**Computer Vision – Exercise 3**

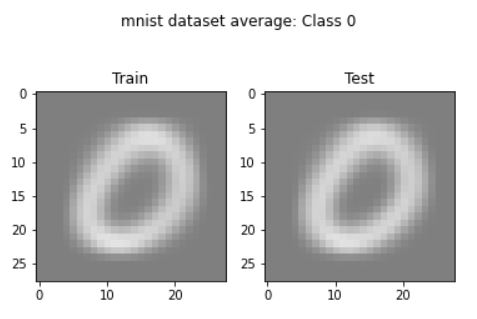
**Kareem Jabareen - 211406343**

**Malik Egbaria - 318585627**

**Part 1:**

* **Q1**:

For this question, we calculated the average image for each class in MNIST both for the training dataset and the test one. After that we normalized the values in the averaged images by finding the minimum and the maximum values of each image.  
Then, using Matplotlib, we plotted the results as follows:

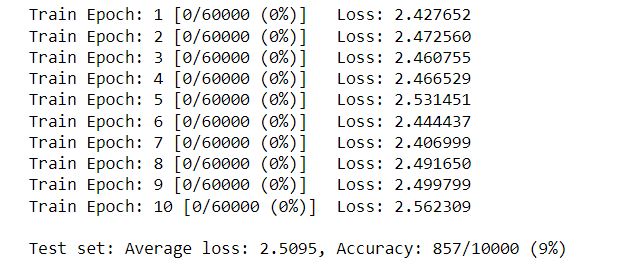


* **Q2**:

A) Using the utilities of **Pytorch**, we created the **DataLoader** objects to load the data and process it, then we trained the CNN and plotted the results including the loss value. Eventually, we called the function called **test** to test our model and plotted its accuracy on the test dataset.

* After that we retrieved the weights from the network parameters and plotted them as normalized images.

B) We created a new method and called it **limited\_train**, which reduced the number of training examples to just 50, then we used it for training our new model and got the following results:

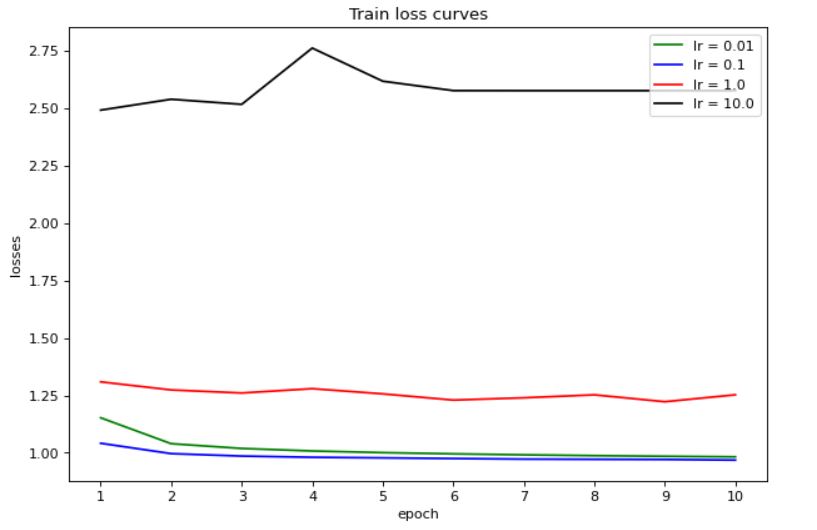


Explanation: As we can see “To be continued” -under fitting.

* **Q3**:

A) We defined a New Class called **MultiLayerNet** in which we added a new hidden linear layer with 1000 Units. In the Forward Function, we called the mentioned layer and applied the tanh non-linearity.  
  
B) We created a new Model using the mentioned Class above and trained it on MNIST four times with different learning rate each time [ 0.01 , 0.1 , 1.0 , 10.0 ] and tested each one of them.

* Eventually we Plotted the training loss curve for each learning rate and got the following results:



For the test accuracy, the learning rate 0.1 and the 0.01 got the best results and they were so close, that’s because **outliers** have less impact in these models than in the other layers with great learning rate.

The same reason explains why a learning rate equals 1.0 gave us a better accuracy than the learning rate equals 10.0 did.

- **Q4**:

When we used the 3x3 convolution Layers, we got the following output:

