Project 5:Recognition using Deep Networks
Pattern Recognition and Computer Vision (CS-5330)

Team Members:
Poojit Maddineni (NUID: 002856550)
Ruohe Zhou (NUID: 002747606)

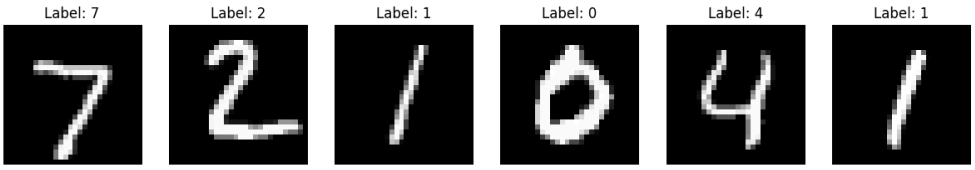
What we have learned from this project:

This project focuses on building, training, and evaluating a deep neural network to recognize digits from the MNIST dataset. We analyzed the 10 filters in the first layer of the network, visualized the diagram of the network, and tested the effects of changing the 3 dimensions – number of convolution layers, size of convolution filters, and dropout rates – on the accuracy of the model. We also implemented a transfer learning on Greek letters and adjusted the shape and intensities of the hand-written images for the trained model to recognize. For the extension, we evaluated the first convolution layer of ResNet18, tried replacing the first layer of the MNIST network with a Gabor filter observed the effect, and also built a live digit recognition application using the trained network.
This project helped us learn and understand how the model behaves and how the model accuracy is affected by changing the hyperparameters. Also, the transfer example for the 3rd task helped us learn how to use an existing model for other datasets. Overall it was a wonderful learning experience!

Task 1: Build and train a network to recognize digits.

A. Get the MNIST digit data set

In this task, we had to download the MNIST digit dataset which consists of 60K 28x28 labeled digits for training and 10K 28x28 labeled digits for testing. We are plotting the first 6 example digits in the test dataset to visualize the digits.



The first 6 example digits of the Test Dataset

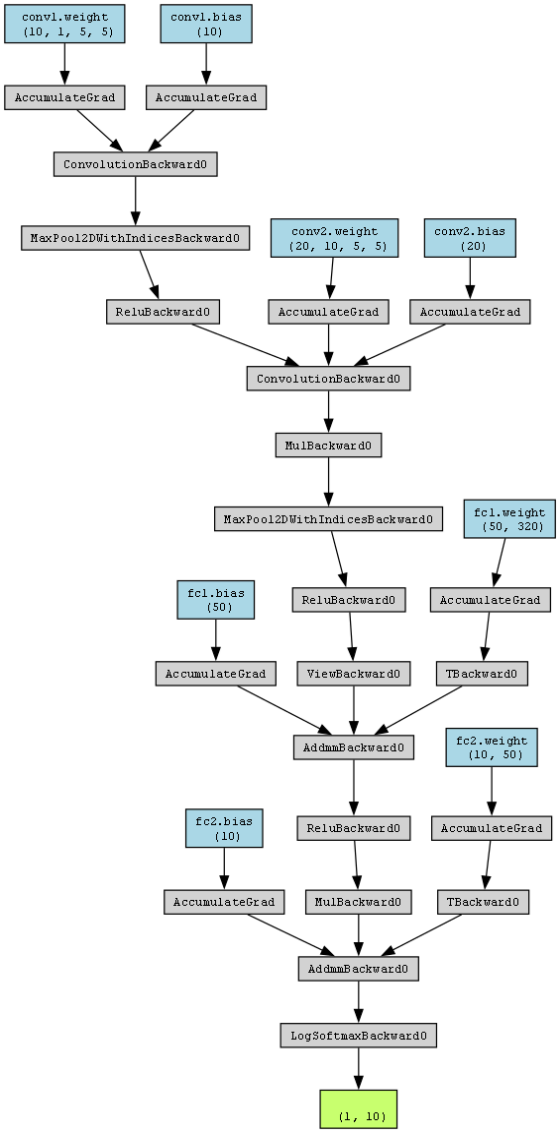
B. Build a network model

Here we built a network model following the prompt given.

A convolution layer with 10 5x5 filters

- A max pooling layer with a 2x2 window and a ReLU function applied.
- A convolution layer with 20 5x5 filters
- A dropout layer with a 0.5 dropout rate (50%)
- A max pooling layer with a 2x2 window and a ReLU function applied
- A flattening operation followed by a fully connected Linear layer with 50 nodes and a ReLU function on the output
- A final fully connected Linear layer with 10 nodes and the log_softmax function applied to the output.

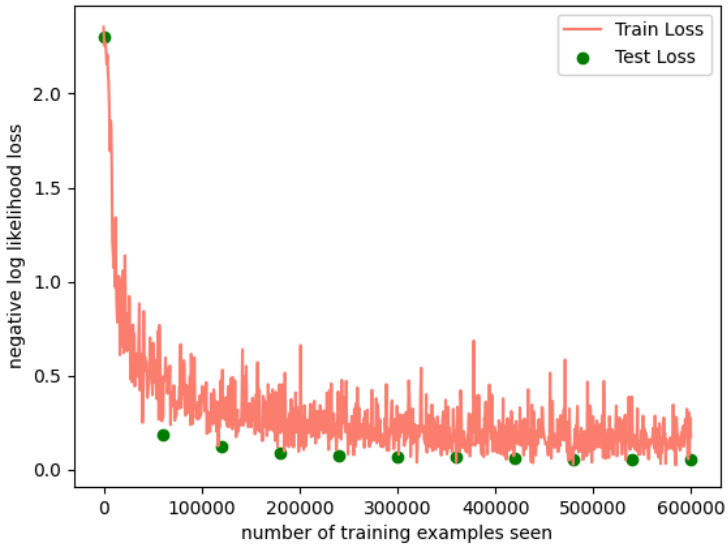
The diagram of the network is as shown:



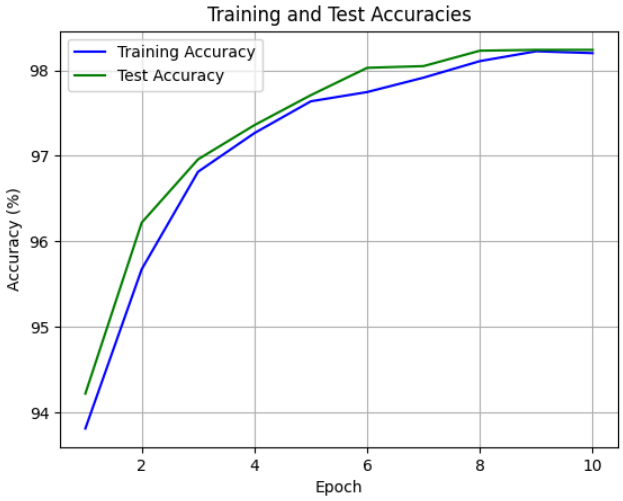
C. Train the model

These are the hyperparameters we used to train the model:

- n_epochs = 10
- batch_size_train = 128
- batch_size_test = 1000
- learning_rate = 0.1
- momentum = 0.5
- log_interval = 10
- random_seed = 1



Plot of Training and Test Error



Plot Of Train And Test Accuracies
We got an accuracy of 98.5%

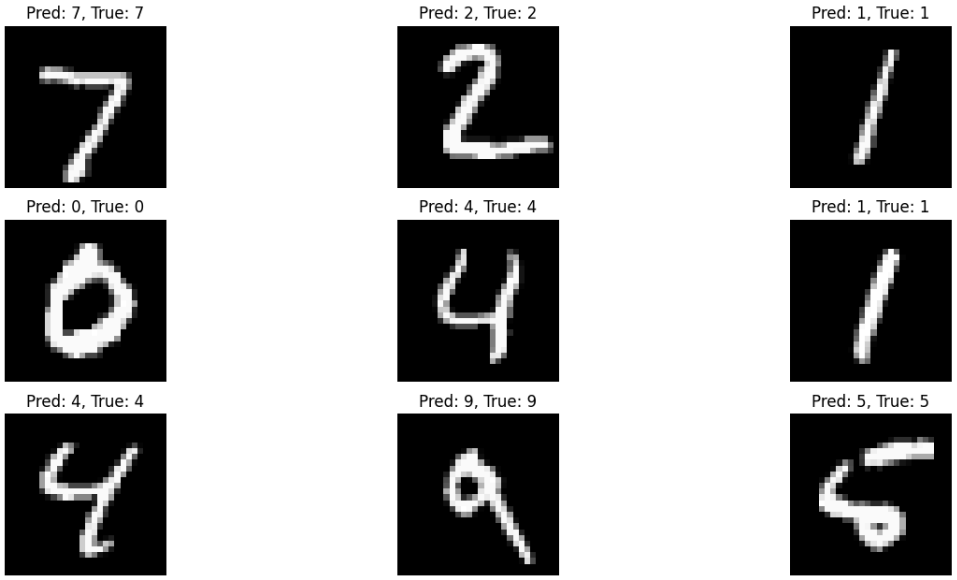
D. Save the network to a file:

We saved the network to the directory and named it model.

E. Read the network and run it on a test set:

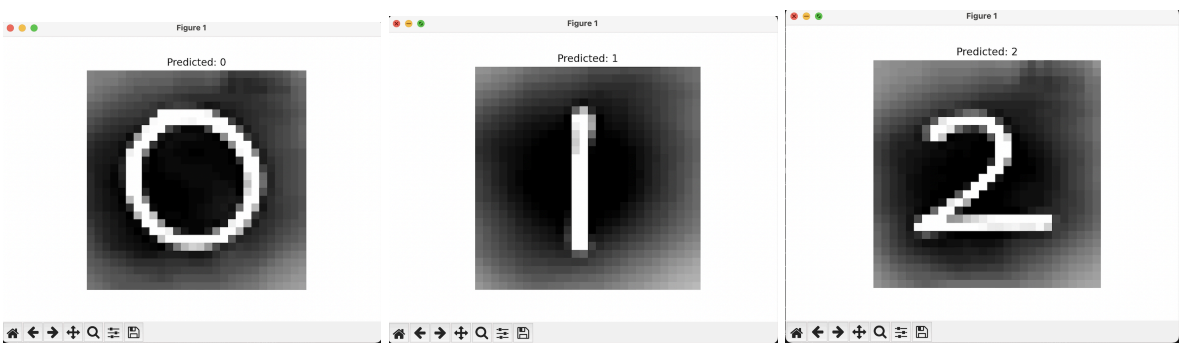
```
(.conda) (base) shuuwakawa@shuuwakawas-MacBook-Pro CS5330_Project5 % python task1_part2.py
Output Values: [-28.28 -25.03 -17.52 -19.95 -28.32 -32.6  -42.8   -0.   -22.96 -17.03] Max Index: 7 Correct Label: 7
Output Values: [-12.21 -13.13  -0.   -15.89 -23.14 -24.8  -17.48 -18.01 -13.17 -25.24] Max Index: 2 Correct Label: 2
Output Values: [-19.94  -0.   -13.27 -17.91 -12.37 -16.37 -15.74 -13.28 -12.81 -17.18] Max Index: 1 Correct Label: 1
Output Values: [ -0.   -21.71 -12.97 -19.32 -19.63 -16.85 -12.4  -16.82 -15.95 -15.51] Max Index: 0 Correct Label: 0
Output Values: [-26.3  -20.88 -19.82 -27.89  -0.   -27.61 -18.83 -20.85 -21.06  -9.35] Max Index: 4 Correct Label: 4
Output Values: [-23.88  -0.   -15.96 -21.24 -14.35 -19.95 -20.15 -14.74 -15.06 -19.8 ] Max Index: 1 Correct Label: 1
Output Values: [-26.37 -13.27 -18.98 -23.79  -0.   -17.34 -17.52 -14.27 -10.15 -10.08] Max Index: 4 Correct Label: 4
Output Values: [-15.55 -14.23 -12.24  -6.05  -4.2   -8.01 -14.57  -9.25  -6.37  -0.02] Max Index: 9 Correct Label: 9
```

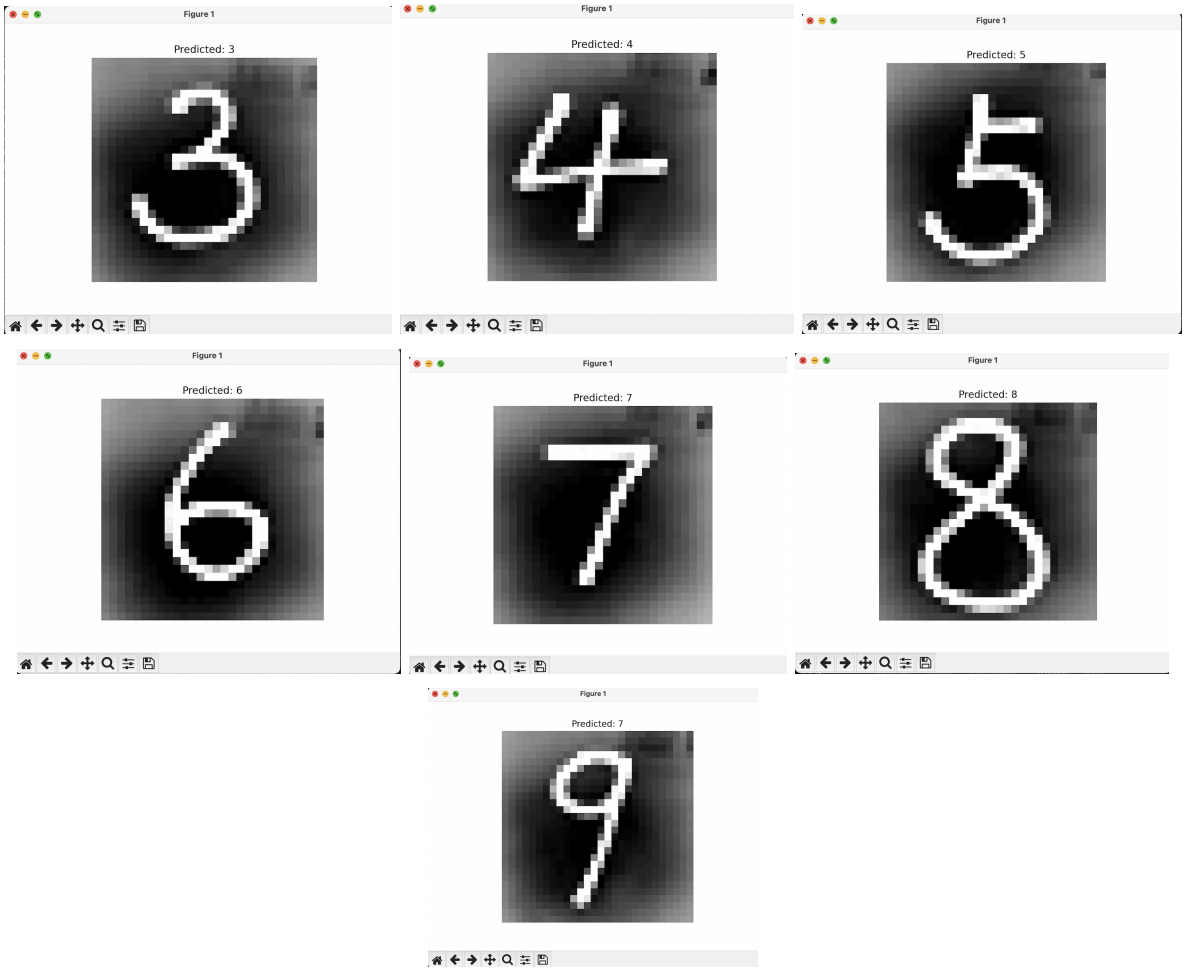
10 network output values, max index value, and correct labels



F. Test the network on new inputs:

The first 9 digits of the test set and the predicted values of the trained model





Results of testing the model on new inputs

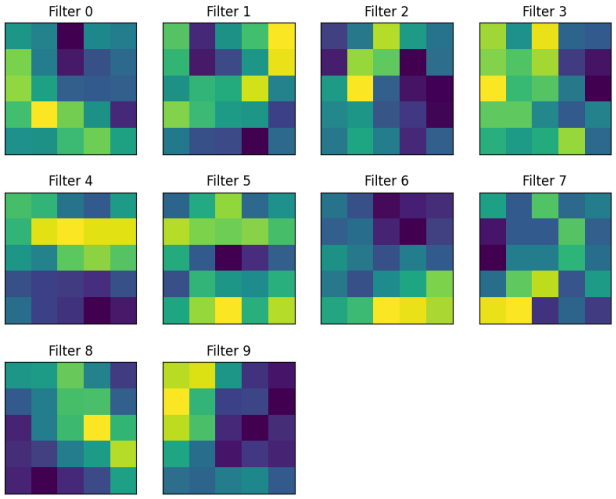
The model predicted 8 out of 9 digits correctly, it did not predict the right label for digit 9. We think since the mode is not 100% accurate in prediction, it was not able to detect.

Task 2: Examine Network

A. Analyze the first layer

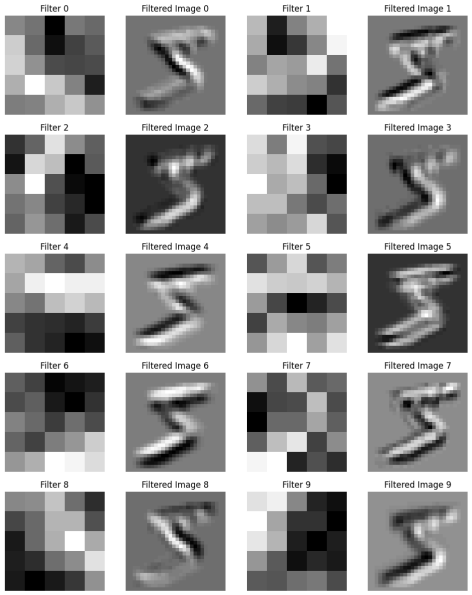
Weights of the first layer

```
weights' shape: torch.Size([10, 1, 5, 5])
filter weights of 1st layer: tensor([[ 0.0451,  0.0506, -0.3274, -0.0068, -0.1007],
        [ 0.2326,  0.0258, -0.2025, -0.0986, -0.0699],
        [ 0.1942,  0.0097, -0.1090, -0.1072, -0.0666],
        [ 0.0886,  0.3322,  0.1881,  0.0065, -0.1984],
        [ 0.0107,  0.0432,  0.1225,  0.1778,  0.1420]])
```



Filters visualization

B. Show the effect of the filters on the first image

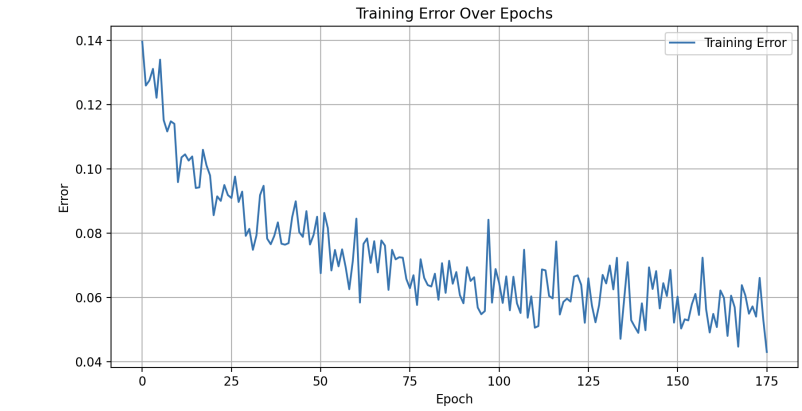


This result makes sense because it matches with the filter. After all, each filter highlights a part of visual features(edges, corners) in the input images, and the resulting images after applying these filters match the highlighted parts.

Part 3: Transfer Learning on Greek Letters

```
Net(
  (conv1): Conv2d(1, 10, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(10, 20, kernel_size=(5, 5), stride=(1, 1))
  (conv2_drop): Dropout2d(p=0.5, inplace=False)
  (fc1): Linear(in_features=320, out_features=50, bias=True)
  (fc2): Linear(in_features=50, out_features=3, bias=True)
)
Epoch 1  Avg Loss: 0.1395  Accuracy: 0.3333
```

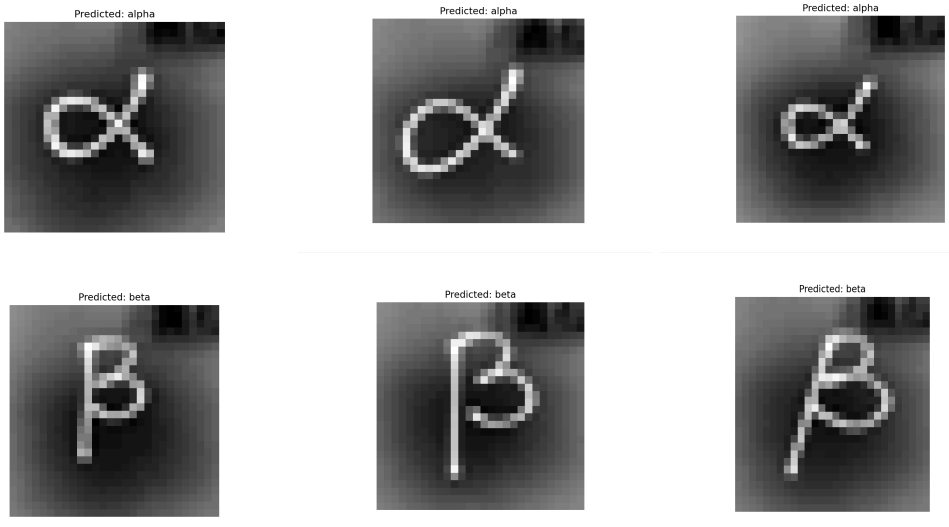
Printout of Modified Network

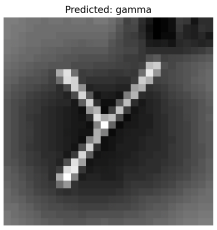


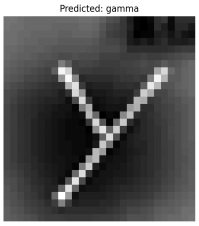
Plot of Training Errors

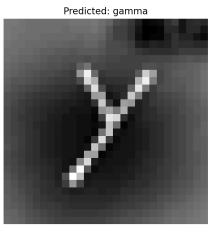
It took around 176 epochs to almost perfectly identify the 27 examples.

Results of additional data:









4. Design your Own Experiment

Develop a plan:

We want to explore how the accuracy of the model on the testing set of MNIST Fashion data will change as the value of the following dimensions change:

Number of convolutionary options: [1, 2]
Size of convolutionary filters: [1, 2, 3, 4, 5]
Dropout rates: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6]

We plan to evaluate 2 * 5 * 5 = 50 Combinations.

Predict the results:

We predicted that the accuracy of the model will increase as the number of convolutional layers increases because it allows the model to learn more complex features. And as dropout rates increase, the accuracy will decrease because the size of the data decreases. We also predicted that the accuracy of the model will increase as the size of the convolutional filters increase because larger filters can capture more global features.

Execute the plan:

Our result matches with our prediction. The accuracy of the model is higher when the number of convolution filters, the number of convolution layers are higher, and the dropout rate is lower.

num_conv_layers=2, num_conv_filters=2, dropout_rate=0.2
Accuracy: 75.1%, Avg loss: 0.713737

num_conv_layers=2, num_conv_filters=1, dropout_rate=0.2
Accuracy: 66.0%, Avg loss: 0.929067

num_conv_layers=1, num_conv_filters=3, dropout_rate=0.2

Accuracy: 75.1%, Avg loss: 0.713737

num_conv_layers=2, num_conv_filters=3, dropout_rate=0.2
Accuracy: 66.0%, Avg loss: 0.929067

num_conv_layers=2, num_conv_filters=3, dropout_rate=0.1
Accuracy: 69.5%, Avg loss: 0.836934

num_conv_layers=2, num_conv_filters=3, dropout_rate=0.3
Accuracy: 66.2%, Avg loss: 0.865088

num_conv_layers=2, num_conv_filters=3, dropout_rate=0.4
Accuracy: 60.8%, Avg loss: 1.070646

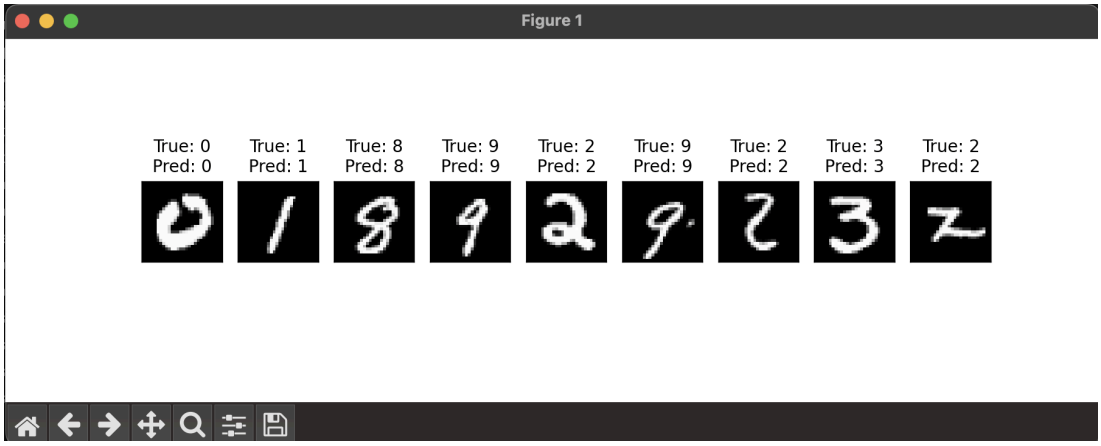
Optimizing: num_conv_layers=2, num_conv_filters=3, dropout_rate=0.5
Accuracy: 56.8%, Avg loss: 1.282435

num_conv_layers=1, num_conv_filters=3, dropout_rate=0.2
Accuracy: 66.9%, Avg loss: 0.860744

num_conv_layers=2, num_conv_filters=3, dropout_rate=0.2
Accuracy: 74.6%, Avg loss: 0.722189

5. Extension -1

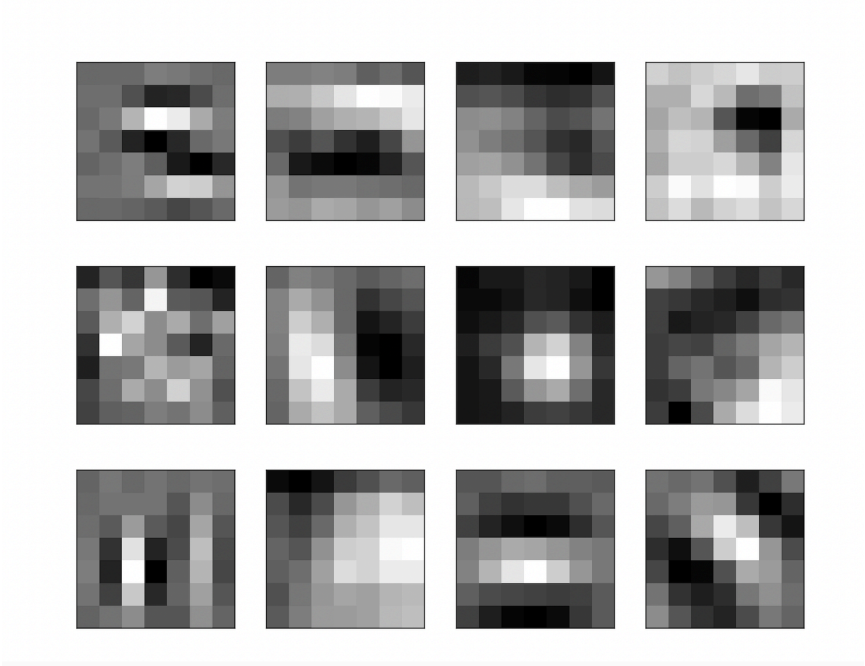
- Replace the first layer of the MNIST network with Gabor filter and retrain the rest of the network:



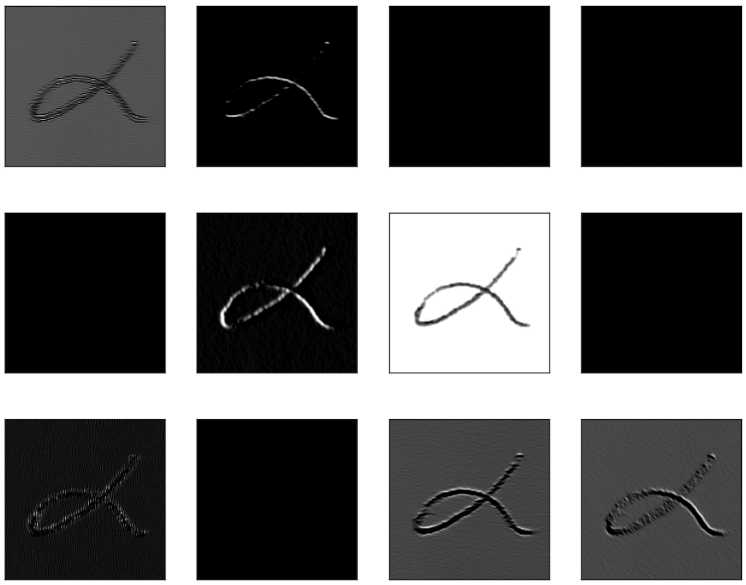
Gabor filter is sensitive to edge detection and texture classification, can help capture critical features from the first layer and improve the accuracy of the model.

Extension - 2

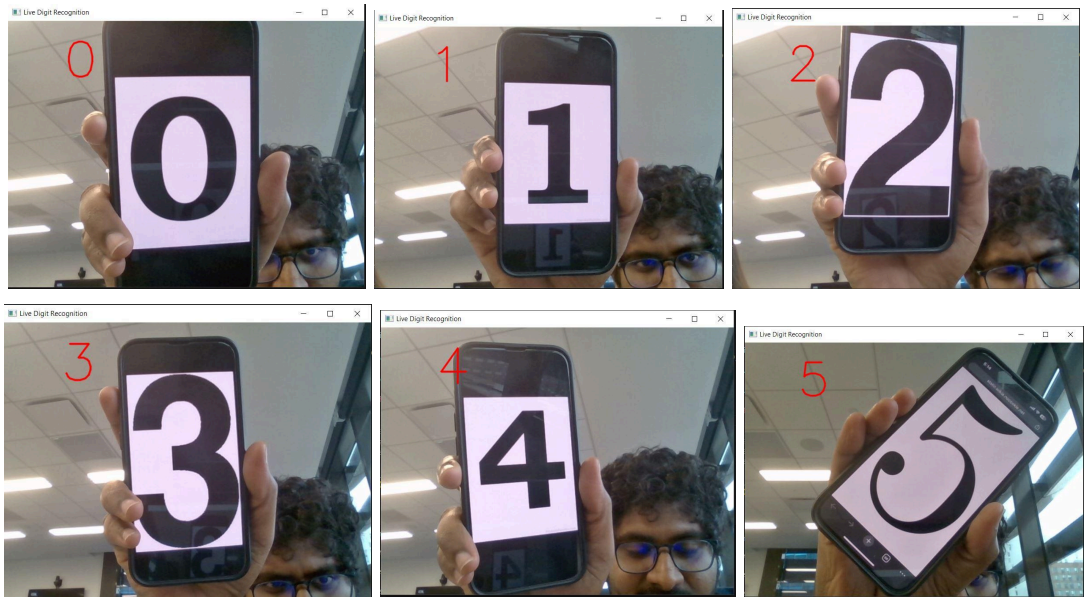
- Evaluate the 1st convolutionaL layer in ‘ResNet18’

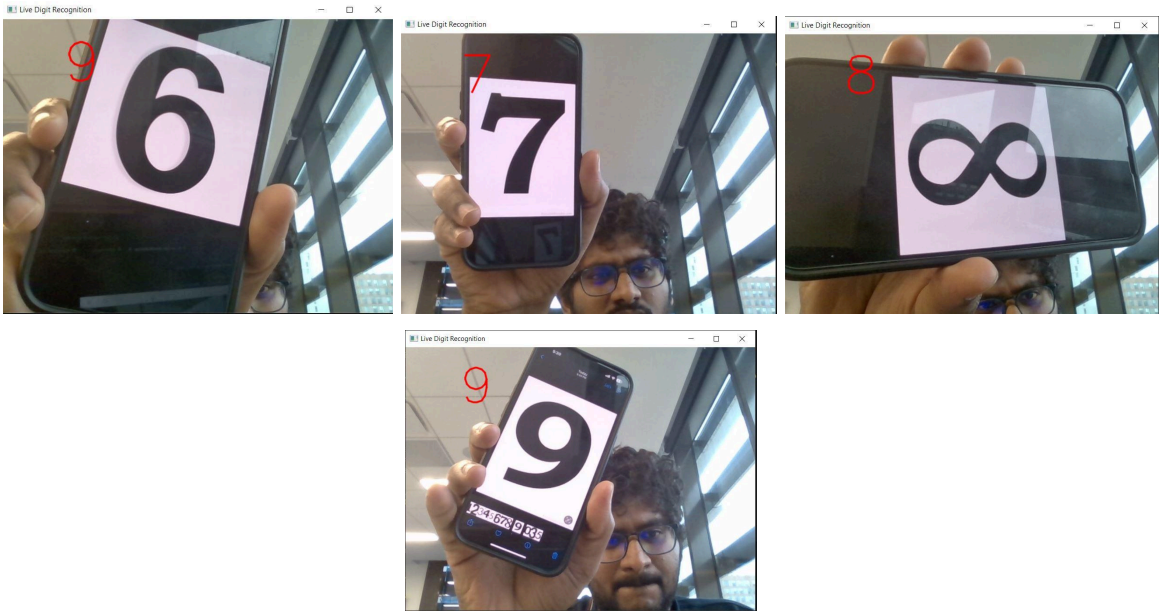


- The effect of the first layer of ‘ResNet18’ on an alpha letter:



Extension - 3 Live digit detection





The model was able to do a live classification of digits but since the model was not completely accurate, it did some false classification too. It can be seen in the case of digit 6 where it was classifying it as 9. This made sense because 6 and 9 are pretty similar.

Reflection:

In this project, we get familiar with the structure of a deep neural network and the process of analyzing the layers and dimensions. We also learned about how to adjust the dimensions and intensities of hand-written images to match with the trained dataset and the effects of network dimensions on the accuracy of the model. In addition, we explored other pre-trained networks such as ResNet18, and tried building a live digit recognition application using the trained network.

Acknowledgement:

We would like to express our gratitude to Professor Bruce Maxwell for his lectures. His teaching was instrumental in helping us think constructively about our solution. We also extend our thanks to Stack Overflow and to all the developers and contributors on the platform. The wealth of resources available there was invaluable during our debugging process. We appreciate the assistance of OpenAI's ChatGPT, which was useful for conducting deeper research on the general topic. Additionally, we recognize the contributions of the numerous technical writers on Medium who have shared their expertise in computer image processing through their writings.

References:

- https://docs.opencv.org/4.x/d4/d94/tutorial_camera_calibration.html
- https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga1019495a2c8d1743ed5cc23fa0daff8c
- Youtube-https://www.youtube.com/watch?v=E5kHUs4npX4
- YouTube - https://www.youtube.com/watch?v=bs81DNsMrnM
- YouTube - https://www.youtube.com/watch?v=pDImLazOPrQ
- https://code.visualstudio.com/docs/cpp/cpp-debug
- https://www.reddit.com/r/computervision/
- https://en.wikipedia.org/wiki/Harris_corner_detector
- Google Scholar- http://www.cs.yorku.ca/~kosta/CompVis_Notes/harris_detector.pdf
- https://www.youtube.com/watch?v=26nV4oDLiqc
- https://www.youtube.com/watch?v=H5qbRTikxI4
- https://www.researchgate.net/figure/Some-images-in-MNIST-dataset-The-whole-dataset-contains-70-000_fig3_315886065
- https://www.youtube.com/watch?v=3xPT2Pk0Jds