Q2)

### Results

|  |  |  |
| --- | --- | --- |
| **Model** | **Parameters** | **Performance** |
| **Custom PyTorch model 1** | * 4 convolutional layers * 4 max pooling layers * 3 linear layers | [1, 200] loss: 1.891  [2, 200] loss: 0.278  [3, 200] loss: 0.121  [4, 200] loss: 0.012  [5, 200] loss: 0.001  [6, 200] loss: 0.000  [7, 200] loss: 0.000  [8, 200] loss: 0.000  [9, 200] loss: 0.000  [10, 200] loss: 0.000  Finished Training  Accuracy of the network on the test images: 100 % |
| **Custom PyTorch model 2** | * 6 convolutional layers * 6 max pooling layers * 3 linear layers | [1, 200] loss: 4.591  [2, 200] loss: 0.959  [3, 200] loss: 0.413  [4, 200] loss: 0.196  [5, 200] loss: 0.132  [6, 200] loss: 0.012  [7, 200] loss: 0.001  [8, 200] loss: 0.049  [9, 200] loss: 0.005  [10, 200] loss: 0.001  Finished Training  Accuracy of the network on the test images: 100 % |

### Findings

Both models give accuracy of 100% and their runtime is also the same. Only difference is in the value of “loss” in each epochs which indicates the difference in predicted and expected output. In first model, values of loss is less. Even tough we added additional parameters in model 2, it increase in value of loss which means that this model overfitted with training data and failed to generalize when exposed to testing data.