The implementation of question 2 was “straight forward” the instruction laid out our goals very clearly however the actual coding involved a great deal of debugging and understanding of how Pytorch works. I’d only ever used Keras up until this point. I have attached another document with the verbose output of all my models. For my loss function I used multi-class cross-entropy as it is considered the default loss function for multiclassification problems making it easy to compare against other results. Overall, the simpler model 1 and model 2 performed well for so small. The presence of overfitting began showing up in the later epochs in model 3 and model 4 as test loss began increasing slightly. Adding regularization with dropout in model 5 did a good job at fixing over fitting but created more unstable test loss between epochs. Overall model 5 achieved the best accuracy of the five models Below I have attached plots of the training loss.

Model #1 - accuracy is 97.82

Shape

Description automatically generated

Very stable decrease in test loss however flattens out fairly high leaving room for more complexity in the model

Model #2 - accuracy is 99.13

Shape

Description automatically generated

Model #2 loss follows a similar curve to model #1 however the two extra convolutional layers allowed the model to better learn the data. There did not appear to be any over fitting in this model meaning more parameters may continue to improve the accuracy.

Model #3 - accuracy is 99.06

A picture containing shape

Description automatically generated

The reduction in learning rate allowed the model to learn the model to fine tune its parameters to the training data which cause test loss to begin increasing in the later epochs. Some of this could also have something to do with changing the activation function to ReLU. This model was a decrease in performance from model #2.

Model #4 - accuracy is 99.13

Chart

Description automatically generated

This model preformed almost the same as model #4, adding the additional dense layer had negligible results with the accuracy improving by only .07.

Model #5 - accuracy is 99.40

A picture containing chart

Description automatically generated

This model achieved the best accuracy by expanding dense layers to 1000 neurons and reducing overfitting by adding regularization (dropout at 50%) to model #4. This model did see test loss increase very slightly in the ending epochs thus 40 epochs would be the max I would train this model, any more was leading to overfitting.