Assignment #4: Deep Learning and Computer Vision

Total score: 100

This assignment can be completed individually or by a team of a maximum of three members.

Due date: See the course page

1. **Simple Deep Learning Exercise using Convolutional Neural Network (CNN) (35 pts)**

(a) Testing the CNN program for digit recognition and GPU usage

- On the Assignment #4 page, you'll find a PyTorch program located at "**L04_digit_CNN.py**" (see Assignment4.zip) that performs digit recognition using both the CPU and GPU. To force CPU usage, remove the comment symbol "#" from line 10 of the code and comment out the line above it. If your local machine does not have a GPU, you may use the **JupyterLab** as discussed in class by selecting the container image: "NRP Deep Learning & Data Science Full, PyTorch (CUDA)".
- Print the classification accuracy and training time for both CPU and GPU execution. Briefly explain any differences in the execution results and the reasons for the differences.
- Modify the network hyperparameters so that the GPU execution becomes faster than the CPU execution and show the results demonstrating a case where the GPU speeds up the training. **Take a screenshot of your code output from this bullet point, put it in your report with a detailed explanation for how you got the GPU to be faster than the CPU, what you changed and why the GPU solution was faster than the CPU. (5 pts)**

(b) Enhancing the image classification model

- There's another script uploaded called "**improved_digit_cnn.py**". Improve the model to achieve at least 99% test accuracy. Save the trained model as "**improved_digit_cnn.pth**".

(c) Handwritten digit prediction

- I've uploaded hand-written digits by using Microsoft Paint with the following names: "**digit2.jpg**" (showing the number 2), "**digit4.jpg**" (number 4), "**digit6.jpg**" (number 6) and "**digit8.jpg**" (number 8).
- Write a Pytorch program, "**predict_my_digits.py**", that loads the pretrained model "improved_digit_cnn.pth" created in (b), predicts the digits in the three images, and prints the prediction results in the format: "Prediction for digitX.jpg: X", where X is the predicted digit. **Take pictures of handwritten digits you wrote out. Were you able to predict the**

handwritten digit the same way as the Microsoft Paint digits? Explain why this did or didn't work for you and what you could implement to improve training accuracy for different types of handwritten digit pictures (Hint: does RGB-style images have anything to do with accuracy? If so, how and what could you do to standardize these images?). (5 pts)

2. **Text extraction from images (25 pts)**

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

Write a Python program "**text_extraction.py**" that extracts text information from receipt image files in the "**/datasets/receipts**" directory (see the Assignment4.zip file) and generates a CSV file, "**shopping_summary.csv**" using the pretrained OCR model "Tesseract OCR". The CSV file should include three columns: store (not uppercased as the screenshot depicts), item, and amount. Every receipt should have a "Total" value under the "item" column to tally up the sum of all prices under the "amount" column. A sample table is provided for reference.

Note: You very likely won't be able to obtain the exact output format present in the screenshot; it's okay if your code has noise in the output. I'll be looking to make sure your output contains items and prices from the receipt that match with the original picture. **Why do you think it would be that the exact output is so hard to extract? What additional preprocessing steps could you utilize to remove the additional noise in the output? If a client of yours was looking for high accuracy, would you recommend this approach? Why or why not? (5 pts)**

3. **Image classification (25 pts)**

Write a PyTorch program "**animal_classifier.py**" that loads and (optionally preprocess) the dataset "animals.zip" (available on the course page), classifies the animal using a pretrained CNN model "ResNet", and prints the following information:

- Basic information about the pretrained model, including the number of layers, number of filters, and any other architectural components not discussed in class
- Total classification time
- Classification accuracy

Note: this program needs to output the model as "**animal_classifier.pth**" which is apart of your submission

What problems did you run into while doing this? Was the model able to accurately predict the animals? Did you run into any issues when trying to use the folder name as the prediction label and if so, why? (5 pts)

4. **Object detection (15 pts)**
   - I uploaded a file under "**/datasets/objects.jpg**" containing three objects for classification.
   - Write a Pytorch program "**object_detection.py**" to detect all objects in the image using the pretrained "YOLO" object detection model. The program should print a list of object names (labels) actually present in the image, a list of predicted object names, and the predicted bounding box coordinates for each detected object.
   - **Briefly discuss the detection results and evaluate the model's performance in the report. Was your output correct? If not, what did you have to do to ensure the model predicted objects accurately? List any additional preprocessing steps (or lack thereof) you needed to use for the model to predict all three objects accurately. (5 pts)**

What to submit

- A report file in word or PDF format that includes the name or names of the team members, and the % contribution of each member. If there is no agreement on individual contributions, briefly describe the tasks assigned and completed by each member. Different scores may be awarded based on individual contributions. If all members contributed equally, simply state "equal contribution." The **report** should also include answers to the **bolded** questions in each section and screenshots demonstrating the functionality of each program.
- **The following files** (**DO NOT** submit any zip file as Canvas cannot open them nor do you need to submit any unit test file [i.e. any file beginning with "test_<script name>.py"]).
  - **Scripts:** L04_digit_CNN.py, animal_classifier.py, improved_digit_cnn.py, object_detection.py, predict_my_digits.py, text_extraction.py
  - **Model Images:** animal_classifier.pth, improved_digit_cnn.pth

Grading criteria

- Does each program successfully perform the required tasks and generate the correct results?
- Does the report demonstrate that the team has a clear understanding of the relevant concepts, programming techniques, functional requirements? Does it show how the team applied this knowledge to successfully complete the tasks? Did the team demonstrate understanding by answering the additional questions in **bold** in each section?
- Is the effort put into the project clearly reflected in both the report and the program files?