7. Pytorch

7.1) Train a neural network in PyTorch

7.1.1)

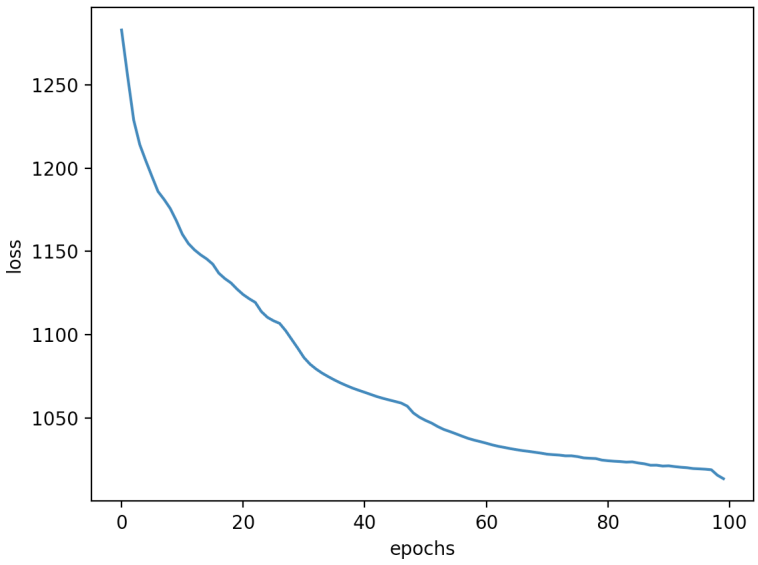

Figure 34: NIST36 Training loss over time (for 100 epochs)

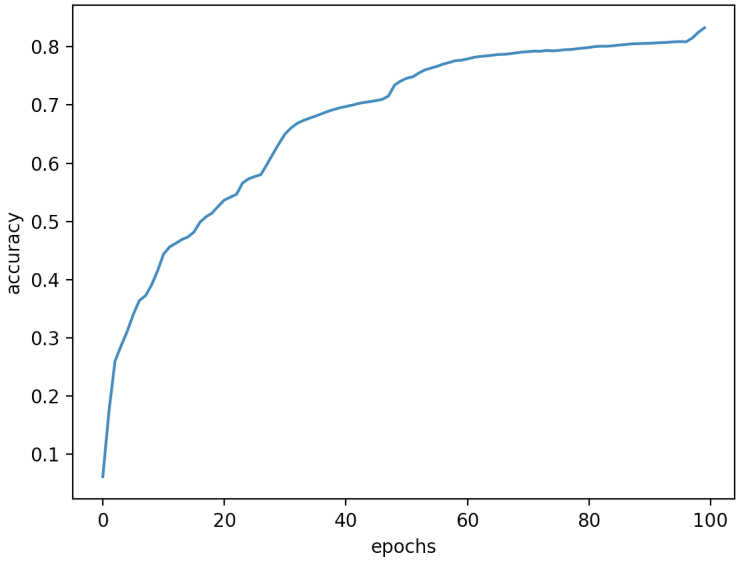

Figure 35: NIST36 Training accuracy over time (for 100 epochs)

7.1.2)

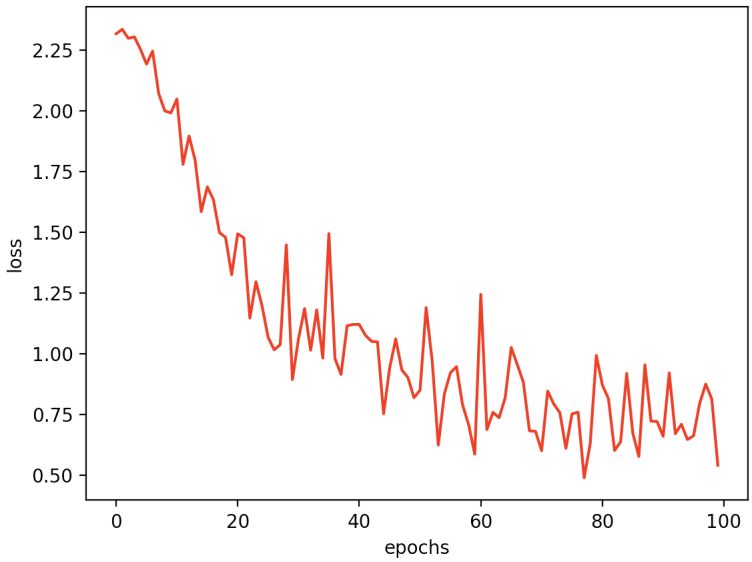

Figure 36: MNIST Training loss over time (for 100 epochs)

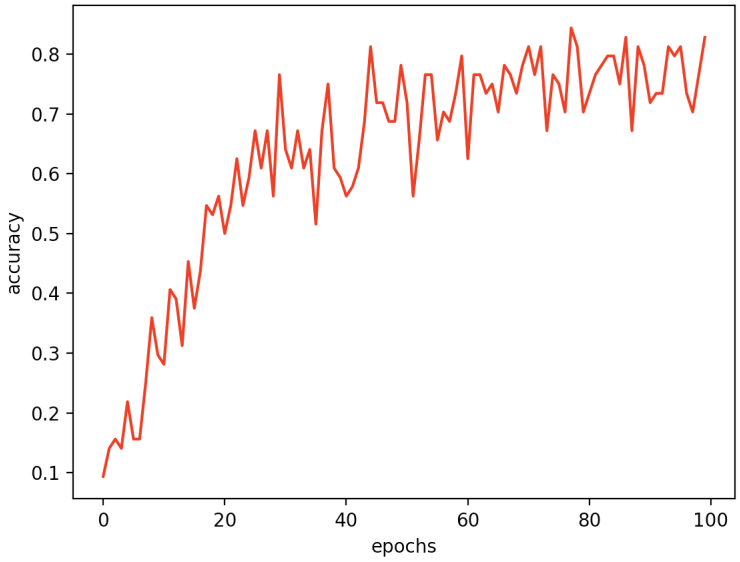

Figure 37: MNIST Training accuracy over time (for 100 epochs)

7.1.3)

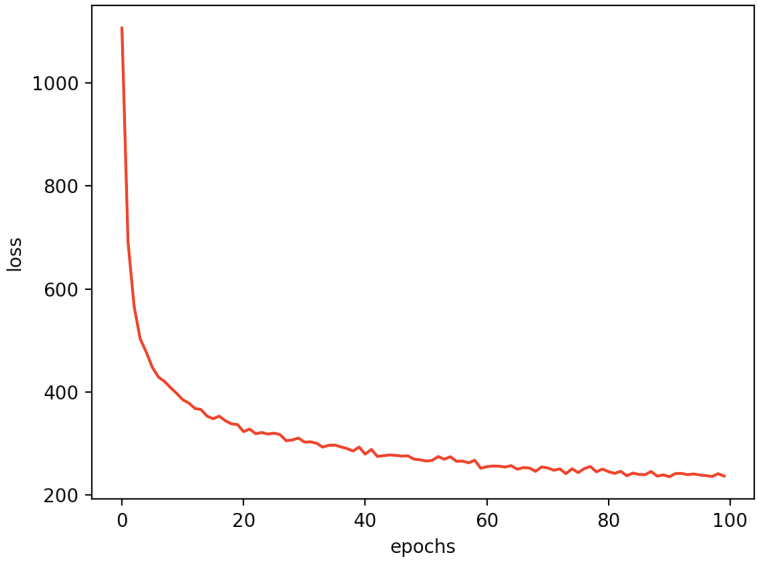

Figure 38: NIST36 Training loss over time (for 100 epochs)

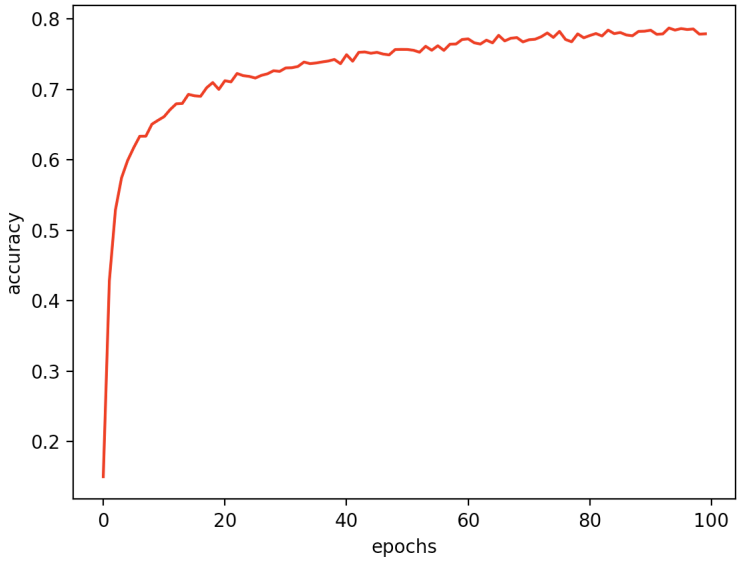

Figure 39: NIST36 Training accuracy over time (for 100 epochs)

7.1.4) The model was trained only for one epoch and thus it doesn't produce good results. The following are the predictions obtained

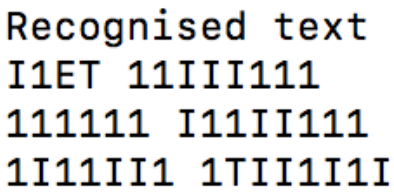

Figure 40

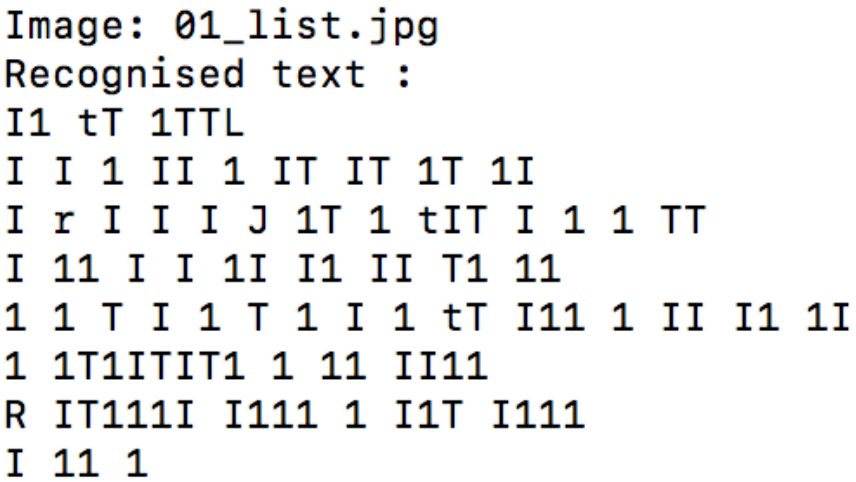

Figure 41

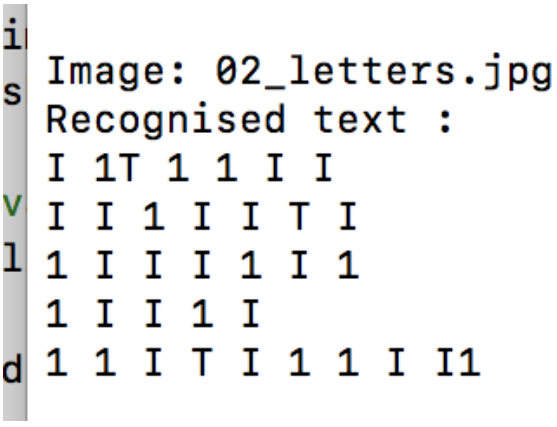

Figure 42

```
Image: 03_haiku.jpg
Recognised text :
111TII ITI I1T1
III 1I11II1T1 1I1I 111I 1TTT JI1I1
IIII111IIII1
```

Figure 43

```
Image: image1.jpg
Recognised text :
7t
q1r1ItIII
ItT1ITI111 T
1II
```

Figure 44

```
Image: image2.jpg
Recognised text :
1II1
T1YIg1IIF1JI131I
I1
```

Figure 45

7.1.5) Residual Networks and Skip Connections

Skip Connections

- Skip Connections refer to the type of connections which involve skipping of layers (i.e) The output of one layer can be fed as an input to one or many layers (Eg DenseNet)
- Skip connection involves stacking/concatenation of the previous layers' output or can also involve addition (like in case of a Residual Net- which is a special case of skip connection but not vice versa).
- Skip Connections can be accomplished in many ways and few good state of the art architectures like DenseNet, U-Net, SegNet help us in easy understanding
- Let's consider U-Net, which contains set of downsampling and upsampling layers. The downsampling layers are concatenated and fed to the upsampling layers' output as from the architecture shown below

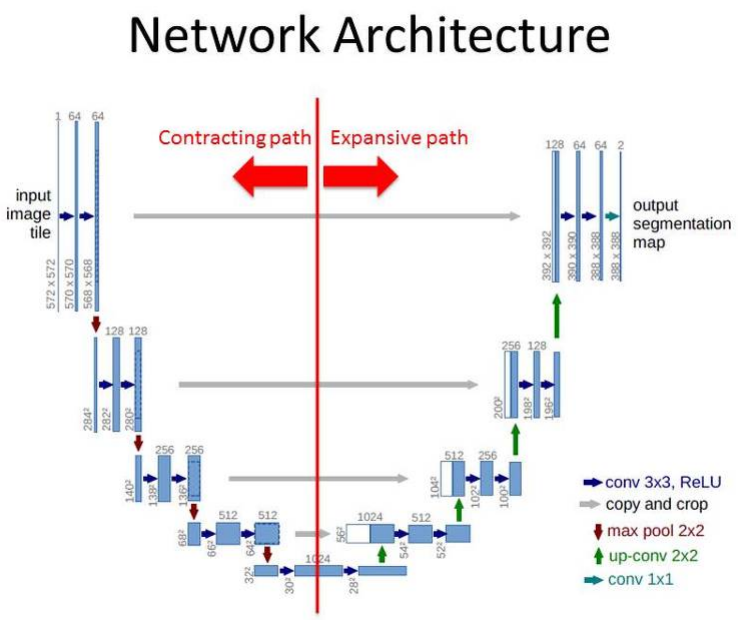

Figure 46: UNet Architecture)

- This type of skip connection/concatenation helps in ensuring that the dimension of the input and output image are same (which is important in semantic segmentation tasks)
- Additionally Skip connections help in overcoming the vanishing gradient problem during backpropagation, specially while dealing with deep neural networks. Skip connections ensure that the initial layers information doesn't get lost as they provide valuable information in determining the features of the input

- The Following skip connection block architecture was used in programming task. It's a very basic skip that I implemented, although in a practical setting which involves many convolution layers, this particular architecture can help in keeping up the initial layer features

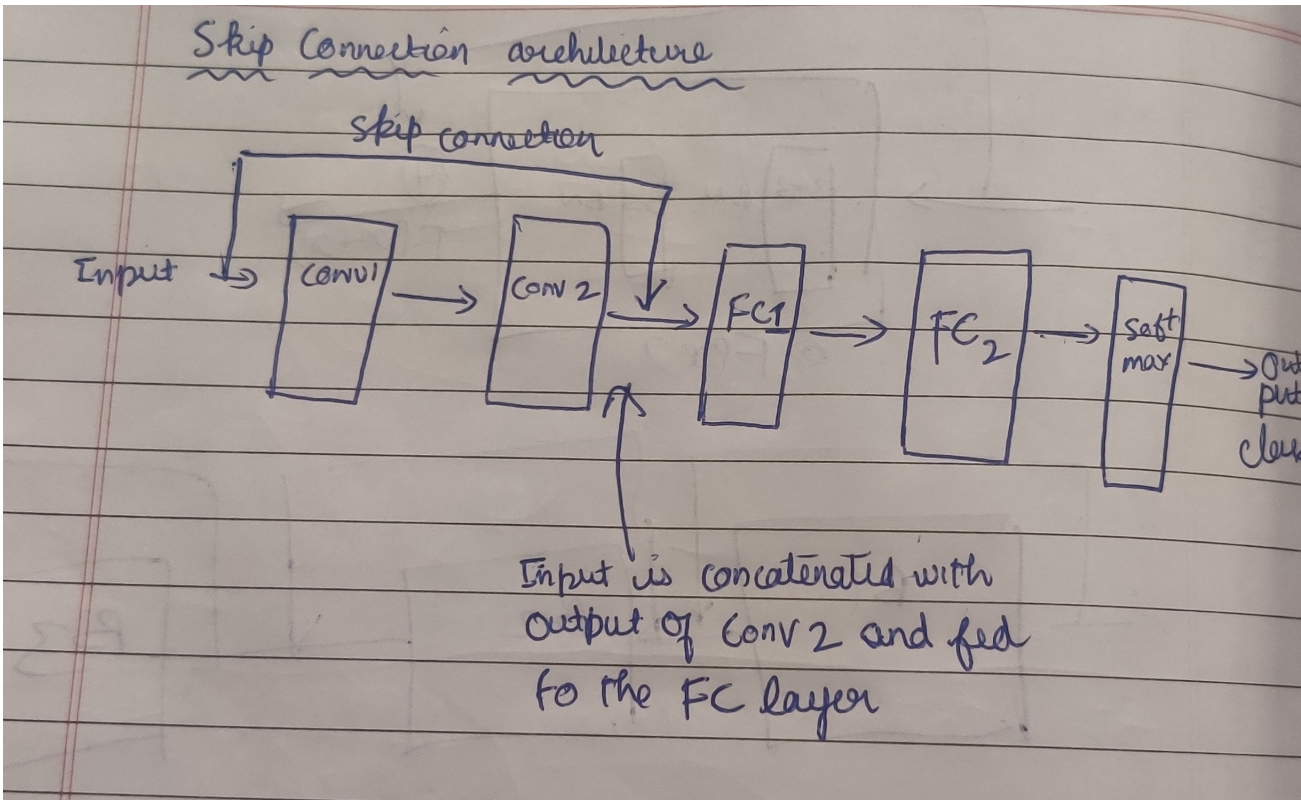

Figure 47: Skip Connection Architecture implemented

- The file **7.1.5_skip.py** contains the implementation of the above mentioned architecture

Residual Connections

- Residual Connections are a type of skip connections where the input is summed along with the output of the Functional block. The following figure is a residual block

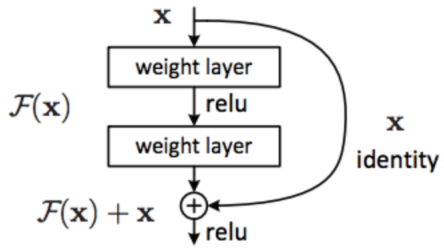

Figure 48: Residual Block)

- Residual blocks connected together gives Residual Network
- ResNets help in avoiding vanishing gradient problem/ Since the input (x) is summed together with the functional block F(x), during backpropagation, even if the gradients from the functional block fall to zero, the input x ensures that the output of the block remains non-zero.
- Thereby the initial layers information can also be prevented from vanishing, specifically in case of deep neural networks
- The Following residual block and network architecture was used in programming task.

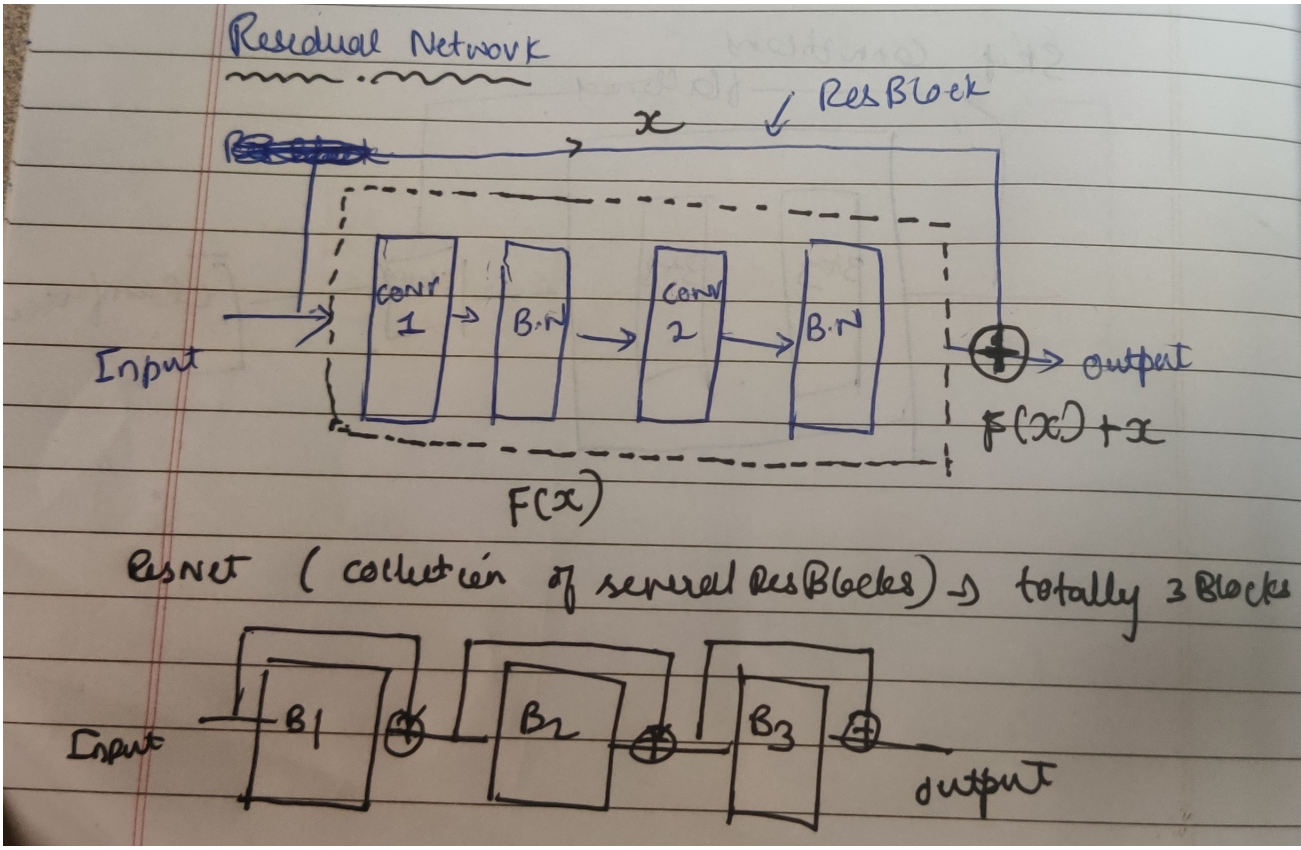

Figure 49: ResNet architecture implemented

- It's a basic ResNet with three layers that has been implemented, as we stack more layers and train it for a longer duration with more images (Like ImageNet), ResNet has provedt to be one of the most efficient srchitectures till date in terms of eradicating vanishing gradients
- The file **7.1.5_resnet.py** contains the implementation of the above mentioned architecture

Creative Network

- While doing literature work about other state of the art skip connection models, and also getting to remember LSTM's and Recurrent Nets which deal with skip connection methodology, I got interested about **HIghway Networks**.
- Highway networks can be in a nutshell understood as LSTM's with Images, as it contains a transform and carry gate (T) and (C)
- The Following architecture was implemented for a Highway Connection, as per the paper.

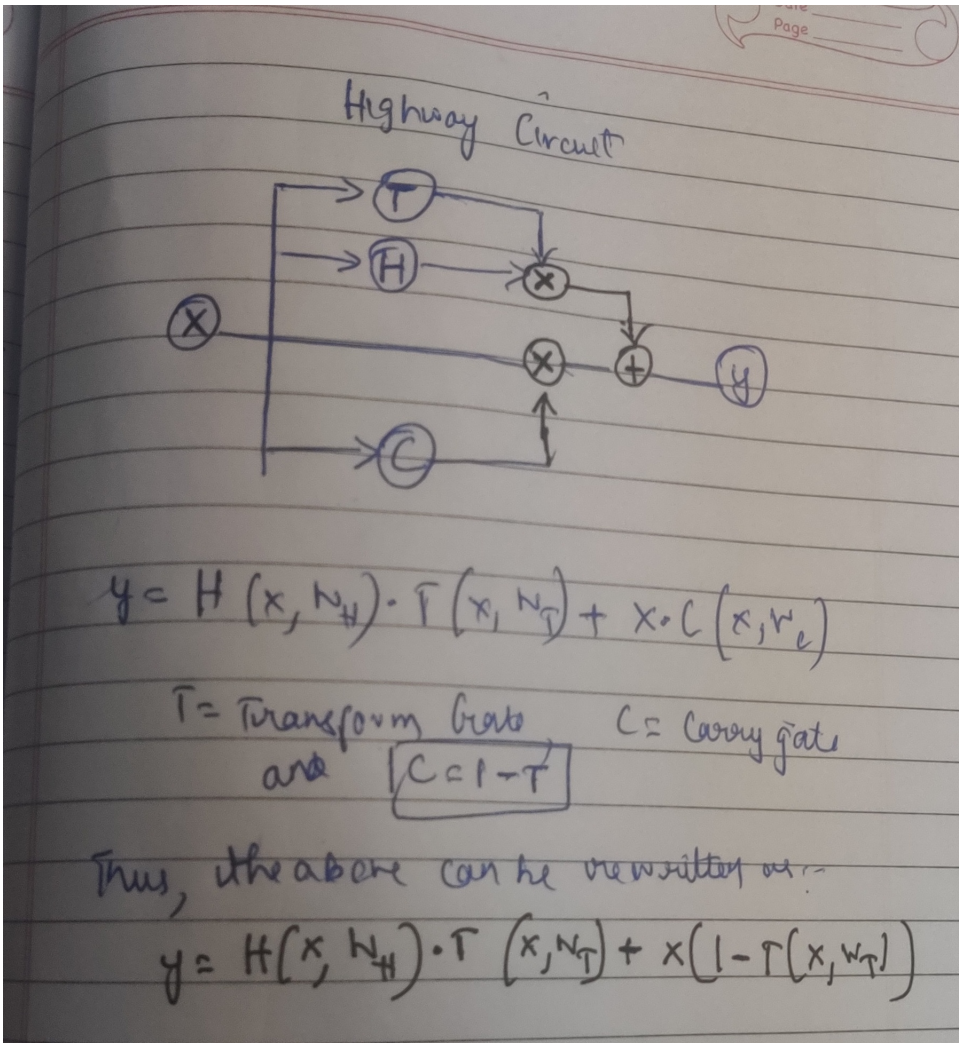

Figure 50: Highway block implemented

- Highway Architecture gates ensure that the previous layer information is carried and helps avoiding vanishing gradient problem like LSTM.

- The file **7.1.5_creation.py** contains the implementation of the architecture

Result Plots

- The follpwing section contains the accuracy and loss plots for skip connections, residual connections and highway connections. Since they take a longer time to train, all three models were trained for just 1 epoch and the results are plotted with respected to batches
- All the plots were generated by training the models on MNIST dataset

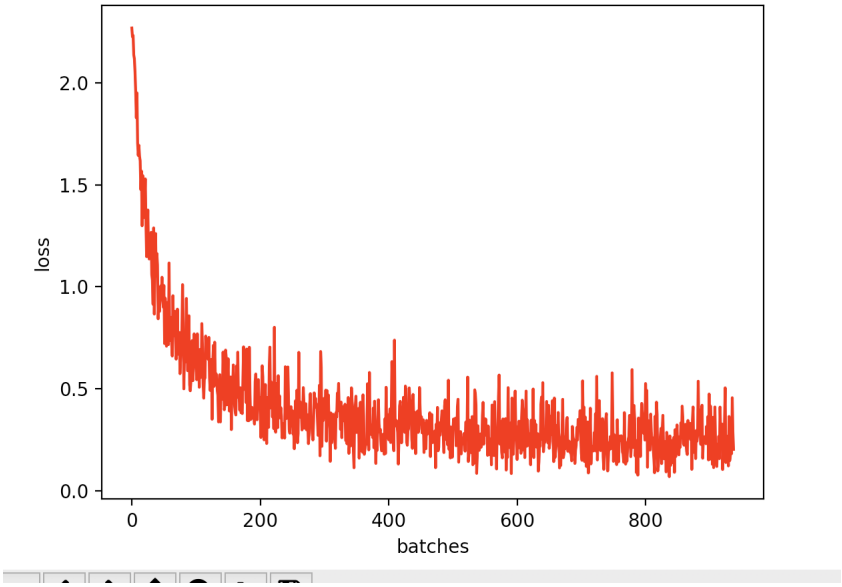

Figure 51: Loss vs Batches for Skip Connection

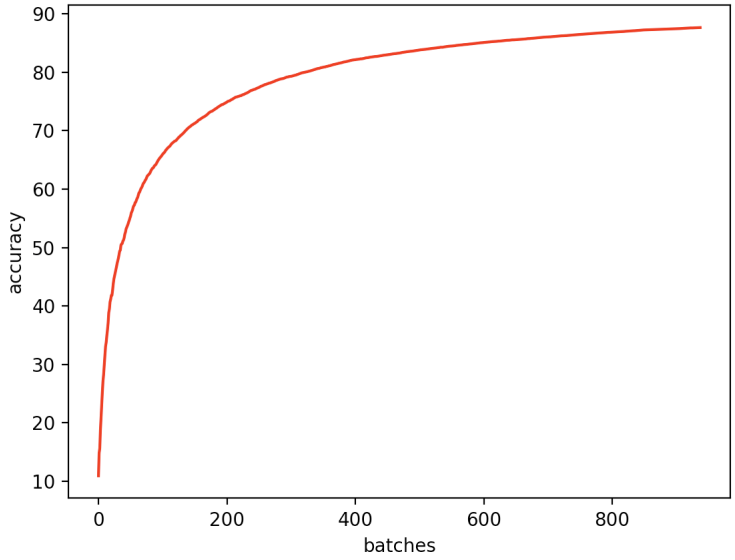

Figure 52: Accuracy vs Batches for Skip Connection

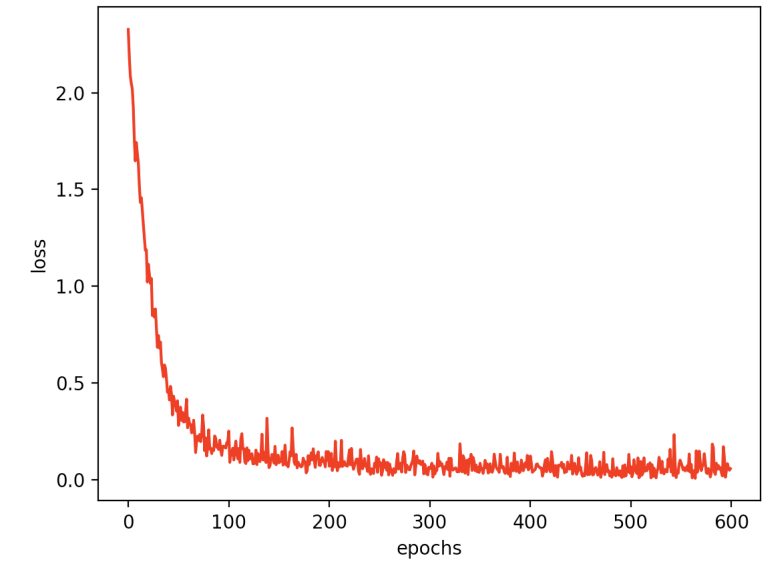

Figure 53: Loss vs Batches for Residual Connection

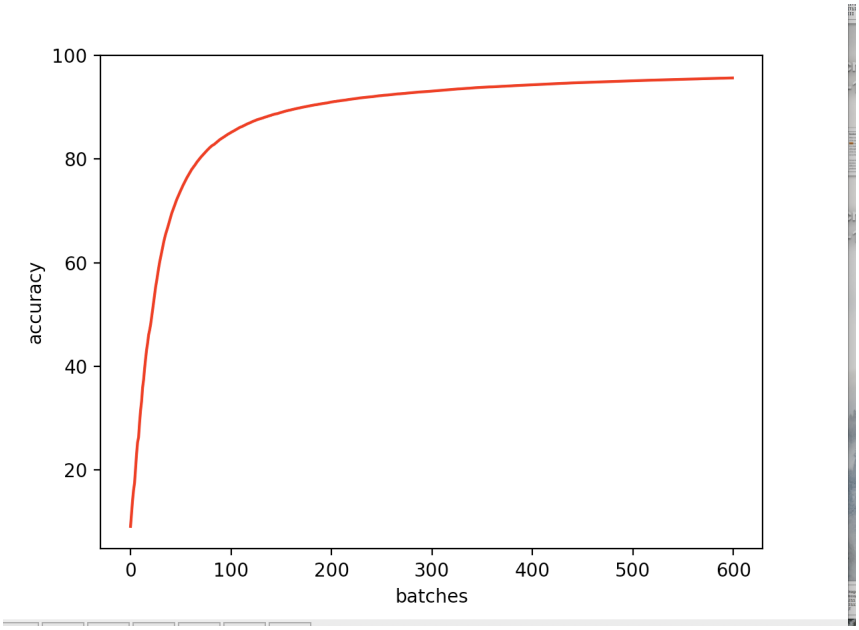

Figure 54: Accuracy vs Batches for Residual Connection)

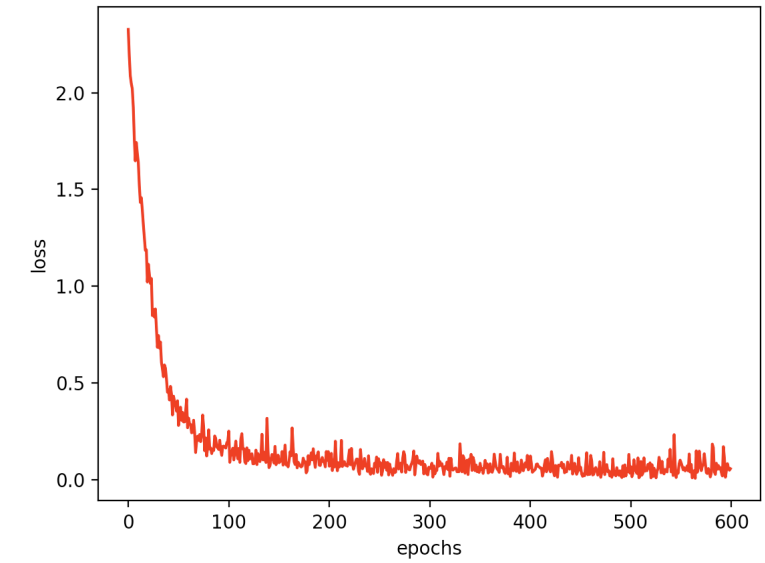

Figure 55: Loss vs Batches for Residual Connection

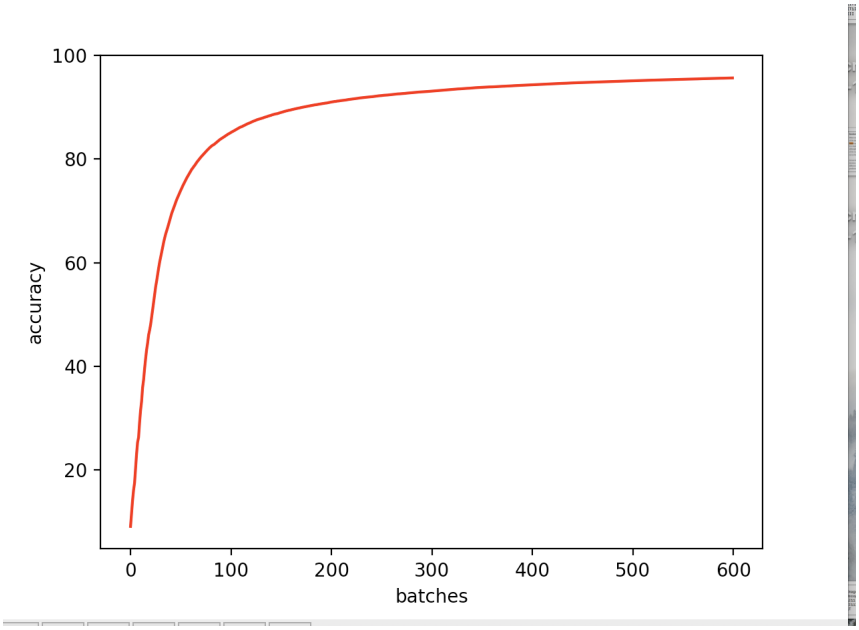

Figure 56: Accuracy vs Batches for Residual Connection)

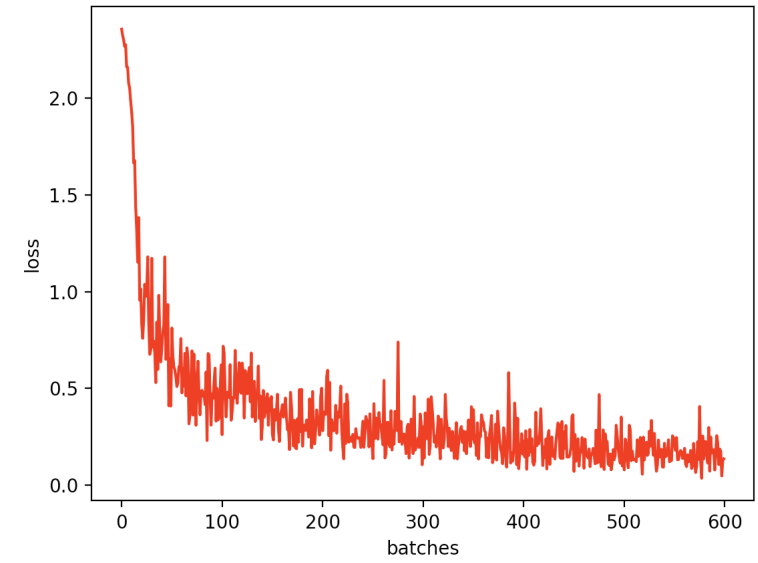

Figure 57: Loss vs Batches for Highway Connection

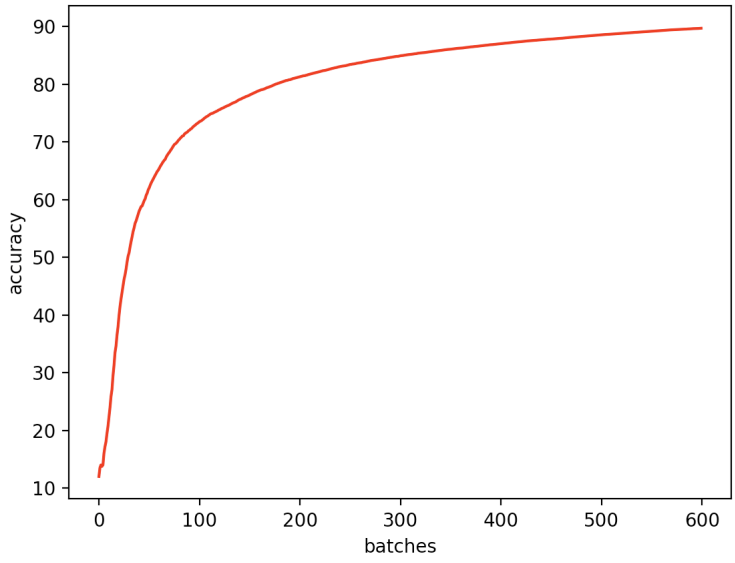

Figure 58: Accuracy vs Batches for Highway Connection)

7.2) Fine Tuning

- The file **7_2.py** contains the implementations for this section (both squeezenet as well as Self-designed (LeNet type architecture)
- It can be observed that the squeezenet fine tuned performed better than the LeNet type self-built architecture
- The pretrained weights of squeenzet helps in generalising and contain information about detecting image cruve and edges which is able to be generalised and turns out to be helpful while fine tuning for flower 17 dataset. The model achieves a good accuracy in very few epochs whilst comparing it with the normal architecture
- This proves that pretrained weights and transfer learning helps in providing better generalisable models and it's faster