Jason Yang
Jyang223
ECE498IoT

Lab 2 Report

What I learned:
This served as my first experience with training models and building neural networks. I found it fun to play with using more in depth Tensorflow tools such as Keras. Additionally, I learned about convolution and the different types of layers in a neural net.

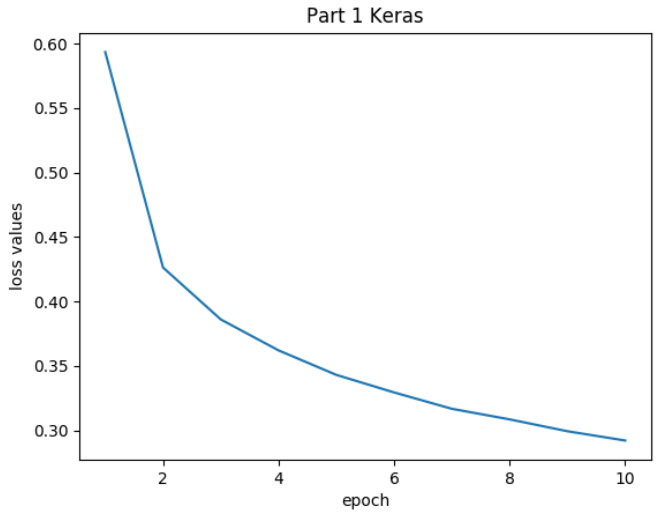
Bugs:
My largest bug was getting Keras to work, as I worked with one of the TAs in office hours for over an hour attempting to fix it. Other than that, I found the Tensorflow part to be interesting with a few bugs but not many.

Part 1: Keras

Test Accuracy and Training Accuracy of Our Part 1 Keras Model:

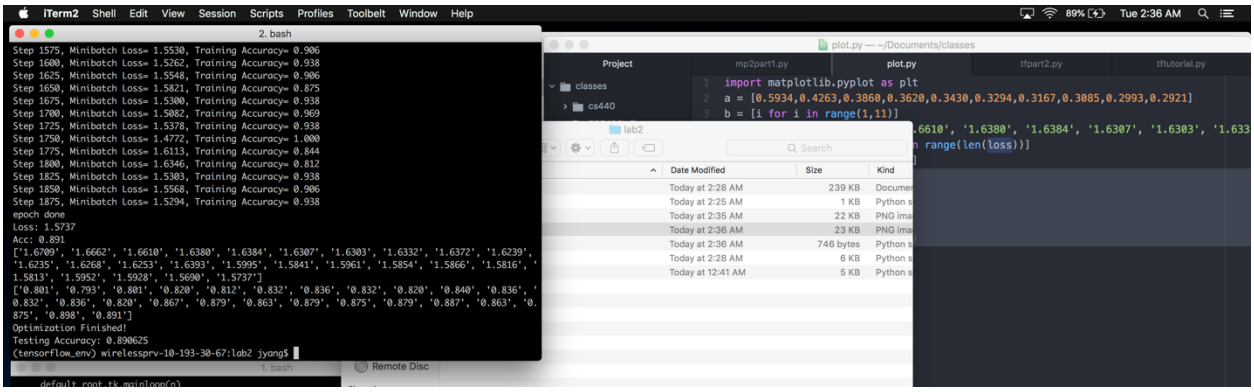
```
Epoch 1/10
2019-02-27 13:35:14.137558: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
2019-02-27 13:35:14.138012: I tensorflow/core/common_runtime/process_util.cc:69] Creating new thread pool with default inter op setting: 8. Tune using inter_op_parallelism_threads for best performance.
60000/60000 [==============================] - 21s 354us/step - loss: 0.5870 - acc: 0.7845
Epoch 2/10
60000/60000 [==============================] - 20s 336us/step - loss: 0.4219 - acc: 0.8462
Epoch 3/10
60000/60000 [==============================] - 19s 314us/step - loss: 0.3791 - acc: 0.8615
Epoch 4/10
60000/60000 [==============================] - 18s 300us/step - loss: 0.3520 - acc: 0.8693
Epoch 5/10
60000/60000 [==============================] - 16s 274us/step - loss: 0.3341 - acc: 0.8752
Epoch 6/10
60000/60000 [==============================] - 16s 273us/step - loss: 0.3178 - acc: 0.8821
Epoch 7/10
60000/60000 [==============================] - 16s 263us/step - loss: 0.3079 - acc: 0.8856
Epoch 8/10
60000/60000 [==============================] - 16s 262us/step - loss: 0.2987 - acc: 0.8888
Epoch 9/10
60000/60000 [==============================] - 16s 267us/step - loss: 0.2883 - acc: 0.8916
Epoch 10/10
60000/60000 [==============================] - 17s 284us/step - loss: 0.2813 - acc: 0.8948
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 24, 24, 3)         78
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 3)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 12, 12, 3)         84
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 6, 6, 3)           0
_________________________________________________________________
flatten (Flatten)            (None, 108)               0
_________________________________________________________________
dense (Dense)                (None, 100)               10900
_________________________________________________________________
dense_1 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_2 (Dense)              (None, 10)                510
=================================================================
Total params: 16,622
Trainable params: 16,622
Non-trainable params: 0
_________________________________________________________________
10000/10000 [==============================] - 1s 138us/step
('Test accuracy:', 0.8828)
(tensorflow_env) wirelessprv-10-193-30-67:lab2 jyang$
```

Below are our plotted loss vs epoch values

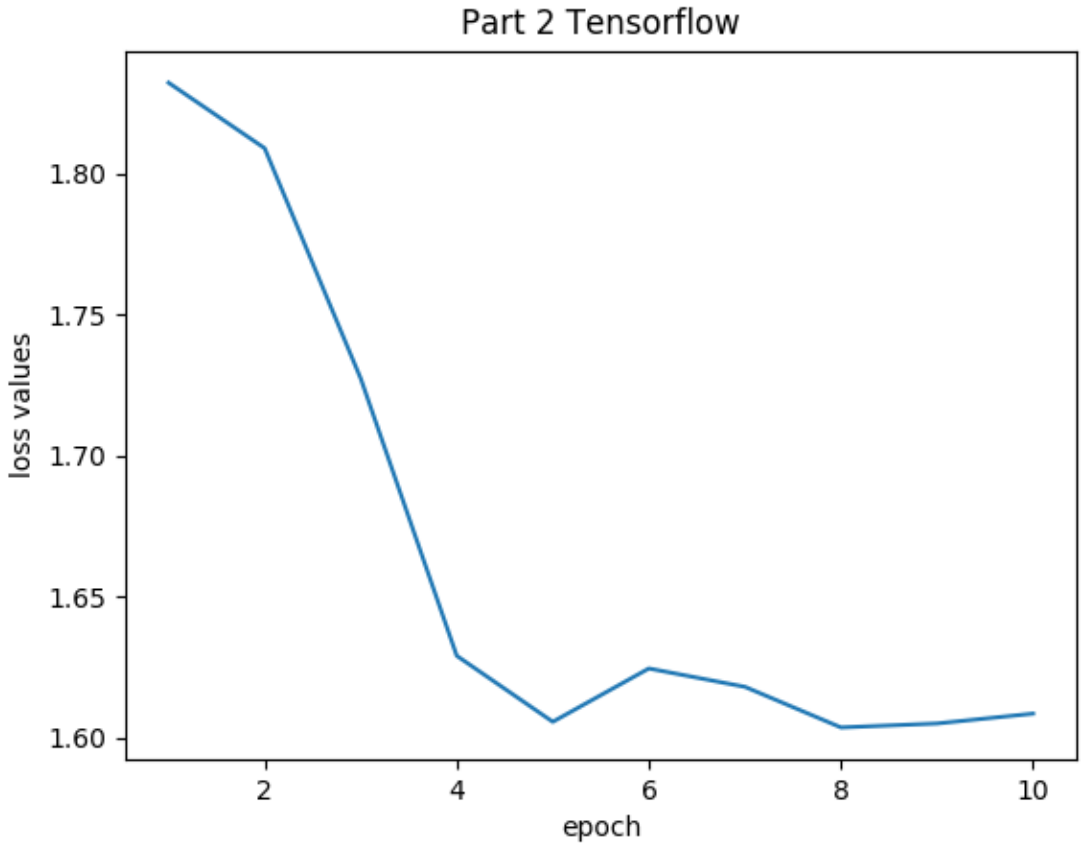


Part 1: Tensorflow

Below is the execution of my code. The final accuracy that can be seen in the terminal is the accuracy when the model runs an evaluate on the TEST dataset.



Below was a sample loss vs epoch curve. To train this one, we ran it for 10 epochs and the result was a bit of overfitting towards the end. However, the grader in the demo said it was also just noise, and I received full credit.



Part 2:

Below are 6 different queries in a row with responses to and from the server. You can see the percentage below for each of them, as two of the six are over 85%. Included also are their testset ID's, as well as the predictions.