DL ASSIGNMENT-2 Part-1

1)

Pickle Files:

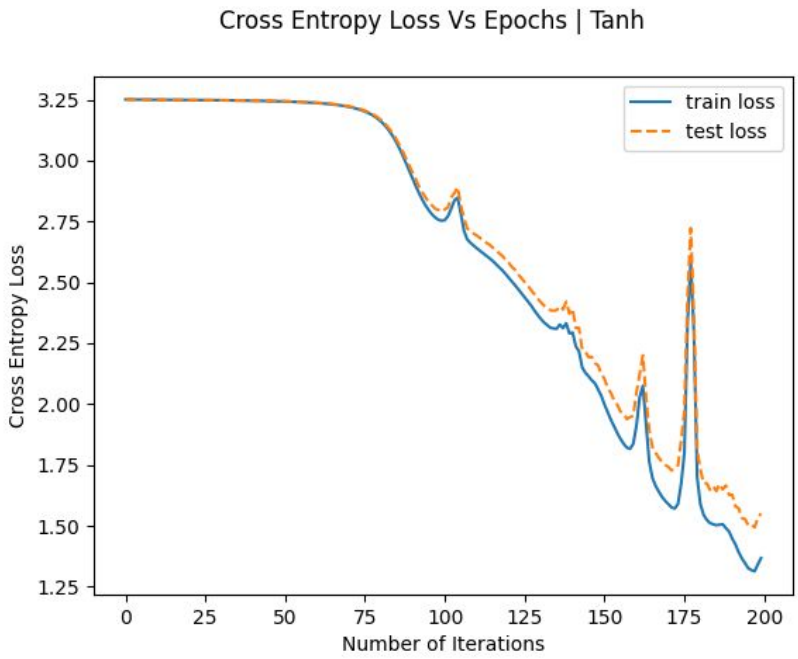
Structure:

```
class MLPClassifier():
    """
    My implementation of a Neural Network Classifier.
    """

    acti_fns = ['relu', 'sigmoid', 'tanh']
    weight_inits = ['random', 'he', 'xavier']

    def __init__(self, layers, num_epochs, dropouts=0, learning_rate=1e-5, activation_function='relu',
        optimizer='gradient_descent', weight_init='random', regularization='l2', batch_size=64, **kwargs):
        """
        Initializing a new MyNeuralNetwork object

        Parameters
        ----------
        - learning_rate: Learning rate of the neural network. Default value = 1e-5.

        - activation_function: A string containing the name of the activation function to be
        used in the hidden layers. For the output layer use Softmax activation function. Default
        value = “relu”.

        - optimizer: A string containing the name of the optimizer to be used by the network.
        Default value = “gradient_descent”.

        - Weight_init: “random”, “he” or “xavier”: String defining type of weight initialization
        used by the network. Default value = “random”.

        - Regularization: A string containing the type of regularization. The accepted values
        can be “l1”, “l2”, “batch_norm”, and “layer_norm” . The default value is “l2”.

        - Batch_size: An integer specifying the mini batch size. By default the value is 64.

        - Num_epochs: An integer with a number of epochs the model should be trained for.

        - dropout: An integer between 0 to 1 describing the percentage of input neurons to be
        randomly masked.

        - **kwargs: A dictionary of additional parameters required for different optimizers.
        """
```

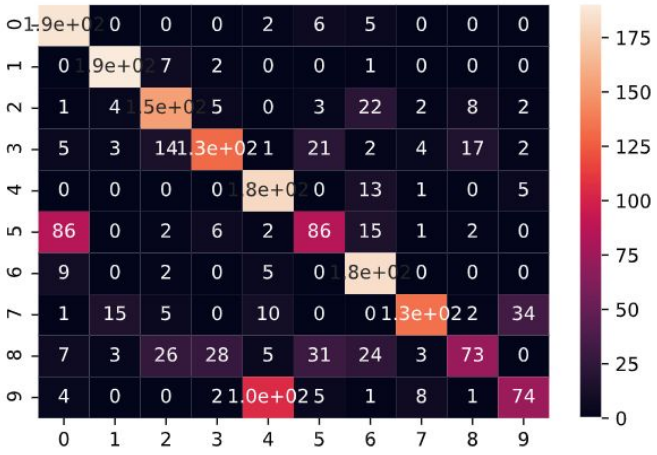
DL ASSIGNMENT-2 Part-1

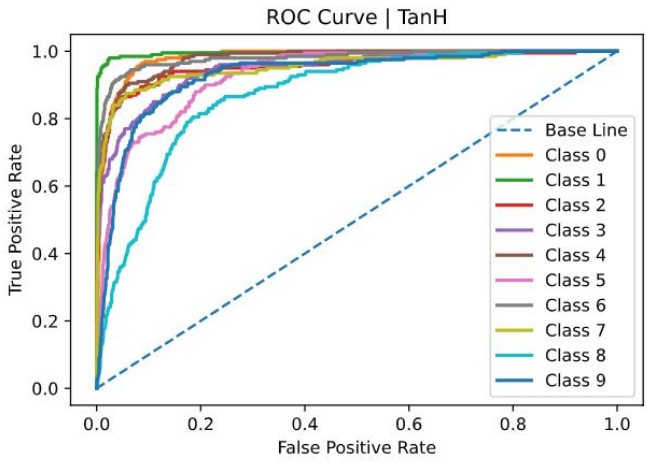
- Tanh with gradient descent

LR=0.1, EPOCHS=200



ACCURACY	
Training	73.96
Validation	69.60





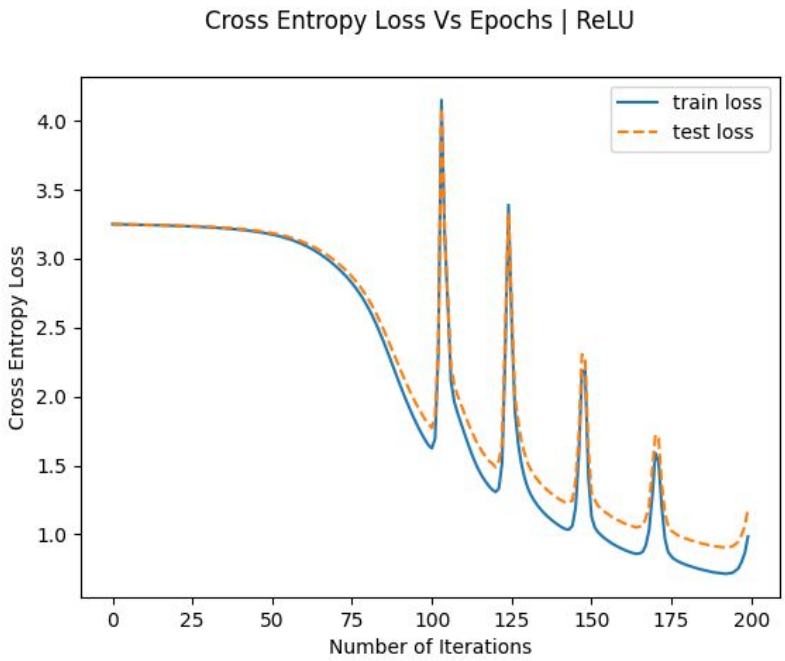
Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

INFERENCE: Tanh works good on the given dataset and achieves an accuracy of around 70%. There are spikes on the Loss vs Epoch curves, which may be due to the large dataset size.

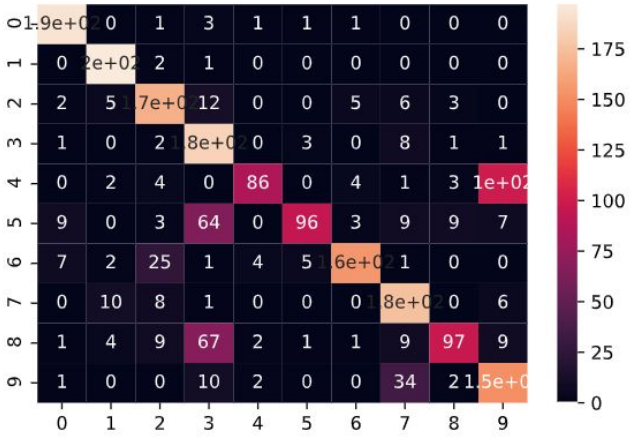
DL ASSIGNMENT-2 Part-1

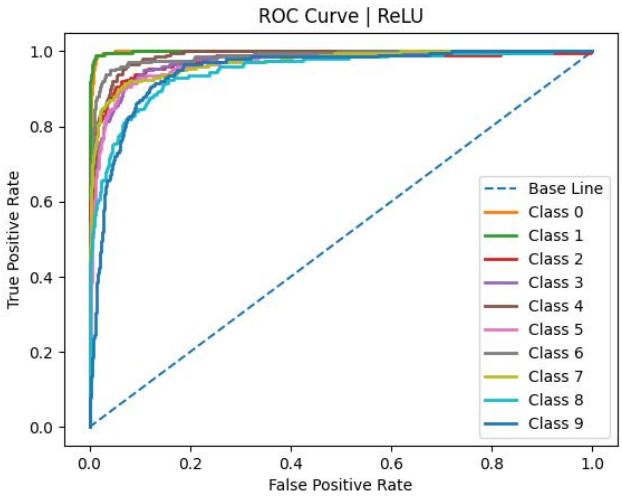
- ReLU with gradient descent

LR=0.01, Epochs=200



ACCURACY	
Training	88.68
Validation	84.40





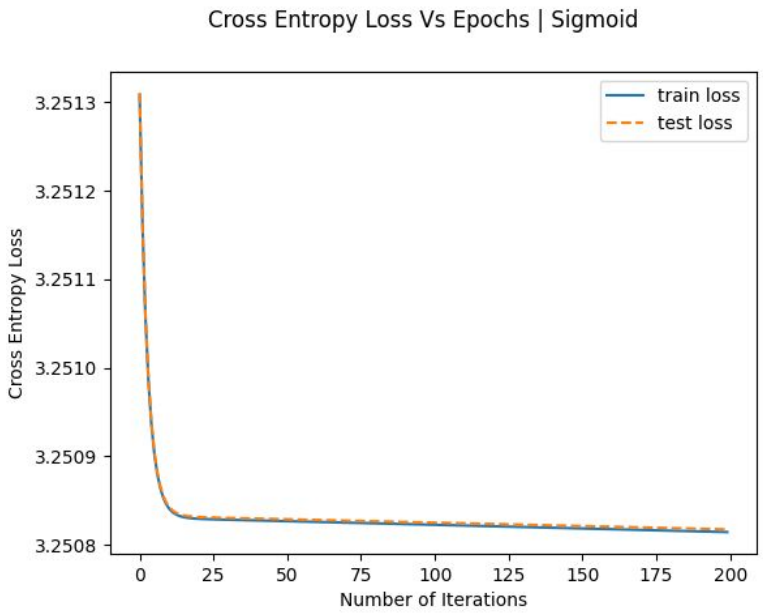
Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

INFERENCE: ReLU works the best on the given dataset and achieves an accuracy of around 80%.

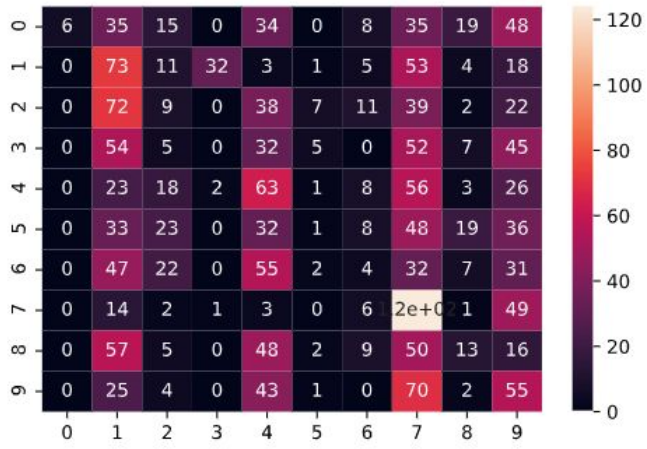
DL ASSIGNMENT-2 Part-1

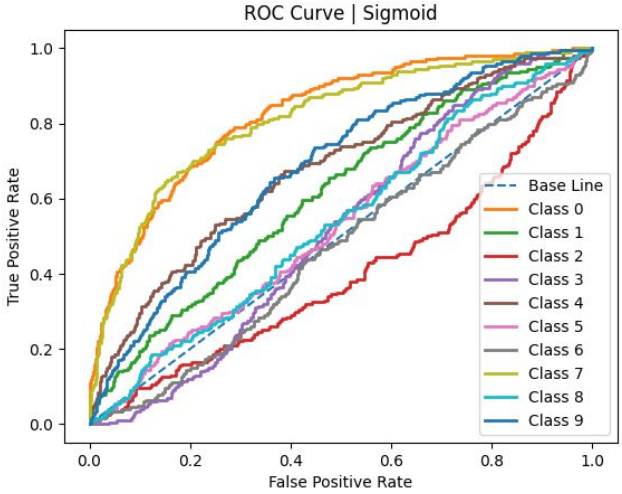
- Sigmoid with Gradient descent

LR=0.1, Epochs=200



ACCURACY	
Training	19.96
Validation	17.4





Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

INFERENCE: Sigmoid does not work at all on the given dataset with gradient descent. It only predicts one class and thus achieves only 10% accuracy.

DL ASSIGNMENT-2 Part-1

	Train Accuracy	Validation Accuracy
Tanh	73.96	69.60
ReLU	88.68	84.40
Sigmoid	19.96	17.40

The best Accuracy was achieved for ReLU with a learning rate=0.01.
Thus for now trying new optimisers, we use ReLU as the activation function.

---

2)Implement the following gradient descent optimizers from scratch and do a thorough analysis of the output of each one of them. Use mini-batch size = 64.

Therefore, we will implement the following:

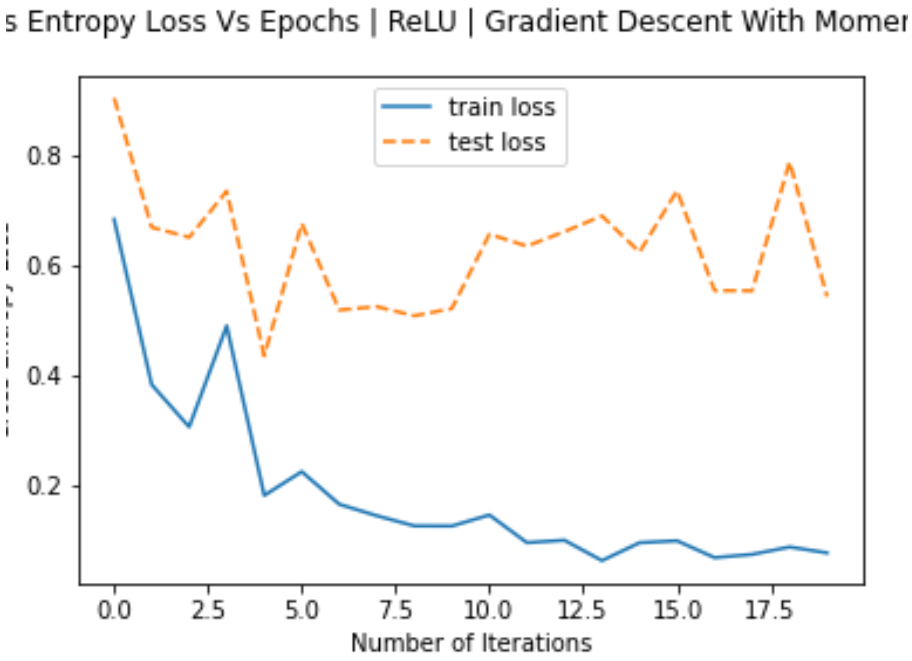
- ReLU with gradient descent with momentum
- ReLU with NAG
- ReLU with AdaGrad
- ReLU with RMSProp
- ReLU with Adam

With LR=0.01 and epochs=20.

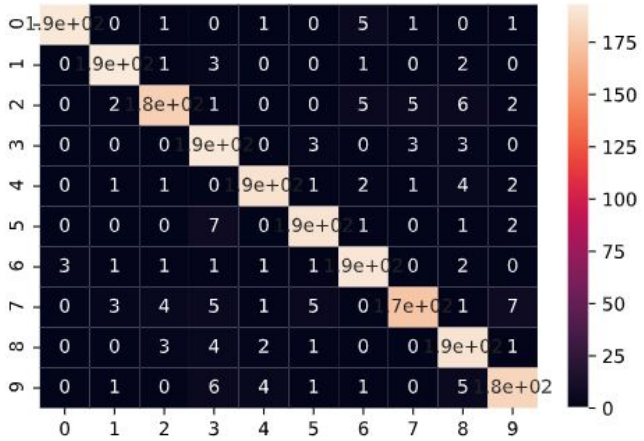
NOTE: The epochs have been reduced as optimisers help the model to converge faster to the minima, as **in just 20 epochs the model converged.**

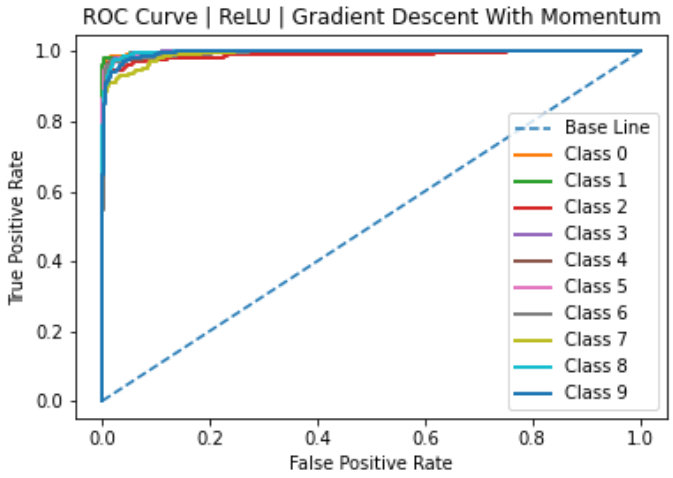
DL ASSIGNMENT-2 Part-1

- Activation=ReLU , Optimiser= gradient descent with momentum
  LR=0.01, Epochs=20



ACCURACY	
Training	98.95
Validation	93.30



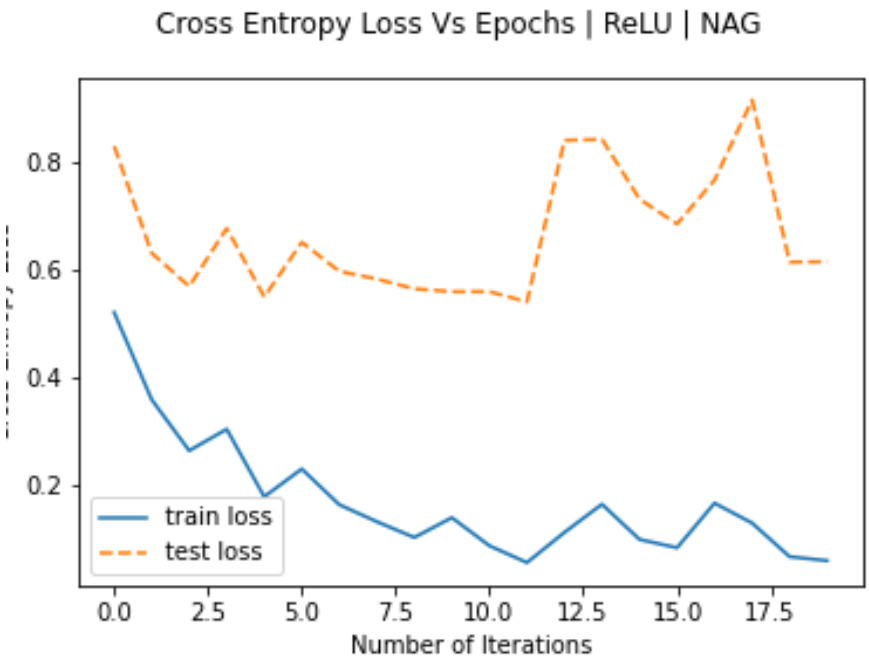


Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

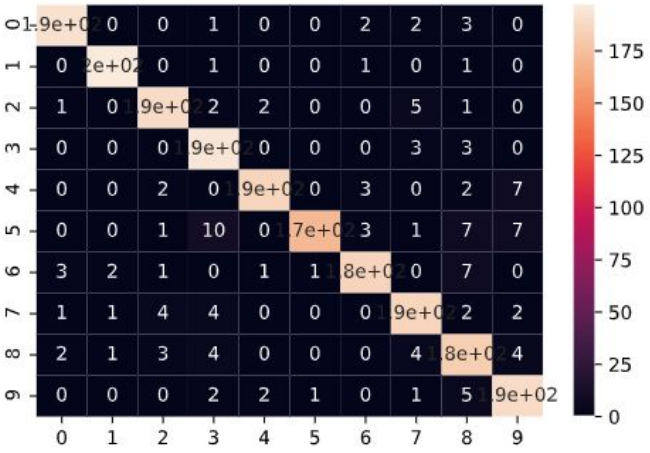
DL ASSIGNMENT-2 Part-1

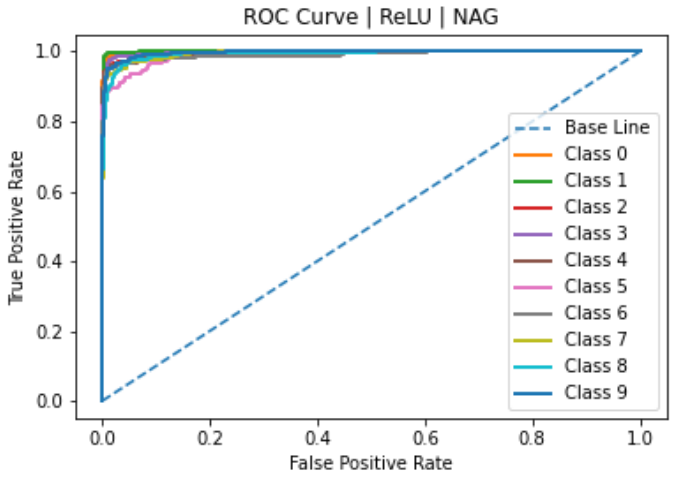
- Activation =ReLU and Optimiser=NAG

LR=0.01, Epochs=20



ACCURACY	
Training	99.11
Validation	93.55



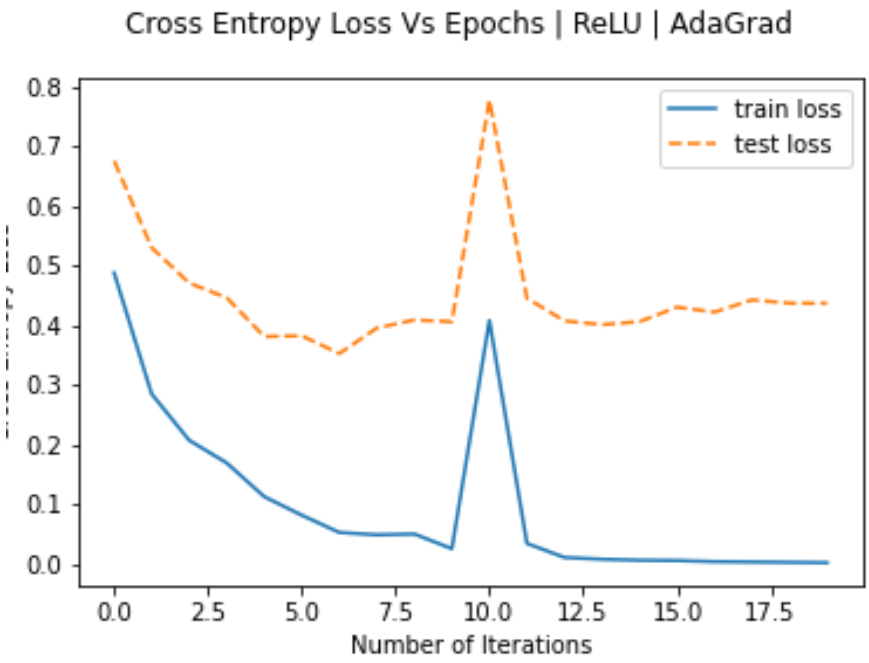


Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

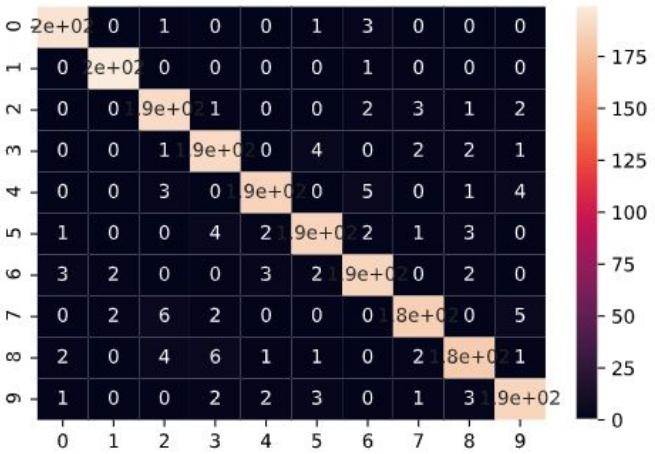
DL ASSIGNMENT-2 Part-1

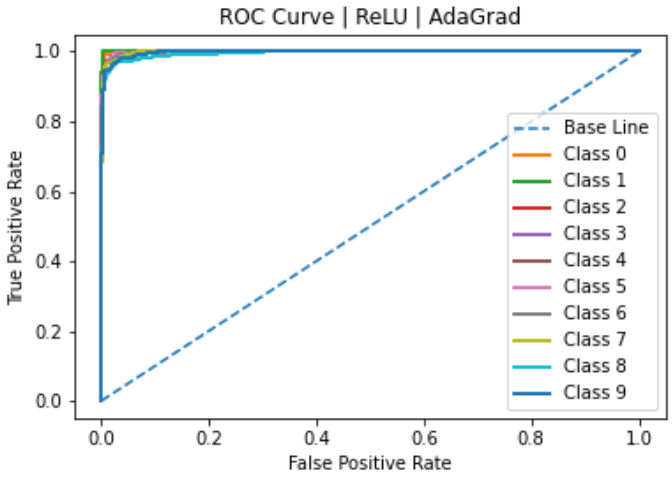
- Activation=ReLU and Optimiser=AdaGrad

LR=0.01, Epochs=20



ACCURACY	
Training	100
Validation	94.65

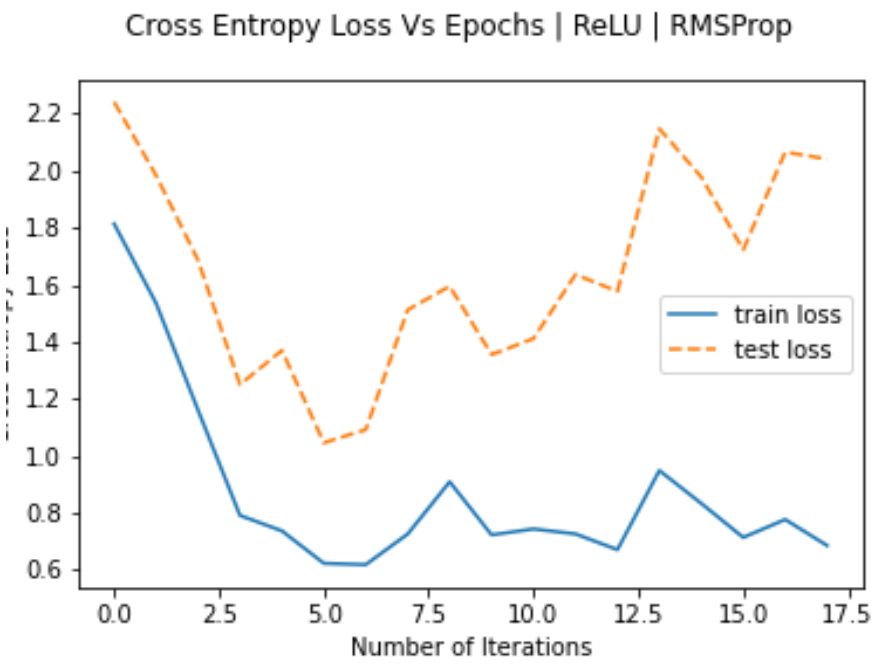




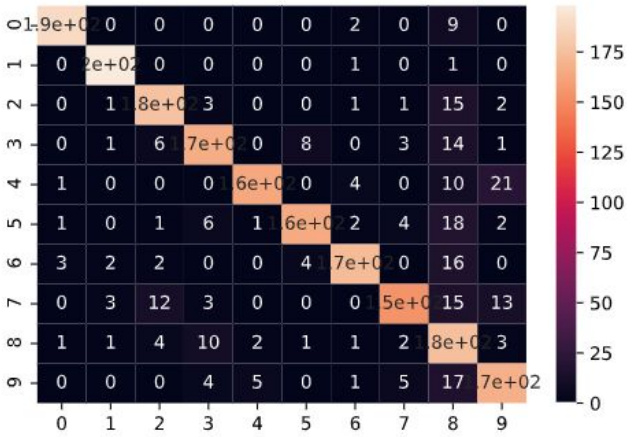
Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

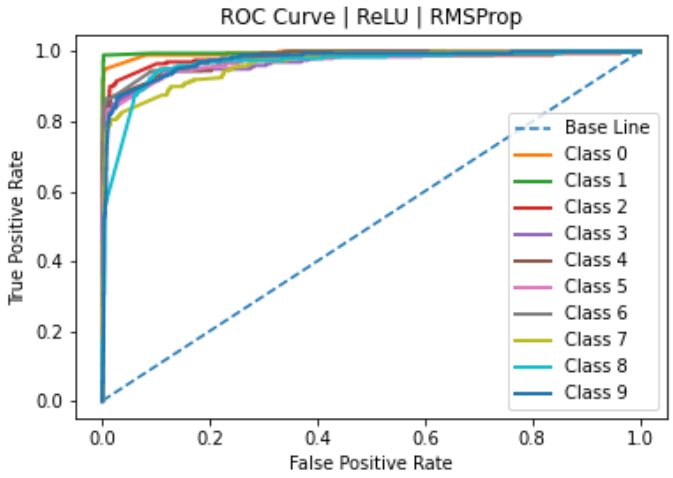
DL ASSIGNMENT-2 Part-1

- Activation= ReLU and Optimiser= RMSProp
  LR=0.01, Epochs=20



ACCURACY	
Training	92.33
Validation	86.5





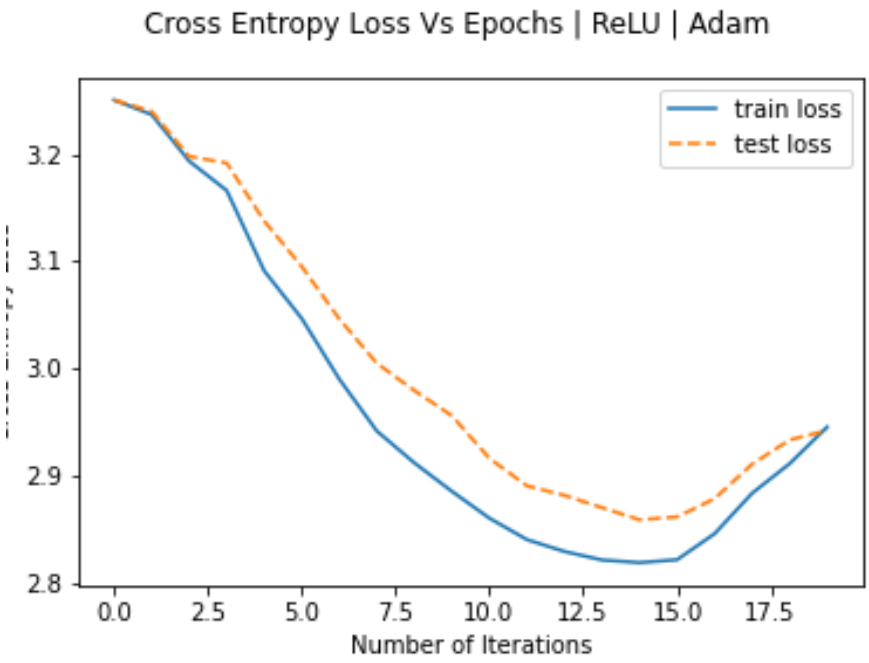
Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

INFERENCE: RMSprop also converges but the accuracy is less compared to other optimisers. This may be due to the high Learning rate etc.

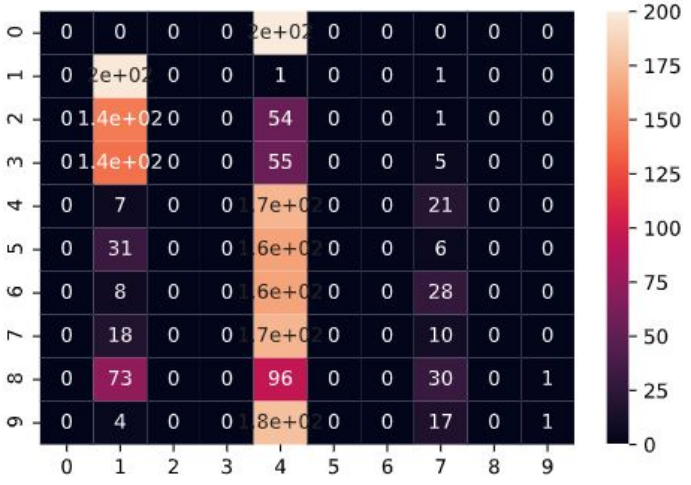
DL ASSIGNMENT-2 Part-1

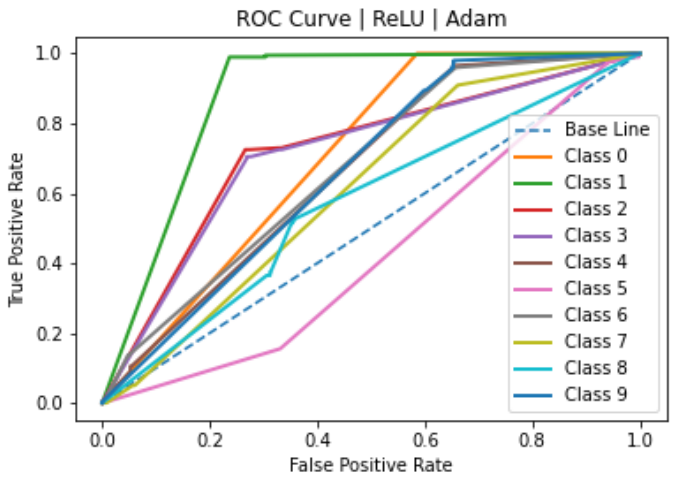
- Activation=ReLU and Optimiser=Adam

LR=0.01, Epochs=20



ACCURACY	
Training	19.05
Validation	19.05





Note: Horizontal axis represents predicted labels while the vertical axis represents the true labels.

INFERENCE: ReLU does not work along with Adam, this is due to high LR or dead neurons as seen in ReLU. It may work with a lower LR or a different activation function.

DL ASSIGNMENT-2 Part-1

INFERENCE:

- ReLU along with gradient descent with momentum optimiser performs better than just gradient descent. Moreover, it converges faster as the learning rate is kept high.
- NAG also performs very well, achieving a training accuracy of 99%.
- AdaGrad also performs well along and with 100% training accuracy.
- RMSprop converges but the accuracy is a bit low due to high LR or Dead Neurons etc.
- The same applies to Adam as it is derived from RMSProp itself. Thus it also does not achieve high accuracy.
- Results with RMSProp may increase if we use tanh as activation or decrease LR.

Activation+Optimiser	Train Accuracy	Validation Accuracy
Tanh	73.96	69.6
ReLU	88.68	84.40
Sigmoid	19.96	17.40
ReLU + momentum	98.95	93.30
ReLU + NAG	99.11	93.55
ReLU+ AdaGrad	100	94.65
ReLU + RMSProp	92.33	86.5
ReLU+ Adam	19.05	19.05

ASSUMPTIONS:

- For part-2 we keep hyperparameters the same except the optimizer