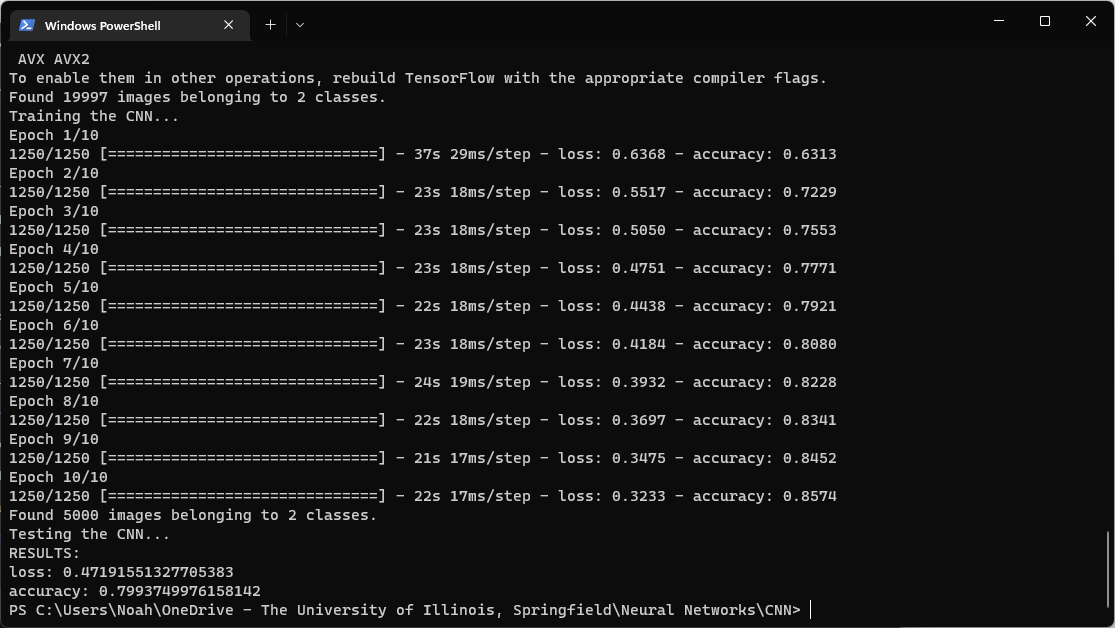
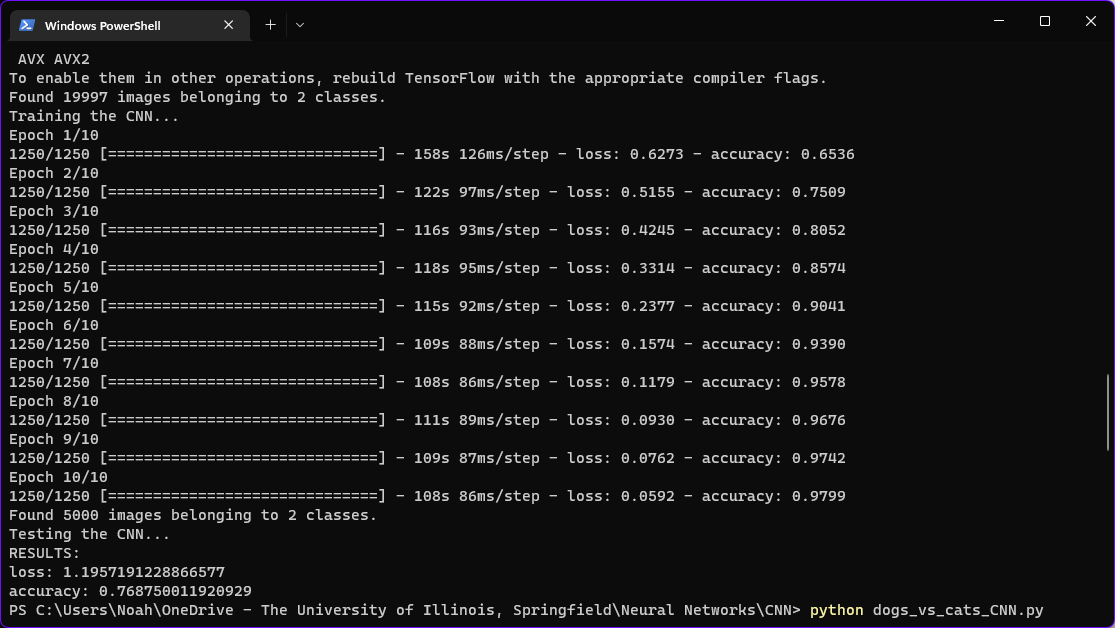
The output I got from running the unaltered script:



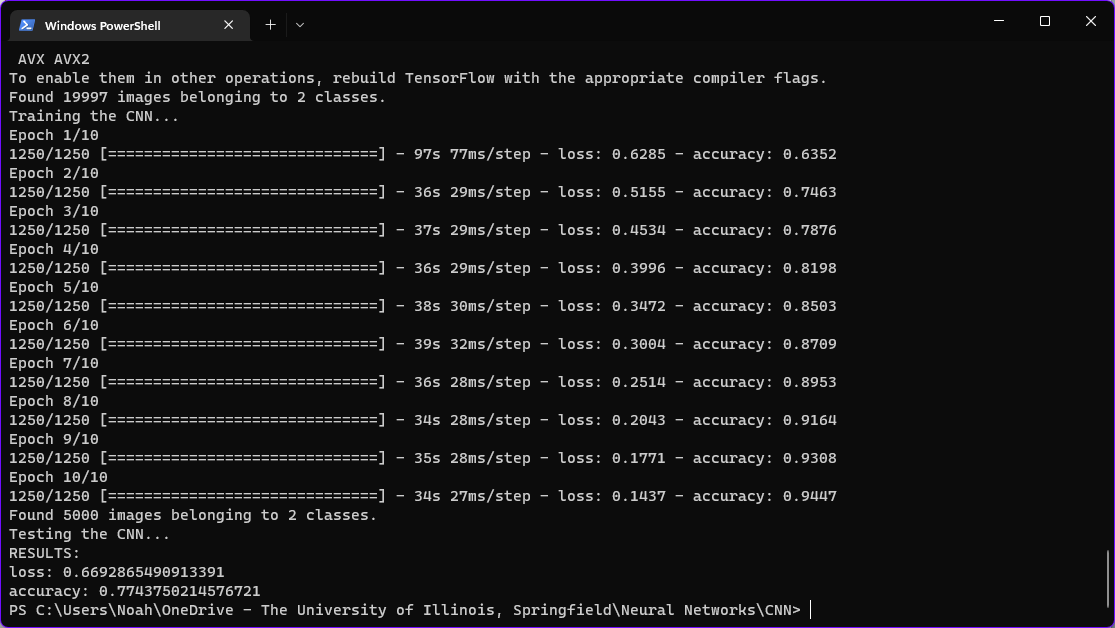
**===========================CHANGING INPUT SIZE===========================**

I changed the input size to 128x128 and got the following results:



The script took much longer to run and actually resulted in lower accuracy, likely due to the inputs being more complex and thus requiring more epochs in order to achieve the same accuracy.

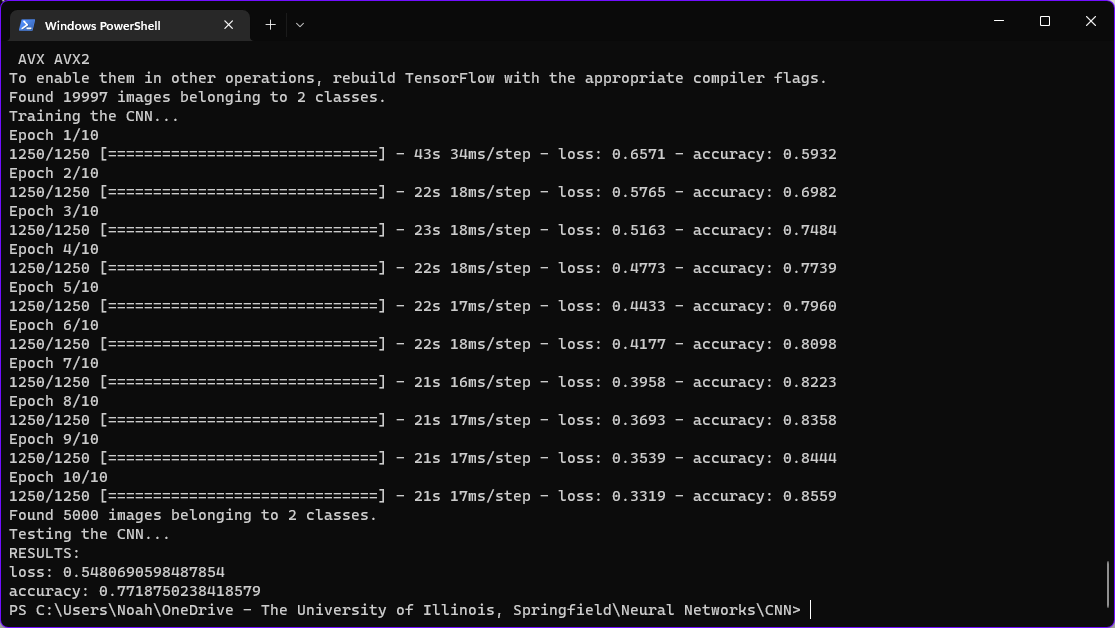
Setting the input size back down to 64x64 produced the following results:



The increased input size still reduces the overall accuracy and increases loss, still likely indicating that the number of epochs need to be increased to compensate for the increased input size.

**==========================ADDING A THIRD LAYER==========================**

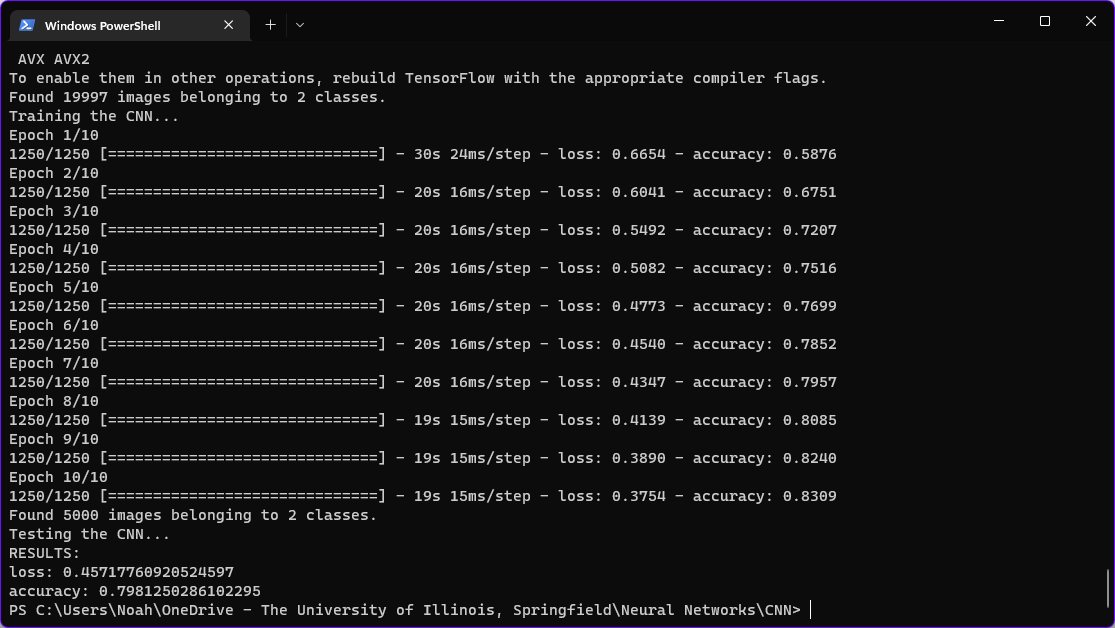
After adding a third convolutional layer (32x32 input size), I got the following output:



The additional layer doesn’t seem to contribute heavily to the accuracy of the network, seeing as the loss increased slightly and the accuracy even decreased slightly, but these can be chocked up to the RNG. Ultimately, though, from this run there does not seem to be a significant impact, positive or negative.

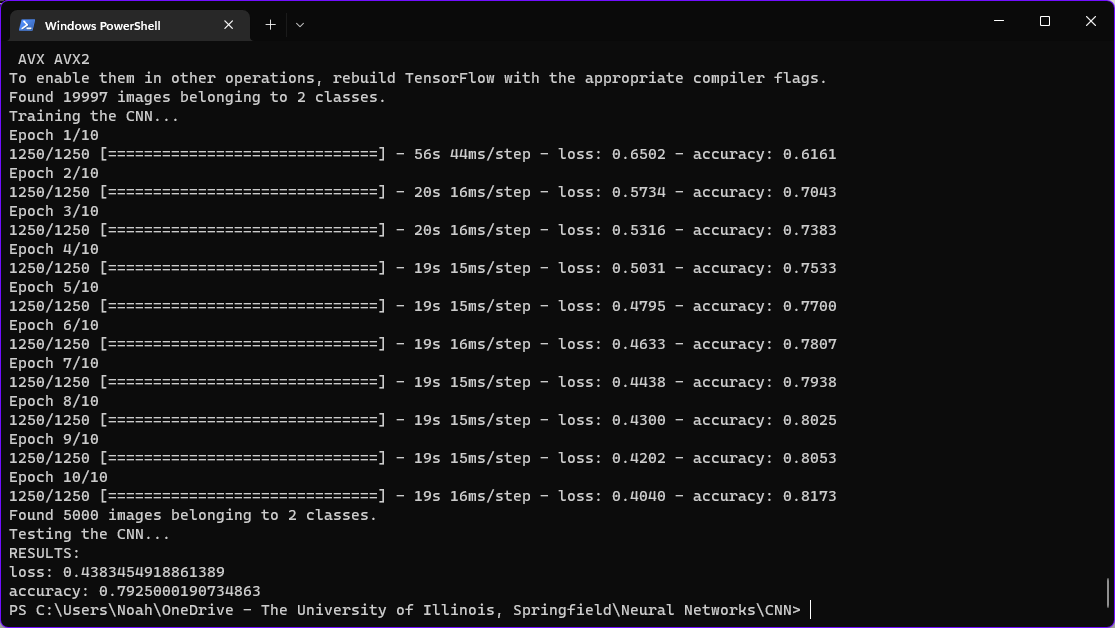
**======================REDUCING NUMBER OF FILTERS======================**

Reducing the number of filters from 32 --> 24 created a CNN that performed on par with that of the original script, meaning the number of filters can be comfortably reduced down to 24 with minimal accuracy loss (at least with the given configuration). Here is the output:



**=================AVERAGE POOLING SUBSAMPLING LAYERS =================**

Below are the results from changing the MaxPooling2D() function call to the AveragePooling2D() function call:



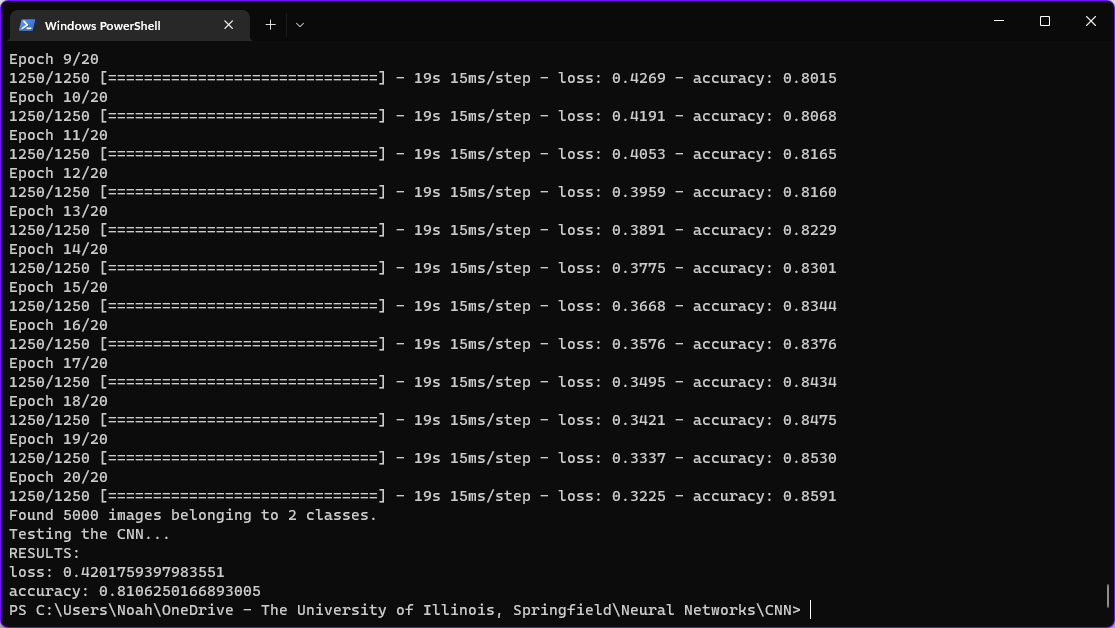
Again, the results seem mostly comparable to the standard script provided in the assignment, albeit with a noticeably lower loss.

I attempted to combine the suggested modifications (including some alterations to other hyperparameters) and was only able to crack the 80% accuracy threshold once, all the while some modifications even doubled the loss at the same ~76-79% accuracy.

The configuration I ended up using was

24 filters  
32x32 input size  
20 epochs  
AveragePooling  
A third 2D convolutional layer and subsampling layer

The resulting output was:



There are definitely other configurations that would result in higher accuracy / lower loss, but training the CNN simply takes too long to justify making minute changes or examining the effects that the RNG has on a given configuration.