AI/ML Engineer Test

TASK1: Object Detection with YOLOv8+

Goal

You have to **train, optimize, and deploy** an object detection model using **YOLOv8+** on a dataset of your choice. This test evaluates your ability to work with object detection pipelines, preprocess data, train models, optimize inference, and deploy an API.

Problem Statement

You have to **train an object detection model** using YOLOv8 (or a newer version) on a dataset of your choice. You can choose:

- A **custom dataset** (prepared by you)
- An **open-source dataset** (e.g., COCO, Pascal VOC, GTSDB, etc.)

You have to **document your choices, reasoning, and steps** in a report and deploy a real-time inference API using **FastAPI, Django, Flask, or any other framework**.

Task Breakdown & Scoring (100 Points)

Task	Description	Points
1. Dataset Selection & Preprocessing	Choose and justify a dataset. Convert it to YOLO format if needed. Apply augmentations.	**20**

2. Model Training	Train a YOLOv8 model. Log training metrics (mAP, loss). Justify hyperparameter choices.	**30**
3. Model Optimization	Convert model to ONNX/TensorRT for faster inference. Benchmark performance.	**20**
4. API Deployment	Deploy an inference API using **FastAPI/Django/Flask**. Return JSON results.	**20**
5. Documentation & Report	Provide a structured report on dataset, preprocessing, training, optimization, and deployment.	**10**

Submission Guidelines

You have to submit the following files in a structured repository (GitHub/GitLab).

1. **Code Repository (GitHub/GitLab)**
    a. Well-organized repository with README. Make sure you write proper commit messages.
    b. If you want to make the repo private, it's fine. We'll ask you for the access after you submit.
    c. Clearly structured folders (e.g., `data/`, `models/`, `api/`)
    d. Requirements file (`requirements.txt`)
2. **Trained Model & Weights**
    a. Submit the final trained model (`.pt` or `.onnx`)
    b. Include a link if hosted on cloud storage (Google Drive, S3, etc.)
3. **Deployment Code**
    a. Provide the complete API implementation
    b. Ensure API endpoints return **JSON responses**
4. **Documentation & Report (Markdown/PDF)**

You have to include:

- **Dataset selection & preprocessing steps**

- **Model training setup, hyperparameters, evaluation**
- **Optimization details (ONNX/TensorRT)**
- **API design, testing, and usage**
- **Challenges faced & improvements**

💡 Bonus Points Work (For Extra 10-20 Points)

- **Live Webcam Inference (10 Points)**: Implement real-time inference via a webcam.
- **Cloud Deployment (10 Points)**: Deploy the model as a REST API on AWS/GCP.

Task 2: Dockerized Jupyter Lab with GPU Access

Goal

You have to create a **Docker image** based on **Ubuntu** that provides a fully configured environment for deep learning. The running container must have **access to all available GPUs** and include **Jupyter Lab** with **TensorFlow/PyTorch** preinstalled. When the container starts, Jupyter Lab should be **accessible over the correct IP** with GPU support enabled.

Task Breakdown & Scoring (100 Points)

Task	Description	Points
1. Dockerfile Creation	Build a Dockerfile using Ubuntu as the base image.	**15**
2. GPU Support	Ensure the container has access to all available GPUs using NVIDIA Docker.	**15**

3. Jupyter Lab Setup	Install and configure Jupyter Lab to launch automatically on container startup.	**15**
4. Deep Learning Frameworks	Preinstall **both TensorFlow and PyTorch** with GPU support.	**15**
5. Correct Network Configuration	Ensure Jupyter Lab is accessible over the correct IP and port.	**15**
6. Docker Hub Deployment	Push the final Docker image to **Docker Hub** and make the repository public.	**15**
7. Run From Docker Hub	Pull and run the image from Docker Hub to verify it works.	**10**

Submission Guidelines

You have to submit the following:

1. **Docker Image on Docker Hub**
    a. Publish your Docker image on **Docker Hub**.
    b. Ensure your **Docker repository is public**.
2. **Build & Run Instructions**
    a. Provide commands to **build** and **run** the container locally.
    b. Provide the **Docker Hub pull command** to run the container from the cloud.
3. **Verification Proof**
    a. **Docker logs or screenshots** showing:
        i. Jupyter Lab running and accessible
        ii. GPU detection inside the container (`nvidia-smi`)
        iii. Successful execution of the container from Docker Hub

---

💡 Bonus Points Work (For Extra 10-20 Points)

- **Container Persistence (10 Points)**: Allow users to mount external storage for Jupyter notebooks.
- **Multi-User Support (10 Points)**: Enable multiple users to access Jupyter Lab securely.

Deadline & Evaluation Criteria

- Estimated time to complete these tasks is **30-35 hours** but you have **10 days** to complete this test.
- Your submission will be evaluated based on **correctness, efficiency, security, and clarity.**
- Late submissions may not be considered unless prior approval is given.