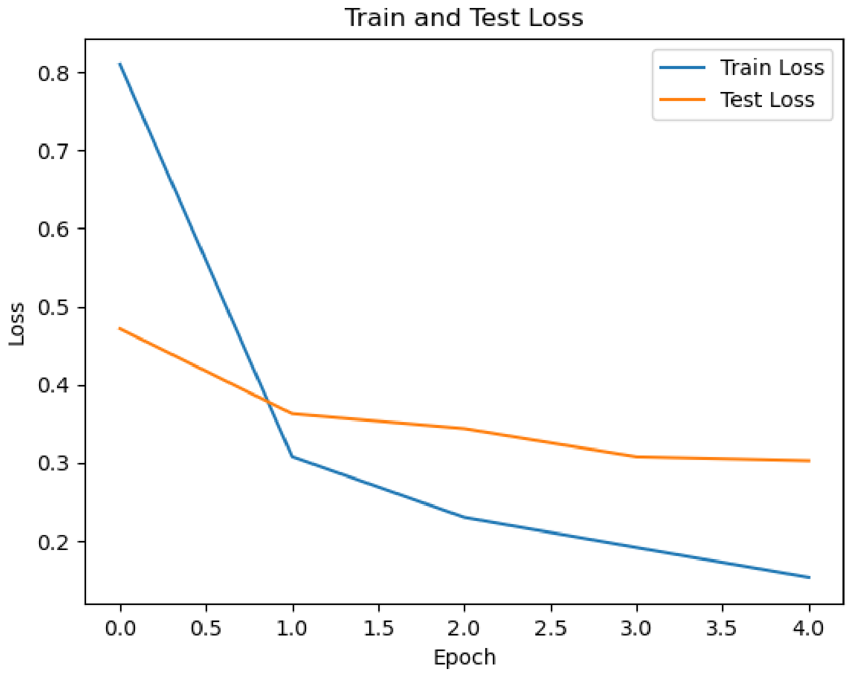
Assignment 3 - Report

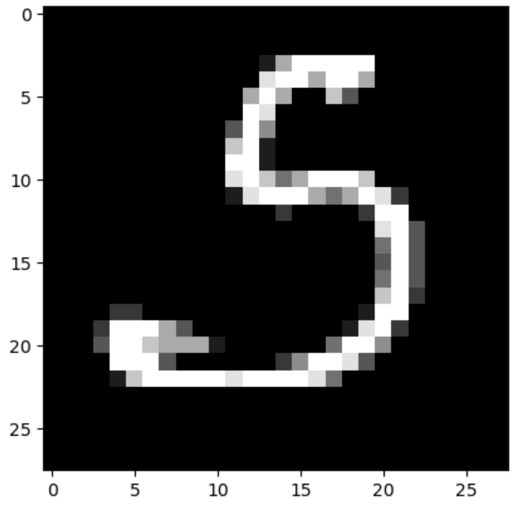
Task 1

```
Epoch [1/5], Train Loss: 0.8105, Test Loss: 0.4716
Epoch [2/5], Train Loss: 0.3076, Test Loss: 0.3627
Epoch [3/5], Train Loss: 0.2298, Test Loss: 0.3435
Epoch [4/5], Train Loss: 0.1911, Test Loss: 0.3074
Epoch [5/5], Train Loss: 0.1528, Test Loss: 0.3024
```

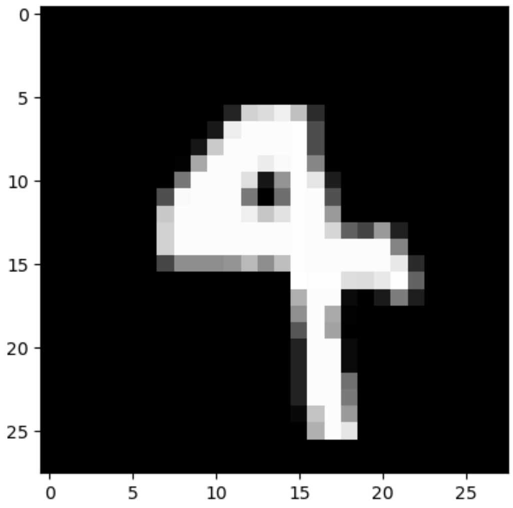


Accuracy of the network on the 10000 test images: 90.88333333333334 %

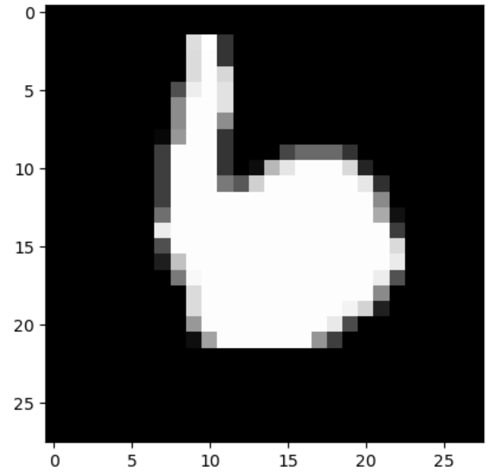
Some misclassified images



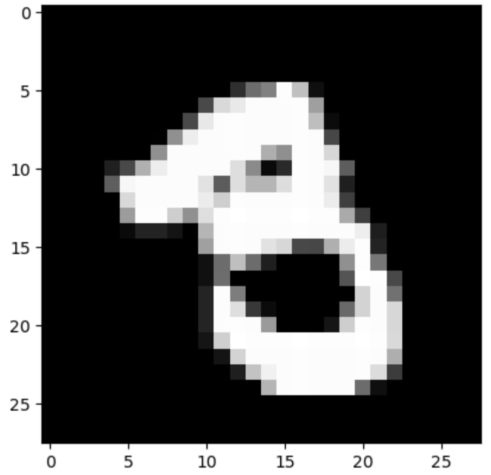
```
Predicted:  3
Actual:  5
```



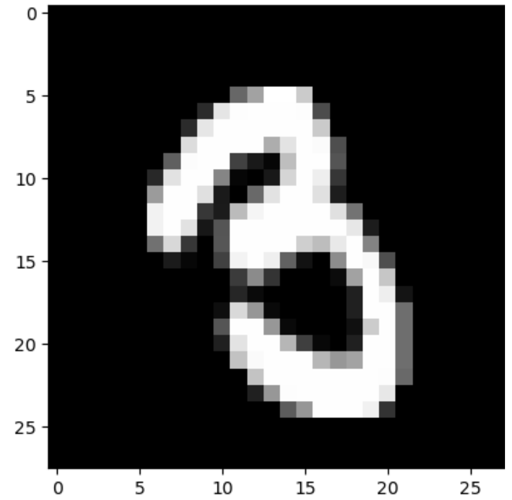
```
Predicted:  9
Actual:  4
```



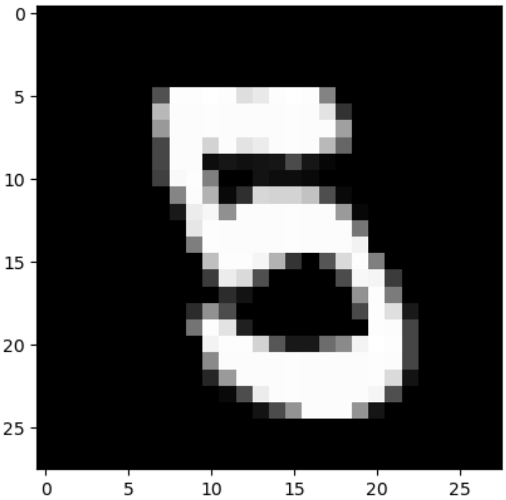
```
Predicted:  4
Actual:  6
```



```
Predicted:  8
Actual:  3
```



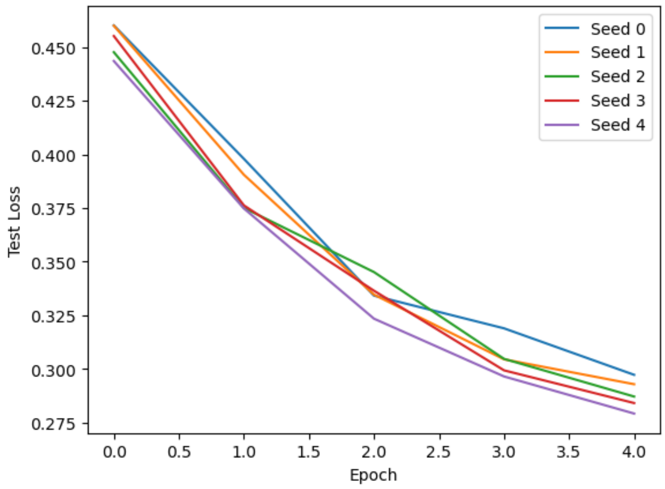
```
Predicted:  8
Actual:  3
```



```
Predicted:  8
Actual:  5
```

Task 2

Test loss for each seed



Our Results:

Test error mean: 0.2882, **Test error std**: 0.0064

Based on the variance of the final test errors for each of the seeds we computed (Variance is the square of std).
As we can see, the standard deviation is low in comparison to the mean, also, in the plot we got we can observe that on all 5 independent runs (for which we trained are model) the graphs representing the test errors are similar to each other.
So, we can conclude that our model is indeed robust to the choice of the seed number.

Task 3

$(\boldsymbol{e}_{te}^*, \boldsymbol{e}_{va}^*)$ such that e_{va}^* is the minimum throughout the training:
[0.2073454052209854, 0.2533122509531677]

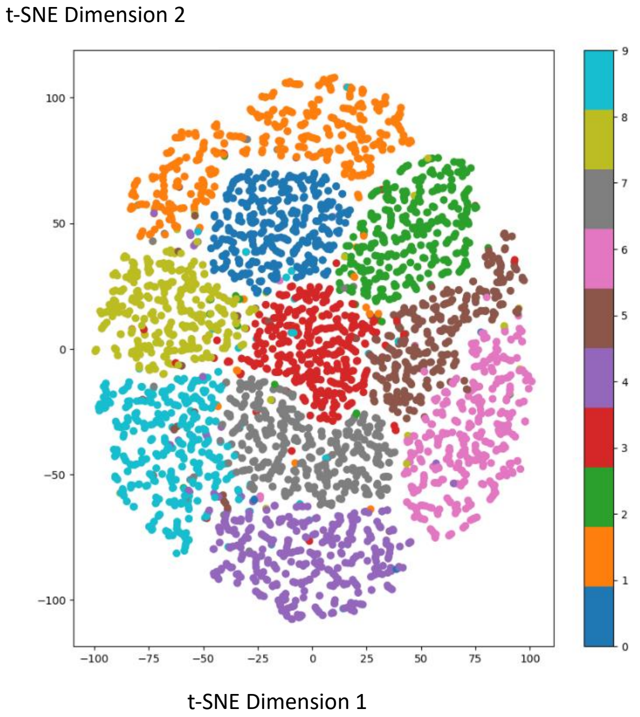
Task 4

Batch Size	Hidden Size	Learning Rate	Best validation error	Test error for same epoch
100	400	0.01	0.28189556	0.316162283
100	400	0.001	0.281020664	0.320208677
100	500	0.01	0.285375091	0.325582019
100	500	0.001	0.283492445	0.319609016
100	600	0.01	0.282584982	0.312651986
100	600	0.001	0.27938626	0.30534839
200	400	0.01	0.250575647	0.350030641
200	400	0.001	0.245074734	0.350435288
200	500	0.01	0.25269933	0.341736364
200	500	0.001	0.244236782	0.341215304
200	600	0.01	0.243746725	0.341965031
200	600	0.001	0.247361264	0.342492678
300	400	0.01	0.288571913	0.385693705
300	400	0.001	0.28823502	0.370264602
300	500	0.01	0.309972566	0.411241569
300	500	0.001	0.302044176	0.380348326
300	600	0.01	0.280688416	0.374043055
300	600	0.001	0.303019803	0.374510198

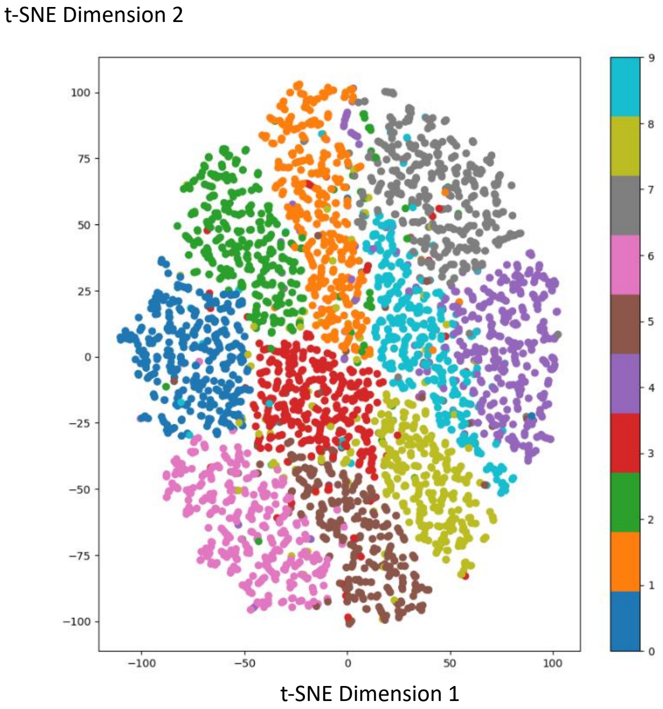
The highlighted combination is the one with the best performance, which is:
Batch size = 100
Hidden size (of the intermediate layer) = 600
Learning rate = 0.01

Task 5

Z Graph



X Graph



As we can observe in these plots, the z_i points are slightly more spread (we can see that in digits 1,6,9) and the "clusters" of the digits are a bit more separable (4 and 9, 3 and 8).

After applying the first layer on the input (and getting z_i from x_i), the $z_i s$ were more separable which means the learned model indeed made a "step" in the right direction of its classification task. By that, we can assume that the next layer will also classify some digits to their correct label even more and lower the test loss.
So, we can say that the learned model does indeed have parameters (weights) (that it obtained during the learning process) and operating function (ReLU in our case) that are compatible with the classification task.