Assignment 5: *Convolutional Neural Networks*

Date: *Mar 27, 2025*
Due : *Apr 8, 2025*

Acknowledgment This assignment has been adapted from the materials of ECE1513 by S. Emara.

Code of Honor Assignments are designed to enhance your understanding and advance your skills, constituting a significant portion of your final assessment. They must be completed individually, as engaging in any form of academic dishonesty violates the principles of the Code of Honor. If you encounter any challenges while solving the assignments, please contact the instructional team for guidance.

How to Submit This assignment has 2 questions. For each question, you need to upload one file on Crowdmark. All questions ask for your code. You need to copy and paste your code in the file that you submit or print your code and its output terminal as PDF.

Grading The grades add up to 100 and comprise roughly 10% of the final mark.

Deadline The deadline for your submission is on **Apr 8, 2025 at 11:59 PM**. Please note that this deadline is strict and **no late submission will be accepted.**

Questions

Question 1 [60 Points] **(Training a Basic CNN)** In this problem, we train a convolutional neural network (CNN) using PyTorch. We work with the CIFAR-10 dataset, which contains 60,000 (32×32)-pixel RGB images of common objects. The images are represented with $32 \times 32 \times 3$ tensors.

1. Load the dataset. The training, validation and test datasets should be transformed from Python Image Library (PIL) image to a tensor in the range $[0.0, 1.0]$. Then, the values should be normalized to be in the range $[-1.0, 1.0]$. Mean and standard deviation can be set to 0.5 to aid this transformation. Recall from Assignment 4 that this can be done using

   `torch.transforms.ToTensor()`

   and

   `torch.transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))`

   when you load the dataset. Split the loaded dataset into the training, validation and test datasets of sizes 40,000, 10,000 and 10,000, respectively.

2. Implement a CNN with two convolutional layers and two fully-connected ones in PyTorch. The first convolutional layer receives 3 input channels for the image, and output 6 channels using 5×5 filters. The second convolutional layer returns 16 channels using 5×5 filters. We apply pooling after each convolution using max-pooling with 2×2 filters and stride 2. The output of the last

convolutional unit (after pooling) is passed through two fully-connected layers of width 120 and 84, respectively. The output layer should produce the probability of the 10 classes in CIFAR-10. For activation use the ReLU function.
Hint: *You can implement all these layers directly using the classes in* `torch.nn` *module.*

3. Using cross-entropy as the loss and mini-batch stochastic gradient descent (SGD) with batch-size 4 and learning rate 0.001, train the CNN for 35 epochs. Plot the training accuracy and validation accuracy vs. number of epoch.

4. Output the test accuracy of the model *on each class*.

QUESTION 2 [40 Points] **(CNN Architecture)** We now train modified versions of the model in Question 1 to understand the impact of model architecture.

1. Modify the model in Question 1 by removing the last fully-connected layer, and train it with the same hyperparameters. In less than 4 sentences, give your insights to justify the difference between the result and what you observed in Question 1.

2. Modify the model in Question 1 by increasing the number of neurons in the first fully-connected layer to 1000, and train it with the same hyperparameters. In less than 4 sentences, give your insights to justify the difference between the result and what you observed in Question 1.

3. In one sentence, comment on if *early stopping would help improve the generalization* of the models trained in Questions 1 and 2.