Homework 3

James Oswald

Note: To keep the zip of the code small, I deleted the all the compressed .pkl and .gz files from my directory before submitting.

Part 0: Downloading training data

I had to modify MNIST_util.py a bit since it was missing various imports, I resolved this by adding imports for `urllib.request`, `skimage.io`, and `gzip`

Part 1: Training TwoLayerNet

I begin by implementing some code to plot loss vs iterations. Once this is done, I run the two layer network with the command:

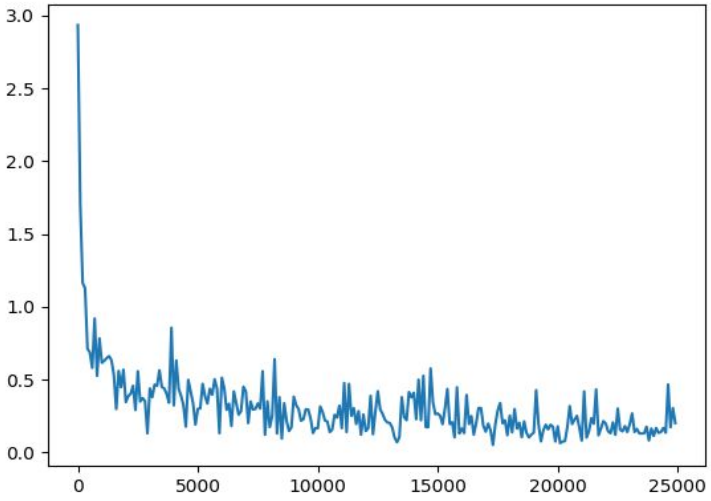
```
python ANN_LeNet_MNIST_demo.py -model TwoLayerNet -iter 25000 -opti SGD
```

This uses 25000 iterations with the SGD Optimizer.

```
Iter 24800 (99.20%), loss = 0.305433
Iter 24900 (99.60%), loss = 0.201452
TRAIN--> Correct: 56897 out of 60000. Acc=0.948283
TEST--> Correct: 9465 out of 10000. Acc=0.946500
```

This produces a training accuracy of ~94.8% with a test accuracy also around ~94.6%
Which is pretty close to the expected accuracies laid out in the code.

I have my plot saved to a file, plotting loss against iterations for 25000 iterations with SGD



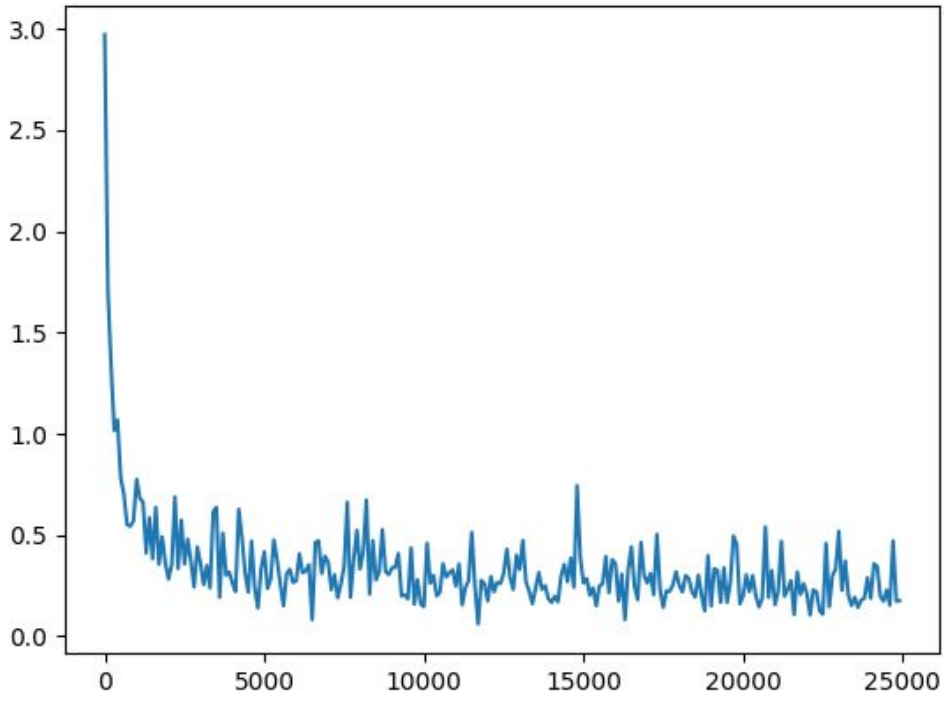
Next I repeat this same process for SGDMomentum with 25000 iterations:

```
python ANN_LeNet_MNIST_demo.py -model TwoLayerNet -iter 25000 -opti SGDMomentum
```

I observe our results using SGDMomentum land us closer to the original expected accuracy laid out in the code comments, around 93.8% for both test and train.

```
Iter 24800 (99.20%), loss = 0.172498
Iter 24900 (99.60%), loss = 0.175326
TRAIN--> Correct: 56283 out of 60000. Acc=0.938050
TEST--> Correct: 9382 out of 10000. Acc=0.938200
```

I plot the loss vs iterations for SGDMomentum as well, getting similar results.



Part 2: Implementing and Training ThreeLayerNet

I begin by implementing code for ThreeLayerNet based on the code from TwoLayerNet in ANN.py. I also edit the H1 and H2 params to be set up as 300 and 100 respectively in ANN_LeNet_MNIST_demo.py

I then run ThreeLayerNet Via

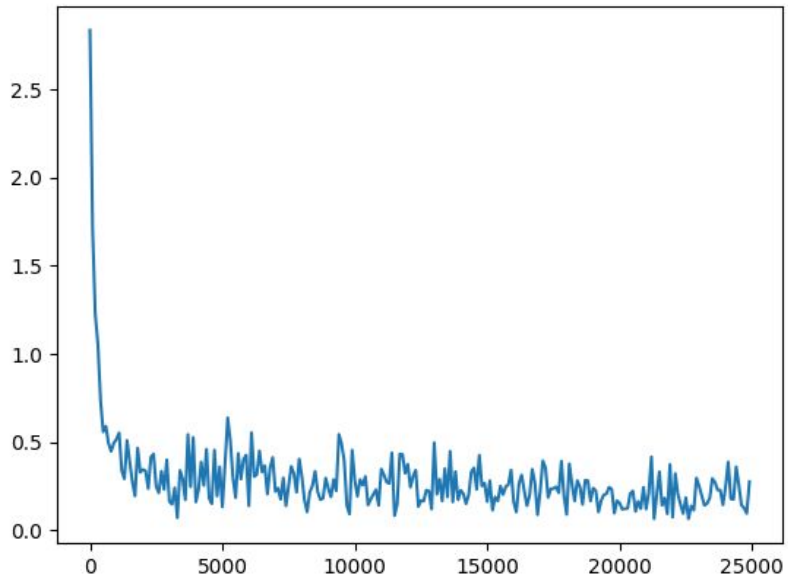
```
python ANN_LeNet_MNIST_demo.py -model ThreeLayerNet -iter 25000 -opti SGDMomentum
```

I wait for a few minutes for it to train and test with ThreeLayerNet and finally obtain the results:

```
Iter 24900 (99.60%), loss = 0.275524
TRAIN--> Correct: 57126 out of 60000. Acc=0.952100
TEST--> Correct: 9509 out of 10000. Acc=0.950900
```

I observe that what I got, ~95% accuracy is very close to the expected values laid out for ThreeLayerNet in the code.

I plot loss vs iterations for ThreeLayerNet and observe much the same thing I observed with TwoLayerNet



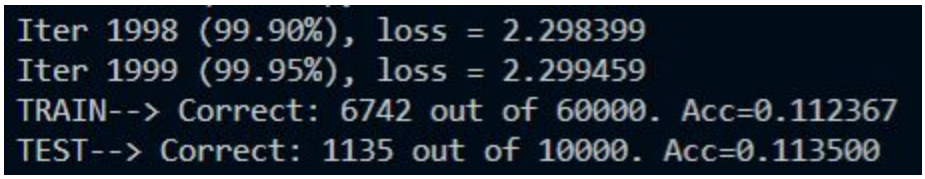
Part 2: Training LeNet5

I then run LeNet5 for 2000 iterations via

```
python ANN_LeNet_MNIST_demo.py -model LeNet5 -iter 2000 -opti SGDMomentum
```

Each LeNet5 iteration takes approximately 1-2 seconds to run, It took 50 minutes to fully train this network. And another 30 minutes to run test results.

LeNet performed rather poorly for over an hour of compute time. However these predictions were well inline with the expected value provided in the comments.



Thus our final accuracy on the testing dataset was around 11.3%