

**CS 203: Software Tools & Techniques for AI**  
**IIT Gandhinagar**  
**Sem-II - 2024-25**

---

**LAB 06**

**Lead TA: Section 1- Eshwar Dhande; Section 2- Himanshu Beniwal**

**Total marks: 100**

**Submission deadline: Tuesday, 25/02/2025 11:59 PM**

**Submission guidelines:**

1. Code should be added to a GitHub repository, and the **repository details should be shared in the pdf.**
2. **Submit the PDF showing screenshots** of all steps involved in the following code.

**Note:** By submitting this assignment solution you confirm to follow the IITGN's honor code. We shall strictly penalize the submissions containing plagiarized text/code.

---

**Objective:**

The goal of this assignment is to learn about experiment tracking, version control, and reproducibility in machine learning workflows. You will set up experiment tracking using **Weights and Biases**.

---

**Section 1: MLP Model Implementation & Experiment Tracking**

**1. Implement a Multi-Layer Perceptron (MLP) Using the Iris Dataset (05%)**

- Load the **Iris dataset** using `sklearn.datasets.load_iris`.
- Extract features and labels, ensuring labels are one-hot encoded.
- Split the dataset into:
  - **Training set:** 70%
  - **Validation set:** 10%
  - **Testing set:** 20%.
- Normalize feature values to **[0,1]** using standard scaling.

## 2. Define and Train the MLP Model (05%)

- Construct a **Multi-Layer Perceptron (MLP) model** with the following architecture:
  - **Input layer:** 4 neurons (for 4 features).
  - **Hidden layer:** 16 neurons, **ReLU** activation.
  - **Output layer:** 3 neurons (for each class), **softmax** activation.
- Train using:
  - **Loss function:** Categorical cross-entropy.
  - **Optimizer:** Adam.
  - **Learning rate:** 0.001.
  - **Batch size:** 32.
  - **Epochs:** 50.
- Track and store both training and validation loss during training.

## 3. Evaluate Model Performance (10%)

- Compute and store the following metrics using the **test set**:
  - **Accuracy**
  - **Precision**
  - **Recall**
  - **F1-score**
  - **Confusion matrix** (visualized using Matplotlib).
- Plot training and validation loss curves over epochs using Matplotlib.

## 4. Set Up Experiment Tracking with Weights & Biases (W&B) (30 %)

- Log the following details:
  - **Model architecture:** Number of layers, neurons, activation functions.
  - **Hyperparameters:** Learning rate, batch size, number of epochs.
  - **Training and validation loss per epoch.**
  - **Final evaluation metrics.**
  - **Confusion matrix and loss curve visualizations.**

## 5. Submission Requirements

- Submit:
  - **Python code** for training, testing, and evaluation.
  - **Screenshots of the W&B dashboard** displaying:
    - Model architecture.
    - Hyperparameters.
    - Logged metrics.
    - Final evaluation results.
    - Confusion matrix visualization.
    - Training and validation loss curves.

---

## Section 2: Hyperparameters

This section aims to perform a hyperparameter search to improve the performance of a **custom model** that distinguishes between any three classes.

### Task 1: Hyperparameter Optimization (20%)

- Use the model trained in the previous section.
- Train the model on the batch size of [2 & 4], learning rate [1e-3 and 1e-5], and epochs [1, 3, and 5].
- Train the model and measure the accuracy and F1 over the test set. Plot the confusion matrix over the test-set predictions.
  - Plot using the truth labels and predicted labels in matplotlib.
- Show the inputs, prediction, and truth values for five samples from the test set.

### Task 2: Automated Hyperparameter Search (20%)

- Use the Grid Search over the parameters defined above, [Random Search](#), and [Hyperband + Bayesian Optimization hyperparameter](#) to search for the hyperparameters defined in Task 1.
- Create a table (Each row with a configuration and column with Accuracy and F1) for Grid, Random, Hyperband, and Bayesian search and compare their accuracy and F1.

## Evaluation Criteria

### Perform hyperparameter optimization using AutoGluon

- Plot the scatter plot for training vs validation loss.
- A relation (direct or inverse) between the hyperparameters and their impact on the performance (Hypothetically, epoch is directly proportional to performance, but batch size is inversely proportional).
- Describe the performance for each hyperparameter combination over accuracy and F1.

### Compare manual tuning vs. automated search

- Which approach is better and why? (At most five lines of explanation)
- Plots for the training vs validation loss for each hyperparameter configuration.

## **Documentation (10%)**

- Proper documentation of code, graphs, and metrics.