Deep Learning, Spring 2020
Assignment 5 - Part 1

Detecting Coronavirus Infections through Chest X-Ray images

NOTE: WE WILL UPDATE THE DETAILS AND ADD REFERENCES. DON'T WAIT FOR THOSE, START WORKING NOW. **THERE IS NO LATE SUBMISSION FOR THIS.**

Submission Guidelines:

Submit all your code and results in a single zip file with name **FirstName_RollNumber_05.zip**.

- Submit single zip file containing
  (a) Code (b) Report
- There should be a **Report.pdf** detailing your experience and highlighting any interesting result. Kindly don't explain your code in the report, just explain the results. Your report should include your comments on the results of all the steps, with images, for example what happened when you changed the learning rate etc.
- The assignment is only acceptable in Jupyter Notebook format (.ipynb).
- In the root directory, there should be **1** Jupyter Notebook and a report.
- Root directory should be named as **FirstName_RollNumber_05**
- Your code notebooks should be named as **'rollNumber_05.ipynb'**
- Follow all the naming conventions.
- For each convention, there is a 3% penalty if you don't follow it.
- Email instructor or TAs if there are any questions. You cannot look at others code or use others code, however you can discuss with each other. Plagiarism will lead to a straight zero with additional consequences as well.
- **100% (of obtained marks) deduction per day for late submission.**

Due Date: 11:59 PM on Saturday, 25th April 2020

Note: For this assignment (and for others in general) you are not allowed to search online for any kind of implementation. Do not share code or look at the other's code. You should not be in possession of any implementation related to the assignment, other than your own. In case of any confusion please reach out to the TA's (email them or visit them during their office hours).

- **You are strongly advised to implement this assignment using Google Colaboratory's free GPU if you don't have a high end GPU optimized machine**

- **Please go through the PyTorch tutorial uploaded on Google Classroom before starting this assignment if you are unfamiliar with its functionality.**

Objectives:

In this assignment you are required to write code for detecting infections such as COVID-19 among X-Ray images:

- Use CNN, pre-trained on ImageNet, to extract basic features from X-Ray images.
- Train the classification layers in order to detect instances of Infected (COVID-19 + Pneumonia) and Normal X-Ray images.
- Fine-tune the entire network to try to improve performance.

This assignment must be completed using PyTorch. Assignments in **Tensorflow/Keras** or any other deep learning library **WILL NOT BE ACCEPTED**.

Chest X-Ray Images Dataset

Background:

New studies [1] have revealed that the damage done to lungs by infections belonging to the family of coronaviruses (COVID-19, SARS, ARDS etc.) can be observed in X-Ray and CT scan images. With a worldwide shortage of test kits, it is possible that careful analysis of X-Ray and CT scan images may be used in diagnosis of COVID-19 and for the regular assessment while they are recovering. In this assignment, we will use an open source dataset of X-Ray images and train a Convolutional Neural Network to try and detect instances of infections containing COVID-19 and Pneumonia.

Dataset Details:

This dataset contains X-Ray images from 2 classes:

Class	# of images in training set	# of images in validation set	# of images in test set
Infected	4,919	615	615
Normal	7,081	885	885

Chest X-Ray images are taken in different views (AP or PA) depending on which side of the body is facing the X-Ray scanner. Images from different views have slightly different features. For this task, we will be using images without considering their views. A few sample images:

COVID-19

Pneumonia

Normal

Fine-tuning in Pytorch:

In PyTorch, each layer's weights are stored in a Tensor. Each tensor has an attribute called 'requires_grad', which specifies if a layer needs training or not. In fine-tuning tasks, we freeze our pre-trained networks to a certain layer and update all the bottom layers. In PyTorch we can loop through our network layers and set 'requires_grad' to False for all the layers that we want to be freezed. We will set 'requires_grad' to True for any layer we want to fine-tune.

PyTorch Fine-tuning Tutorial: Link

GitHub Repository:

Besides code and report, you are all required to make a public GitHub repository where you will upload your code and results. Following conventions are for the repository only:

1. Name your code notebook as covid19_classification.ipynb
2. Name your repository as rollNumber_COVID19_DLSpring2020
3. Show your results in **readme.md** (confusion matrices and accuracy)
4. Create a heading of Dataset and provide the link that was shared with you on Classroom.
5. Create a folder named 'weights' and upload fine-tuned models. Naming convention of models is mentioned in each respective task.
6. Add the following description to repository

   "This repository contains code and results for COVID-19 classification assignment by Deep Learning Spring 2020 course offered at Information Technology University, Lahore, Pakistan. This assignment is only for learning purposes and is not intended to be used for clinical purposes."
7. You may refer to the following repository to see how they have organized their results and description:

   https://github.com/kevinhkhsu/DA_detection

Task 1: Load pretrained CNN model and fine-tune FC Layers

- In this task you will fine-tune two networks **(ResNet-18 and VGG-16)** pretrained on ImageNet weights.
- Load these models in PyTorch and freeze all the layers except the last FC layers.
- Replace the FC layers with 2 FC layers. First FC layer will have neurons equal to:
  - (Last 2 digits of your roll number x 10) + 100
- The Last FC layer will have neurons according to the number of classes
- You may try different learning rates
- Save your model and name it as **'vgg16_FC_Only.pth'** and **'res18_FC_Only.pth'**

Task 2: Fine-tune the CNN and FC layers of the network

- In this task you will fine-tune two pre-trained networks (ResNet-18 and VGG-16 pretrained on ImageNet weights)
- Perform different experiments where you first unfreeze only a few Convolutional layers and then the entire network and fine-tune on your dataset
- Compare the performance of training in different experiments. Show what effect it has on accuracy when you fine-tune just FC layers, then a single Conv layer, then a few Conv layers and then the entire network.
- Save your model where you fine-tune the whole network and name it as **'vgg16_entire.pth'** and '**res18_entire.pth'**

Requirements:

In your report, for each task, you are required to provide the following:

1. Confusion Matrix for train, test and validation sets
2. Loss and accuracy curves for train and validation sets
3. Experimental setup (learning rate, number of layers fine-tuned etc.)
4. Two well classified images and two worst classified images from both classes
5. Final accuracy and F1 score for each experiment
6. GitHub Repository link
7. Analysis on each task and comparison of experiments to each other

Report

- Share the loss and accuracy curves on the train and validation for both.
- Use the same scale of axis for both the tasks so that they are comparable
- Discuss Task-1 vs Task-2, why and how it effects, which works better and why?

Please perform all tasks in the same notebook. Do NOT create separate notebooks for each task.

References:

[1] Zu, Zi Yue, et al. "Coronavirus disease 2019 (COVID-19): a perspective from China." *Radiology* (2020): 200490.