Part-2:

Datasets:

Two datasets were used.

First, the dataset used for Part-1 was taken and images belong to classes 0-9 were filtered out and made a Custom MNIST Dataset , which is similar to standard MNIST Dataset

Second, Standard MNIST Dataset was used
(Reference: http://yann.lecun.com/exdb/mnist/)

Data Preprocessing:

Random rotation was enabled and Custom MNIST Dataset was resized from 900 X 1200 to 28 X 28 because the pretrained model then had to train on standard MNIST Data.

Approach:

This part consists of two experiments

Experiment-1:

The model was trained on the Custom MNIST Dataset that was prepared manually. The model weights were stored. Then this pretrained network was trained on the Standard MNIST Train Set and tested on Standard MNIST Test Set. Results and observations were noted

Experiment-2:

The model was trained from scratch using random weight initialisation on the Standard MNIST Train Set and tested on Standard MNIST Test Set. Results and observations were noted.

Training:

Experiment-1:

Out of the 400 images available (40*10) from Custom MNIST Dataset, 342 were used for training and 58 for validation.

Adam Optimizer with a learning rate of 3e-4 was used and the model was trained for 80 epochs.

This pretrained network was then used to train on Standard MNIST Train Set and tested on Standard MNIST Test Set. The train set was split and 50000 were used for training and 10000 for validation. Test Set consisted of 10000 images. Results and observations were noted. Adam Optimizer with a learning rate of 3e-4 was used and the model was trained for 30 epochs.

Experiment-2:

Since both the training (the one with scratch and other with the help of a pretrained network) have to be compared, the learning rate and number of epochs were the same for both Experiment-1 and Experiment-2.

The model was trained from scratch on Standard MNIST Train Set and tested on Standard MNIST Test Set. The train set was split and 50000 were used for training and 10000 for validation. Test Set consisted of 10000 images. Results and observations were noted.

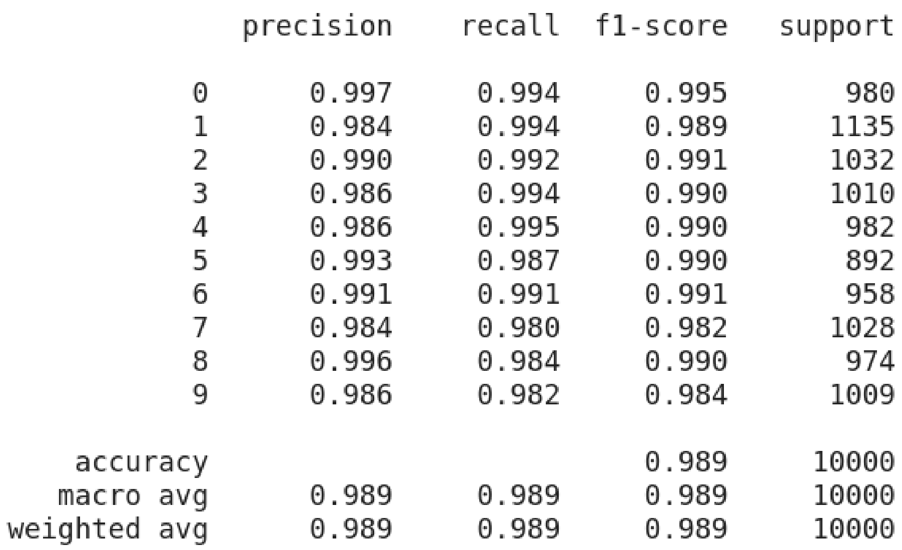
Results:

Experiment-1:

The best validation accuracy was 99% with train accuracy at 98.8%

Test accuracy was 98.9%

Precision, Recall and F1-Score obtained on Test Set



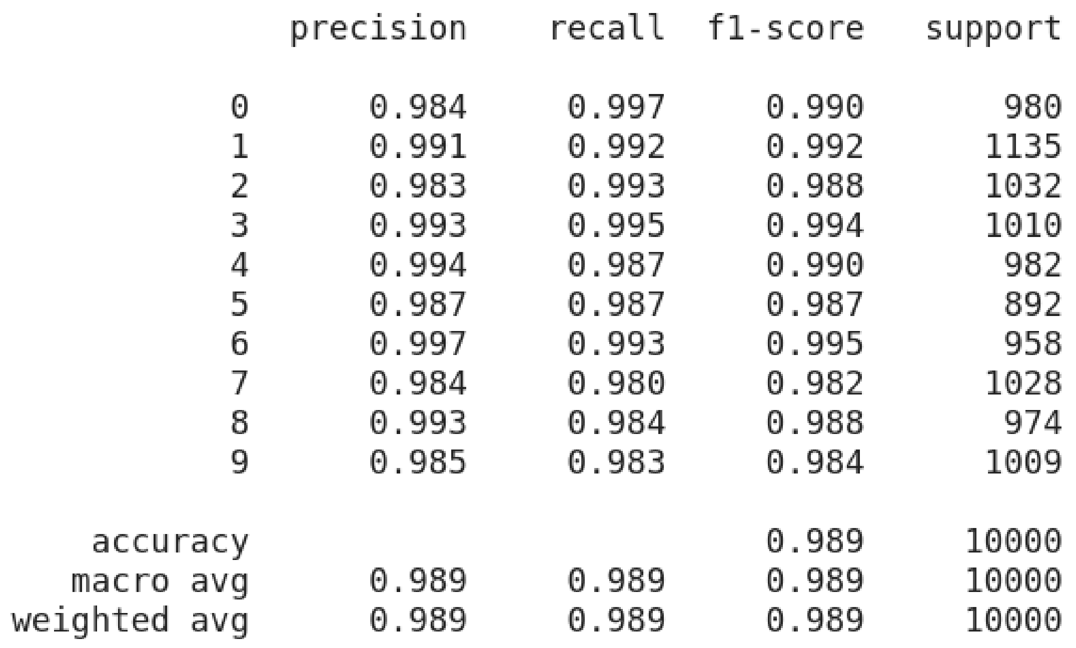
```
              precision    recall  f1-score   support

           0      0.997     0.994     0.995       980
           1      0.984     0.994     0.989      1135
           2      0.990     0.992     0.991      1032
           3      0.986     0.994     0.990      1010
           4      0.986     0.995     0.990       982
           5      0.993     0.987     0.990       892
           6      0.991     0.991     0.991       958
           7      0.984     0.980     0.982      1028
           8      0.996     0.984     0.990       974
           9      0.986     0.982     0.984      1009

    accuracy                          0.989     10000
   macro avg      0.989     0.989     0.989     10000
weighted avg      0.989     0.989     0.989     10000
```

Experiment-2:

The best validation accuracy was 98.87% with train accuracy at 99%

Test accuracy was 98.9%

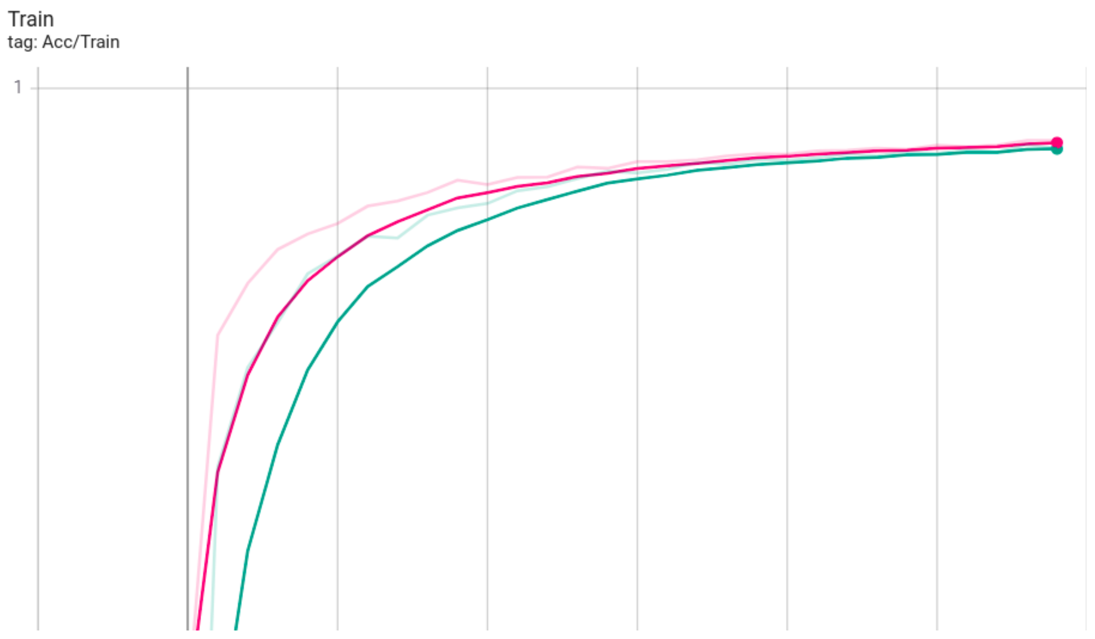
Precision, Recall and F1-Score obtained on Test Set

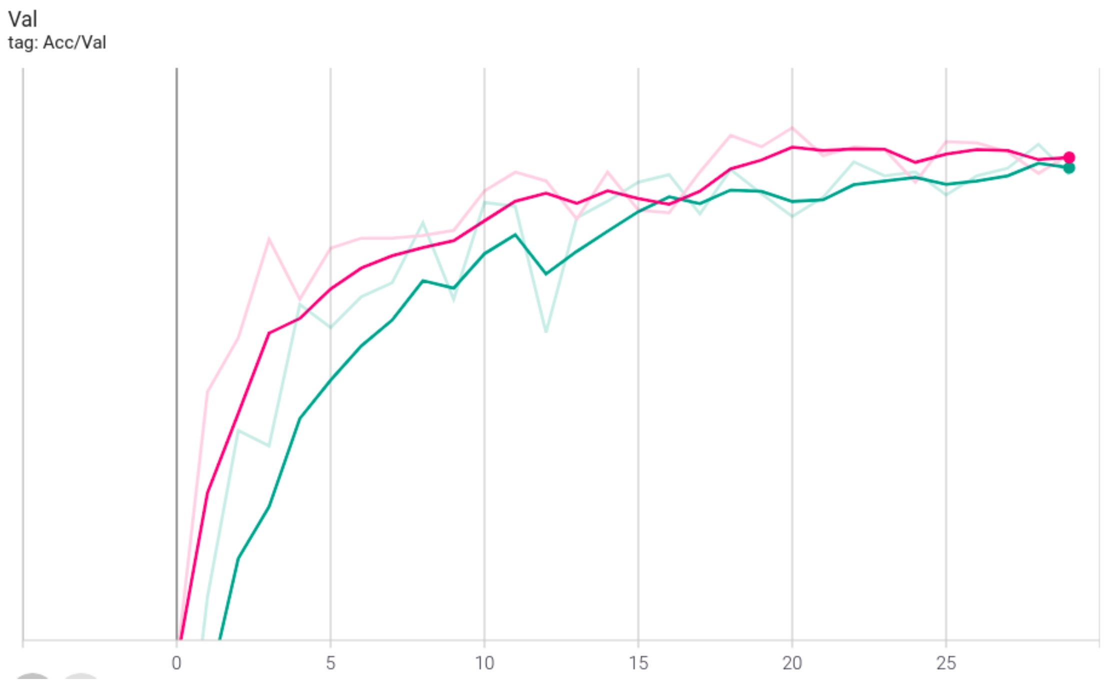


```
              precision    recall  f1-score   support

           0      0.984     0.997     0.990       980
           1      0.991     0.992     0.992      1135
           2      0.983     0.993     0.988      1032
           3      0.993     0.995     0.994      1010
           4      0.994     0.987     0.990       982
           5      0.987     0.987     0.987       892
           6      0.997     0.993     0.995       958
           7      0.984     0.980     0.982      1028
           8      0.993     0.984     0.988       974
           9      0.985     0.983     0.984      1009

    accuracy                          0.989     10000
   macro avg      0.989     0.989     0.989     10000
weighted avg      0.989     0.989     0.989     10000
```

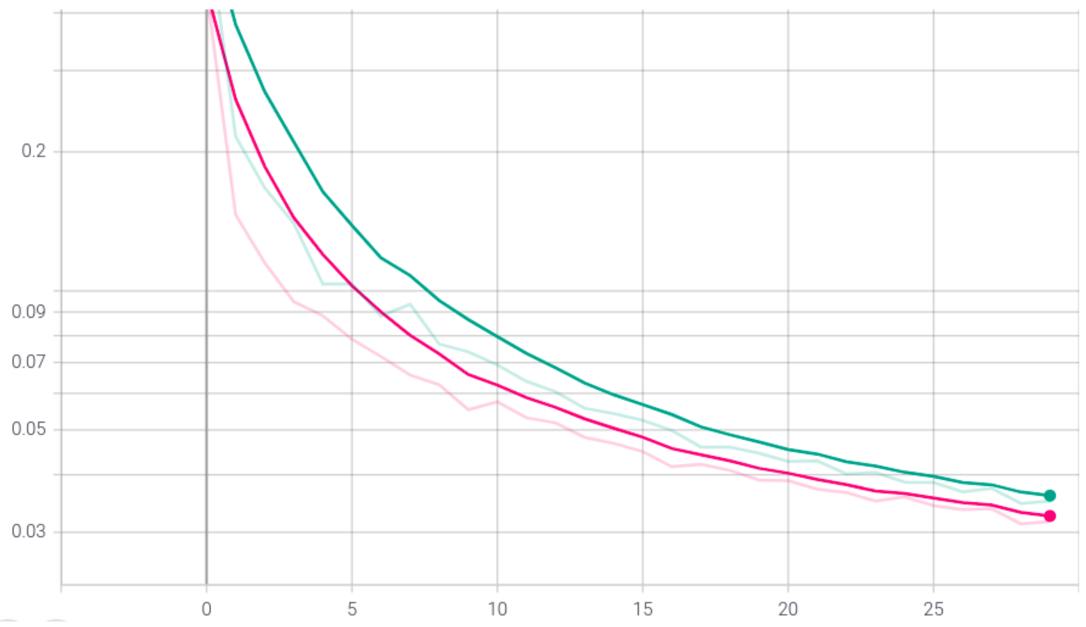
Comparison between Two Experiments:

Pink -- Model trained using Pretrained Network
Green -- Model trained from Scratch

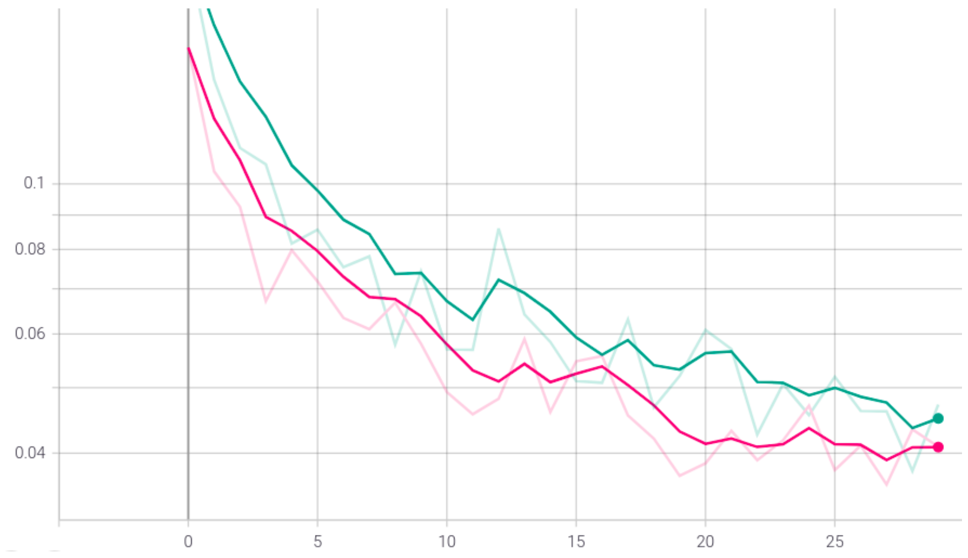




Train
tag: Loss/Train



Val
tag: Loss/Val



As the plots suggest, the pretrained model had better accuracies during the initial stages and convergence was quicker in it. After some time, the accuracies ended up being almost the same after training for a lot of epochs. Since the dataset was simple there was not much of difference but still pretrained model outperformed the model that was trained from scratch