Project Part 3: Classification Using Neural Networks and Deep Learning (SVHN Dataset)

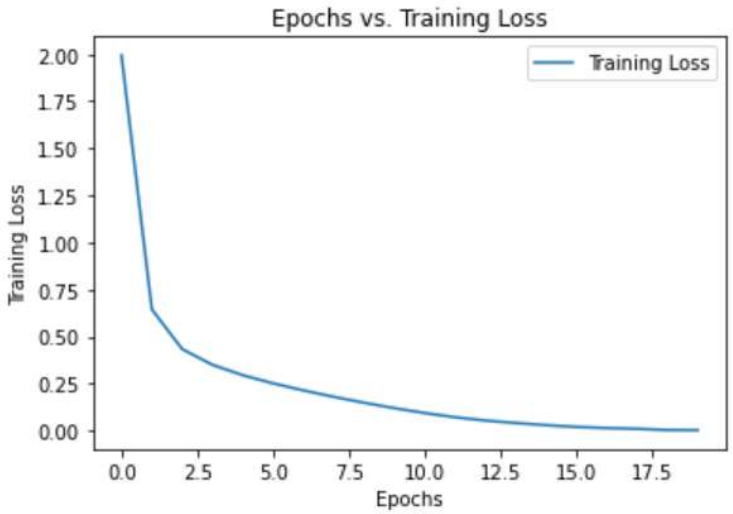
The steps involved in implementing the model are:

1. Loading the data from test and training files.
2. Normalize the training and testing data.
3. Perform one hot encoding for training and testing labels using categorical encoder.
4. Implement CNN using given architecture.
5. Use SGD optimizer with learning rate as 0.01.
6. Evaluate the model using the testing dataset.

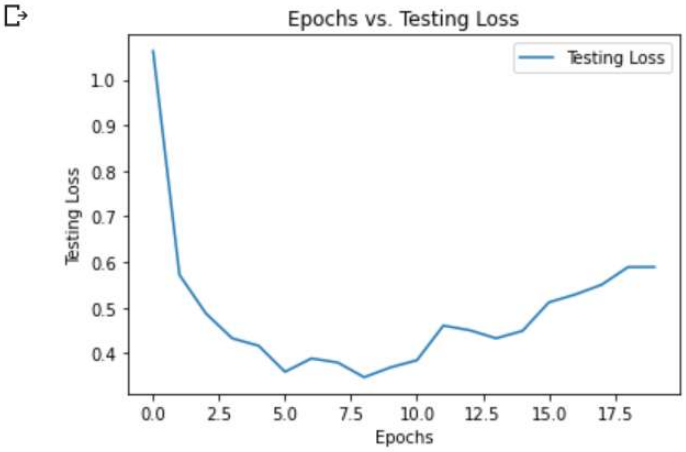
The CNN is implemented using the below architecture and parameters.

- The first hidden layer is a convolutional layer, with 64 output feature maps and kernel size of 5x5, stride 1 and activation as ReLU.
- The second layer is a max pooling layer with pooling as 2x2 and with stride 2.
- The third layer is a convolutional layer with 64 output feature maps and kernel size of 5x5 with stride 1 for convolution and activation as ReLU.
- The fourth layer is a max pooling layer with pooling as 2x2 and stride as 2.
- The 5th layer is connected to another convolutional layer, with 128 output feature maps. The convolution kernels are of 5x5 in size with stride 1 for convolution. The activation is ReLU.
- We apply flattening before the fully connected layer.
- The next layer is a fully connected layer with 3072 nodes and ReLU activation function.
- It is followed by another fully connected layer with 2048 nodes and ReLU activation function
- The last layer is a fully connected layer with 10 output nodes.

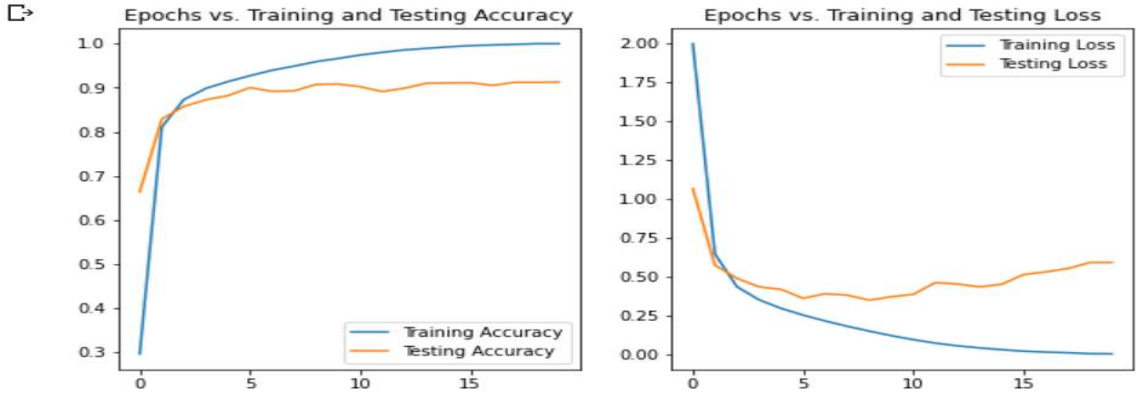
The graph between Epochs and Training loss is given in the below graph.



The graph between Epochs and Testing loss.



Epochs vs Training and Testing loss and Training and Testing accuracy.



Results:

The final testing accuracy reported is:

Test accuracy is: 91.295 %

Classification on SVHN dataset was implemented using the given architecture for the CNN and using the approach the amount of preprocessing to be done is minimized when compared to other algorithms and the model is trained with categorical cross entropy and SGD optimizer with a learning rate of 0.01 and the graphs were plotted for Epochs and Training loss and Testing loss along with Epochs and Training accuracy. The algorithm was run for 20 epochs with a batch size of 32. The loss values were stored and the plots are drawn. The training accuracy changes from 20 % to 99.99 % over the 20 epochs where as the testing accuracy changes from 66 % to 91.30 % over the 20 epochs. Making changes to the architecture layers as well as changing the learning rate and batch size will effect the final outcome for the accuracy. As a future work different architectures can be considered and the model which gives the highest accuracy can be chosen as the final output.