CS5330 Project 5: Recognition using Deep Networks

A short description of the overall project:

In this project a model was built and trained to recognise digits using the MNIST dataset measuring all the accuracies and losses in various levels. Later using transfer learning greek letters alpha, beta and gamma were also recognised.In the ending we tried to tweak some parameters and observe the corresponding accuracies.

Task-1:

(A)Include a plot of the first six example digits

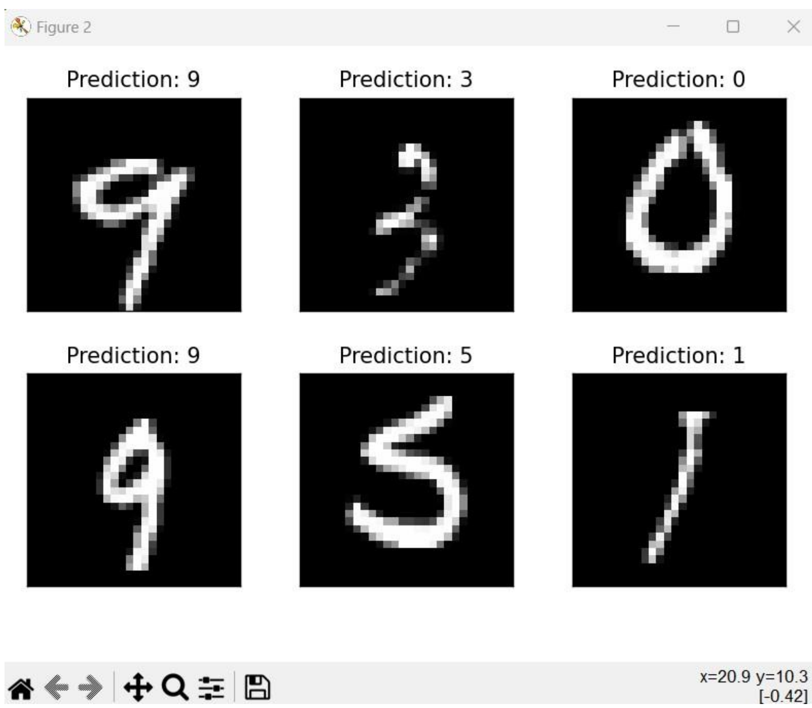


Fig-1 plot of first 6 example digits

(C) Put a diagram of your network

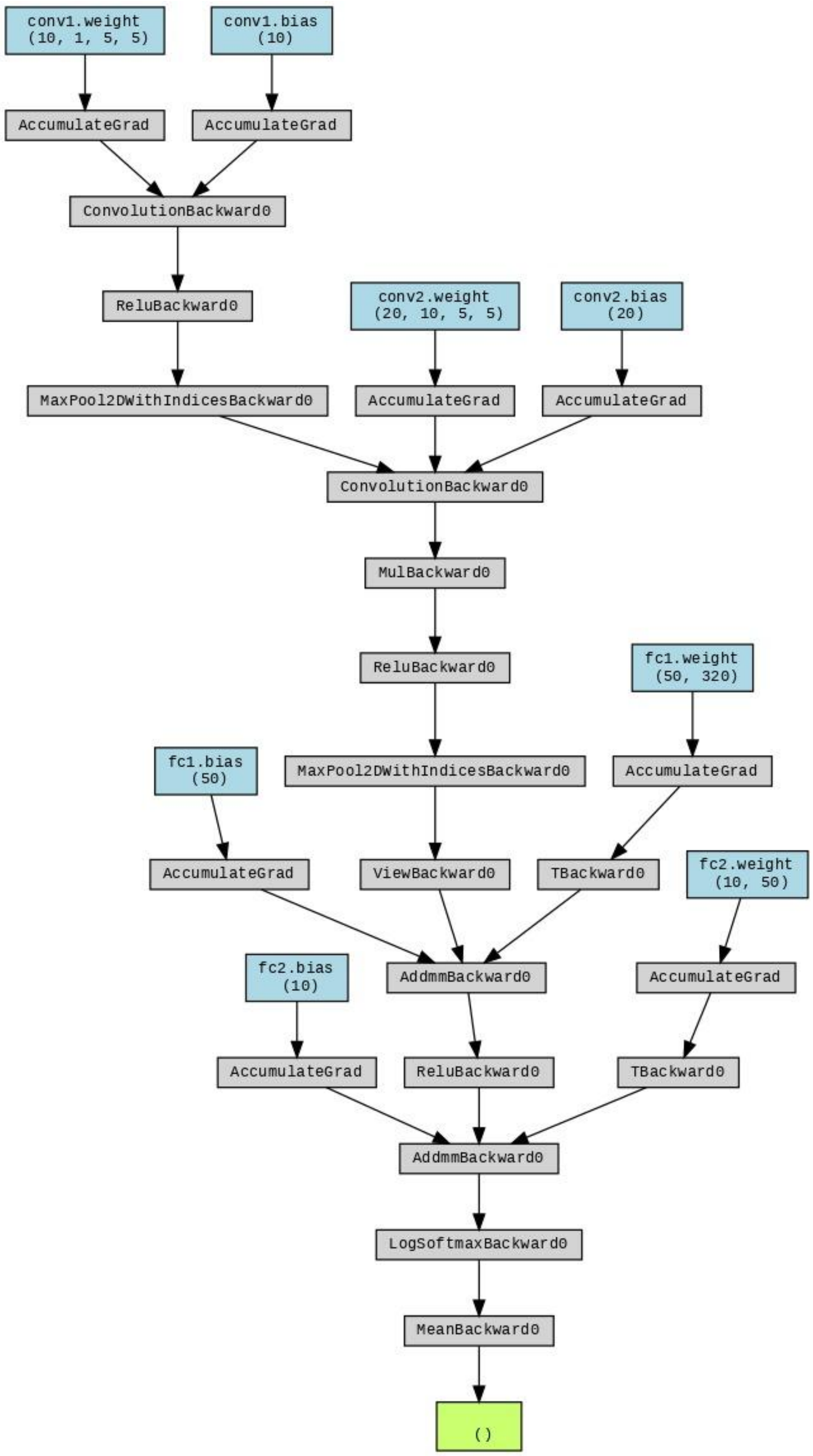


Fig-2 Diagram of the Network

Summary:

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1           [-1, 10, 24, 24]             260
              ReLU-2           [-1, 10, 24, 24]               0
         MaxPool2d-3           [-1, 10, 12, 12]               0
            Conv2d-4             [-1, 20, 8, 8]           5,020
         Dropout2d-5             [-1, 20, 8, 8]               0
              ReLU-6             [-1, 20, 8, 8]               0
         MaxPool2d-7             [-1, 20, 4, 4]               0
            Linear-8                   [-1, 50]          16,050
              ReLU-9                   [-1, 50]               0
           Linear-10                   [-1, 10]             510
================================================================
Total params: 21,840
Trainable params: 21,840
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.13
Params size (MB): 0.08
Estimated Total Size (MB): 0.22
----------------------------------------------------------------
```

(D) Collect the accuracy scores and plot the training and testing accuracy in a graph. Include this plot in your report.

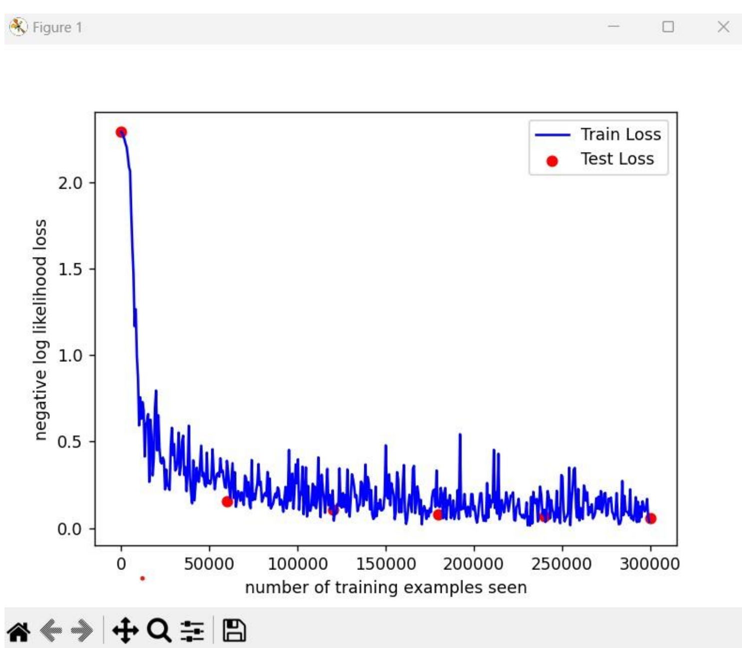


Fig-4 Training and testing accuracy plots

(F) Include a table (or screen shot) of your printed values and the plot of the first 9 digits in your report.

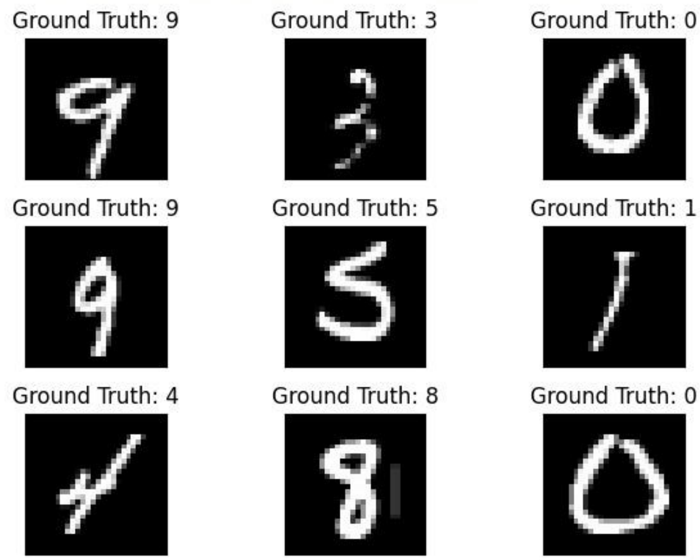


Fig-5 plot of first 9 digits

(G) Display how well the network performed on this new input in your report

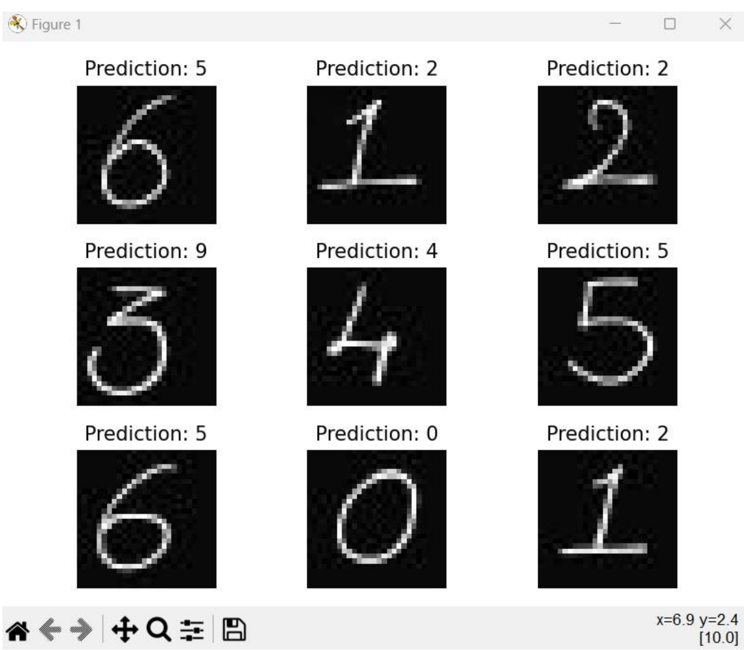


Fig-6 plot of first 9 digits with the new input

Task-2

(A)

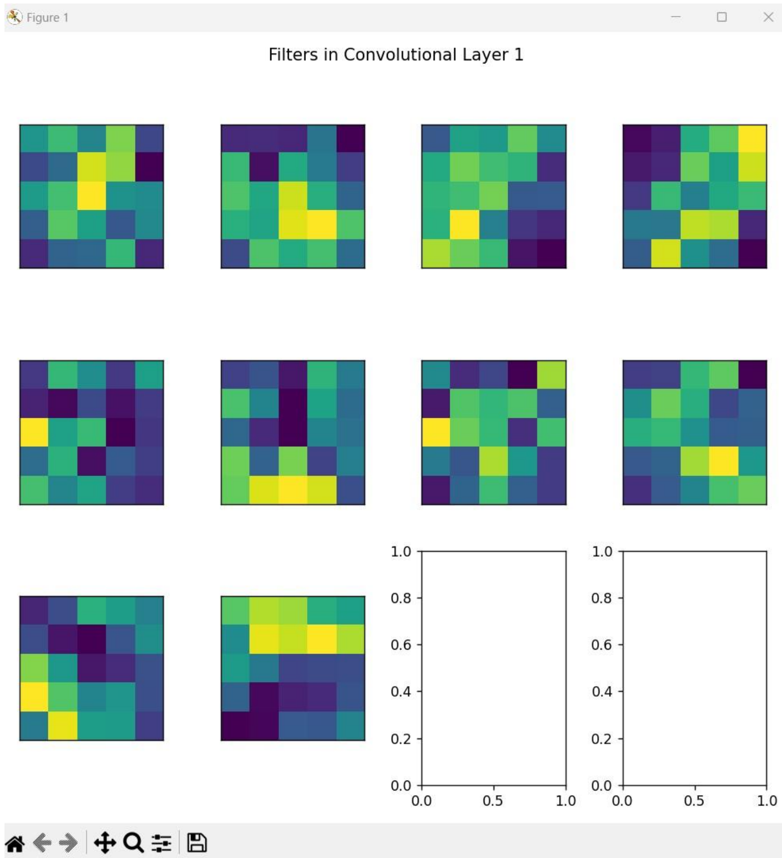


Fig-7 First layer visualization

(B) In your report, include the plot and note whether the results make sense given the filters.

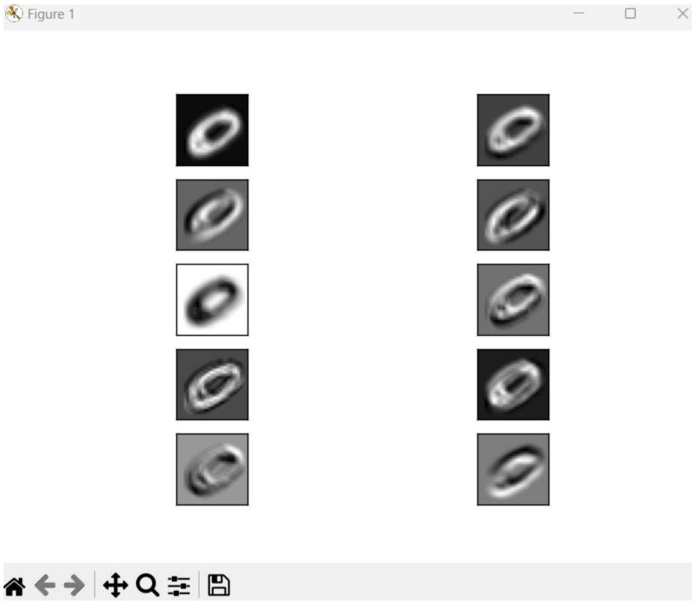


Fig-8 filtered image plot

Task-3:

A plot of the training error, a printout of your modified network, and the results on the additional data.

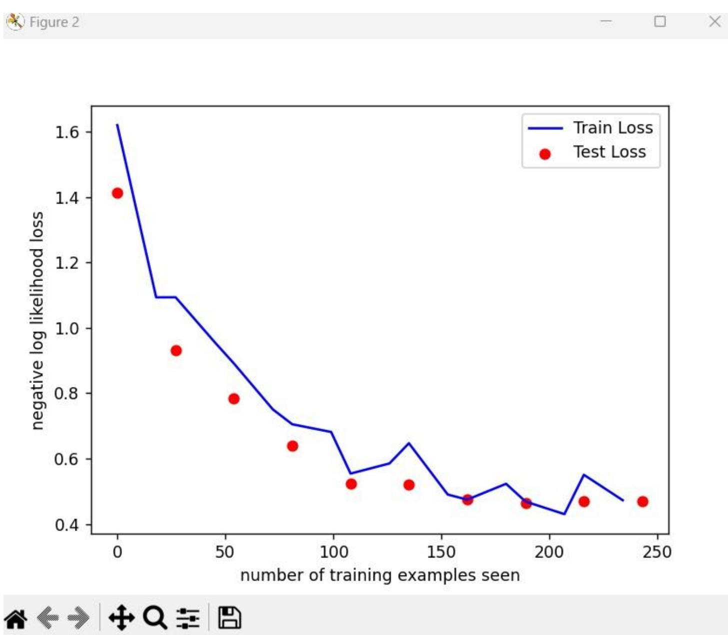


Fig-9 Plot of training error

```
MyNet(
  (conv1): Conv2d(1, 10, kernel_size=(5, 5), stride=(1, 1))
  (pool1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv2d(10, 20, kernel_size=(5, 5), stride=(1, 1))
  (dropout): Dropout2d(p=0.5, inplace=False)
  (pool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (fc1): Linear(in_features=320, out_features=50, bias=True)
  (relu): ReLU()
  (fc2): Linear(in_features=50, out_features=3, bias=True)
)
```

Fig-11 Modified network parameters

```
Test set: Avg. loss: 1.4120, Accuracy: 8/20 (40%)
Train Epoch: 1 [0/27 (0%)]      Loss: 1.619970
Train Epoch: 1 [18/27 (60%)]    Loss: 1.093157

Test set: Avg. loss: 0.9312, Accuracy: 12/20 (60%)
Train Epoch: 2 [0/27 (0%)]      Loss: 1.093402
Train Epoch: 2 [18/27 (60%)]    Loss: 0.956874

Test set: Avg. loss: 0.7831, Accuracy: 13/20 (65%)
Train Epoch: 3 [0/27 (0%)]      Loss: 0.890944
Train Epoch: 3 [18/27 (60%)]    Loss: 0.750632

Test set: Avg. loss: 0.6402, Accuracy: 14/20 (70%)
Train Epoch: 4 [0/27 (0%)]      Loss: 0.705056
Train Epoch: 4 [18/27 (60%)]    Loss: 0.681535

Test set: Avg. loss: 0.5251, Accuracy: 16/20 (80%)
Train Epoch: 5 [0/27 (0%)]      Loss: 0.554258
Train Epoch: 5 [18/27 (60%)]    Loss: 0.585219

Test set: Avg. loss: 0.5215, Accuracy: 17/20 (85%)
Train Epoch: 6 [0/27 (0%)]      Loss: 0.647121
Train Epoch: 6 [18/27 (60%)]    Loss: 0.490091

Test set: Avg. loss: 0.4765, Accuracy: 17/20 (85%)
Train Epoch: 7 [0/27 (0%)]      Loss: 0.474656
Train Epoch: 7 [18/27 (60%)]    Loss: 0.523063

Test set: Avg. loss: 0.4636, Accuracy: 17/20 (85%)
```

```
Train Epoch: 8 [0/27 (0%)]      Loss: 0.468671
Train Epoch: 8 [18/27 (60%)]    Loss: 0.430235

Test set: Avg. loss: 0.4709, Accuracy: 17/20 (85%)
Train Epoch: 9 [0/27 (0%)]      Loss: 0.550590
Train Epoch: 9 [18/27 (60%)]    Loss: 0.472743

Test set: Avg. loss: 0.4708, Accuracy: 17/20 (85%)
```

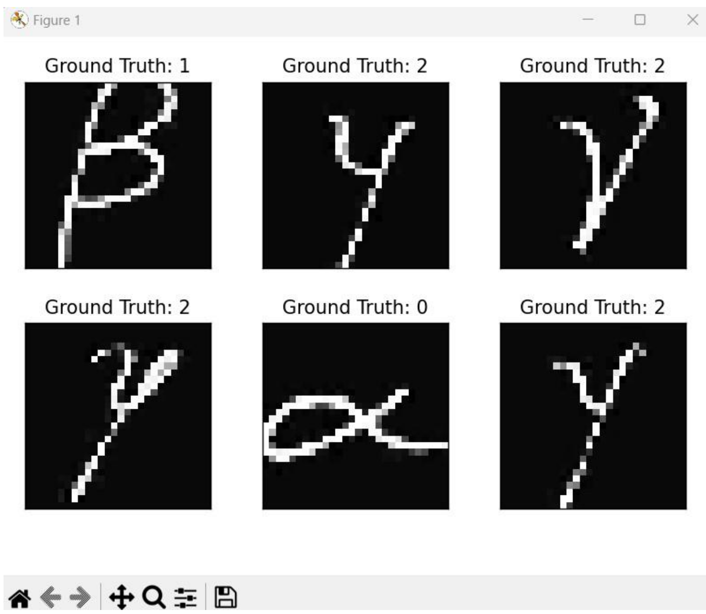


Fig-12 Results-1(Ground truths)

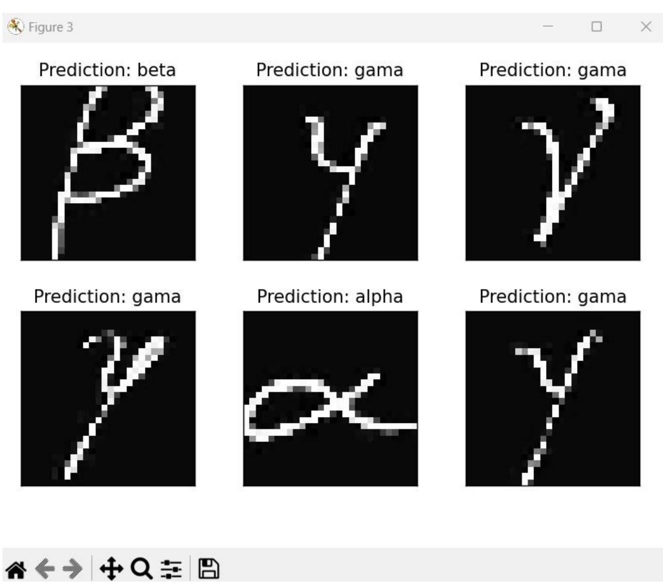
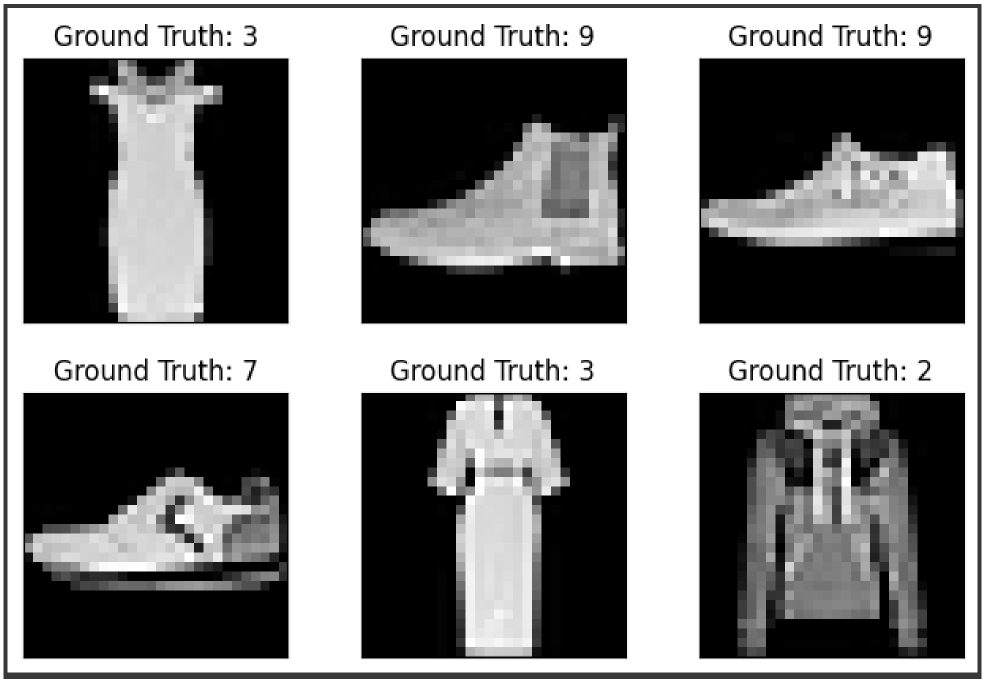


Fig-13 Results-2(Predictions)

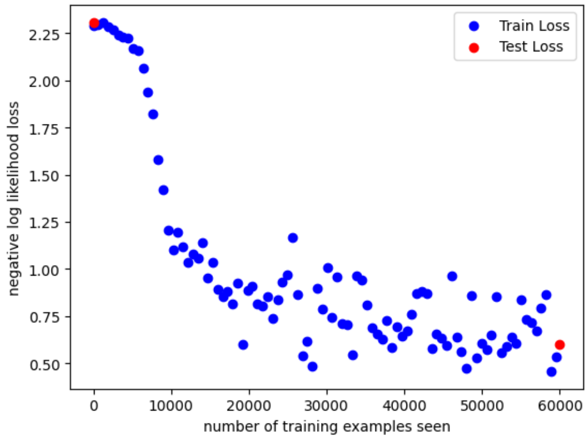
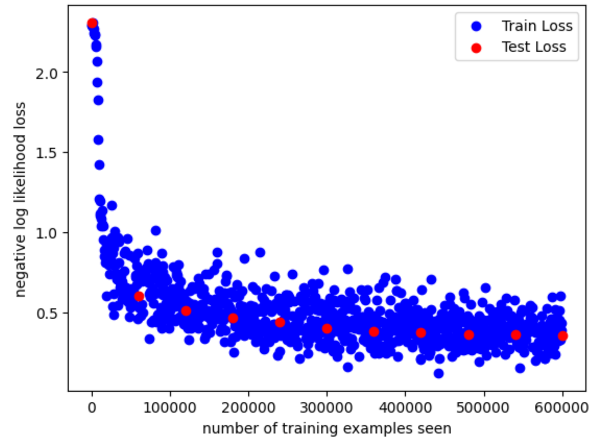
Task-4:



Parameter varied	**Hypothesis / Expected output**
Number of Epochs	As we increase the epoch number the accuracy has to increase since the number of times the model is being trained is more.
Training Batch size	Since more training data will be provided the accuracy has to be increased
Dropout rate in drop out layer	A specific value which doesnt overfit the model has to be existing.By varying the rate a suitable drop out rate value can be identified based on better accuracy.

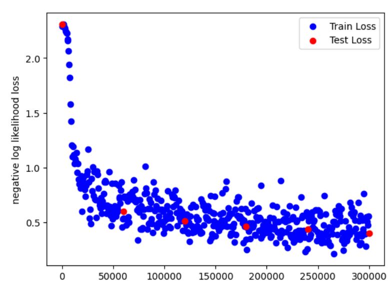
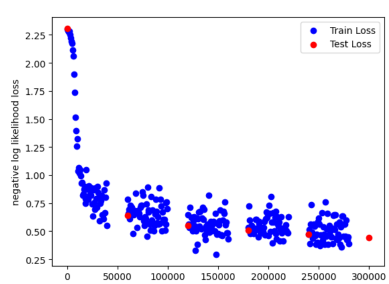
Parameter-1:

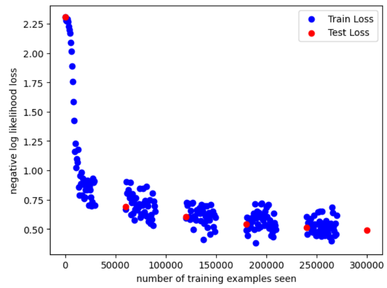
Epoch varying from 1 to 10

Average loss for Epoch-1	**Average loss for Epoch-10**
	

Parameter-2

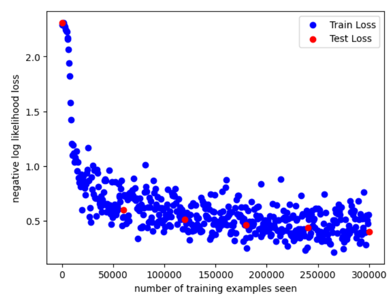
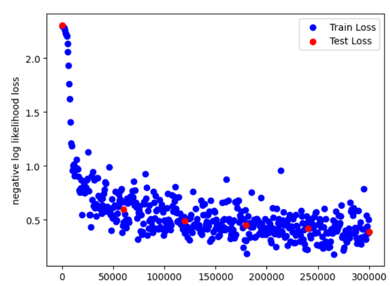
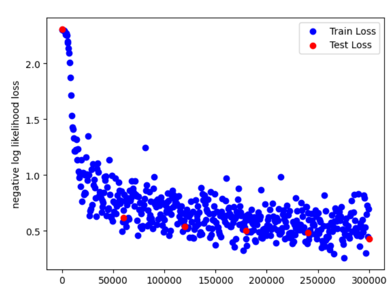
Training Batch size

Batch size	**Average Loss**	**Result**
64		Avg. loss: 0.3993, Accuracy: 8531/10000 (85%)
96		Avg. loss: 0.4426, Accuracy: 8425/10000 (84%)

128		Avg. loss: 0.4883, Accuracy: 8130/10000 (81%)

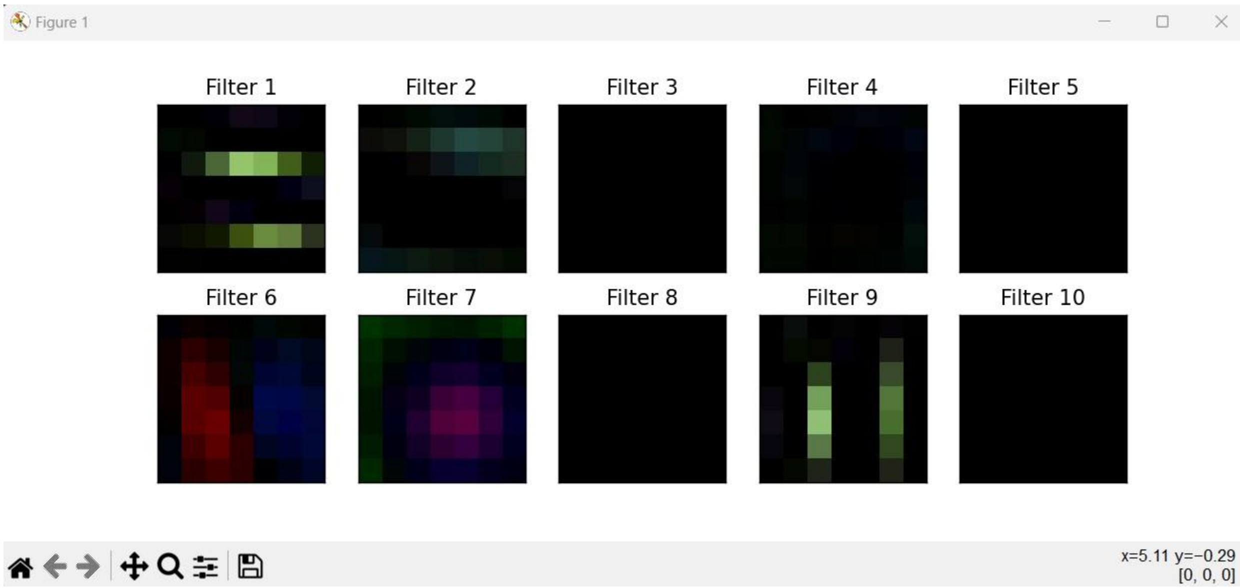
Parameter-3

Dropout rate in drop out layer

P-value	**Average Loss**	**Result**
p=0.5		Avg. loss: 0.3993, Accuracy: 8531/10000 (85%)
p=0.3		Avg. loss: 0.3868, Accuracy: 8550/10000 (86%)
p=0.7		Avg. loss: 0.4313, Accuracy: 8449/10000 (84%)

Extensions:

-> There are many pre-trained networks available in the PyTorch package. Try loading one and evaluate its first couple of convolutional layers as in task 2.



ResNET-18 neural network filter outputs

A short reflection of what you learned.

Learned how to build and train models and got an experience dealing with deep networks and getting hands-on experience with pytorch and working with datasets MNIST and FashionMNIST. Understood how transfer learning works.

Acknowledgement of any materials or people you consulted for the assignment

https://pytorch.org/tutorials/beginner/basics/intro.html

https://nextjournal.com/gkoehler/pytorch-mnist

https://github.com/martisak/dotnets