Short Answer Questions

### See if you can improve the MNistResNetwork architecture using more ResNetBlocks. What's the highest accuracy you achieve? What is the architecture (you can paste the output from print(network)).

I attempted to add more ResNetBlocks and run the training in affordable time. Adding 2 more ResNetBlocks achieved accuracy 98.3% after 20 epochs and adding 3 more ResNetBlocks achieved accuracy 98.6% after 20 epochs. The original MNISTResNetwork (See the network architecture shown below) gives an accuracy of 98.8% after 20 epochs which is the best accuracy I could get.

MNISTResNetwork:

(layers): SequentialLayer:

(0): TorchConvLayer: Kernel: (5, 5) In Channels 1 Out Channels 6 Stride 1

(1): MaxPoolLayer: kernel: 2 stride: 2

(2): ReLULayer:

(3): TorchConvLayer: Kernel: (5, 5) In Channels 6 Out Channels 16 Stride 1

(4): ResNetBlock:

(conv\_layers): SequentialLayer:

(0): TorchConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1

(1): ReLULayer:

(2): TorchConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1

(add\_layer): AddLayer:

(relu2): ReLULayer:

(5): ResNetBlock:

(conv\_layers): SequentialLayer:

(0): TorchConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1

(1): ReLULayer:

(2): TorchConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1

(add\_layer): AddLayer:

(relu2): ReLULayer:

(6): MaxPoolLayer: kernel: 2 stride: 2

(7): ReLULayer:

(8): FlattenLayer:

(9): LinearLayer: (784, 120)

(10): ReLULayer:

(11): LinearLayer: (120, 84)

(12): ReLULayer:

(13): LinearLayer: (84, 10)

(loss\_layer): SoftmaxCrossEntropyLossLayer:

### Do you get any improvement using a different non-linearity? Be sure to change it back to ReLU before you turn in your final code.

I didn’t get any evident improvement using a different non-linearity.

### Can you come up with an architecture which gets even higher accuracy? Again, include the output from print(network).

Using the following network design, I got an improved accuracy of 98.9% after 20 epochs. This surprised me since I reduced one ResNetBlock from the original design.

MNISTResNetwork:

(layers): SequentialLayer:

(0): TorchConvLayer: Kernel: (5, 5) In Channels 1 Out Channels 6 Stride 1

(1): MaxPoolLayer: kernel: 2 stride: 2

(2): ReLULayer:

(3): TorchConvLayer: Kernel: (5, 5) In Channels 6 Out Channels 16 Stride 1

(4): ResNetBlock:

(conv\_layers): SequentialLayer:

(0): TorchConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1

(1): ReLULayer:

(2): TorchConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1

(add\_layer): AddLayer:

(relu2): ReLULayer:

(7): MaxPoolLayer: kernel: 2 stride: 2

(8): ReLULayer:

(9): FlattenLayer:

(10): LinearLayer: (784, 120)

(11): ReLULayer:

(12): LinearLayer: (120, 84)

(13): ReLULayer:

(14): LinearLayer: (84, 10)

(loss\_layer): SoftmaxCrossEntropyLossLayer: