**CSC 4760- Homework 7**

**Objectives:**

1. Build a convolution neural network – use it to classify handwritten digits (show your code, training and testing information, testing results.
2. Show the network architecture of your convolutional neural network

**Results:**

The good news is that the assignment already contains instructions on where to find an example for handwriting digit classification. The bad news is that it is fairly uncommented, which makes altering it tricky. The first step is to run the model to make sure it works.

A screenshot of a computer

Description automatically generated

Figure 1 – Default accuracy performance of the model. Can we improve this further?

The CNN was successfully downloaded and ran for pytorch. I also added a lot of comments.

A screenshot of a computer program

Description automatically generated

Figure 2 – The comments that I added to the original code. The hope was that by taking it apart and commenting everything that it would be possible to alter the code. It looks like we simply need to define an additional layer within the initialization function, and then connect it to the forward layer in the forward function. Now a 5x5 feature map seems a bit elaborate to me, for example I know for a fact that we only need 3x3 to create a visually distinct version of every digit (again the pooling and neural networks mean that this likely looks different in reality, but I don’t just want to use the default model – that seems lazy).

A screenshot of a computer

Description automatically generated

Figure 3 – Proof that digits can be reasonably typed using only a 3x3 grid. Now if you squint really closely at the picture of the comments, currently the feature map is 7x7. Applying a 2x2 maxpool should reduce the size of the feature maps to 5x5. I think?

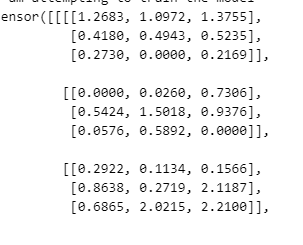


Figure 4 – I should have read my own comments that suggested that I print my output tensor when I had issues with counting. It is clearly a 3x3.

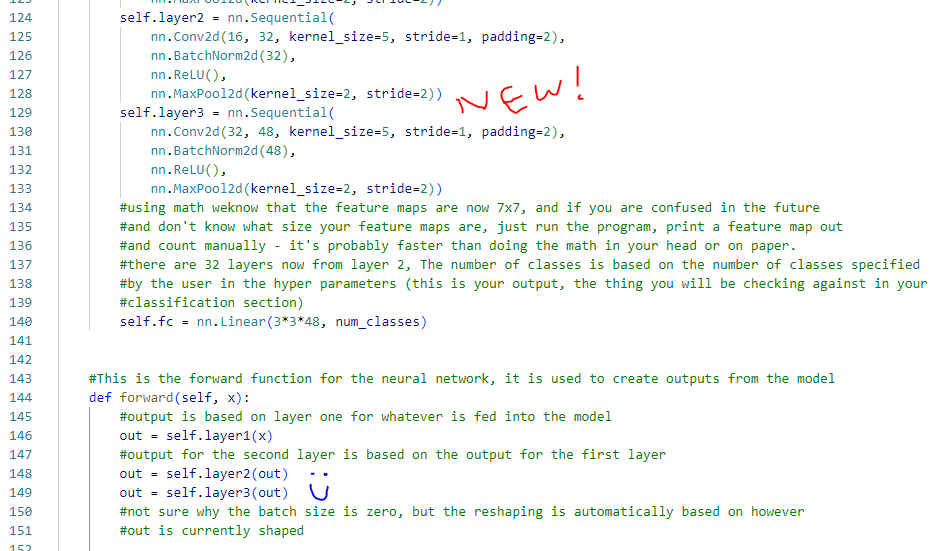


Figure 5 – The key changes to my code that create the additional layer that transforms the 7x7 to a 3x3.

A screenshot of a computer

Description automatically generated

Figure 6 – Look at that! We got 99.11% accuracy vs. 98.34% accuracy! That’s an improvement, my changes worked!

A diagram of a graph

Description automatically generated

Figure 7 – Map depicting the new layer tacked on, along with the size of each layer. All of that for a single classification. Wild. Alright, time to stare at my comments.

A screenshot of a computer

Description automatically generated

Figure 8 – Code lines 1-31 Device Configuration, Library Imports, and Parameters.

A screenshot of a computer program

Description automatically generated

Figure 9 – Code lines 34-77 Dataset download and data loader.

A screenshot of a computer program

Description automatically generated

Figure 10 – Code lines 79-118 Creation of the model class PART 1

A screenshot of a computer code

Description automatically generated

Figure 11- Code lines 117 -140: Creation of the model class PART 2 – Pay close attention to the addition of the new layer and how the fully connected final layer had to be altered to accommodate it.

A screenshot of a computer

Description automatically generated

Figure 12 - Code lines 141-169 Creation of the model class PART 3 – I had teething issues determining the size of the input layers, this was because I cannot do math good about determining how much pooling and convolution would reduce the size of the arrays being considered. This is why the PRINT commands are there, so I can lazily read the size of the tensors.

A screenshot of a computer program

Description automatically generated

Figure 13 – Code lines 173-197 Loss Function and Optimizer Definition. I did not make any changes here because I wanted to compare how the addition of a new layer affected the accuracy of the model while changing nothing else. I am pleased to see that accuracy improved further, I wonder if tampering with the optimizer and loss function could improve it even more.

A screenshot of a computer program

Description automatically generated

Figure 14 – Code lines 200-243. This is the training section It changes the parameters of the model object we create.

A screenshot of a computer program

Description automatically generated

Figure 15 – Code lines 250-278 – This is the testing and saving phase. After I had commented the original code to understand it, I noted that I should go back and add more layers to see what it would do. It made it more accurate! Yay!