CSCI-GA 3033: Vision Meets ML

Assignment 1: CNN from Scratch

Introduction

This assignment gets you hands-on with:

- **Part 1:** Image Filtering and Gradient Operators
- **Part 2: Building a tiny Convolutional Neural Network stack** (no autograd!) to understand how the PyTorch framework works under the hood.

Learning Objectives

- Explore Image filtering and gradient operators.
- Implement forward and backward passes for core CNN primitives: Conv2D, MaxPool2D, ReLU, Linear, Cross-Entropy

Part 1: Image Filtering and Gradient Operators [30 Pts]

In this part of the assignment, you will work with image filtering and edge detection using the provided Jupyter notebook (`Part1_Filtering.ipynb`). You are expected to apply different padding techniques, implement convolution operations from scratch, and experiment with gradient-based edge detection using 1D filters. Use the given images (`zebra.png`, `cameraman.png` and `cameraman_median.png`) as instructed in the questions. Follow the instructions in the notebook, display all results, and include brief written analysis (1–2 lines) in text cells where required.

Make sure that the outputs for every section of the assignment are visible when you submit.

Note: You can complete this part of the assignment on Google Colab.

Part 2: Build CNN from Scratch [70 Pts]

Goal: Implement the core layers and train a small CNN on MNIST.

Task:

Write a minimal DL stack (pure PyTorch tensors; no autograd on params):

- **Layers**: `Conv2D`, `Linear`, `MaxPool2D`, `ReLU`, `Flatten`
- **Loss**: `CrossEntropy`
- The `Sequential` class that chains multiple layers together
- The **SGD** training loop

Tests:

A provided test script checks:

- Numeric gradient sanity for layers & loss
- Forward shape checks
- Tiny end-to-end training sanity (loss drops)

You must pass **all the tests** and demonstrate a brief MNIST training run.

Deliverables

Please submit a **single zip file** as your final submission that contains the following deliverables:

- **Part 1:** Completely implemented Jupyter notebook with all the **outputs visible**.
- **Part 2:**
  - **`cnn_from_scratch.py`** script with your implementation and a tiny MNIST training run.
  - **Plots**: Plots of training curves and final test set evaluation results.
  - **Screenshots** of results of executing `tests.py` scripts and completed `cnn_from_scratch.py`.
    - * You can either submit the output file resulting from running a .SBATCH script or even the screenshot of execution on the cloud burst's compute node works.