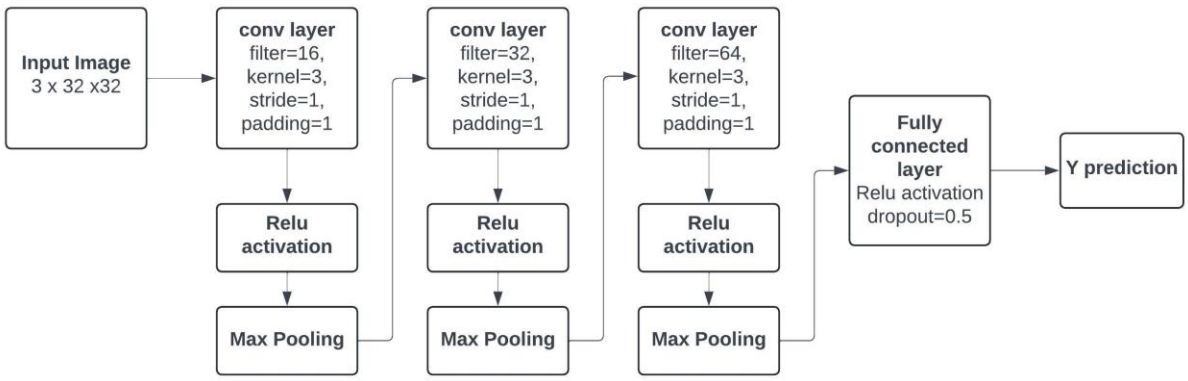
Name: S.M. Faiaz Mursalin **MNumber: M03537966**

Homework 2 Report

Architecture:

The architecture of the model is shown below:



Training Process Documentation:

We implemented a 3-layer CNN model to classify images in CIFAR-10 dataset. The dataset has 10 classes, and the dataset was split into training and test as instructed. The Model was trained over 20 epochs with varying learning rates. With multiple learning rates, this helps us analyze the effects of learning rate on training loss and accuracy. The results are shown below:

Learning rate = 0.001:

In the plot below we can see that the training loss decreases as epoch progresses, this indicates that the model is learning effectively with this learning rate. The training loss decreases a lot over the epochs and the test accuracy increases.

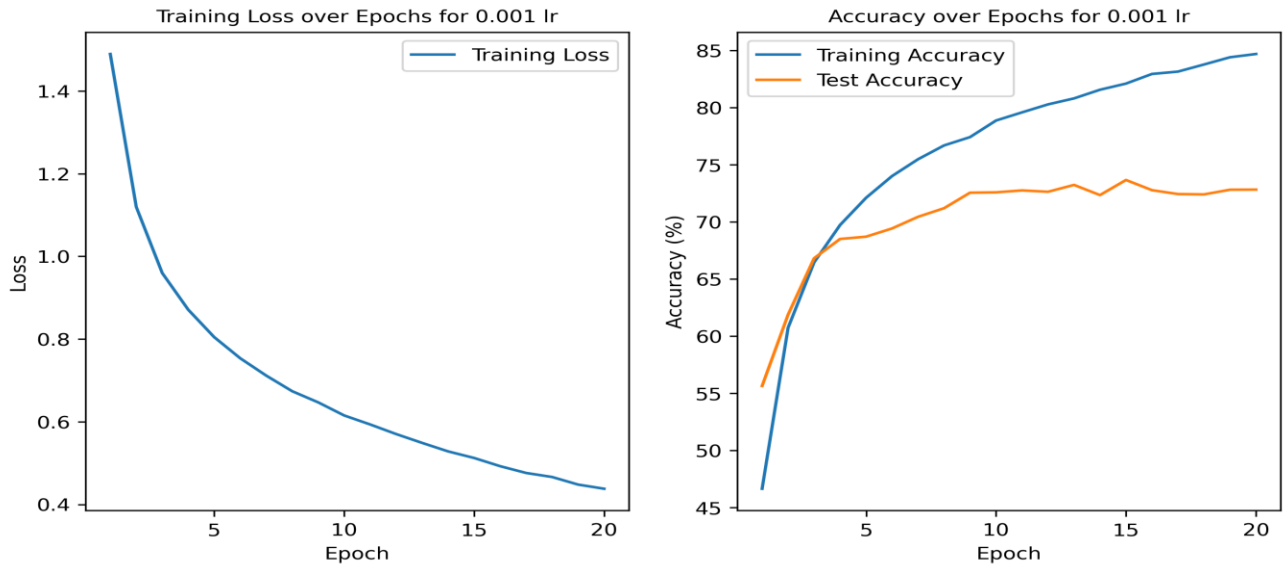


Fig 1.1: Training loss vs epochs, and Training Accuracy vs Testing Accuracy for LR=0.001

Learning Rate = 0.01:

Similarly in the plot below we see that even though the learning rate increases it is still showing a good curve. In the plot below we see that though the loss decreases smoothly but the test accuracy does not improve after a certain point which is after epoch 5. This makes us conclude that this learning rate has started to converge more quickly.

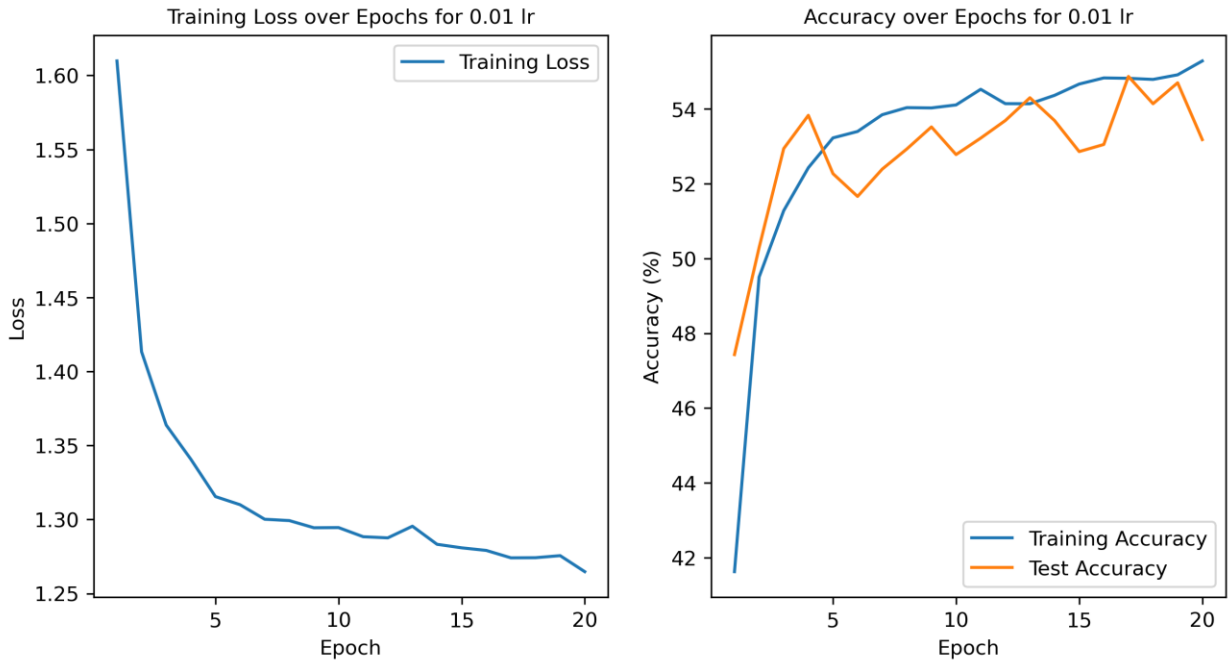


Fig 1.2: Training loss vs epochs, and Training Accuracy vs Testing Accuracy for LR=0.01

Learning Rate = 0.1:

The result below for learning rate 0.1 shows a clear issue. While the training loss drops very quickly, both training and testing accuracy reaches a plateau at around 10 percent. This indicates the model failed to learn effectively. The rapid fluctuations also tell me that the model is fluctuating and bouncing. This concludes that the model is stuck into a local minimum, and it is failing to converge.

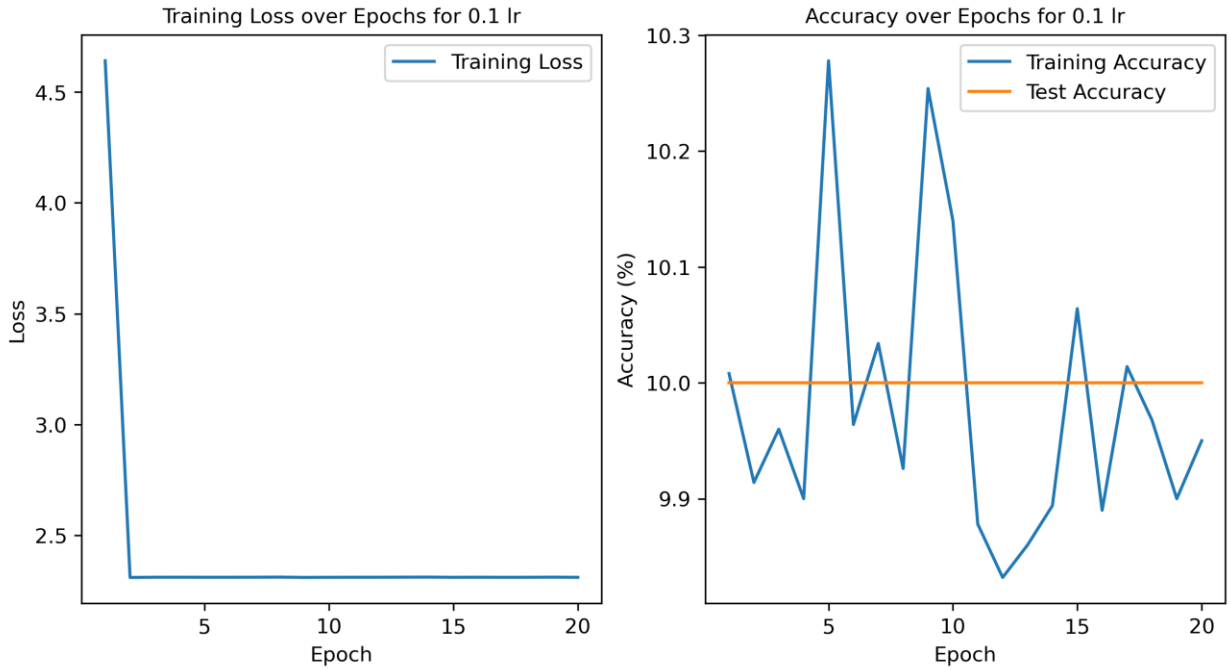


Fig 1.3: Training loss vs epochs, and Training Accuracy vs Testing Accuracy for LR=0.1

Effects of Different layers on learning and performance:

Convolution layer:

The convolution layer helps to extract features from the image. They extract basic patterns such as edges. While we go deep into the convolution layer, we see that the deeper the convolute layer is the more features like texture are extracted from the image. Increasing the depth of the layer can help to improve the model's accuracy but there is always a risk of overfitting the model which will improve the training accuracy but will not increase the testing accuracy.

Pooling layer:

We utilized the max pooling layer which helps us sample the feature maps by downgrading the sample dimension. But too many pooling layers can lead to spatial information loss.

Fully Connected layer:

The layer acts as a decision-making layer. This layer reduces the high dimensional feature maps into 1D output to perform classification. In this layer we cannot make it too large, and we must match the inputs from the previous layer with the 1D output for decision making.

Challenges faced and how they were overcome:

- **Overfitting:**
  One of the major challenges faced was the risk of overfitting. In case of big learning rate like 0.1 the step size is so big that the model converges to local optima. This hampers our test accuracy. To solve this problem, we tested different learning rates, and we could see that 0.001 learning rate with 20 epochs perform well. We can also utilize the concept of dropout such that our model performs well. We also added random flip of images and cropping for the transformation to increase the training size. And added l2 normalization as weight decay in the optimizer. The model gives exceptionally good output but only for learning rate 0.1 which is big step size it still gets stuck in a local optimum. The graphs with the changes are provided below for learning rate 0.001 and 0.1 for comparison purpose:

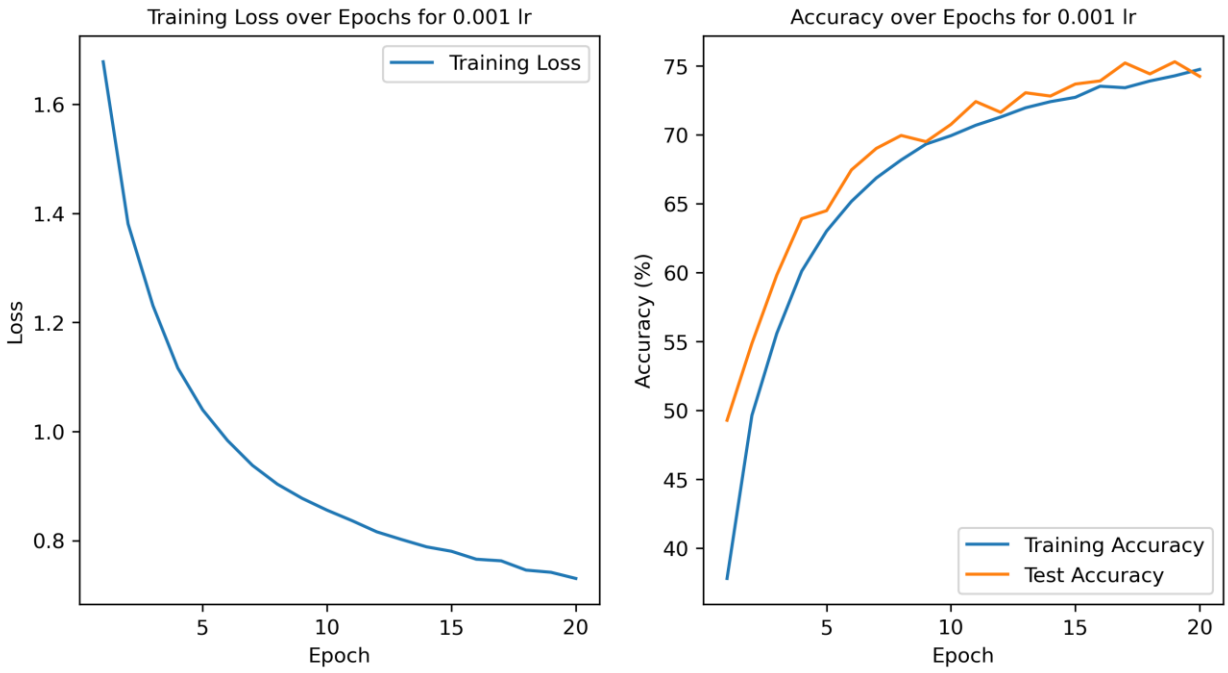


Fig 2.1: Training loss vs epochs, and Training Accuracy vs Testing Accuracy for LR=0.001

For learning rate 0.001 it gets better with the changes of dropout and l2 normalization.

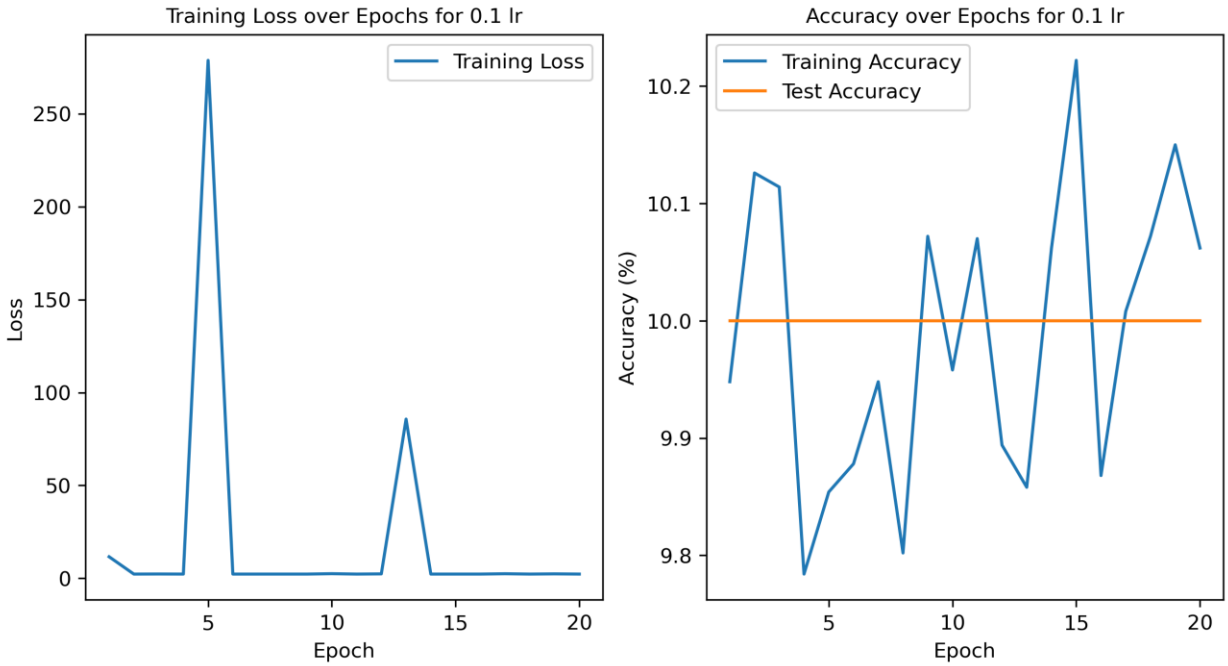


Fig 2.2: Training loss vs epochs, and Training Accuracy vs Testing Accuracy for LR=0.1

For learning rate 0.1 with changes of dropout and l2 normalization the model is still stuck in local optima as before.

- **Hyper parameter tuning**:
  Choosing the correct learning rate is critical. If the learning rate is too big the model converges quickly, and again a low learning rate would take a lot of time to converge. The optimal learning rate we could see was 0.001.
- **Balancing the complexity of the model:**
  Another challenge faced was model complexity. As we are using 3 hidden layers that can help in training for feature extraction, but computation cost was increasing making it more prone to overfitting. To solve this, we utilized the max pooling layers to reduce spatial dimensions and later drop out was introduced as above.