**Name:** Suyog Bodke  
**Roll No:** 381006  
**PRN:** 22310072

**Assignment No: -** 7  
**Title: -** Object Detection using YOLO and Pretrained Model

**Problem Statement:**

Implement **Object Detection** using **YOLO (You Only Look Once)** architecture and a **pretrained YOLOv8 model**. The system should detect multiple objects in real-time using a camera feed and display bounding boxes with class labels.

**Objective:**

* To understand **YOLO architecture** and its real-time detection capability.
* To use a **pretrained YOLOv8 model** for object detection tasks.
* To detect and classify objects using **bounding boxes and confidence scores**.
* To implement **real-time camera-based detection** with option to stop detection loop.

**S/W Packages and H/W apparatus used:**

* **Operating System:** Windows/Linux/MacOS
* **Kernel:** Python 3.x
* **Tools:** Google Colab / Jupyter Notebook
* **Libraries:** TensorFlow, Ultralytics YOLO, OpenCV, NumPy, Matplotlib
* **Hardware:** GPU/CPU with camera support

**Theory:**

**Object Detection:**

* It is a Computer Vision technique to identify objects of interest within an image or video.
* Outputs: Class labels, bounding boxes, and confidence scores.

**YOLO (You Only Look Once):**

* A **real-time object detection algorithm**.
* Processes the entire image in a **single forward pass**.
* Divides the image into a grid and predicts bounding boxes + class probabilities.
* **YOLOv8** (latest version from Ultralytics) is faster and more accurate than earlier YOLO versions.

**Pretrained Model:**

* A model trained on a large dataset (e.g., **COCO dataset with 80 classes**).
* Can be directly used for inference without retraining.

**Methodology:**

1. **Install Ultralytics YOLO library**.
2. **Load Pretrained YOLOv8s model**.
3. **Access camera using JavaScript integration in Colab**.
4. **Capture image frames** in real-time.
5. **Run YOLO model** for detection (yolo.track() with stream=True).
6. **Draw bounding boxes and labels** on detected objects.
7. **Stop button provided** to terminate detection loop.

**Results:**

* Real-time object detection achieved with YOLOv8s.
* Bounding boxes drawn with labels and confidence scores (e.g., *person 0.85, car 0.76*).
* System detected multiple objects simultaneously.
* Loop could be **stopped manually** using stop button.

**Advantages:**

* **Real-time detection** with high speed.
* **Multiple object detection** in a single frame.
* **Pretrained models** reduce training time.

**Limitations:**

* Requires **GPU** for real-time performance.
* Limited to classes in the pretrained dataset (e.g., COCO 80 classes).
* Sensitive to **lighting conditions and camera quality**.

**Applications:**

* **Self-driving cars** (detect pedestrians, vehicles, traffic lights).
* **Security systems** (detect intruders, weapons, abnormal activity).
* **Retail analytics** (customer counting, product detection).
* **Healthcare** (detecting instruments or abnormalities in scans).

**Conclusion:**

Object Detection using **YOLOv8 pretrained model** was successfully implemented in Google Colab with real-time camera input. The model was able to detect multiple objects with bounding boxes and confidence scores, achieving high accuracy and speed. This demonstrates the power of **deep learning in real-time computer vision applications**.