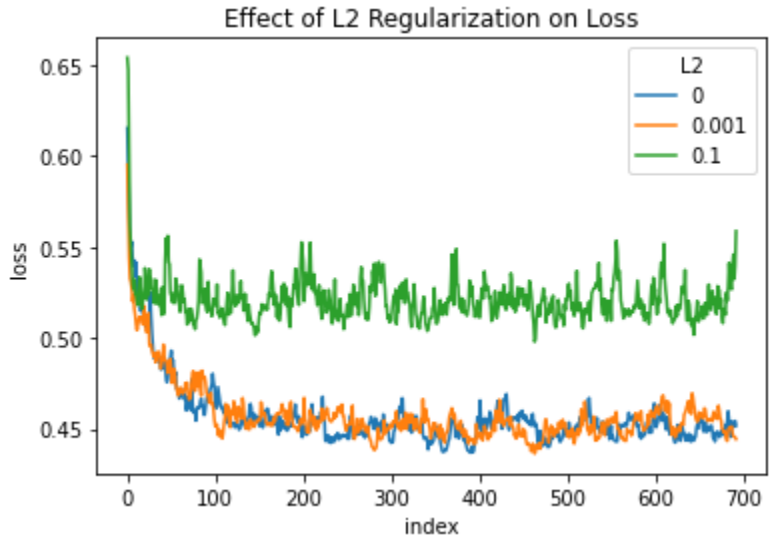


Figure 2: Plot of Binary Cross Entropy Loss for every 50 updates

Experiment 2:



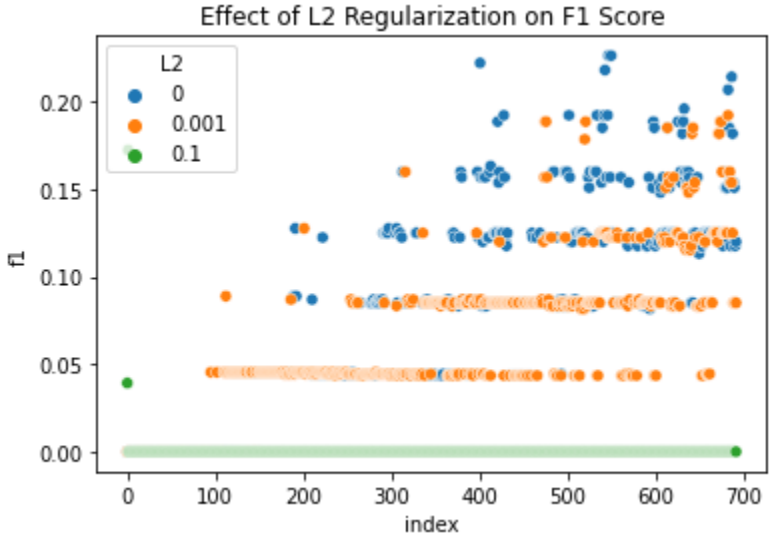
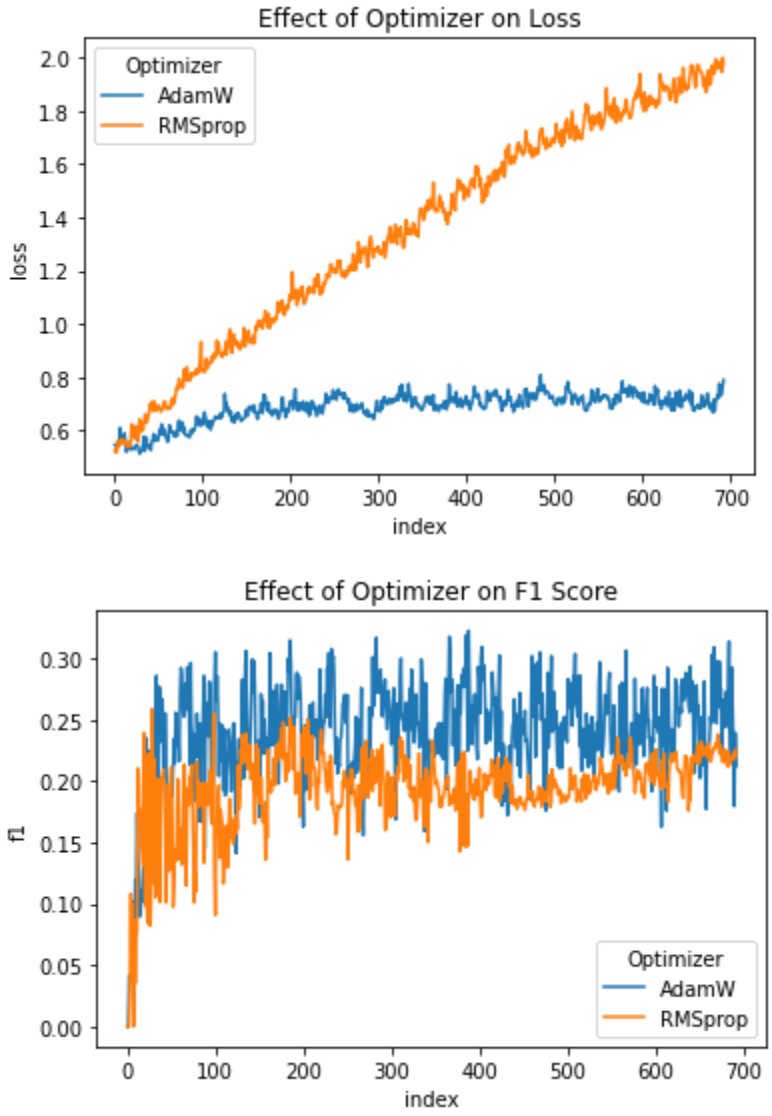


Figure 3: Effect of L2 Regularization on loss and F1 Score

What effect does L2 have on the convergence speed and overall model performance?

From these plots it seems like a higher regularization coefficient leads to higher loss i.e worse model performance. In terms of convergence speed, they seem to be mostly similar but the 0.1 regularization is more “bumpy” with higher peaks and deeper valleys as compared to 0 and 0.001

Experiment 3:

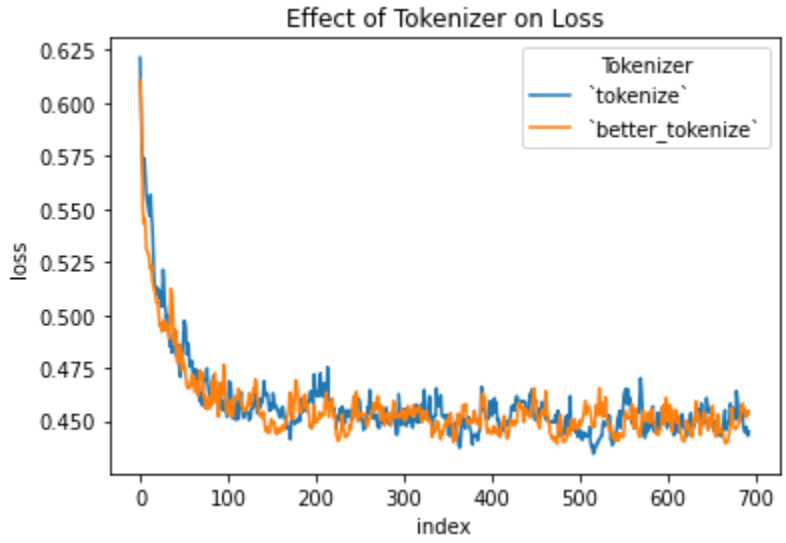


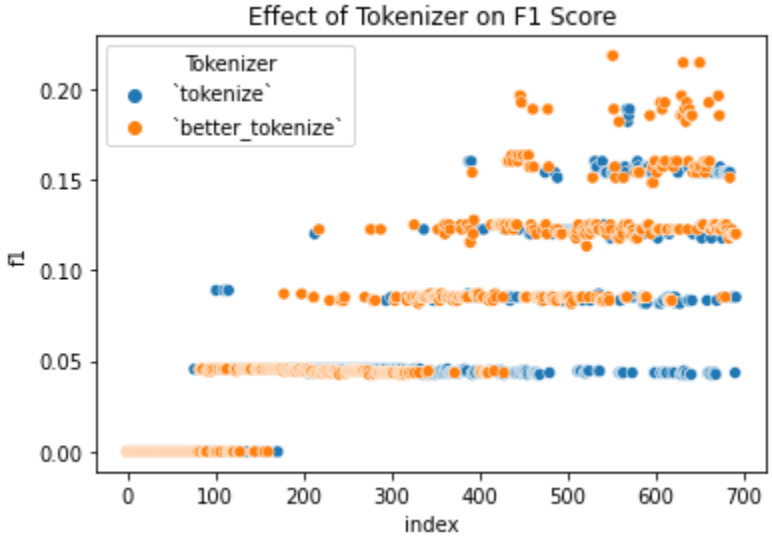
What effect does the choice in optimizer have on the convergence speed and overall model performance?

I'm not sure if this is an error or something specific with the optimizer, but it seems like the loss keep growing with every iteration for the RMSprop optimizer, and is fairly stationary with AdamW. This would suggest that model performance is worsening over time (at least with the RMSprop)

I'm also not sure about the convergence as it seems like neither curves have fully plateaued.

Experiment 4:

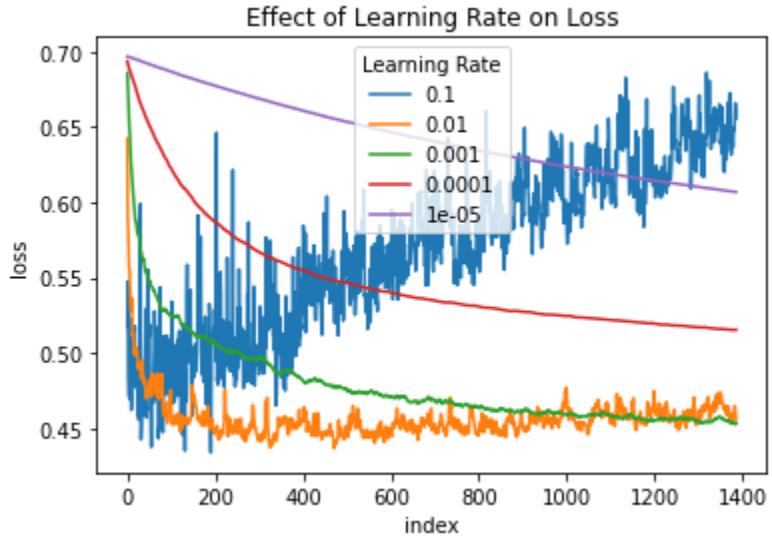


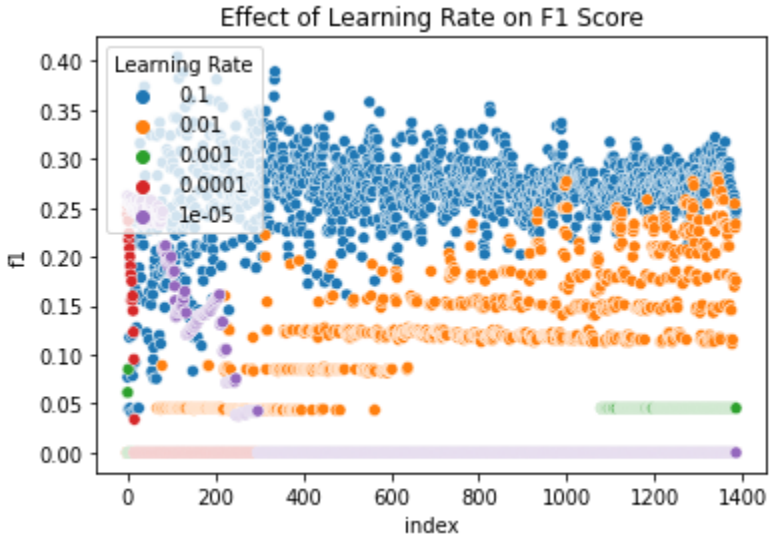


What effect does tokenization have on the overall model performance?

In this case, the two variations of the tokenizer don't seem to have much of a difference in affecting performance of the model. This is potentially due to the fact that my better_tokenize is not very sophisticated. It only removes punctuation and carriage return / new line characters. I would expect a more significant difference if advanced techniques like stopword removal or stemming were used and compared against the trivial tokenizer.

Experiment 5:





What effect does the learning rate have on the model's convergence speed?

A very low learning rate implies that convergence is very slow (purple line) whereas a higher learning rate implies quicker convergence (orange line). However, too high of a learning rate fails to converge as the size of updates are probably too big and the loss does not hit the minima