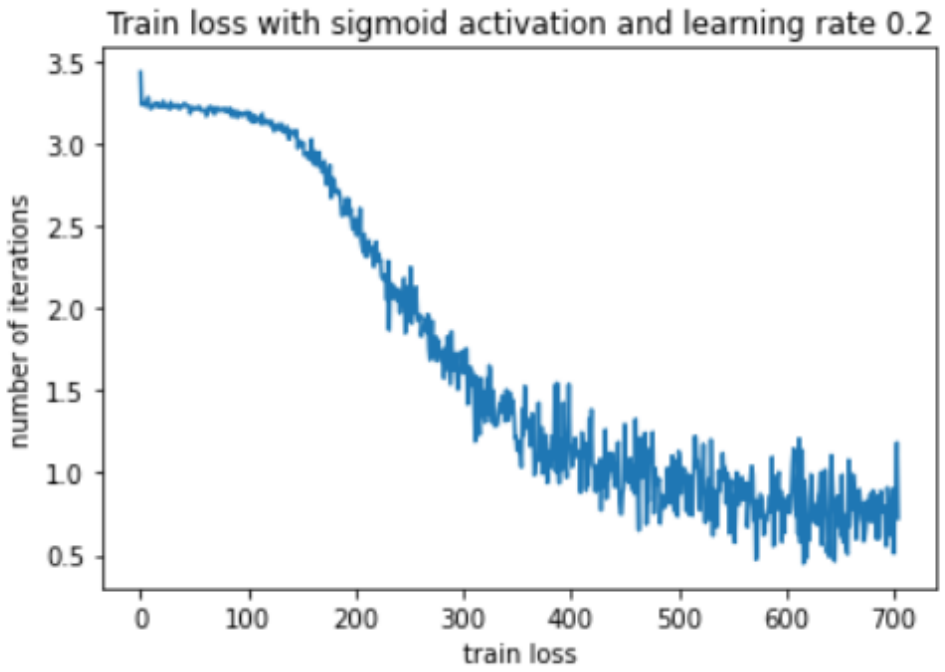
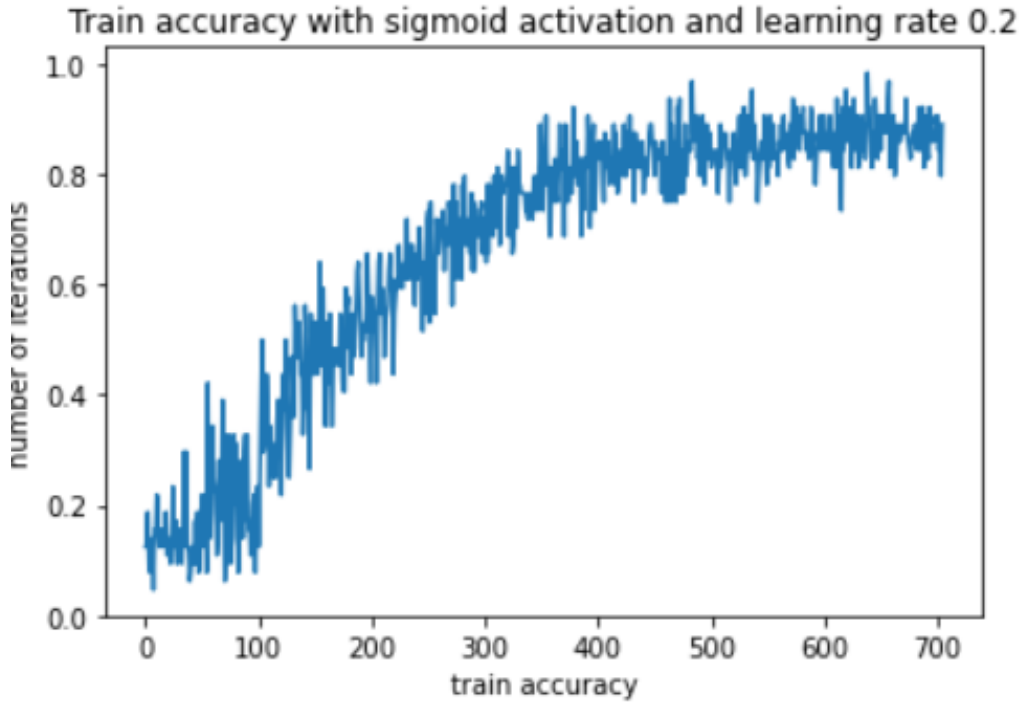
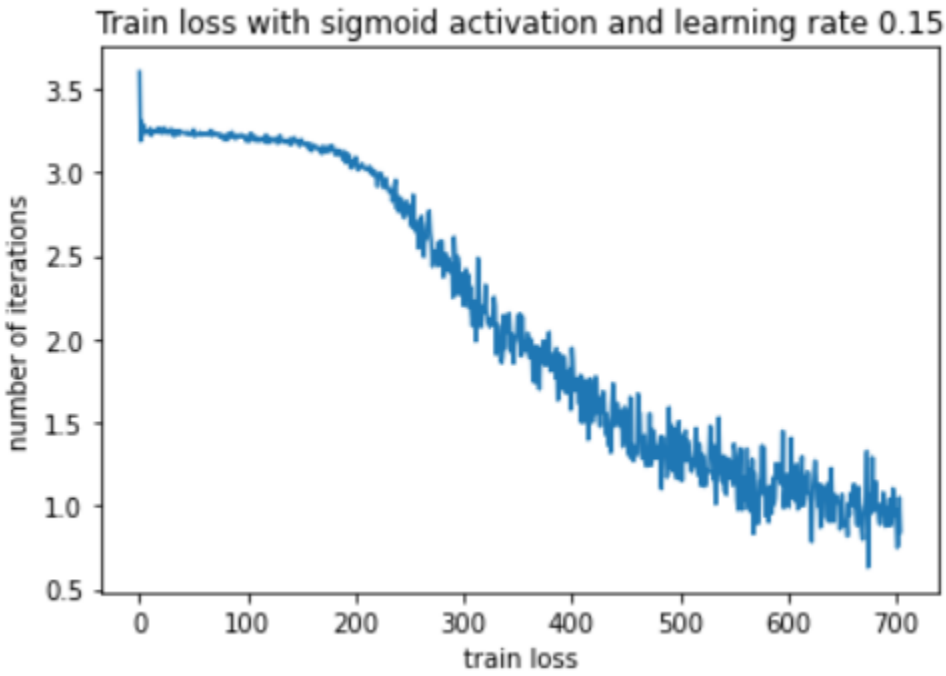
EE5179 Programming Assignment 1 - Report

1. Training loss and accuracy plots using sigmoid activation function with learning rate 0.2

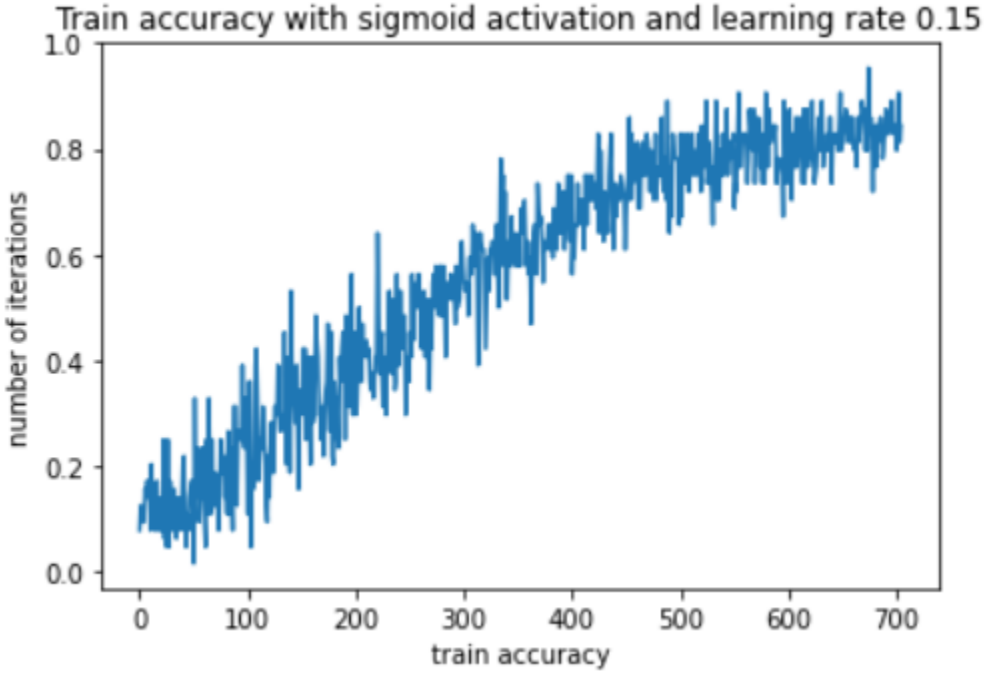




2. Training loss and accuracy plots using sigmoid activation function with learning rate 0.15



Training loss and accuracy plots using sigmoid activation function with learning rate 0.15



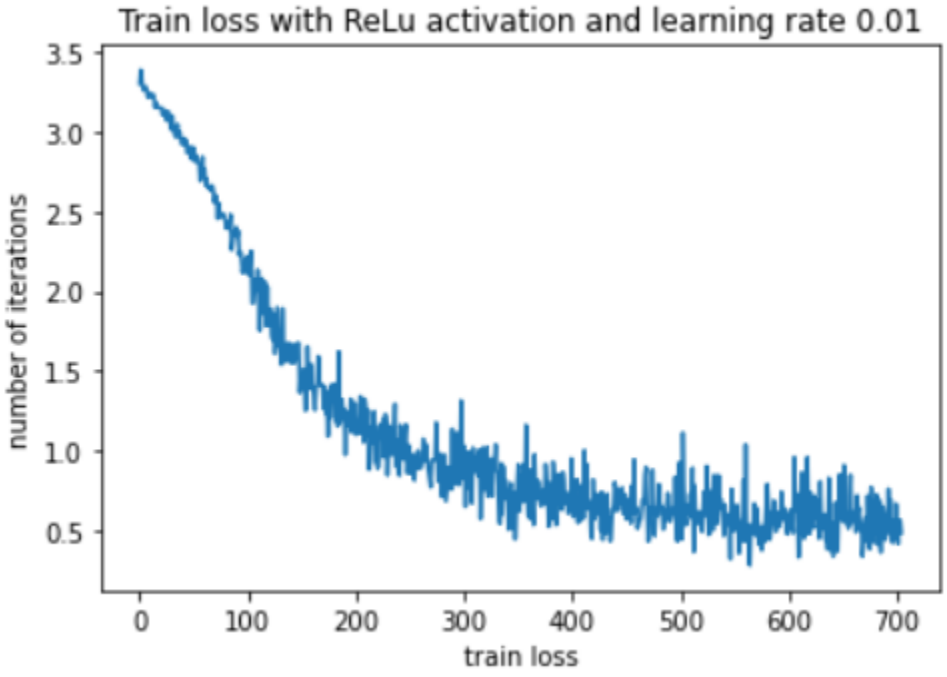
<u>Observations:</u>

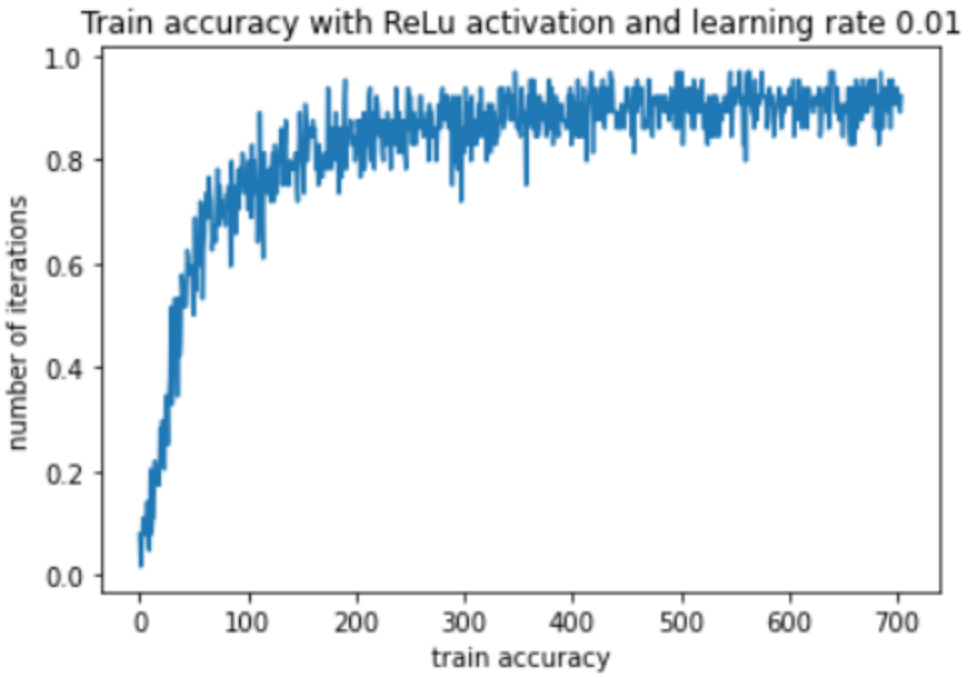
The model was performing bad with low learning rate as 0.01 and as learning rate increases to 0.15 to 0.3 the performance of the model has been improved.

Accuracies:

Learning Rate	Train Accuracy	Test Accuracy
0.15	85.45%	85.72%
0.2	92.34%	92.01%

3. Training loss and accuracy plots using ReLu activation function with learning rate 0.01





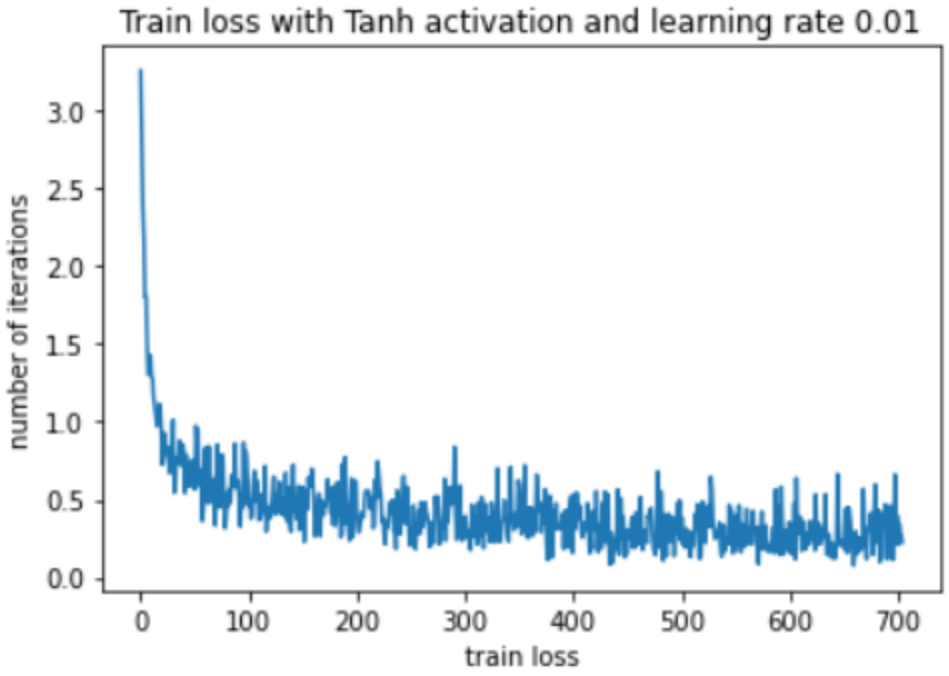
Observations:

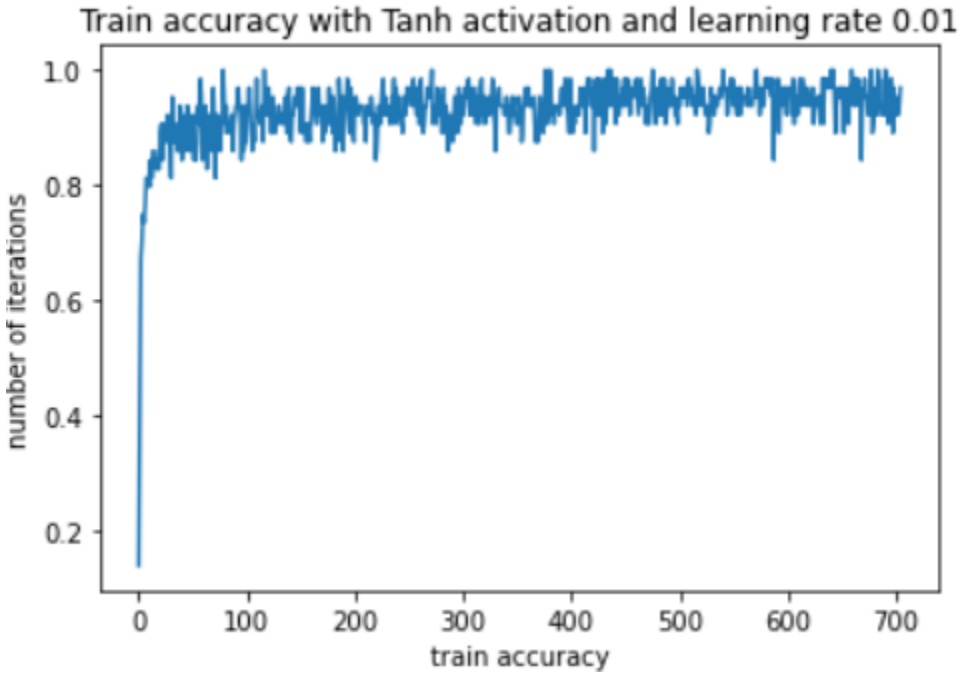
The model was performing better with Relu activation function compared to sigmoid activation function with learning rate of 0.01

Accuracies:

Learning Rate	Train Accuracy	Test Accuracy
0.01	92.19%	91.17%

4. Training loss and accuracy plots using Tanh activation function with learning rate 0.01





Accuracies:

Learning Rate	Train Accuracy	Test Accuracy
0.01	96.875%	95.49%

Steps followed for implementation of MLP from scratch:

1. Defined activation functions and their derivative functions.
2. Weights and biases initializations
3. Feed forward pass based on the given activation function and store the output
4. Calculate gradients to update the parameters(back propagation)
5. Update weights and biases (gradient descent)
6. Train the model in batches - model has been trained with 15 epochs and in each epoch all the training data has been trained in the batches of 64
7. Predicted the outputs on the test data by performing forward pass using trained parameters.
8. Plotted the confusion matrix with predicted outputs and actual outputs of the test data.
9. Plotted the loss vs number of iterations and accuracy vs number of iterations plots for the training data.
10. Calculated the train and test accuracies.
11. Performed different experiments by changing the learning rate and activation functions.
12. Due to computational constraints couldnt perform experiments by changing the number of iterations and number of epochs.

MLP implementation using pytorch:

To implement MLP using pytorch I have used the following online resources

- https://github.com/milindmalshe/Fully-Connected-Neural-Network-PyTorch/blob/master/FCN_MNIST_Classification_PyTorch.py

- https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwisi7udueb5AhU6zzgGHWvaADwQFnoECAkQAQ&url=https%3A%2F%2Fnextjournal.com%2Fgkoehler%2Fpytorch-mnist&usg=AOvVaw2JpN80D41L-qXVTfWRW8iB

- https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwisi7udueb5AhU6zzgGHWvaADwQFnoECCAQAQ&url=https%3A%2F%2Fwww.kaggle.com%2Fcode%2Ffranklemuchahary%2Fmnist-digit-recognition-using-pytorch&usg=AOvVaw1Xo3MfXs0-oFk3mKU90Oqc