**February-June 2022 Semester**
**CS671: Deep Learning and Application**
**Programming Assignment 3**

**Date: April 19, 2022**
**Deadline for submission of code and report: Monday, May 9, 2022, 10:00 PM**

The objective of this programming assignment is to deepen your understanding of the functional blocks of a convolutional neural network using the means of coding. The primary task of building a convolutional neural network for image classification is broken down into smaller hierarchical tasks. You are given the subset of Caltech-101 dataset for the same. Each group is given 3 classes. Given dataset is train-validation-test separated. The Caltech-101dataset consists of colour images with varying sizes. **You must resize the images to a size of 224 x 224**. The tasks for this assignment are as follows:

1. Pick any one image from each of the three classes and convert it into a single channel grayscale image. Initialize a 3x3 convolutional filter using Kaiming initialization. Traverse the convolution filter over all the pixels of the image with stride 1, padding 0 and obtain the final output as a feature map. Calculate the expected dimension of the feature map using the mathematical formula and verify it is the same as obtained by you. Report should include these observations, image considered, filter values and the feature maps obtained etc.

2. Write a class that implements the convolutional layer with ReLU activation function. Instantiate two convolutional layers having 32 and 64 3x3 filters using the self-coded convolutional layer class. Initialize convolutional filters using Kaiming initialization. Pass images (considered in the Qn. 1) in a feedforward manner to the two stacked layers and print the feature map obtained after each layer. For convolution in both the layers, consider stride 1, padding 0. Calculate the expected dimension of the feature map using the mathematical formula and verify it is the same as obtained by you. Report should include these observations, any 10 filters as images from first convolutional layer and any 10 filters as images from second convolutional layer and feature maps of corresponding filters

3. Write a code to implement a convolutional neural network. The input layer should take 224 x 224 3 channel image, stack two convolutional layers with 32 and 64 3x3 filters one after the other, down-sample the feature map using the 2x2 max pooling operation (stride 1), and flatten it. For convolution in both the layers, consider stride 1, padding 0. Next, add two fully connected layers. Consider 128 hidden nodes with rectified linear activation function in the first hidden layer and 3 neurons with a softmax activation function in the second layer (output layer). Initialize convolutional filter using Kaiming initialization and fully-connected layers with weights randomly sampled from a zero mean normal distribution with a standard deviation inversely proportional to the square root of the number of units. The weights should be updated using a backpropagation algorithm with stochastic gradient descent.

    a. Observe the classification accuracy for training, validation and test data. Present the accuracy with confusion matrix.
    b. Consider the same image from each of the classes as taken in Qn. 1. Plot the 10 filters as images from first convolutional layer and 10 filters as images from second convolutional

layer (Consider the filters at the same position/location index as taken in Qn. 2) and feature maps of corresponding filters.

c. For each of the same 3 images, find out 5 neurons in the last convolutional layer that are maximally activated. Trace back to the patch in the image which causes these neurons to fire. Visualize the patches in each of the 3 images which maximally activate these 5 neurons.

4. Leverage Tensorflow Keras API, use VGG19 pretrained on ImageNet. Modify the classification layer of VGG19 to 3 output nodes. Retrain only the classification layer.
   a. Observe the classification accuracy for training, validation and test data. Present the accuracy with confusion matrix and compare with that of the performance obtained in Qn. 3a.
   b. For each of the same 3 images considered in Qn. 1, find out 5 neurons in the last convolutional layer that are maximally activated. Trace back to the patch in the image which causes these neurons to fire. Visualize the patches in each of the 3 images which maximally activate these 5 neurons.
   c. Leverage Tensorflow Keras API and visualize the influence of input pixels on any of the 5 neurons in the last convolutional layer of a pretrained VGG19 network after passing any one image from each of the 3 classes (same images considered in Qn. 1). Consider guided-backpropagation algorithm to find the influence. In the report give the selected image, position of each of the 5 neurons in last convolutional layer and resultant gradient images. Write your observations clearly.

**Instructions:**

- **Each group of students must use the dataset assigned for that group.**
- **Use python/MATLAB for programming.**
- **For tasks 1, 2, and 3 use of deep learning (DL) APIs (Tensorflow, PyTorch, MATLAB libraries for DL etc.) is not allowed.**
- **Codes should be self-written. If copied, zero will be awarded.**
- **Report should be in PDF form and report by a team should also include the observations about the results of studies.**
- **Your codes and report should be uploaded as a single zip file on Moodle.**
  - **Give the name of the folder as Group<*number*>_Assignment3, Example: Group*01*_Assignment3**
  - **Give the name of the zip file as Group<*number*>_Assignment3.zip. Example: Group*01*_Assignment3.zip**

**We will not accept the submission, if you don't follow the given instruction.**