1. **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)).

Due to memory limitation on my laptop, I need to double the downsampling for the first maxpool layer. As a benchmark to the new reference ResNetBlock is now at 96% after 7 epochs. So, now I can go deeper with more ResNetBlocks.

The highest accuracy I can achieve with more ResNetBlock is 96% with a test loss of 0.129. I couldn’t improve the network with more ResNetBlocks.

Settings:

Epochs=10

batch\_size = 100

lr = 0.01

The architecture is

*CustomNetwork:*

*(layers): SequentialLayer:*

*(0): ConvLayer: Kernel: (5, 5) In Channels 1 Out Channels 16 Stride 1*

*(1): MaxPoolLayer: kernel: 4 stride: 4*

*(2): ReLULayer:*

*(3): ConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1*

*(4): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(5): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

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

*(7): ConvLayer: Kernel: (1, 1) In Channels 16 Out Channels 16 Stride 1*

*(8): ReLULayer:*

*(9): ConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1*

*(10): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(11): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(12): ReLULayer:*

*(13): FlattenLayer:*

*(14): LinearLayer: (144, 120)*

*(15): ReLULayer:*

*(16): LinearLayer: (120, 84)*

*(17): ReLULayer:*

*(18): LinearLayer: (84, 10)*

*(loss\_layer): SoftmaxCrossEntropyLossLayer:*

1. **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 chose LeakyReLU as the non-linearity using the new reference network. I observed an improvement in the accuracy. Using the same architecture but a different non-linearity and settings as in problem 1, the accuracy that I get is 97.3% with a test loss of 0.083 which is a 1.3% additional accuracy compared to the reference network after 7 epochs.

Settings:

Epochs=10

batch\_size = 100

lr = 0.01

*LeakyReLUNetwork:*

*(layers): SequentialLayer:*

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

*(1): MaxPoolLayer: kernel: 4 stride: 4*

*(2): LeakyReLULayer:*

*(3): ConvLayer: Kernel: (3, 3) In Channels 6 Out Channels 16 Stride 1*

*(4): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): LeakyReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): LeakyReLULayer:*

*(5): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): LeakyReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): LeakyReLULayer:*

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

*(7): LeakyReLULayer:*

*(8): FlattenLayer:*

*(9): LinearLayer: (144, 120)*

*(10): LeakyReLULayer:*

*(11): LinearLayer: (120, 84)*

*(12): LeakyReLULayer:*

*(13): LinearLayer: (84, 10)*

*(loss\_layer): SoftmaxCrossEntropyLossLayer:*

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

The highest accuracy that I get for other architectures is 94%.

Settings:

Epochs=10

batch\_size = 100

lr = 0.01

*ProNetwork:*

*(layers): SequentialLayer:*

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

*(1): MaxPoolLayer: kernel: 4 stride: 4*

*(2): LeakyReLULayer:*

*(3): ConvLayer: Kernel: (3, 3) In Channels 6 Out Channels 16 Stride 1*

*(4): LeakyReLULayer:*

*(5): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(6): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(7): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(8): MaxPoolLayer: kernel: 2 stride: 2*

*(9): LeakyReLULayer:*

*(10): ConvLayer: Kernel: (3, 3) In Channels 16 Out Channels 16 Stride 1*

*(11): LeakyReLULayer:*

*(12): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(13): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(14): ResNetBlock:*

*(conv\_layers): SequentialLayer:*

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

*(1): ReLULayer:*

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

*(add\_layer): AddLayer:*

*(relu2): ReLULayer:*

*(15): LeakyReLULayer:*

*(16): FlattenLayer:*

*(17): LinearLayer: (144, 120)*

*(18): LeakyReLULayer:*

*(19): LinearLayer: (120, 84)*

*(20): LeakyReLULayer:*

*(21): LinearLayer: (84, 42)*

*(22): LeakyReLULayer:*

*(23): LinearLayer: (42, 10)*

*(loss\_layer): SoftmaxCrossEntropyLossLayer:*