**Part 1: MNIST Logistic Regression – Report**

For this part one of the first things I decided to do was change to optimizer to the “adam” optimizer. There was no particular reasoning for it, I had simply seen other examples of its usage when researching similar problems, and it lead to the biggest increase in accuracy compared to all other optimizer I experimented with.

I changed the number of epochs throughout, however eventually I was satisfied with using 8 epochs with my improved model.

Next I added another network layer to the model. I experimented with the number of neurons and dropout regularization, eventually using 100 neurons with a retain percentage of 10%.

After this I decided to try and add some convolutional layers. I played around with the number of layers I wanted to add and found that adding two layers each with 28 filters and a 3x3 kernel size took me above requirement the accuracy requirements of the assignment. I also used max pooling with a size of 2x2 per convolutional layer to further optimize the model.

To run the code simply upload new notebook in google colab, using the MNIST.ipynb file and run each cell in the notebook.

The results I recorded were:

--Fit model--

Train on 60000 samples

Epoch 1/8

60000/60000 - 38s - loss: 0.1996 - accuracy: 0.9391

Epoch 2/8

60000/60000 - 37s - loss: 0.0629 - accuracy: 0.9807

Epoch 3/8

60000/60000 - 38s - loss: 0.0451 - accuracy: 0.9858

Epoch 4/8

60000/60000 - 37s - loss: 0.0357 - accuracy: 0.9888

Epoch 5/8

60000/60000 - 37s - loss: 0.0270 - accuracy: 0.9915

Epoch 6/8

60000/60000 - 37s - loss: 0.0240 - accuracy: 0.9921

Epoch 7/8

60000/60000 - 37s - loss: 0.0215 - accuracy: 0.9931

Epoch 8/8

60000/60000 - 38s - loss: 0.0184 - accuracy: 0.9940

--Evaluate model--

10000/1 - 3s - loss: 0.0204 - accuracy: 0.9904

Model Loss: 0.04

Model Accuray: 99.0%