EEE 443/543 - Project #3

a-e)

In this project the aim was to design and train a PTA for digit classification. The MNIST database, which is used for this project, involves 60k train and 10k test images with the corresponding labels.

Each image is 28×28 pixels, therefore the input layer of our neural network will consist of 28×28=784 nodes. The output layer will have 10 nodes, corresponding to the digits 0 through 9. The biases are not included. The goal was to determine 784×10=7840 weights so that the network produces an output of $[1\ 0\ 0\ \cdots\ 0]^T$ when the input image is a 0, $[0\ 1\ 0\ \cdots\ 0]^T$ when the input image is a 1, and similarly for the other digits.

```
Training images shape: (60000, 784)
Training labels shape: (60000,)
Test images shape: (10000, 784)
Test labels shape: (10000,)
```

After downloading and loading the MNIST dataset the PTA algorithm was built and implemented following the given steps in the project manual. The results with different parameters were observed and discussed in the subsequent sections.

In the following steps the numpy random seed is set to 29 to be able to get the same results after each run:

```
np.random.seed(29)
```

In this project there are 3 parameters that we dealt with:
First one is n (n<60k) which denotes the first n elements of the training images and training labels that are used in the PTA process.
Second parameter is η which is the learning rate. It determines the how much the update rule depends on the past. If η I large, the weights will be more susceptible to the latest iteration.
Last parameter is ϵ, it is the ratio of misclassifications to the total number of test samples (10000 in our case). For smaller train sets ϵ may converge to 0, however, for larger sets it way harder, sometimes impossible, to obtain reach ϵ = 0.
In this project the affect of different parameters are observed.

f)

In this step the task was to run the train and test algorithms with n = 50, η = 1, and ϵ = 0. In the training step number of misclassifications converged to 0 after 5 epochs.

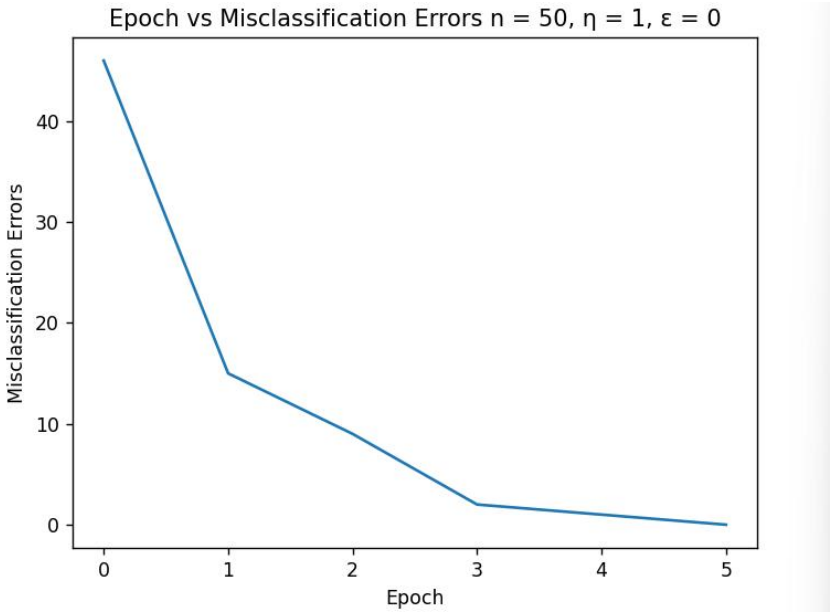

Fig.1 Epoch vs Misclassification Errors n = 50, η = 1, ε = 0

The algorithm is tested using the test set that contains 10k instances with the trained weights and the error rate is calculated as the following:

```
n = 50, η = 1, ε = 0, Test Error: 42.94%
```

When n = 50, 42.94% error was observed. Even though, there were 0 misclassifications in the training set, the algorithm failed in the test set because the number of samples that are used in the training process was not sufficient to classify a large dataset.

g)

In this step the task was to run the train and test algorithms with n = 1000, η = 1, and ϵ = 0. In the training step number of misclassifications converged to 0 after 32 epochs.

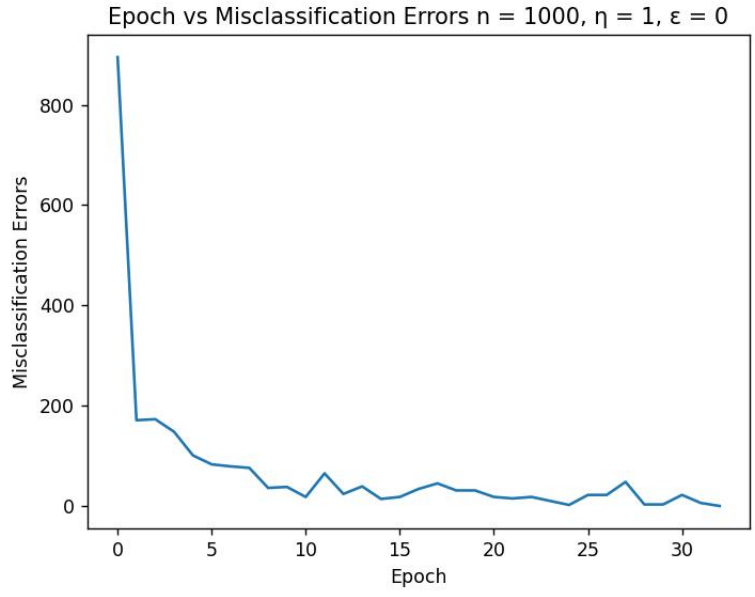

Fig.2 Epoch vs Misclassification Errors n = 1000, η = 1, ε = 0

The algorithm is tested using the test set that contains 10k instances with the trained weights and the error rate is calculated as the following:

```
n = 1000, η = 1, ε = 0, Test Error: 17.22%
```

When n = 1000, 17.22% error was observed. In this case the algorithm was relatively successful compared to previous case (n = 50). This is because now the algorithm trained with a set that is relatively sufficient.

h)

In this step the task was to run the train and test algorithms with n = 60000, η = 1, and ϵ = 0. In the training step number of epochs did not converge since the number of misclassifications did not converged to 0. Instead, as it can be seen from *Fig.3,* the number of misclassifications oscillated around 8.5k. Since the number of epochs diverged, the code is arranged to terminate after 50 epochs as it is a sufficient number to make observations.

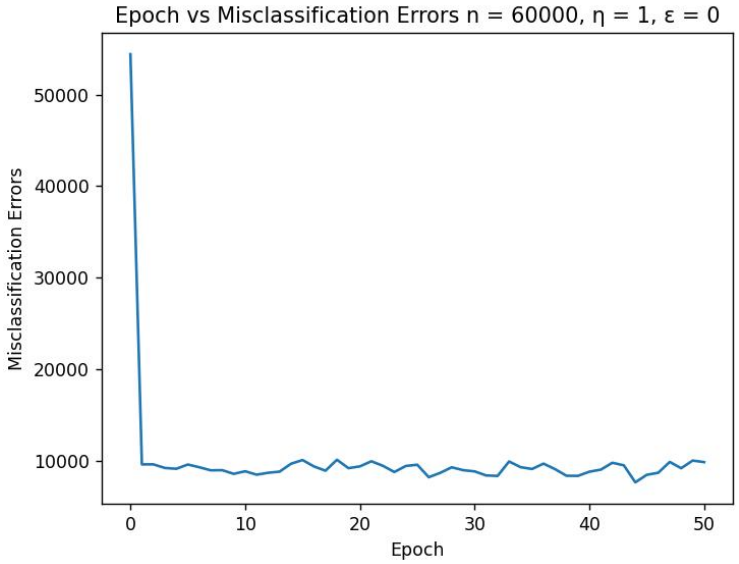

Fig.3 Epoch vs Misclassification Errors n = 60000, η = 1, ε = 0

The algorithm is tested using the test set that contains 10k instances with the trained weights and the error rate is calculated as the following:

```
n = 60000, η = 1, ε = 0, Test Error: 17.23%
```

When n = 60000, 17.23% error was observed. In this case, the success of the algorithm was nearly the same compared to previous case (n = 1000). This means from this point the success of the algorithm may be improved by adjusting other parameters (η and ϵ) instead of increasing the size of the dataset.

i)

According to observations from the previous step the parameters are chosen as n = 60000, η = 0.5, and ϵ = 0.142.

In the previous part after certain number of epochs the algorithm is oscillates around 8500 that is visible in *Fig.3*. Therefore the ϵ is chosen as:

$$\epsilon = \frac{8500}{60000} \approx 0.142$$

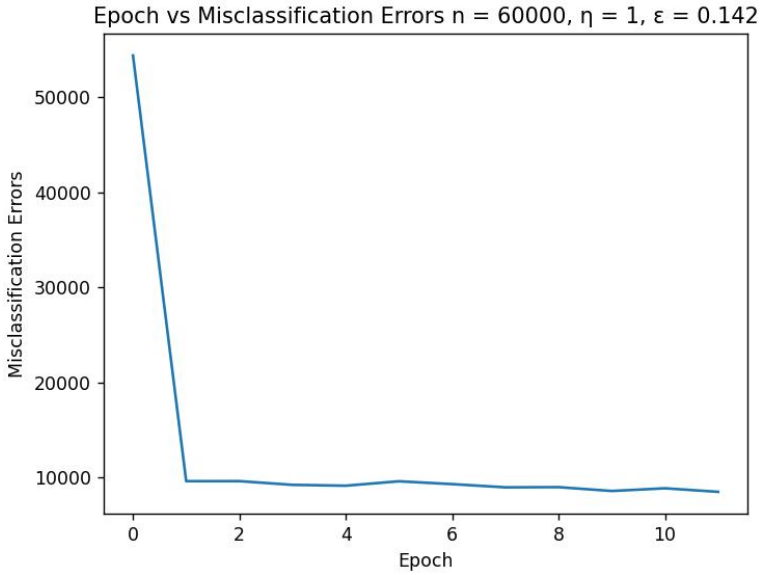

Fig.4 Epoch vs Misclassification Errors n = 60000, η = 1, ε = 0.142

In *Fig.4,* it is visible that the algorithm has terminated after 11 epochs due to increased ϵ. This increased the efficiency of the training process.

The algorithm is tested using the test set that contains 10k instances with the trained weights and the error rate is calculated as the following:

```
n = 60000, η = 1, ε = 0.142, Test Error: 14.63%
```

When ϵ = 0.142, the 14.63% error was observed. This is due to the training process is terminated on a controlled and relatively small misclassification number. In the previous case *Fig.3* it was terminated when the epoch number reached to 50 which was basically random therefore it may end up in a suboptimal value.

Now, with the fixed n and ϵ values, different η values are tested to chose the optimal one. The results are below in *Fig 5&6.*

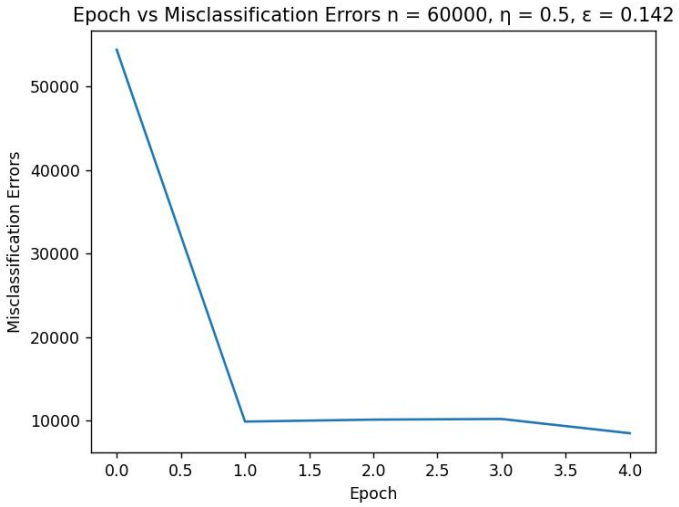

Fig.5 Epoch vs Misclassification Errors n = 60000, η = 0.5, ε = 0.142

```
n = 60000, η = 0.5, ε = 0.142, Test Error: 14.65%
```

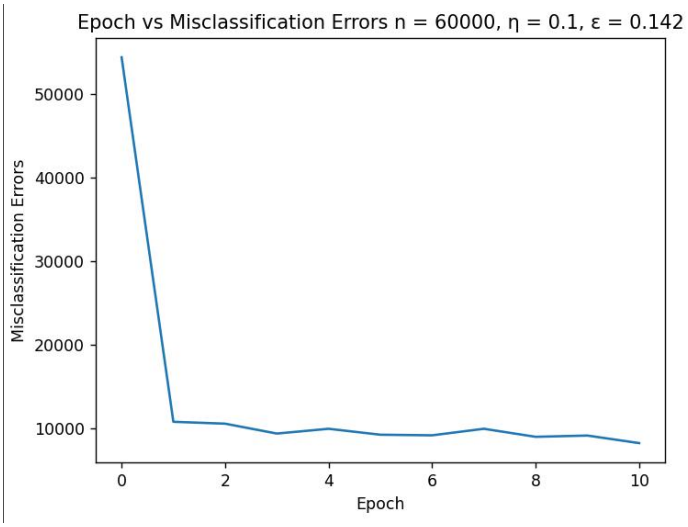

Fig.6 Epoch vs Misclassification Errors n = 60000, η = 0.1, ε = 0.142

```
n = 60000, η = 0.1, ε = 0.142, Test Error: 14.19%
```

The learning rate η = 0.5 is chosen because it reaches the desired misclassification rate fastest and the test error does not change much with different η. Larger η may result unnecessary oscillations where as smaller η may result in unnecessary delays.

Therefore the optimal values determined as:

$$n = 60000, \eta = 0.5, \varepsilon = 0.142$$

The algorithm has trained and tested using different initial weights 3 times and the results have been obtained.

```
Iteration: 1
n = 60000, η = 0.5, ε = 0.142, Test Error: 14.59%
Iteration: 2
n = 60000, η = 0.5, ε = 0.142, Test Error: 13.41%
Iteration: 3
n = 60000, η = 0.5, ε = 0.142, Test Error: 14.63%
```

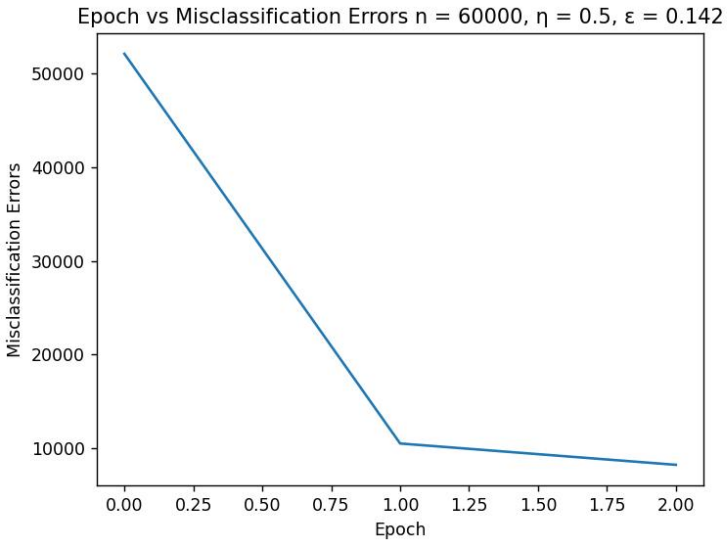

Fig.7 Epoch vs Misclassification Errors with optimal values, iteration 1

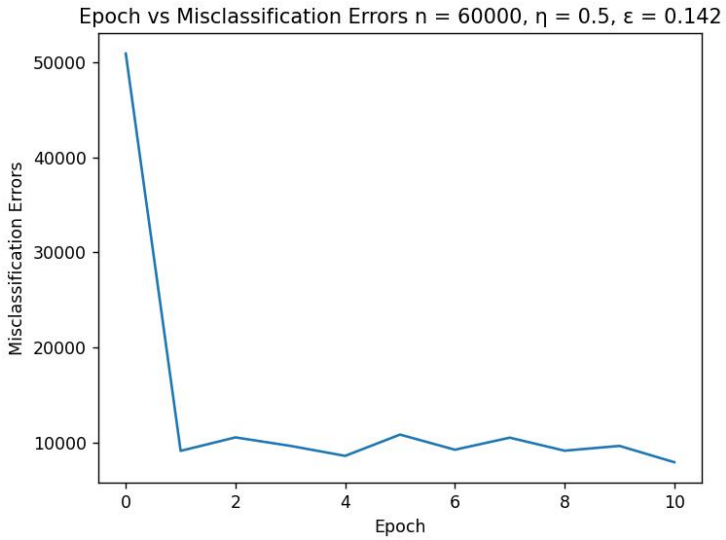

Fig.8 Epoch vs Misclassification Errors with optimal values, iteration 2

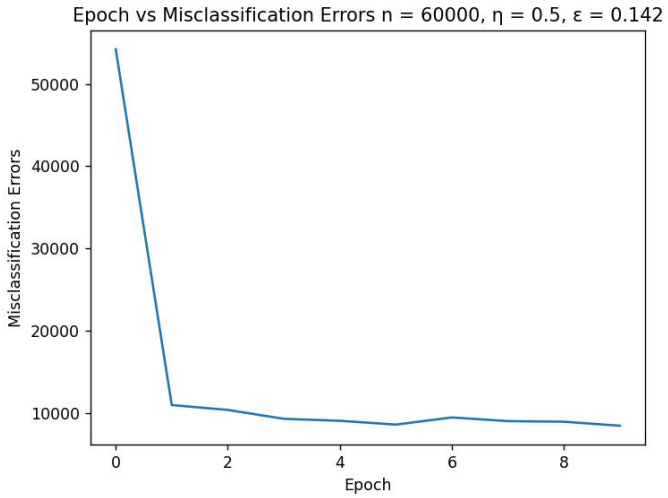

Fig.9 Epoch vs Misclassification Errors with optimal values, iteration 3

$$W \leftarrow W + \eta\left(d(x_i) - u(Wx_i)\right)x_i^T$$

From *Fig. 7-9* and the test error results it can be observed that different initial weights gave totally different results with the same parameters. This means the algorithm is highly depends on the initial weights. This is due to the update rule, that is give above, depends on the initial weights.