Assignment: VisDrone-MOT Real-Time Object Detection and Tracking

1. Overview

Build a real-time object detection system using the VisDrone-MOT dataset, and evaluate its performance on video data. You may optionally extend your work with multi-object tracking to maintain consistent identities across frames.

2. Introduction

Develop a reproducible detection system using PyTorch or a PyTorch-based framework. Your code should reflect good engineering practices and be easy for others to reproduce and extend.

3. Minimum Requirements

This is an open-ended assignment intended to evaluate your ability to design and deliver a practical detection system with minimal guidance. You are free to make any technical and architectural choices, but your submission must meet the following minimum requirements:

Detection System: A working (and fairly tuned) object detector trained on the VisDrone-MOT dataset, along with a quantitative evaluation and a brief qualitative analysis of results.

Reproducibility: A clear setup and run guide (`README.md`) that allows another engineer to reproduce your results.

Report: A concise technical summary describing: (a) model design, (b) training setup and model selection, (c) experiments and key findings.

4. Object Detection

Train and evaluate a real-time-capable object detector on the VisDrone-MOT dataset. You are free to choose your model, training setup, and evaluation methodology. Document your reasoning, trade-offs, and key design decisions. Provide train and eval scripts that reproduce the results you claim.

Model Selection & Explainability

Evaluate and compare different model configurations or architectures, and justify your final selection. Your process should reflect how you would approach model selection in a production setting — balancing accuracy, robustness, and real-time performance. Clearly explain the reasoning behind your chosen model, weights, and inference parameters.

Qualitative Analysis Examples

Include a qualitative analysis that illustrate your model's strengths and weaknesses on representative examples.

5. Tracking (Bonus)

You may enrich your system by extending it with multi-object tracking, combining detection with a tracking component to maintain track ids. You may use a traditional computer vision approach or a ML-based tracker algorithm. Integrate tracking into your existing detection pipeline. Provide a short evaluation of the combined (detector + tracker) system, both quantitative and qualitative.

6. Demonstration & Reproducibility

Provide a minimal visualization (playback) tool that displays the system's detections on a video sequence using your detector. If a tracker is also implemented, the visualization should support running the combined detector–tracker system for a complete perception playback.

7. Runtime Requirements and Deployment

Make sure that you document your runtime environment and GPU setup clearly enough to reproduce inference. Also, include a lightweight, reproducible inference setup (e.g. Docker container). Your system should be runnable with minimal configuration.

8. Deliverables and Submission

Your deliverable should be organized as a GitHub repository containing:

- All source code, configs, and scripts needed to reproduce your results.
- a `README.md` with environment setup, training, evaluation and inference instructions, as well as usage of the visualization module.
- A report (PDF) summarizing your approach, experiments and key findings.