Missouri University of Science & Technology — Department of Computer Science

Spring 2024 — **CS 6406: Machine Learning for Computer Vision (Sec: 101/102)**

Homework 3: Convolutional Neural Networks

Instructor: *Sid Nadendla* — **Due:** *April 15, 2024*

Goals and Directions:

- The main goal of this assignment is to implement CNNs using PyTorch.
- Comprehend the impact of hyperparameters and learn to tune them effectively.
- A template Jupyter notebook will be provided for each problem to develop your solution.
- You may obtain your compute power from S&T's Foundry, Google Colab, or AWS SageMaker Studio Lab.

Problem 1 Convolutional Neural Networks *4 points*

Build a CNN of your choice from scratch and train it on CIFAR-10 dataset. Divide the CIFAR-10 dataset into 10 folds (batches), and perform the classical 10-fold cross validation of your trained CNN model for a 80-20 train-test split of your dataset. Your CNN should produce an accuracy of at least 75%.

Problem 2 Transfer Learning *6 points*

Import three different pretrained models (e.g. AlexNet, VGG, ResNet-50) from the PyTorch library, which is originally trained on ImageNet. Use transfer learning techniques and retrain the model on CIFAR-10 dataset. Perform 10-fold cross validation, and compare your CNN model performance with the three retrained models in terms of accuracy.

Problem 3 Object Detection *15 points*

Load two different pretrained object detection models, Faster-RCNN and YOLO, from PyTorch and fine-tune the models on Pascal-VOC dataset (download link - `http://host.robots.ox.ac.uk/pascal/VOC/voc2012/index.html#devkit`). The dataset consists of 17125 images along with corresponding annotations in XML format for each image. Preprocess the images to appropriate size/color and compute the bounding box coordinates for each image from the given XML annotations.

Create a function which loads the pretrained model and replace the pretrained head with a new one depending on the number of classes in Pascal-VOC. Train the modified model with your choice of optimizer (SGD or Adam) for 80-20 train-test split of the dataset. Do the same with both the pretrained models and compare their performances in terms of average precision (AP) and mean average precision (mAP) metrics. For your reference, please visit this link: `https://www.kaggle.com/code/yerramvarun/fine-tuning-faster-rcnn-using-pytorch`