Inmind Academy

Computer Vision Project

Edition: 1

Issuing Date: 20 March 2025

inmind .ai | contact@inmind.ai | +961 03 500 166 | Mkalles, Main Road, Bldg. 757, Lebanon

Table of Contents

GUIDELINES

This project focuses on object detection tasks in computer vision. The deadline for submission is **Friday, March 28, 2025**. The presentation will take place during the academy session.

Students are required to create a private GitHub repository and share it with the instructor (view access) by **March 23, 2025**. They must develop their solution on GitHub and continuously update the repository. Commits made after **March 28, 2025, at 2:00 PM** will not be evaluated.

Students who are unable to attend the presentation must contact the instructor before **March 27, 2025**, to schedule an alternative meeting.

The cleanliness of GitHub commits will be considered in the evaluation. Students should keep their commits neat and well-structured by following consistent best practice. Avoid large, disorganized commits.

Plagiarism will not be tolerated.

ASSIGNMENT DESCRIPTION

The dataset consists of a collection of labeled **tuggers, cabinets, STRs, boxes, and forklifts** in the **BMW JSON format.**

You are required to:

PART 1: DATA PREPERATION & VISUALIZATION

1. Load the dataset and labels using **PyTorch DataLoader**.
2. Write a function to **visualize** some of the labeled images.
3. Explore the possibility of **augmenting the dataset** using **Albumentations**.
4. Split the training dataset into **train** and **validation** sets.

PART 2: MODEL TRAINING & EVALUATION

1. Train an **object detection model** based on **YOLOv5**.
2. Evaluate the model on the **testing dataset**, then **relaunch training with different hyperparameters**
3. Use **TensorBoard** to **visualize the evolution** of both models' metrics during training.
4. Compare the performance of both models by analyzing metrics such as **accuracy, IoU, training time, inference speed, and hyperparameters**.

PART 3: MODEL DEPLOYMENT & INFERENCE

1. Export both models into an **inference-friendly format** (e.g., **ONNX** or **OpenVINO**).
2. Visualize both networks using **Netron**.
3. Create an **Inference API** using **FastAPI** that runs on **CPU**, supporting the following endpoints:
    a. **Model listing endpoint**: Returns a list of available models.
    b. **Inference endpoint (JSON output)**: Accepts an image and returns detected bounding boxes with class labels and confidence scores.
    c. **Inference endpoint (image output)**: Accepts an image and returns the same image with bounding boxes drawn.
    d. *(Optional)* **Dockerize the API** into a runnable container.