**Deep Learning**

**Term Project Report**

**Name:** Shahnila Rahim

**Roll Number:** 2020327462

**Architecture/Model:**

**VGG:**

Initially, I have used the VGG model for cifar-10 dataset from scratch and the accuracy was 88.78%. Even though after tweaking the parameters and optimizer from SDG, the accuracy was not improved much.

Before parameter tuning:

Learning rate: 0.001

Optimizer: SGD

Mean (RGB normalized mean): (0.4914, 0.4824, 0.4467)

Standard deviation (RGB normalized mean): (0.2471, 0.2436, 0.2616)

Obtained Accuracy: 88.78%

I changed the model parameters as follows:

Learning rate: 0.008

Optimizer: Adam

Mean (RGB normalized mean): (0.4911, 0.4824, 0.4467)

Standard deviation (RGB normalized mean): (0.2471, 0.2436, 0.2612)

Obtained Accuracy: 89.94%

**VGG-19:**

In this project I have used **VGG-19**. It is a convolutional neural network that is 19 layers deep. I have loaded a pretrained version of this network which is trained on more than a million images from the ImageNet database. The pretrained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 224-by-224. I have used pytorch framework to train and test cifar-10 dataset using VGG-19 mode.

The reason of using VGG-19 model was I only need 10 categories of images, so I though VGG19 is enough for CIFAR-10 and architecture is so simple, if you understand the basic CNN model, you will instantly notice that VGG19 looks similar. Moreover, it is 19 layers deep which make it more complex and can learn in high dimensional space and helps you to extract more features.

**Improvements and Parameter Tuning:**

Text

Description automatically generatedVGG-19 is used in this project and pretrained weights and layers are not changed. This network have 47 layers; 19 layers with learnable weights, 16 with convolutional layers and 3 fully connected layers. What more I have added is that; as in fig, I have added one extra last layer and fine-tuned it and modify the classes (i.e., 10) according to cifar-10 dataset. Moreover, I have changed the size.

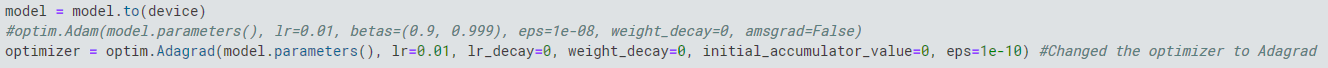
Even after changing the above-mentioned changes, the accuracy did not improve.

Text

Description automatically generatedSo, I normalized the data and tweaked the values of mean and standard deviation (as shown in below image).

Moreover, I changed the optimizer from SGD to Adaptive Gradient Algorithm (Adagrad) which is used for gradient based optimization. I have also used adam optimizer but adagrad performed well. The main advantages of Adagrad optimizer is that,

* It eliminates the need to manually tune the learning rate means that it adapts the learning rate.
* Convergence is faster and more reliable than SGD when the scaling of weights is unequal



**Accuracies**:

I have used Kaggle to run the code and used GPU.

**Number of epochs = 10:**

Learning rate: 0.001

Optimizer: SGD

Mean (RGB normalized mean): (0.4914, 0.4822, 0.4465)

Standard deviation (RGB normalized mean): (0.2023, 0.1994, 0.2010)

Obtained Accuracy: 90.1%%

Then I changed:

**Number of epochs = 10:**

Learning rate: 0.001

Optimizer: Adagrad

Mean (RGB normalized mean): (0.4914, 0.4822, 0.4465)

Standard deviation (RGB normalized mean): (0.2023, 0.1994, 0.2010)

Obtained Accuracy: 95.63%



**Increasing the number of epochs to 50 with same parameters:**

**Obtained Accuracy: 95.88%**

Text

Description automatically generated

The code for epochs 150 and 250 is still running