EEE 419/591
Project Final
Classification of CIFAR10

Introduction

The CIFAR10 dataset is an image-based labeled dataset with 10 different classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). Each image is 32x32 pixels for each of the 3 RGB colors (Red, Green, Blue). Our customer is asking us to develop an algorithm that takes an RGB image and declares a cat or a dog. The customer shared with us that they will never use the algorithm to classify any other types of images.

Problem Statement

Towards that goal, we decided to depend on the CIFAR10 dataset to build the algorithm and test it. Our main objective is to develop a code with the highest accuracy when tested on the CIFAR10 testing dataset.

Your duty is to write a machine learning (ML) code that calculates the classification accuracy for "cats" and "dogs" classes. Images in the CIFAR10 dataset are provided in RGB format, which has more data than a single colored (aka "Grayscale"). You will write two algorithms; one of them processes the RGB images, while the other converts the entire dataset to grayscale first before training the model. You will compare both algorithms' performance in terms of "Accuracy" and "Runtime". It is up to you to define what a Grayscale image is as long as it has a single color with 32x32 pixels. Keep in mind that your definition of Grayscale might affect the accuracy.

Accuracy: it is the percentage of images of the testing datasets that were accurately classified for the two classes (Cats and Dogs).

Runtime: it is the is the total amount of time needed to run the entire training + testing code of a given case (RGB or Grayscale).

The command to invoke the training portion of the CIFAR10 dataset is

```
from torchvision import datasets, transforms
trainset = datasets.CIFAR10(root='~/CIFAR10_data', train=True,
                            download=True, transform=transform)
```

The testing potion can be invoked similarly.

Code Inputs: Your code should not expect any input from the grader.

Code Outputs: Your code should output the following: Accuracy, in percentage, and Runtime, in seconds, for the RGB and Grayscale. In addition, it should recommend to the grader which

algorithm you recommend based on the overall performance. Since there is a tradeoff, there is no correct answer to this statement and will, thus, have no weight in the grading rubric. However, you need to let your code print it as it will be used for academic integrity violations. To prevent academic integrity, you only need to make sure that your statement prints your name. Hence, this statement can be hard coded into your code. A sample output of your code needs to have the following format:

```
RGB Accuracy: 64.4%
Grayscale Accuracy: 62%
RGB Runtime: 65.95904 seconds
Grayscale Runtime: 40 seconds
Ahmed Ewaisha recommends RGB algorithm
```

Hints

1. The code uploaded on the project's page on Canvas is recommended as a starting point. However, you are allowed to depend on any code that uses any ML algorithm.
2. Visually inspect the dataset, especially its dimensions, before starting to write your code.
3. Make sure your testing and training datasets are exclusive to the two required classes.
4. Play with the values of the parameters that your algorithm is using to improve accuracy. The parameters include (dropout ratio, number of layers, relu/activation function, number of in_channels/out_channels, kernel size, batch size, epochs…etc.).
5. The accuracy of your code might not be that high. Do not expect a 99% accuracy.
6. The values of parameters that give you the highest accuracy for RGB might also work well for Grayscale. Optimize them if you have time.
7. Most of the runtime will be spent on training not testing. Use a small training dataset while you are developing your code to reduce the runtime. You don't want to spend a lot of time waiting for your code to run to realize you forgot to change a parameter.
8. If a small training dataset does not sacrifice the performance significantly, you can submit a code that uses only a portion of the training dataset.

Allowed:

1. The use of AI (ChatGPT…etc.), especially if you will use it to:
    a. Ask it to recommend some commands that do a specific task.
    b. Paste (parts of) your non-working code in the AI-chat along with the error message to understand why it is not working and what is recommended.

    In both cases, you should not use AI's suggestions blindly since AI hallucinates A LOT.
2. The use of any resources (internet/books/videos/forums) except if you are involved in direct communication with another human being outside this class (tutor, friend…etc.).
3. General discussions outside/inside Ed Discussion board.
4. Sharing commands. Limit your screenshots/shares to 2 consecutive commands per share

5. Sharing final answers (accuracy/runtime).
6. Sharing ranges of parameter values you tried/recommend even if the values you plan to submit are included in these ranges. Sharing the exact parameter values you plan to submit is not allowed.
7. Sharing the name of the ML algorithm you plan to submit / work on / have tried but failed.
8. Sharing screenshots of codes that don't work or give errors to ask your peers for feedback. In this case, there is no limit on the number of commands but make sure it is a screenshot not a copy-paste of your code.
9. Sharing fundamental ideas that would work or would not work.

Not Allowed

1. Pasting the entire code that AI gives you without understanding what it does. AI is allowed because it helps you understand, not so that it does the work for you.
2. Seeking help from another human being outside this class (private tutors, friends…etc.)
3. Sharing screenshots/command-pastes with 3 or more commands with your peers
4. Sharing the exact parameter values you plan to submit with other students.
5. Working on the project together. This project is an individual effort.

Notes

1. Make sure the code you submit is ML-based. Non-ML-based will get no credit.
2. Average computers on the market nowadays should run your entire code in 2-4 mins, without a GPU. If your code needs more than 5 mins in total, try to optimize it or seek help.
3. You need to submit only one code that prints the entire required output.
4. Make sure your code does not need a GPU to run. Your grader might not have a GPU. If you want to use a GPU while developing your codes, make sure to change your code to use CPU before you submit.
5. Name your code as "Final_Project_FirstName_LastName.py"