**Assignment No. 7**

**[YOLO Object Detection]**

**Name: Pranav Zaware**

**Batch: TY-A A2**

**PRN: 22311462**

**1. Problem Statement**

To perform object detection using **YOLO (You Only Look Once)** with a pretrained YOLOv8 model. The implementation will detect and classify objects in a given image using bounding boxes and confidence scores.

**2. Objective**

* Implement YOLO object detection using a pretrained model (yolov8s.pt).
* Understand the architecture and working of YOLO.
* Perform detection on input images and visualize bounding boxes with labels.
* Evaluate the accuracy and effectiveness of pretrained YOLO on real-world images.

**3. Software and Hardware Packages Used**

**Software**

* Python 3.10 or later
* Google Colab / Jupyter Notebook
* Ultralytics YOLOv8 (pretrained model)
* OpenCV (for visualization)
* NumPy (for array operations)
* Matplotlib (for displaying results)
* PIL (for handling image formats)

**Hardware**

* GPU-enabled machine (NVIDIA CUDA for acceleration)
* Minimum 8 GB RAM
* Image dataset or single image for testing

**4. Libraries Used**

* **Ultralytics** → for YOLOv8 pretrained models
* **OpenCV** → for image preprocessing and drawing bounding boxes
* **Matplotlib** → for visualization of detection results
* **PIL** → for handling image formats
* **NumPy** → for numerical operations

**5. Theory**

YOLO (You Only Look Once) is a real-time object detection algorithm. Unlike traditional methods that scan the image multiple times, YOLO applies a single convolutional neural network to the entire image, dividing it into grids and predicting bounding boxes and class probabilities for each cell.

**YOLOv8**, the latest version from Ultralytics, provides higher accuracy and faster inference. It uses pretrained weights on large datasets like COCO, enabling detection of 80+ object categories.

Key concepts:

* **Single-pass detection** → Fast and efficient.
* **Bounding Boxes** → Predict object location and size.
* **Confidence Scores** → Probability of object presence.
* **Class Probabilities** → Assign detected object to a category.

**6. Methodology**

1. **Data Preparation**
   * Collect images or videos for testing.
   * Preprocess images by resizing to YOLO input size (e.g., 640×640).
2. **Model Loading**
   * Load pretrained YOLOv8 model (yolov8s.pt).
3. **Inference**
   * Input image is processed by YOLO.
   * The model outputs bounding boxes, class labels, and confidence scores.
4. **Post-Processing**
   * Apply **Non-Maximum Suppression (NMS)** to remove overlapping boxes.
   * Filter detections below a confidence threshold (e.g., 0.5).
5. **Visualization**
   * Draw bounding boxes with class labels and confidence scores.
   * Display the annotated image with detected objects.

**7. Advantages**

* Real-time detection, suitable for video feeds.
* High accuracy due to pretrained weights.
* Easy to use with Ultralytics library.
* Can detect multiple objects in one pass.

**8. Limitations**

* Struggles with detecting small or overlapping objects.
* Performance may degrade on CPU (requires GPU for real-time use).
* Trade-off between speed and accuracy depending on model size (YOLOv8n vs YOLOv8x).

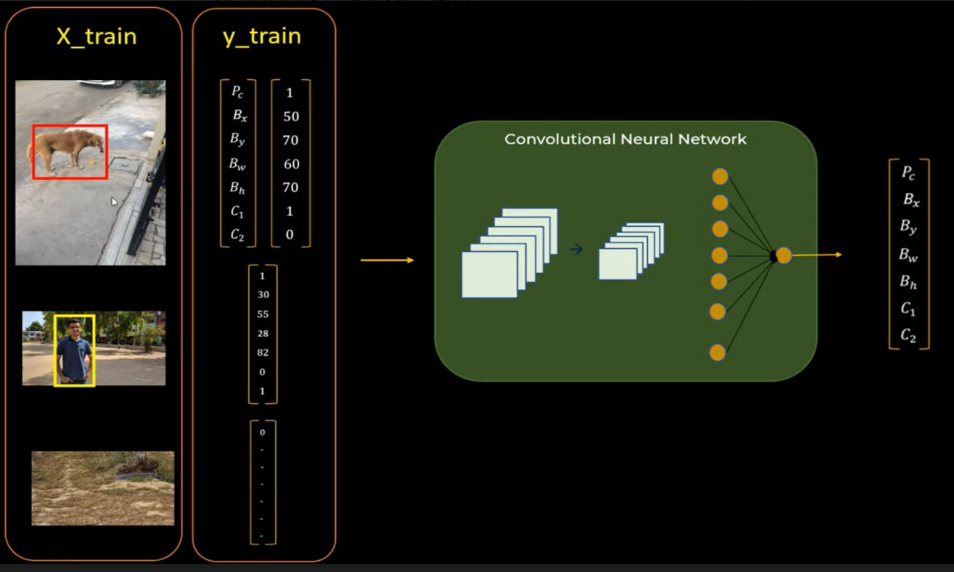
**9. Applications**

* **Autonomous Vehicles**: Detecting pedestrians, cars, and traffic signs.
* **Surveillance**: Monitoring public and private spaces.
* **Healthcare**: Detecting abnormalities in medical scans.
* **Retail**: Product detection and inventory management.
* **AR/VR and Gaming**: Object tracking in real-time environments.

**10. Working / Algorithm**

1. **Initialization**: Load pretrained YOLO model weights.
2. **Preprocessing**: Resize image, normalize pixel values.
3. **Prediction**: Model predicts bounding boxes, confidence scores, and class probabilities.
4. **Non-Maximum Suppression (NMS)**: Keeps only the most confident bounding boxes.
5. **Post-Processing**: Remove detections below a confidence threshold.
6. **Visualization**: Draw bounding boxes and labels on the input image.
7. **Output**: Annotated image with detected objects.

**11. Diagram**



**12. Conclusion**

This assignment demonstrates object detection using **YOLOv8 pretrained models**. The model efficiently detects multiple objects in a single image with bounding boxes and confidence scores. YOLO is highly effective for real-time applications in fields like autonomous driving, healthcare, and surveillance. Despite minor limitations with small or overlapping objects, it remains one of the most popular and reliable object detection methods.