CIFAR – 10 Image Classification Report

Name – Shrayansh Bhardwaj

Student Number – 230524675

Module Identifier – Neural Networks and Deep Learning (ECS659P/ECS7026P)

Introduction: - The CIFAR - 10 dataset is composed of 60000 colour images and these images is divided into 10 classes in which each class has 6000 images. The classes include airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck images. These 60000 images are divided into training dataset composed of 50000 images and testing dataset composed of 10000 images. My project implements the convolutional neural network based on the specified basic architecture which include to create the data loader, then formation of neural network architecture composed of sequence of intermediate blocks followed by the output blocks, Training and Testing of a neural network and enhance my model to achieve accuracy on the testing dataset.

Architecture: - For CIFAR-10 dataset, which holds approximately 60,000 colour images in 10 categories, the study is based on the task with predefined transformations. These transformations comprise tensorization of images and having pixel values with normalized output values such that the network can learn the most effectively. The dataset is split into training and testing dataset, for each set, I have created the data loader. These data loader helps in shuffling of data in training dataset and batching due to which we can group image together to streamline network training and make processing more efficient.

Primary Architecture: - My primary architecture closely follows the architecture describe in the coursework. First, I implemented the intermediate block same described in the coursework which takes image as input and produces another image as output. This block consists of several convolutional layers, each performing a specific operation on the input image independently, Here I have implemented 3 intermediate block each having 2 convolutional layers. In each of these convolutional layers, I provide the same input into the block, these convolutional layers apply their own transformations and produces an intermediate image. The output image is created by combining all the intermediate images produced by the convolutional layers. Each intermediate image is multiplied by a corresponding weight and then added together to form the final output image.

Each intermediate image contributes toward the final picture by using a weight vector a. This weight vector is obtained by first weighing the average value of each channel in the input picture. The mean vector obtained by this operation, named ‘m’, has the same number of dimensions, channels included, as the input image. Finally, this vector is submitted to a fully connected layer that generates the weight vector ('a'), the individual weight elements of which determine how much the final image gets influenced by each of the intermediate image (which is obtained from the output of individual convolutional layers).

The output of the final intermediate block is passed to the output block where final intermediate image is also converted into a logits vector(‘o’) to be used for classification. From primary architecture I have achieved the accuracy of 42 % over 60 epochs.

Enhancement in Primary Architecture: - I have applied several techniques in my Architecture to increase the model performance and accuracy. These Techniques were either mentioned in the coursework or discussed in the lectures.

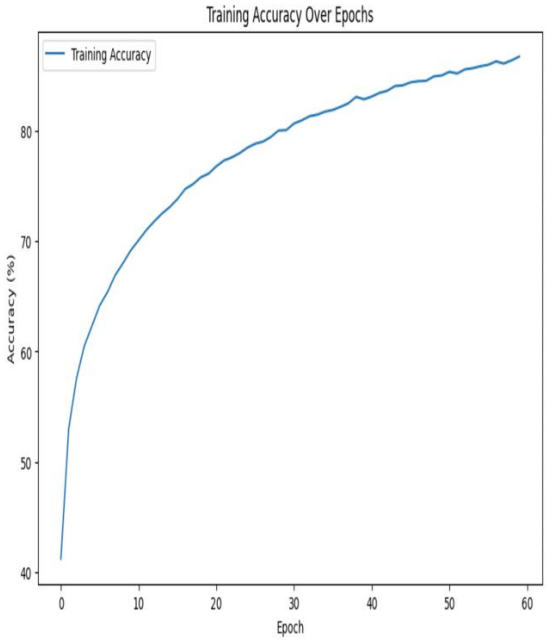
1. **Modular neural network design: -** In my Intermediate Blocks, I have introduced Batch normalization, ReLU activation and dropout techniques which is very beneficial in solving the vanishing gradient problem, improving model generalization and they provide a robust way to train deeper networks.
   In my output block I have used 'AdaptiveAvgPool2d' to reduce the spatial dimensions to 1*1 before feeding to a fully connected layer which is a good choice to deal with varying input sizes and reducing the number of trainable parameters in increase the model efficiency. I have also added the hidden layers in the output block which helps in learning complex and abstract representations from the data, avoid the overfitting, enhance model performance, and introduces nonlinearity in the model which are crucial for learning non-linear and complex patterns.

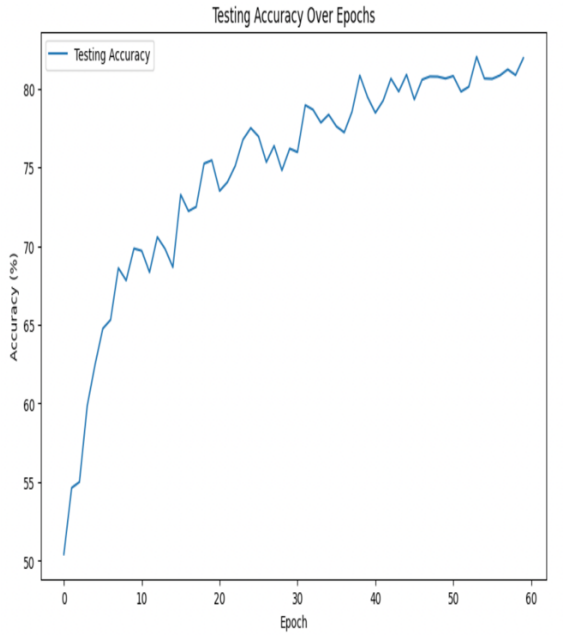
2. **Enhanced Data Handling: -** Here I used the normalization approach at the data pre-processing stage(transforms.Normalize) which helps in standardize the data and therefore converges become faster during the training process. I have also used the data loader to perform shuffle and batches to perform better training.

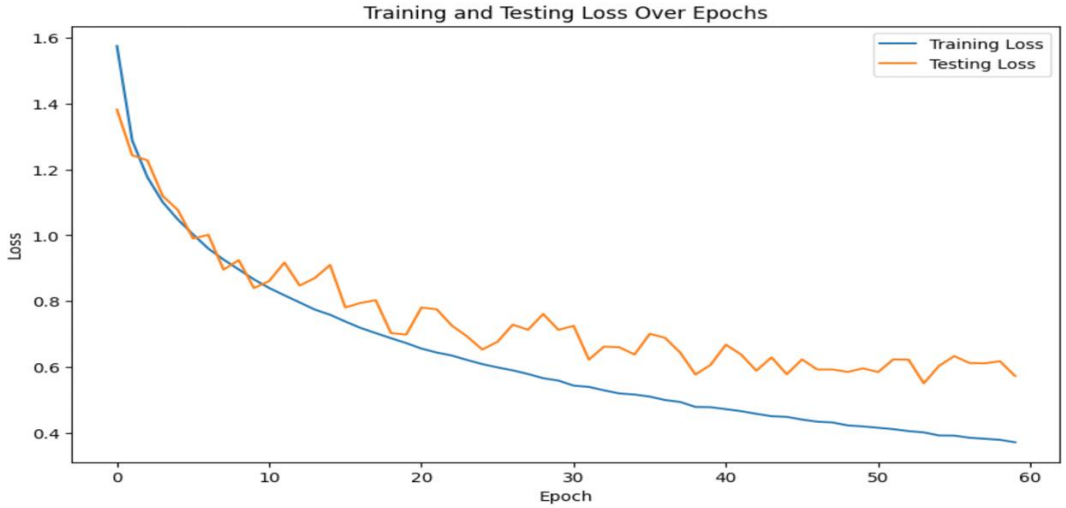
3. **Training and validation step and hyperparameter tuning: -** Here I have written the code to check the GPU availability and used the GPU if it is present in the system which increases the speed of training process due to faster computations. Here I also use the Adam optimizer with the learning rate of 0.001. This optimizer adjusts learning rates for each parameter individually based on estimates of first and second moments of the gradients, enabling efficient convergence. I have also tracked the training losses, testing losses over 60 epochs and accuracies per batch which helps in monitoring of model performance and quick adjustment of test parameters.

4. **Dropout: -** I have used the dropout in my network layers which is set to 0.5. It helps in preventing the overfitting of the model.

Result: - After using the enhancement techniques in my primary architecture, I am able to achieve the accuracy of 82.02% over 60 epochs.








```
Epoch 56 - Train Loss: 0.39, Train Accuracy: 86.02 %, Test Loss: 0.63, Test Accuracy: 80.61 %
Epoch 57 - Train Loss: 0.38, Train Accuracy: 86.32 %, Test Loss: 0.61, Test Accuracy: 80.83 %
Epoch 58 - Train Loss: 0.38, Train Accuracy: 86.13 %, Test Loss: 0.61, Test Accuracy: 81.21 %
Epoch 59 - Train Loss: 0.38, Train Accuracy: 86.42 %, Test Loss: 0.62, Test Accuracy: 80.85 %
Epoch 60 - Train Loss: 0.37, Train Accuracy: 86.77 %, Test Loss: 0.57, Test Accuracy: 81.93 %
Average Test Accuracy: 74.73 %
Best Epoch: 54 with Test Accuracy: 82.02 %
```