Part-3:

Datasets:

Two datasets were used.

First dataset used has images classified into 10 classes.

(Reference: https://www.dropbox.com/s/otc12z2w7f7xm8z/mnistTask3.zip)

Something unique about this dataset is that for a particular class, all images that don't have that digit belong to it.

Eg: All images belonging to class 0 folder are images that have digits 1,2,3,4,5,6,7,8,9. Similarly for other class folders.

For testing , Standard MNIST Test Set was used.

Random rotation was applied from -30 degrees to 30 degrees to the train dataset

Training:

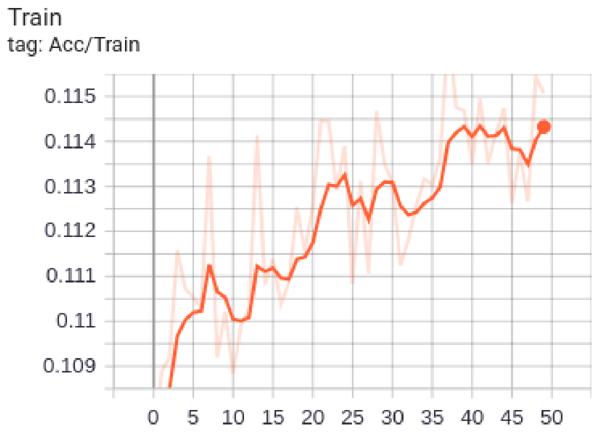
Training was done using two methods. In first, the pretrained model that was used for Part-2 was used and then it was trained on the dataset made for this part and tested on standard MNIST test dataset. Similar one was done for experiment-2 but the model here was trained from scratch

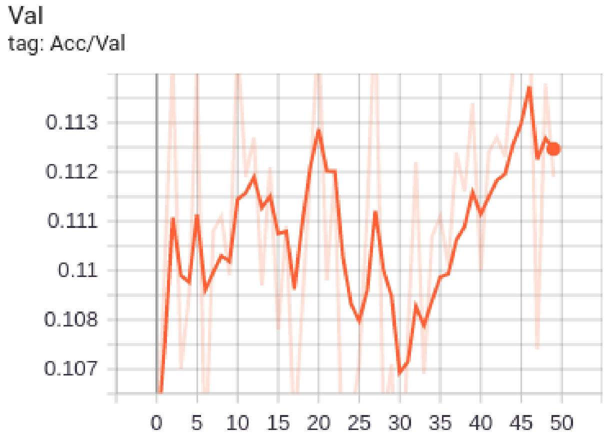
Adam Optimizer with learning rate of 3e-4 was used and training was done for 50 epochs

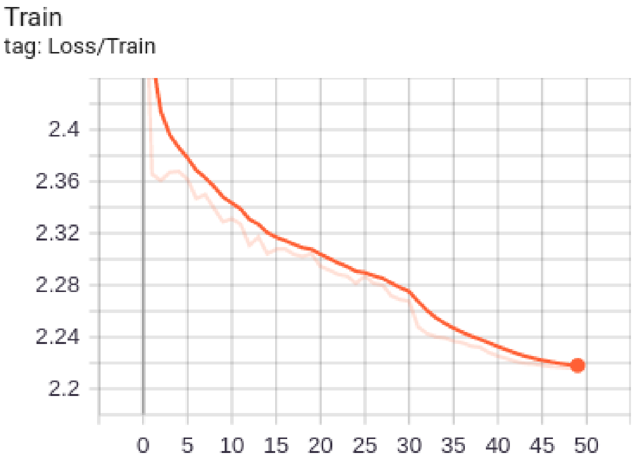
Results:

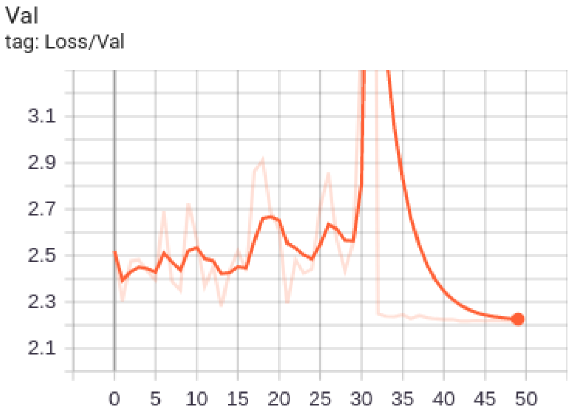
Using Pretrained Model:

Training accuracy was 11.4 and Validation Accuracy was 11.52 and Test Accuracy on MNIST Test set was 0.16 %

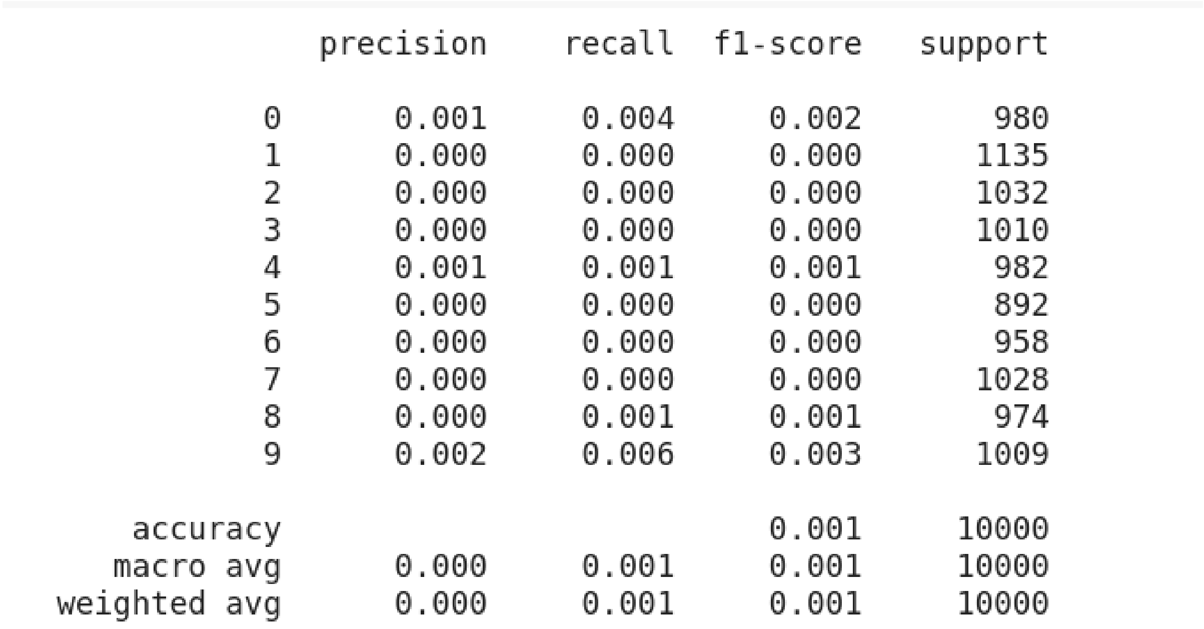








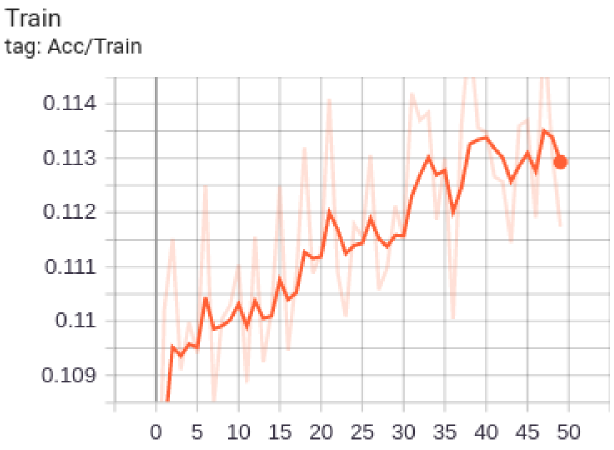
Precision, Recall and F1-Score obtained on Test Set

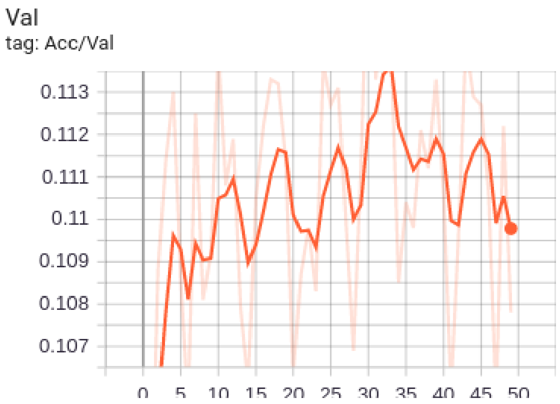


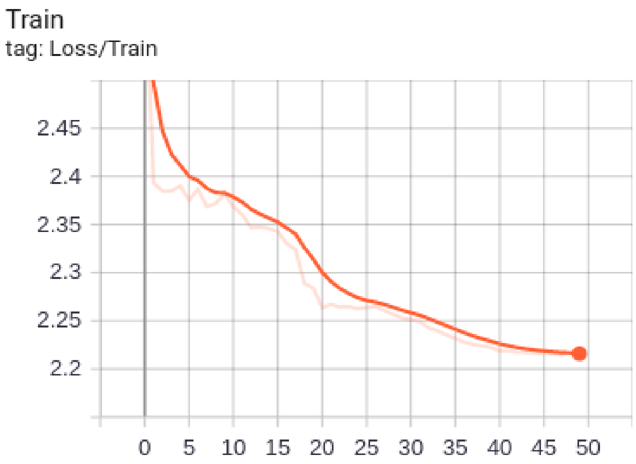
```
              precision    recall  f1-score   support

           0      0.001     0.004     0.002       980
           1      0.000     0.000     0.000      1135
           2      0.000     0.000     0.000      1032
           3      0.000     0.000     0.000      1010
           4      0.001     0.001     0.001       982
           5      0.000     0.000     0.000       892
           6      0.000     0.000     0.000       958
           7      0.000     0.000     0.000      1028
           8      0.000     0.001     0.001       974
           9      0.002     0.006     0.003      1009

    accuracy                          0.001     10000
   macro avg      0.000     0.001     0.001     10000
weighted avg      0.000     0.001     0.001     10000
```

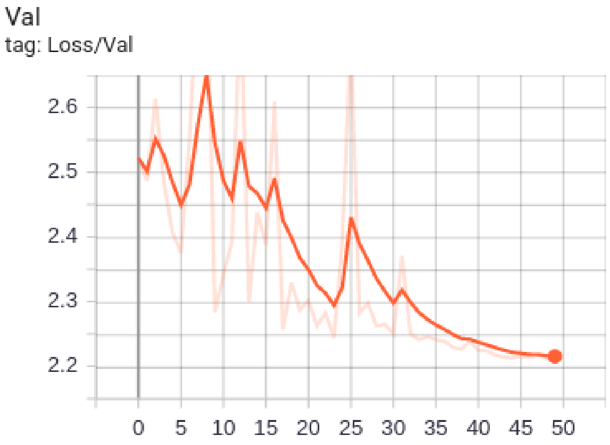
Using Model trained from Scratch:

Training accuracy was 11.2 % and Validation Accuracy was 11.72% and Test Accuracy on MNIST Test set was 0.12 %

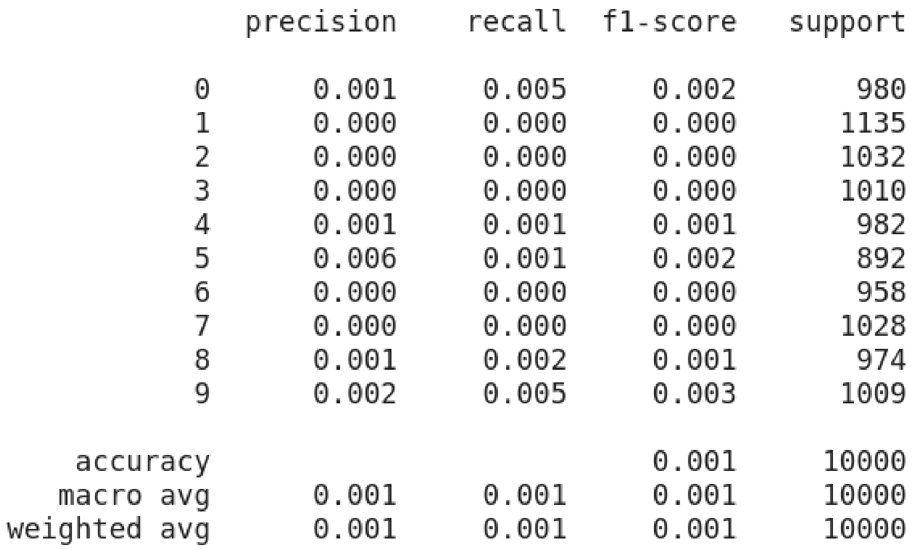








Precision, Recall and F1-Score obtained on Test Set



```
              precision    recall  f1-score   support

           0      0.001     0.005     0.002       980
           1      0.000     0.000     0.000      1135
           2      0.000     0.000     0.000      1032
           3      0.000     0.000     0.000      1010
           4      0.001     0.001     0.001       982
           5      0.006     0.001     0.002       892
           6      0.000     0.000     0.000       958
           7      0.000     0.000     0.000      1028
           8      0.001     0.002     0.001       974
           9      0.002     0.005     0.003      1009

    accuracy                          0.001     10000
   macro avg      0.001     0.001     0.001     10000
weighted avg      0.001     0.001     0.001     10000
```

Analysis on the accuracies of both the models:

The training and Validation accuracies didn't increase beyond 12% in most of the epochs for both experiments. The Test Set accuracy was also very less.

The training and validation set as discussed before contain classes that don't have the images belonging to the class in it. But the Test Set i.e MNIST Dataset is completely opposite to it. In MNIST the image containing a particular digit is assigned to that class. So that explains why the Test Set Accuracy is very low. It being very low means the model is learning properly because it is trained on the dataset which is completely opposite to that of the test set.

Coming to Training and Validation set, even multiple experiments didn't cause the accuracy to increase beyond the 11% mark on average. Also one thing I noticed while training is, 9 images are fed into the model and are labelled as belonging to the same class. So this is pretty much like random guessing. For Eg: According to the Training data, an image containing digit 0 can be assigned to either of 1,2,3,4,5,6,7,8,9. So I infer that the reason for such low accuracy is due to the random guessing of the model which may be due to the randomness of the train dataset.