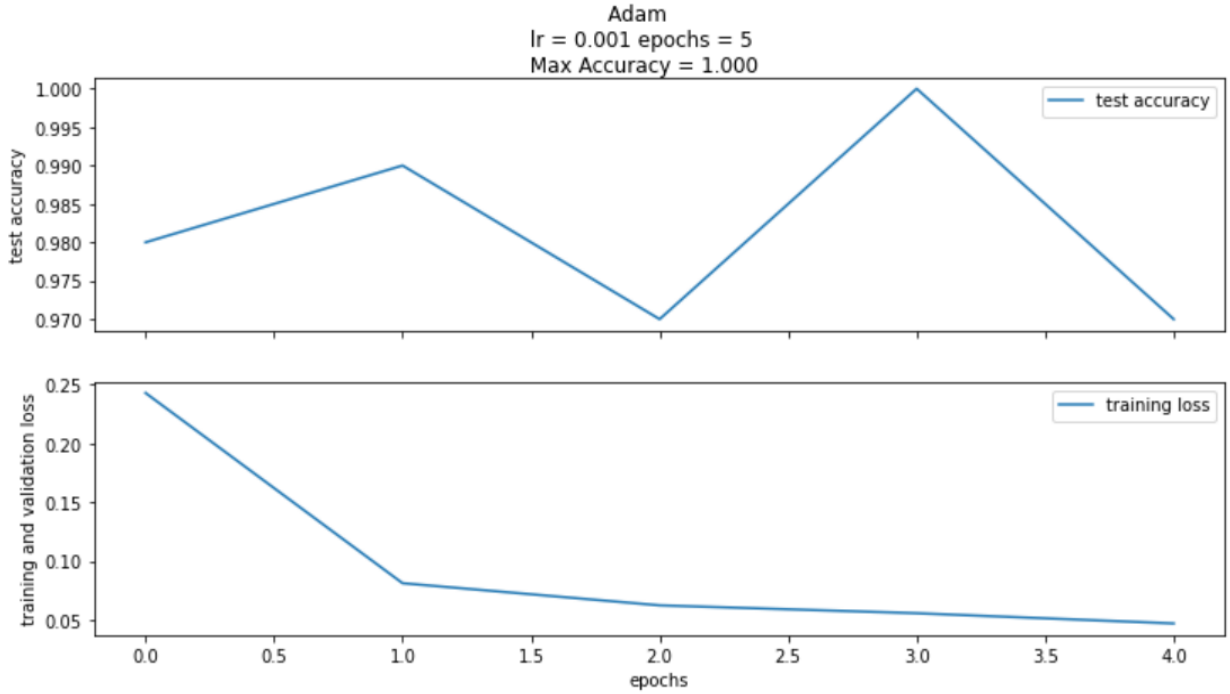
Eric Gagliano (egagli)
Inferring Structure from Complex Systems

Lab 4

Github link: https://github.com/egagli/amath563/blob/main/labs/4/lab4.ipynb

1. Implement LeNet in PyTorch with tanh activation and Adam Optimizer, achieve greater than 95% accuracy on the MNIST classification task.



LeNet is a popular network often used in classifying letters and numbers. It is a 7 layer network with 3 convolutions and 2 average poolings, and 2 fully connected linear layers with a final softmax for classification. We implement this in pytorch using the attached code, and I use tanh as the activation function and Adam as the optimizer. This one isn't too bad to code up as (as networks go) this one is relatively intuitive. It runs pretty fast too, with 5 epochs on collab it took 2.5 mins. One thing to remember is that this is a 32x32 input network and our dataset is not, so in my dataloader statement I added a transform.

2. What values did you use for following hyperparameters when training LeNet that meets the accuracy requirement?

1) Learning Rate

Using Adam with a learning rate of 0.001

2) Batch Size

Training and Test batch sizes of 100

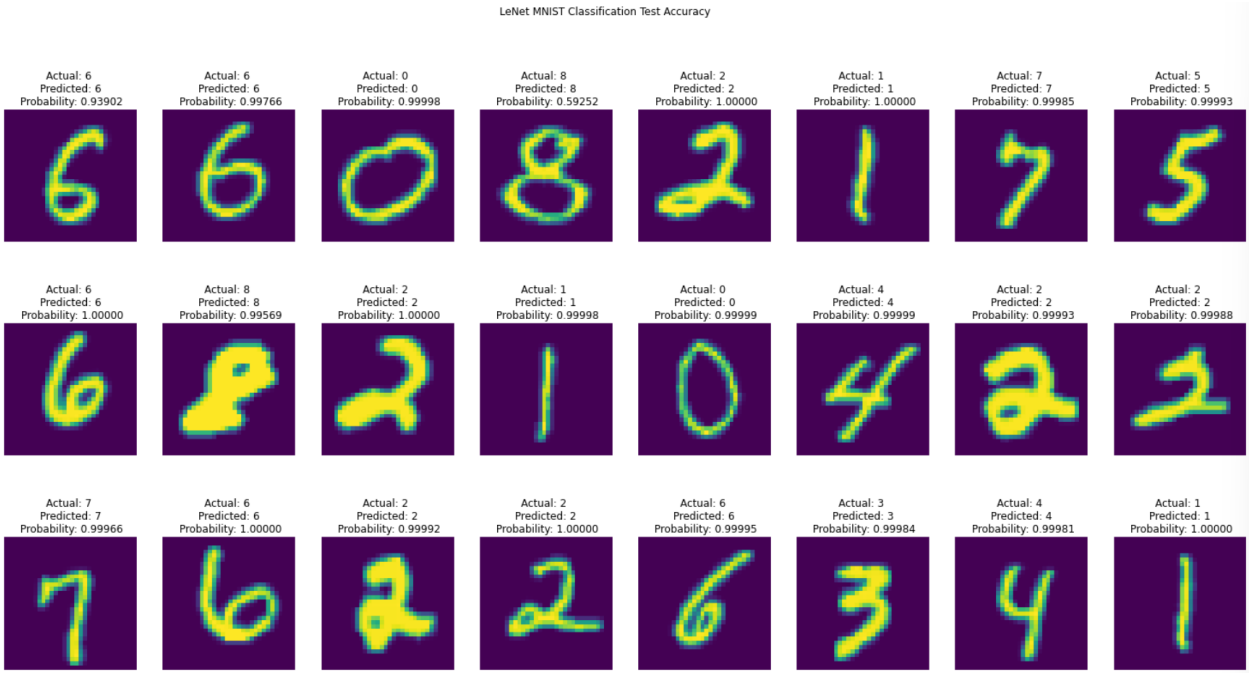
3) Dropout (if any) after your convolutional layers

No Dropout used

4) Any other hyperparameters you set manually (regularization, normalization, etc.)

I use 5 epochs. No other manually set hyperparameters!

3. Please plot the first 10 test images with a title that shows the probability of the predicted class.



Looks like all are classified correctly! I'm surprised at how high these probabilities are, we should all aspire to have the confidence of this classifier. As expected, some of the more visually ambiguous and confusing numbers have lower probabilities. It looks like the 4th number, the 8 has a lower probability. Perhaps it is getting confused with an 8 or 0. This is something we can check by looking at the probabilities. Pretty great performance all around though!