GNR638: Machine Learning for Remote Sensing - II

(Spring 2025-26)

Programming Assignment: Design a Deep Learning Framework

1 Introduction

This programming assignment requires designing and implementing a custom deep learning framework from scratch and using that framework train a convolutional neural network (CNN) for multiclass classification on the provided datasets.

You must first build your own framework and then use only that framework in the training and evaluation code for submission. The use of existing deep learning frameworks is strictly prohibited. Also, using any additional data apart from the provided ones are strictly prohibited.

Please use your IITB email ID to access any of the links provided.

2 Learning Outcomes

Upon successful completion of this assignment, students will be able to:

- Design and implement a basic deep learning framework from first principles
- Build and train convolutional neural networks without relying on external deep learning libraries
- Integrate C++ backends with Python frontends
- Understand and apply automatic differentiation and backpropagation
- Train an image classification model
- Analyze performance trade-offs between Python and C++ implementations
- Analyze and report model complexity in terms of parameters, MACs, and FLOPs

3 Assignment Requirements

3.1 Broad Implementation

You must implement a custom deep learning framework that supports:

- Tensor abstractions with gradient tracking and backpropagation
- Core mathematical operations required for neural networks
- Convolutional layers, activation functions, pooling layers, and fully connected layers
- Loss computation and gradient-based optimization

The framework must be importable and usable from Python 3.12. You are strongly encouraged to implement the computational backend in C++ for performance reasons; however, a Python backend is permitted. The use of existing deep learning frameworks or automatic differentiation libraries is strictly prohibited.

3.2 Dataset Loading

You have to download both the datasets from the folder Assignments 1 Datasets from the provided link `https://drive.google.com/drive/folders/1mX8kaByeedwpp9-4-enuJffGChgmdy9n?usp=drive_link`. The datasets provided is in an image-folder format, i.e., parent folder consists of multiple subdirectories, each containing images in PNG format. Provided the parent folder, you are required to:

1. Load images from the folder
2. Convert them to tensors of size **32x32** using your own framework
3. **(Optional)** Apply data augmentation of your choice
4. Infer labels from the folder names
5. Support batching during training and evaluation

You must measure and report the dataset loading time (in seconds) when reading each dataset from the disk. This value must be printed during training as well as evaluation and clearly documented in the report.

3.3 Model Requirement

After loading the dataset, you are required to build a convolutional neural network (CNN) model that must contain at least:

- One convolution layer
- One activation function
- One pooling layer
- One fully connected layer

You must calculate **number of parameters**, **MACs** and **FLOPs** of your network. The efficiency metrics should be printed during training, evaluation and should be clearly documented in the report.

All layers and operations must be implemented from scratch and used through your own framework. You may compose these layers in any valid configuration to form a CNN architecture and include any additional layers or operations of your choice, provided all the constraints in this assignment are satisfied.

3.4 Evaluation

Hidden test datasets will be used for evaluation. The test datasets will follow the same directory structure as the training datasets and will contain fewer samples than the training datasets.

3.5 Constraints

- For each dataset training must complete within 3 hours on the evaluation machine
- For each dataset valuation must not take more than 1 hour on the evaluation machine
- The training script must use only the student's framework and allowed libraries listed in Section 4

4 Languages and Tools

- **Frontend API:** Python 3.12
- **Target OS:** Linux, MacOS & Windows 11
- **Backend Language:** C++ or Python (either may be used)

S.No.	Assignment Component	Weight (%)
1	Successful training and evaluation of your network	25
2	Report	10
3	Reproducibility of training and evaluation results	20
4	Beating the chosen benchmark	10
5	Listing down all the sources used for assignment	5
	Total	**70**

Table 1: Grading Breakdown

You may use standard tools for Python bindings, such as pybind11, ctypes, or cffi and only the standard libraries provided with the chosen programming language and their full functionality. In addition, OpenCV (Python or C++) is permitted exclusively for image loading from disk, and its usage must be strictly limited to image reading and basic image processing. No other third-party libraries may be used. In particular, the use of deep learning frameworks (e.g., PyTorch, TensorFlow, JAX, Keras), automatic differentiation libraries, numerical or scientific computing libraries (e.g., NumPy, SciPy), model analysis or profiling tools, or any equivalent or analogous libraries in C++ or other languages, as well as any third-party library not included in the language's standard distribution, is strictly prohibited in all parts of the assignment. Violation of these restrictions may result in disqualification of the submission.

5 Evaluation & Grading Strategy

Submissions will be evaluated based on correctness, completeness, reproducibility, and compliance with all assignment constraints. A detailed breakdown is listed in Table 1. You must submit a GitHub repository link containing the complete, working codebase. The repository must include clear and precise instructions describing how to:

- Build the framework (if applicable)
- Run the training script
- Run the evaluation script

At the time of training, the grader will provide only the path of the training dataset along with the path to the model configuration file. For evaluation, only the parent directory path of the hidden test dataset would be provided along with the path of saved weights after training. The evaluation script must be able to run using only these inputs and must not require any code modifications.

In addition to the code, you must submit a written report that clearly documents:

- Model architecture design and rationale
- Total number of trainable parameters
- MACs and FLOPs per forward pass
- Dataset loading time
- Training and validation performance metrics across epochs
- Associated plots for the logs (such as loss, training accuracy etc.)
- Any additional key indicators relevant to the implemented architecture and assignment components
- Also, explain one failed design decision and why it did not work

Failure to report the required metrics or omissions in documentation may result in loss of marks. If the submitted code fails to run as per the provided instructions, or cannot be evaluated using the supplied test directory and weights, the submission will receive a grade of 0. You must also cite all the resources used for doing this project, including any AI tools and follow the honour code as mentioned on the course website.

6 Bonus Performance Marks:

Up to 50 bonus marks may be awarded based on:

- C++ backend implementation **(20 points)**
- Efficiency in terms of data loading, memory consumption and model complexity vs performance **(10 points)**
- The best performing model in image classification provided the constraints **(10 points)**
- Any key insights into the working of the model **(10 points)**