CS776: Deep Learning for Computer Vision
Assignment 1
Due Date: February 3, 2023 (11:59 PM)

General Instructions:

- This is a group assignment with TWO members in a team. (You can also submit the assignment individually; however, it is advisable to form a group.)
- Only electronic submissions will be accepted. your solution has to be submitted via the mookit (Hello IITK) platform.
- Late submissions will not be accepted.
- Any sort of plagiarism will be penalised. If you are referring to any materials online or books, please cite them accordingly otherwise it will be considered as plagiarism.
- You should not use built-in functions from Numpy or PyTorch or other image processing libraries for image augmentation/transformation. You need to write your own functions for this assignment. For more details read the instructions in the question.
- Please ensure that the report is written as per the instructions given in the assignment.
- You are free to use systems from KD Lab or Google Colab or Kaggle to train the models.
- Instructions for KD Lab machine login:
    1. ssh [iitk_user_name]@gpu3.cse.iitk.ac.in
    2. enter password : cs776_user
    3. Please change the password for security concern
    4. When you login, you will land in /[user_name] directory, where you can save all the data within this directory.
    5. For accessing internet, run (command : python3 ../cs776/authenticator.py) and provide iitk credentials (not cse)
    6. To keep the remote terminal running even after closing the local terminal use screen or tmux
    7. To init a screen: use screen -t [some name: alphanumeric]
    8. To get back to the last screen: use screen -x (if you have initiated multiple screen then screen -x will return some ids, then to go back the corresponding screen use: screen -x [id])

Question (100 Marks)

In this assignment, your goal is to implement and train a multi-layer perceptron (MLP) on the CIFAR-10 dataset with data augmentation (Two hidden layers with 64 neurons each MLP model).

1. Download the CIFAR-10 dataset (https://www.cs.toronto.edu/~kriz/cifar.html). Hint: Use pickle library to load the dataset. [5 marks]

2. Implement the image transformation methods mentioned below: [5x4=20 marks]

   (a) Image Enhancement
   Instruction for enhancing the image:
   for each pixel i in the image x use the formula: $(\frac{(i-min)}{max-min}) * 255$
   Here, min and max are the minimum and maximum pixel values in the image x.

   (b) Posterization of Image (A posterizing image is the one in which you reduce the number of colors in the image to provide a visual art). Follow the instructions below to posterize the image:

   - Select a desired minimum and maximum pixel value in the range of [0-255].
   - Now, for each pixel i in image x:
     i. Calculate the range r by subtracting selected minimum and maximum pixel value
     ii. Get a divider for the colors using $divider = \frac{255}{r}$
     iii. Get the level of colors by $i = \frac{i}{divider}$
     iv. Finally, apply the color palette on pixel by $i = i + min$
   - Make sure, the final image x should have the pixel values in the range of [0, 255]

   (c) Random Rotate [-180°, 180°].

   (d) Contrast & Horizontal flipping. (First, change the contrast of the image with a factor of α randomly selected from the range $(0.5, 2.0)$ and then flip the image horizontally with a probability of 0.5)

   Instructions for changing contrast of an image:
   Given an image x with pixel values in range [0, 255], you can change its contrast by a factor of α using the following steps:

   i. Change all the pixel values with formula $x'(i, j, c) = \alpha \cdot (x(i, j, c) - 128) + 128$. Here, $x(i, j, c)$ refers to the c-th channel value of the (i, j)-th pixel.
   ii. Clip the pixel values in x' so that the final values are in the range [0, 255]

   Note that $\alpha < 1$ will decrease the contrast of the image while $\alpha > 1$ will increase the contrast.

   For each transformation, define a python function that takes an input image and returns the transformed image. Use of any built-in functions (such as pillow, sklearn, PIL, CV2, etc.) is prohibited for this part. For all the padding operations as well as the gaps created due to the image transformation operations, fill with zero values. Demonstrate each transformation method by applying the transform functions on at least one example image.

3. Create the augmented training set using the transformation functions implemented in the previous part. Randomly select one of the four transformations for each image in the training set and apply it to that image. Combine the transformed images with original training set to get the augmented training set. Note that the number of examples for the augmented training set will be twice that of the unaugmented training set. [10 marks]

4. Use the **feature_extractor.py** file provided with the assignment on the original (unaugmented) CIFAR-10 dataset and on the augmented dataset to get 1-dimensional input vectors. You can ignore the implementation of **feature_extractor.py** and use it directly. [10 Marks]

   Instructions for using feature-extractor.py:

   - Refer to this page : https://pytorch.org/get-started/locally/ for installing the required dependencies of PyTorch. However, if you are using Google Colab or Kaggle for running your code then you do not need to install PyTorch there, as these environments supports PyTorch.
   - The feature_extractor.py accepts images of size $(3 \times 224 \times 224)$ [Channel $\times$ Height $\times$ Width]. Use image processing libraries like PIL, CV2 to resize the CIFAR images from $(3 \times 32 \times 32)$ to $(3 \times 224 \times 224)$.
   - Pass the resized images to **feature_extraction** function of BBResNet18 class to generate feature vectors
   - **feature_extraction:** function expects each image is a **numpy.ndarray** of dtype: **numpy.float32** and shape: **[None, 3, 224, 224]**, where: **None** represents a variable size. It returns a **numpy.ndarray** of dtype: **numpy.float32** and shape: **[None, 512]**.

5. Implement a multi-layer perceptron (MLP) for classification of CIFAR-10 images. Use two hidden layer with 64 neurons each and ReLu activation function. The input to this MLP will be the 1-dimensional vectors generated in the previous step. [10 Marks]

6. Implement the back-propagation algorithm and use it to train the MLP model on: [20 marks]

   (a) original training set

   (b) augmented training set

   You are not allowed to use built in functions that performs back propagation directly. You should write your own back propagation algorithm.

7. Implement the following Classification algorithms: [10 marks]

   (a) SVM Classifier

   (b) KNN Classifier

   (c) Logistic Regression Classifier

   (d) Decision Tree Classifier

   You can use any built in library functions for these ML classifiers.

8. Evaluate the performance of the above four classifier and the following trained MLP models on both the original (unaugmented) test set and augmented test set for top 5 classification : [5 marks]

9. Submit a report clearly explaining how you have built the MLP models and chosen the hyper-parameters like learning rate, epochs used for training, evaluation metrics and the instructions for running the models. Derive the gradients and update expressions needed to implement back-propagation, Compare the performance of the MLP model and the specified ML classifiers on the original (unaugmented) test set and the augmented test set and justify the observed behaviour. [20 marks]