Homework 1: training and analyzing CNN on FashionMNIST

In homework 1, you will complete the PyTorch code to train a CNN model on FashionMNIST and analyze the impact of training parameters. We will provide a starter code that you can modify. Your homework submission includes (1) the training code for the model you submit, (2) the trained CNN model, and (3) a PDF writeup reporting the model loss/accuracy, and the impact of hyperparameter choices by answering the questions below. You will submit all of these via Gradescope.

Due date: **April 7 (Mon) 11:59pm on Gradescope**. **No late submission**.

As mentioned in lecture 1, each homework is an individual assignment. Thus you are NOT allowed to collaborate with other students in any capacity. Everything related to this homework needs to be done individually.

Setup:

- Download starter file: Canvas -> Files -> Homework ->hw1 -> `hw1_starter.py` or use the link: https://canvas.uchicago.edu/files/13212111/download?download_frd=1

- Install the following packages: `numpy`, `torch`, `torchvision`, `matplotlib`
    - You can use `$pip install package_name` in the environment you create.

- Access to the FashionMNIST dataset
    - We provide the code for accessing the FashionMNIST dataset in `main`.
    - The specific access code differs when you work on the server or on your local machine. So make sure to choose the one that is compatible with your work mode.
    - The FashionMNIST dataset includes a training and a validation dataset.

- Read all the comments in the starter PyTorch code carefully. Even for comments not marked with **TODO**, they contain important information about the functionality of the functions and how you may modify them (such as saved model names, saved plot names etc.).

Part 1 (70 pt):

Complete the training code in `hw1_starter.py` to train a model

- Code to be filled have instructions under **`TODO`**
- You can explore image transformations and training parameters to train the model
- You will submit a model with the name `model.p` to Gradescope. Our grading system **does not** accept models with any other name.
- Your model will be graded on the validation dataset with transformations (`RandomHorizontalFlip` and `RandomVerticalFlip`).

    - Evaluate on the original validation data (40 pt)
        - If accuracy > 70%: 40pt
        - Else if accuracy > 60%: 35 pt
        - Else if accuracy > 50%: 30 pt
        - Else if accuracy > 40%: 20 pt
        - Else if accuracy > 30%: 10 pt
        - Else if accuracy > 20%: 5 pt
        - Else 0 pt
    - Evaluate validation dataset with `RandomHorizontalFlip` (15 pt)
        - If accuracy > 60%: 15 pt
        - Else if accuracy > 50%: 12.5 pt
        - Else if accuracy > 40%: 10 pt
        - Else if accuracy > 30%: 5 pt
        - Else: 0 pt
    - Evaluate validation dataset with `RandomVerticalFlip` (15 pt)
        - Rubric is the same as above.
- You will see the points you get on this part every time you submit to Gradescope
- We will run the training code you submit after the due date
    - Your final submitted training code should correspond to the final model you submit.
    - If we run your training code and it has substantial difference from the model you submit, this will result in **violation of academic integrity**.

Part 2 (30 pt):

Analyze model performance: write a report to answer the following questions. At this point, you should have already trained a model (`model.p`) that can classify any image from FashionMNIST. This will be your **original model**.

Q1: The starter code includes a module to plot the loss-accuracy trend. Paste this loss-accuracy plot of your **original model** to your report. Describe the plot on how training loss and validation accuracy change with training epochs in your **original model**. (10 pt)

Q2: Describe the transformation you applied to training data and explain the purpose of this data augmentation step in your **original model**. (5 pt)

Q3: Change the batch size to be ¼ of the batch size in your **original model**, with all the other training parameters the same as in your **original model**, train a new model. We will call this model M2. How does M2's performance compare to your **original model**? (5 pt)

Q4: Modify the learning rate of M2, and train a new model M3 such that M2 and M3 only differ in learning rate. (M3 has a batch size that is ¼ of your **original model**.) What change to learning rate do you find so that M3 achieves a comparable performance to your **original model**? (5 pt)

Q5: When you change the batch size to be ¼ of the batch size in your **original model**, what change to training time of each epoch do you notice? (5 pt)

Bonus Q: In the architecture of `FashionCNN`, the first linear layer is `nn.Linear(in_features=16*4*4, out_features=120)`. Explain why in_feature is 16*4*4. (5 pt)