**Assignment No: - 7**

**Name :- Isha Ghorpade**

**Enrollnment :- 22420020**

**Roll :- 381066**

**Problem Statement:**

Object detection using YOLO and Pretrained Model.

**Objective:**

The objective of this practical is to implement an object detection system using a pretrained YOLO model. The specific objectives are:

* To understand the architecture of YOLO.
* To load and fine-tune a pretrained YOLO model.
* To detect objects in images or video streams with bounding boxes and class labels.
* To evaluate the model’s performance in terms of accuracy, precision, and recall.

**Software and Hardware Packages Used:**

Software Packages:

* Python 3.10 or later
* Jupyter Notebook / Google Colab
* Anaconda (for environment management)
* YOLOv8 pretrained model weights

Hardware Packages:

* GPU-enabled machine (e.g., NVIDIA CUDA GPU) for faster inference
* At least 8 GB RAM for processing
* Web camera / external camera (for real-time detection)

**Libraries Used:**

* ultralytics → For implementing YOLO models
* NumPy → Numerical operations
* OpenCV → Image and video processing
* torch, torchvision → Deep learning framework support
* Matplotlib → Visualization of outputs
* PIL (Python Imaging Library) → Image handling and preprocessing

**Theory:**

* YOLO (You Only Look Once):  
  YOLO is a state-of-the-art real-time object detection model. Unlike sliding window-based methods, YOLO uses a single neural network to process the entire image and divide it into grids. Each grid cell predicts bounding boxes, confidence scores, and class probabilities.
* Key Concepts:
  + YOLO Architecture: Based on CNN for feature extraction and bounding box prediction.
  + Pretrained Models: YOLO models trained on datasets like COCO can detect common objects out of the box.
  + Real-Time Detection: Optimized for high-speed image processing.

**Methodology:**

1. Data Preparation
   * Collect images/videos containing objects.
   * Resize images to YOLO’s required input size (e.g., 640×640).
2. Model Loading
   * Load YOLOv8 pretrained weights (yolov8s.pt) from ultralytics.
3. Inference
   * Pass images through the YOLO model.
   * Model predicts bounding boxes, class probabilities, and confidence scores.
4. Post-Processing
   * Apply Non-Maximum Suppression (NMS) to remove redundant overlapping boxes.
   * Set a confidence threshold (e.g., 0.5) to filter weak predictions.
5. Visualization
   * Use OpenCV to draw bounding boxes and labels on detected objects.
6. Evaluation
   * Calculate metrics: accuracy, precision, recall, and F1-score.
   * Perform qualitative analysis on diverse test images.

**Advantages:**

* Real-time detection for video and camera input.
* High accuracy with pretrained models.
* Scalability: Works across multiple domains (autonomous driving, healthcare, etc.).

**Limitations:**

* May struggle with small objects.
* Trade-off between speed and accuracy.
* Less effective with overlapping objects.

**Applications:**

* Autonomous Vehicles: Detecting pedestrians, vehicles, and obstacles.
* Surveillance Systems: Identifying people and objects in security footage.
* Healthcare: Detecting abnormalities in medical scans.
* Retail: Automated inventory and product tracking.
* AR/VR & Gaming: Real-time object tracking.

**Working / Algorithm:**

1. Initialization: Load pretrained YOLOv8 weights.
2. Preprocessing: Resize images to 640×640, normalize pixel values.
3. Prediction: Model outputs bounding boxes, confidence scores, and class probabilities.
4. Post-Processing: Apply NMS and filter by confidence threshold.
5. Visualization: Draw bounding boxes with class labels and confidence on images/videos.
6. Output: Annotated image/video stream with detected objects.

**Conclusion:**

In this practical, object detection was implemented using the YOLO model. YOLO’s ability to process images in a single forward pass makes it suitable for real-time applications. The pretrained YOLOv8 model detected multiple objects with good accuracy and speed. Although it has limitations in detecting very small or overlapping objects, YOLO remains one of the most efficient and widely used object detection frameworks.