Assignment #3

This assignment can be completed individually or by a team (maximum of 3 members).

Total score: 150

Due date: refer to the Canvas page

Objectives

Deep learning for image classification and object detection

Required activities

Complete the following tasks as specified in each question, strictly following the instructions, and write a brief report. The report should include:

- Team member names, optionally specifying each member's percent contribution or stating "*Everyone contributed equally*" if applicable. If your team does not reach an agreement on individual contribution, briefly write a task description for each member. Different grades may be assigned based on individual contributions.
- Modeling results and answers to any questions asked in each task

You may reuse publicly available source codes from the Internet or get help from Large Language Model; however, sharing your code with other teams or students in the class is strictly prohibited. Any student or team violating this policy will receive a zero for this assignment and may face penalties on all the remaining assignments.

Only Python programs written using Python 3.0 or higher will be accepted. **NO Jupyter notebook or any other Python variant will be accepted** for efficient grading.

1. Simple deep learning exercise using CNN

- Modify the PyTorch program "Assignment3_digit_CNN.py" (available on the course page) for digit recognition to improve the CNN model by adding more layers and/or adjusting parameters to achieve 100% test accuracy, and save the trained model as "improved_digit_model.pth".
- Create three hand-written digit images: "digit3.jpg" (for the number 3), "digit4.jpg" (for 4), and "digit5.jpg" (for 5). Use the pretrained model "improved_digit_model.pth" to predict the digits in these images and print the results in the format:
  "Prediction for digitX.jpg: X", where X is the predicted digit

2. Image classification using CNN

Download a dog dataset “Assignment3_dogs.zip” (available on the course page), split it into a training set (80%) and a test set (20%).
Design your own CNN model for classifying dog breeds, providing a brief justification for your design choices, including the number of layers, number of filters, use of skip connections, use of batch normalization, and any other architectural components you included.
Write a PyTorch program “**dog_classifier.py**”, to train your model and print the following modeling results:

- Total training time
- Classification accuracy

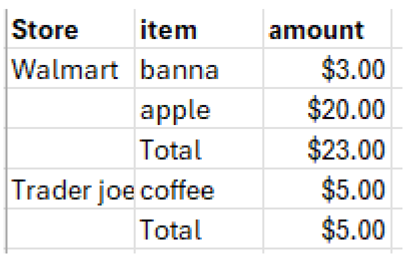
3. Image classification using a pretrained model

Write a Pytorch program “predict_compressed_object.py” that performs the following tasks:

- Adding Gaussian noise $\boldsymbol{\varepsilon} \sim (\boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{1})$ to the image I = “Assignment3_dog_to_compress.jpg”, creating a noisy image $\boldsymbol{I'} = \boldsymbol{I} + \boldsymbol{\varepsilon}$
- Compress the noisy image $\boldsymbol{I'}$ using an (pretrained) Autoencoder, where $z = f(I')$
- Reconstruct the image $\hat{x}$ from the compressed representation $\hat{x} = g(z)$
- Classify the reconstructed image $\hat{x}$ using the pretrained “ResNet” model (without fine-tuning)
- Print the predicted class.

4. Text extraction from images

Write a Python program “**text_extraction.py**” that extracts text information from receipt image files “Assignment3_receipts.zip” (available on the course page) and generates a CSV file, “shopping_summary.csv” using the pretrained OCR model “Tesseract OCR”. The CSV file should include three columns: store name, item name, and amount, along with the total amount spent at each store. A sample table is provided for reference.



Store	item	amount
Walmart	banna	$3.00
	apple	$20.00
	Total	$23.00
Trader joe	coffee	$5.00
	Total	$5.00

5. **Object detection**

Write a Pytorch program “**object_detection.py**” to detect and recognize all objects in the image “Assignment3_street.jpg” using two pretrained models: “**YOLO**” (any YOLO model) and “Detectron2” (based on faster R-CNN). Also count the number of people in the image “Assignment3_people.jpg” using each model.

- For object detection and recognition, the program should print a list of the actual object names (labels) in the image, a list of the predicted object names, and their corresponding locations (coordinates).
- For object counting, the program should print the actual number of people and predicted number.

Briefly describe the architecture of each pretrained model used for the above tasks, compare the results in terms of accuracy and speed of detection, and recommend one of the models for the tasks of object detection and recognition, providing a brief justification.

What and how to submit

- One report in Word or PDF format per team
- Only the program file(s) created by the team (excluding any third-party packages).
- Upload each file separately. DO NOT submit a ZIP file as Canvas cannot open ZIP files.

Grading criteria

- 90% based on the overall quality of work, including the modeling process and results, code implementation, the depth of understanding demonstrated in the report
- 10% based on the effort reflected in the report and program development