MIDAS - IIITD (Internship Task, 2021)
Task - 2 (Computer Vision)

Intro To Task

Part 1:

- Using the provided database, build a CNN model to train on the dataset.
- Report accuracy of training and explain design choices.

Part 2:

- Using the dataset in Part 1 (Only 0-9 digits), retrain a new CNN model.
- Take this pretrained model, and retrain on MNIST Train Set. Furthermore, train a randomly initialised model of the same architecture, and train on MNIST Train Set.
- Compare performance of both on MNIST Test Set.

Part 3:

- Once again, using the pretrained model on 0-9 digits from part 1, retrain the same on the new provided dataset of 0-9 digits.
- Similarly, train a randomly init model on the new dataset.

Solution - Part 1

EDA and Dataset Properties:

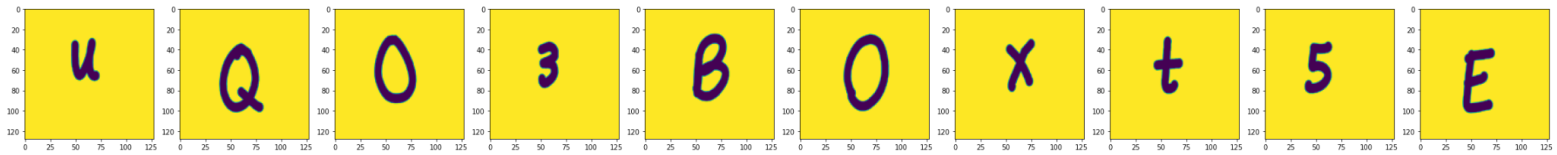
1. First, the data was loaded and exploratory analysis of the data was conducted.
2. Multiple observations were made which are described below:

a. Shape of Images - (900, 1200, 3) -> 3 Channels

b. 62 classes. **No Imbalance.** 2480 total images.

c. Since images were of high resolution, there was no apparent pixelation.

3. A view of a few samples is given below:



Preprocessing:

Based on the EDA, the following preprocessing measures were adopted for all images:

1. Using Pillow, each image was reduced to one dimension to reduce computational expense.
2. Each image was resized to (128, 128, 1).
3. Normalization was done on all images. For this, each pixel was divided by 255.

Data Augmentation:

To increase data size and improve performance, data augmentation was performed on the database. On running experiments with multiple data augmentation including **ZCA Whitening, Rotation, Flip, Shear, Fill Mode and Shifts,** the following were found to work well at lowest computational cost.

1. Pixel Standardization
2. Rotation - 30 Degres - Included to account for the tilt in few alphabets and numbers.

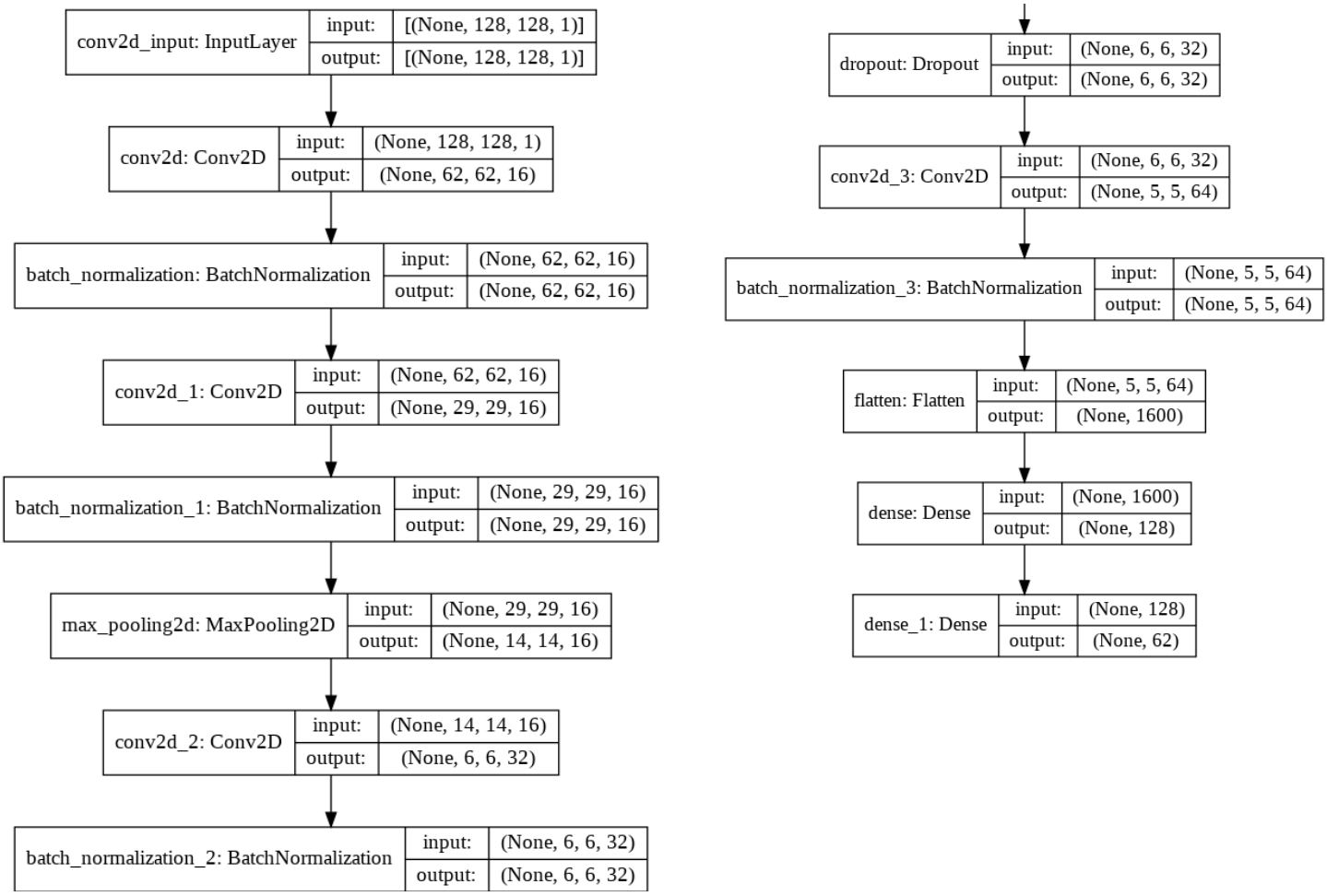
The idea was to find a model which gave enough accuracy in a cost efficient manner.

CNN Model Details:

Accounting for the high quality of images, no data imbalance and addition of augmented, a relatively heavy model was chosen. This was also due to the 64 categories.

1. Stride is set as 2 for the CNN models, to **downsample the image from (128,128,1) to lower dimensions** as it goes down the model.
2. Batch Normalization and Dropouts are added to prevent overfitting.
3. **Filter size is varied from (5,5) to (2,2) as the image is downsampled.**

The model structure is shown below. It is split into two for ease of viewing.



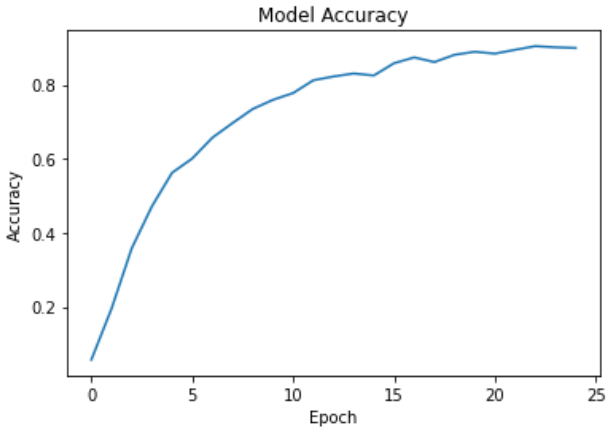
Training Details

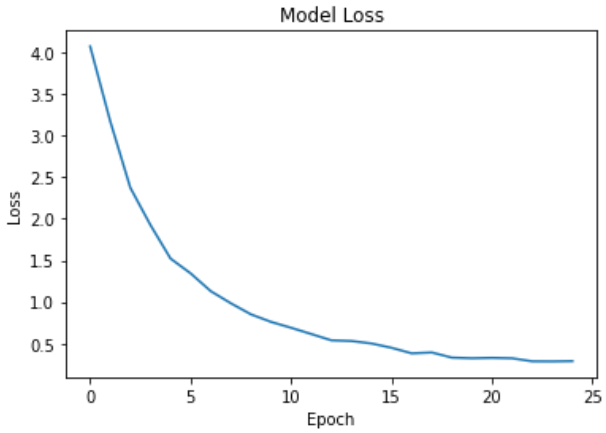
1. Batch Size - 20
2. For better results, **Model Checkpoints** were used alongwith **Early Stopping and ReduceLROnPlateau.** This results in only saving the best model, and also causes the training to stop if model accuracy does not improve. Furthermore, the learning rate is reduced depending on the loss.
3. Adam optimizer was used with categorical crossentropy as the loss function.
4. Model was trained for 25 epochs.

Results

Following are the results of the training:

Parameter	Value
Accuracy	90.457%
Loss	0.2599
Convergence Time	215 seconds





Solution - Part 2

EDA and Dataset Properties:

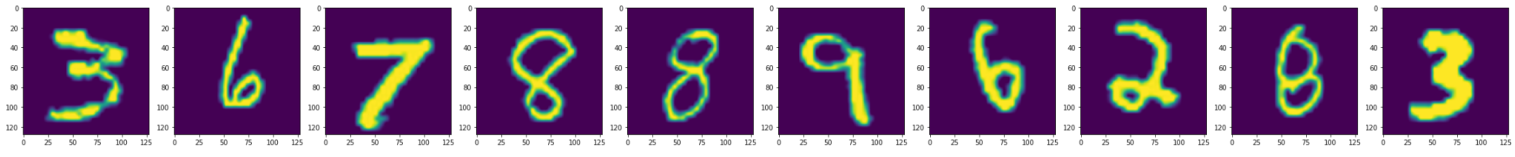
1. Here, two datasets were loaded - The data from Part 1 (But only 0-9 Samples) and MNIST train and test splits.
2. While the conclusions drawn are the same for part 1 data, features of the MNIST data are listed below.

a. Shape of Images - (28, 28, 1) -> 1 Channels

b. 10 classes. **Insignificant Imbalance.** 60000 total training images.10000 testing images.

c. Visible pixelation in contrast to dataset 1.

3. A view of a few samples is given below:



Preprocessing:

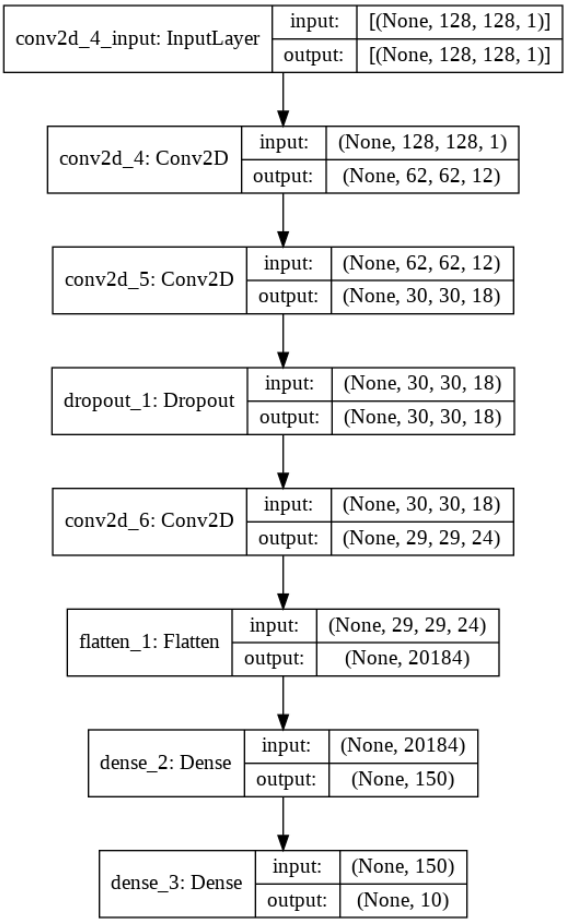
For the Part 1 data, preprocessing measures were the same.
For MNIST, the following measures were adopted.

1. Each image was resized to (128, 128, 1).
2. Normalization was done on all images. For this, each pixel was divided by 255.

Data Augmentation methods were the same as the one described in Solution 1.

CNN Model Details:

Here, a relatively **lighter model** was used in comparision to the first part due to -> **Less classes to predict (10) and more training images(60000).** Other minor features are the same as before. Note that the same model structurer is used for both the pretrained and random training.
The model structure is shown below.



Training Details

A) Pretrained Model

1. First the model was trained on 400 images of classes 0-9 of Part 1 data. This was done at the same settings as described in part 1, except that 30 epochs were used and the model was as shown above.
2. The retraining was done for 5 epochs, since the model converged within those epochs.
3. For retraining, a validation split was also created and that was used as parameters for early callback and learning rate reduction.
4. Other settings were the same as before.

B) Random Init Model

1. The model structure was same and other details were same as above.
2. In this case weights were not pretrained and instead **random_normal distribution was used for the weights.**

Results and Comparision (MNIST Test Set)

Following are the comapraritive results of the training:

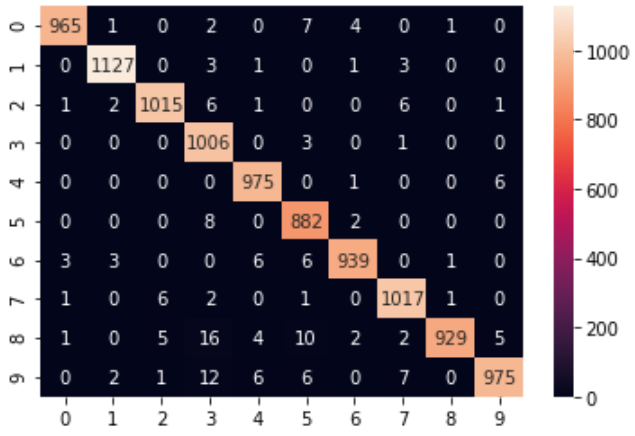
Model	Accuracy	F1 Score (Weighted Avg)	Recall (Weighted Avg)	Loss	Convergence Time
Pretrained	98%	98%	98%	0.318	12 min 13 sec
Random Init	98%	98%	98%	**0.299**	**11 min 47 sec**

We see the following conclusions:

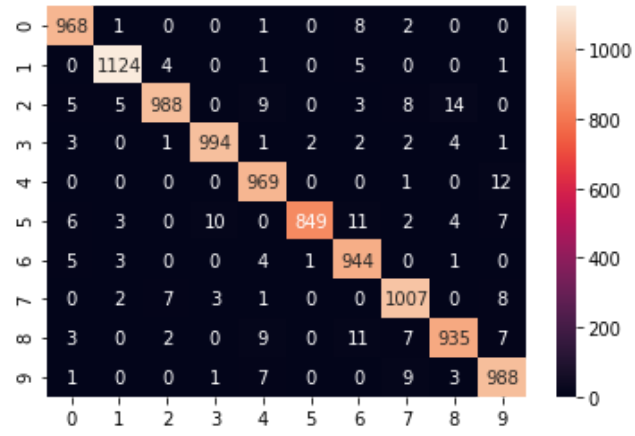
- The models perform equally well on the test set.
- Both models have excellent weighted recalls showing that there is no visible bias towards a class.
- The randomly initialised model converges faster. This could be due to the qualitative difference in quality of the part 1 data and the MNIST images. Since the pretrained model is trained on the part 1 dataset, it took a longer time to converge.

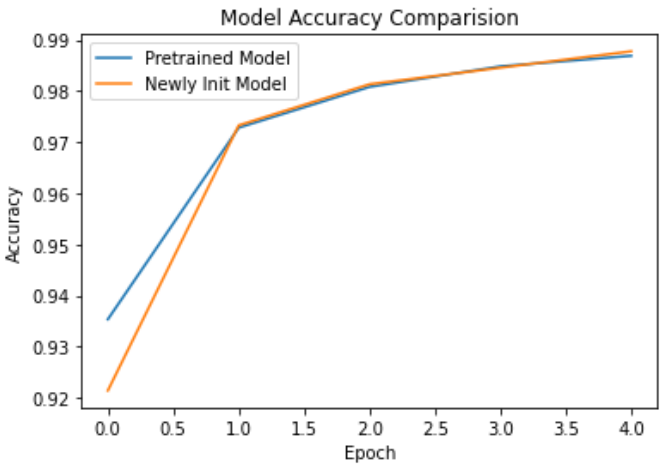
To further analyse the classwise accuracies, we plot the heatmap and history of the models.

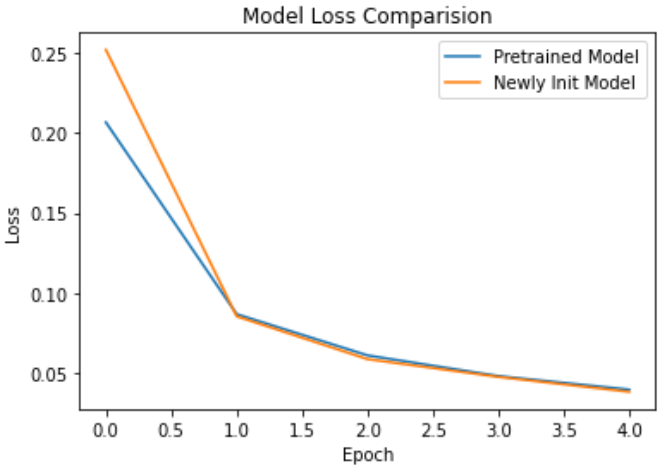
Pretrained Model



Random Init Model







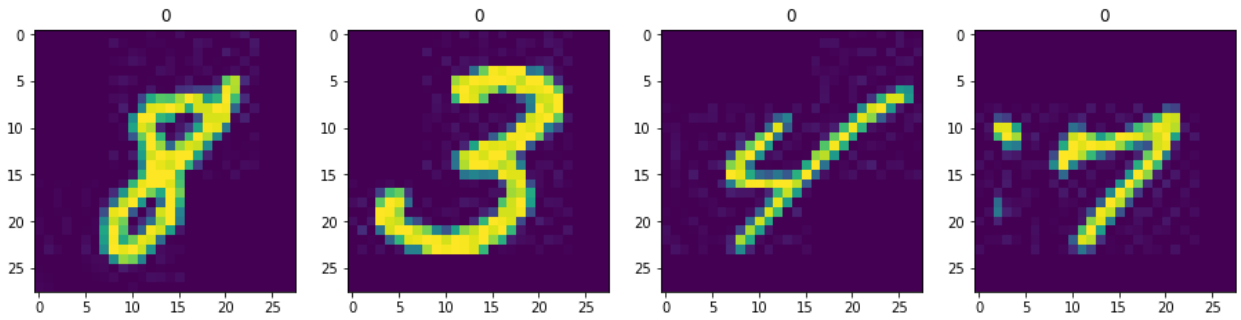
Other conclusions we can draw are:

- While both models have almost no bias, the randomly init model has a more even distribution of class representation.
- The random model starts at a higher loss and lower accuracy but manages to converge before the pretrained model.

Solution - Part 3

EDA and Dataset Properties:

1. Here all three datasets were loaded. This was done to compare all three.
2. The new data for this part was different due to a fallacy -> **The dataset was incorrectly labelled.** Each folder contained random distribution of digits. This is shown below wherein **all labels are 0 for random images.**



3. Hence, a good accuracy was not expected but still training and testing was done.
4. A qualitative comparision of the three datasets is given below

Dataset	Sample Size	Shape	Imbalance
Part 1 Dataset	2480	900 * 1200 * 3	None
MNIST Dataset	60000 Train, 10000 Test	28 * 28 * 1	Minimal
Part 3 Dataset	60000	28 * 28 * 1	**INCORRECTLY LABELLED**

All other settings, data augmentations and models were the same as part 2. The only difference was that the model was pretrained only on the new dataset. The random model was the same.

Results and Comparision (MNIST Test Set)

Since the data was incorrectly labelled, the results were expected to be extremely bad as shown below.

Model	Accuracy	F1 Score (Weighted Avg)	Recall (Weighted Avg)	Loss	Convergence Time
Pretrained	0.02%	0%	0%	4.56	12 min 13 sec
Random Init	**9%**	**2%**	**1%**	**2.23**	16 min 45 sec

REFERENCES:

1. ZCA Whitening, Martin Thoma
2. Optimal Window Size for CNNs, Swarnima Pandey
3. Visualisation of Filter Initializers, Pawan SJ
4. A Comparison of Weight Initializers in Deep Learning-based Side-channel Analysis, Li et al., IACR