Tutorial Assignment IV

Instructions: Please read the below instructions carefully to complete this assignment.

1. This assignment has two parts (Part 1 and 2) and will be graded out of 25 points.
2. Please work on this assignment individually.
3. Please submit all your code in one python (*.py) file; name it as <*your_mac_id*>_part1-2.py.
4. Please submit your report (*.pdf) file; name it as <*your_mac_id*>_report.py.
5. Submit these files on Avenue.
6. Please DO NOT upload ANY datasets. The datasets you upload will NOT be used for assignment evaluation.
7. Please import only the below functions in your code. The assignment will NOT be graded if any other function from any other library is used.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import sklearn
import torch
import torch
import torch.nn as nn
import torchvision
import torch.nn.functional as F
from torch.utils.data import RandomSampler,DataLoader
# you may not need the below functions, but just in case
from torchvision.transforms import Compose,RandomHorizontalFlip,Grayscale,Resize,RandomCrop,ToTensor
```

8. The coding assignment will be evaluated on both **functionality** and **code quality**.
   a. Functionality refers to correctness of solution, sanity checks, stress tests, robustness, catching edge cases, comprehensiveness in code architecture, conciseness, and performance on the held-out test set (where applicable).
   b. Code quality refers to cleanliness/ readability, well-chosen data structures, program design for optimal efficiency, and clear and meaningful comments existing for *at least* 75% of the lines of code.

9. Please do not hard code any aspects of your model, except model parameters.

10. Please do not use AI to generate your code. For this class, it will count as academic integrity violation. While AI is good at generating text, it is still not good at producing quality code for building ML models. I have observed several students make fundamental mistakes on their ML models because they prompted AI to complete their task. *Most importantly, to know whether AI has given you the correct solution, you should be able to know what the correct solution is.* Using AI to learn how to build AI is a dystopian idea that will harm your learning and mastering AI.

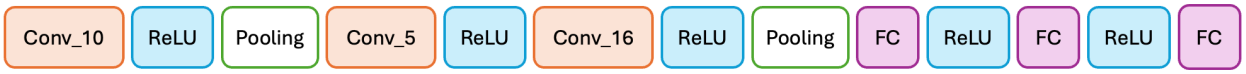
Part I (Max 12.5 points)

Context: Image classification tasks have benefitted significantly from neural networks. In the early days of computer visions, researchers painstakingly crafted features that resembled the real world. However, the advent of neural networks eliminated that effort. With the massive amount of image data available, anyone can build an image classification model using neural networks. Please refer to the lecture slides for more details on why neural networks are a game changer for computer vision applications.

Dataset: The dataset was collected to serve as an alternative to EMNIST which was quite popular, but the only choice. Fashion-MNIST comprises 28x28 grayscale images of 70,000 fashion products, gathered from Zalando, Europe's largest online fashion platform. It has 10 categories, with 7,000 images per category. The training set has 60,000 images and the test set has 10,000 images. The dataset is freely available at this link, and you must download it for this assignment.

Challenge: In this ML challenge, you will implement a lightweight convolutional neural network with the below specifications that can run on your CPU. Your goal is to use the images of various clothing to correctly identify their class. For this assignment, select all 10 categories reported in Table 2 of this research paper. Notice Table 3 of the same paper? It gives you a detailed benchmark of different models and their accuracy measures on this dataset. This is a common practice in ML, and for this assignment, you will familiarize yourself with this practice. You will evaluate your model against this benchmark, and report model performance as compared to other models of different families on the same task.

Goals: Implement a convolutional neural network to classify images from different training sets. The given convolutional architecture is adopted from the CNN architecture we discussed in class but is smaller in depth and width so that you do not need a high-performance GPU to train it. The layers must be stacked in left to right order shown below:



1. Build the model with above architecture and following specifications:
    a. Input size: 28x28 These images have only 1 channel.
    b. Conv_10: Convolution layer with 10 computational units
    c. Conv_5: Convolution layer with 5 computational units
    d. Conv_16: Convolution layer with 16 computational units
    e. Filter/kernel size for convolutional layers: 3
    f. ReLU: Layers with ReLU Activation function.
    g. Pooling: Performs Max Pooling operation and uses a filter size of 2.
    h. FC: This is fully connected layer (nn.linear can do the job for you).

2. For training, use a cross-entropy loss (nn.CrossEntropyLoss) and stochastic gradient descent (torch.optim.SGD) to train the model.
3. Run the model for at least 15 epochs (might take a few minutes depending upon your machine configuration).
4. Feel free to select your own values for batch size and learning rate for the neural network.
5. Generate a graph to show training and validation loss. You may select the validation set from the training set data by splitting it randomly. Do not hard code the sample indices.
6. Evaluate your model using test set. Remember from the lecture that while training the model you need to do both forward pass and backpropagation, however, while testing the model, no backpropagation is necessary.
7. Recall we discussed in class that models like neural network cannot be evaluated using cross validation. Therefore, you will benchmark the model accuracy against the Table 3 in this paper.
8. Report the model performance by showing validation set loss and test set accuracy numbers in percentage.
9. In the report answer the following questions:
    a. Which techniques from Table 3 report **better** performance than your neural network? List all of them as table with their respective accuracy numbers.
    b. Which techniques from Table 3 report **worst** performance than your neural network? List all of them as table with their respective accuracy numbers.

Part II (Max 12.5 points)

Context: Algorithmic decision making has been widely adopted in public sector lately, with promises made by ML organizations that the automation of decision making will make our lives better. For instance, Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) was used to assess risks that a criminal offender will likely commit a crime again. In fact, it was used to compute not just the risk of recidivism but also nearly two dozen so-called "criminogenic needs" that relate to the major theories of criminality, including "criminal personality," "social isolation," "substance abuse" and "residence/stability." Defendants are ranked low, medium or high risk in each category. If you have watched Mindhunter on Netflix, then COMPAS is the outcome of profiles the protagonist in the show creates. Unfortunately, the people were so focused on building the system and incorporating technology into the social fabric, that they did not bother to evaluate it before deployment. ProPublica made a startling discovery in 2016 that the COMPAS was causing more harm than doing good. Check out this article on the expose.

Dataset: For this part of the assignment, you will use the dataset released by ProPublica as part of their "algorithmic expose". The dataset comprises of data from 11,757 individuals who were assigned a "risk of recidivism" score using COMPAS indicating whether the person has a low/medium/high risk of reoffending, and a "risk of violence" score associated indicating whether they have low/medium/high risk of committing a violent crime.

Challenge: In this ML challenge, **the goal is to understand how to measure biases and then mitigate** them**.** First implement a simple logistic regression model that predicts the probability of an individual who is likely to reoffend. **This is your biased model same as built by ProPublica researchers to investigate the issue.** You may assign a binary label of 0 (will not reoffend) when they have low/medium risk of recidivism, and a binary label of 1 (will reoffend) when they have high risk of recidivism score. Build a classification model using logistic regression to find probabilities of an individual being assigned 1 or 0. After you have built the model evaluate model performance on test set (extracted from the same data file). Remember, this model is taking raw data from COMPAS scores, and no bias mitigation strategy is added. Notice that the bias occurred not because you made an active decision to create a biased model, but the bias was in data. By using the sensitive attribute of "race", compute the likelihood of an individual being assigned a "high risk" score as compared to other individuals with similar other attributes. Use the measure of Equalized Odds to compute the bias. If you find that the model is biased, implement a simple sampling strategy to balance the training dataset based on race. **This strategy should lead to a less biased model**. In this strategy, reduce the training set size by selecting a balanced dataset based on sensitive attribute of race. Measure the accuracy of the new model, and the Equalized Odds measure to report if the bias was mitigated. Below are the goals you must accomplish.

Goals: Implement a logistic regression model to classify whether an individual

1. Assign labels to individual samples based on recidivism criteria.
    a. Assign 0 (will not reoffend) when an individual has low/medium risk of recidivism score assigned by COMPAS.
    b. Assign 1 (will reoffend) when an individual has high risk of score assigned by COMPAS.
2. Split the data set 80-20 to get test and train set.
3. Perform necessary operations to handle categorical and continuous data features, using techniques like one-hot coding and normalization. Notice that you have Time as a variable too. Here is a list of techniques you can use to process time as a feature. You may use sklearn to perform feature preprocessing (although it is not at all required and you can easily preprocess your data without sklearn). If you decide to use sklearn, you must store the extracted features in a tensor, and then use pytorch to build your model using these features as described in the next step.
4. Train a Logistic Regression model using training set. Here you must use pytorch to build the model (not sklearn). An example of this was discussed in class before the reading week and is uploaded on Avenue.
5. Measure accuracy of the model, and measure if the model is biased using Equalized Odds (see slides and demos on how to do this). Report these measures in the **pdf and** printing it out on your **console**.
6. Next, change the training dataset by reducing class imbalance; that is, use a small subset of data samples in which all races have near equal class membership based on race (class 1 has the same race distribution as class 0). Do not hard code this, instead, write a function to perform this selection.

7. Retrain the logistic regression model with same parameters, and compute performance and bias. Report these measures again in the **pdf and** printing it out on your **console**.
8. Answer the following questions in your report:
    a. Provide a brief reflection (max 100 words) on model performance and bias measurement, before mitigation strategy and after training the second model with biased mitigation strategy.
    b. As part of **identifying** algorithmic biases in ML models that are trained without any bias mitigation strategy, which racial group had **overall** high likelihood of being predicted to reoffend?
    c. As part of **identifying** algorithmic biases in ML models that are trained without any bias mitigation strategy, which racial group had **unreasonably** high likelihood of being predicted to reoffend for individuals labelled "no recidivism"? Remember that no recidivism is the score predicted by COMPAS and directly input into the model. Unreasonably high likelihood means that COMPAS assigned risk score with "no recidivism" but the logistic regression technique still assigned it high probability of recidivism.
    d. As part of **mitigating** the identified algorithmic biases in ML models, what is the difference in responses to (b) and (c) after training the second model in step 7? Did the accuracy drop, and bias score improve with this technique? Was balancing the dataset able to mitigate any biases you observed?