# Assignment 7 – Object detection using YOLO and Pretrained Model

Name : Aayush Bokde

Roll No: 382016

PRN: 22310316

## Problem Statement

Implement an Object Detection system using YOLOv8 (You Only Look Once, version 8) to detect objects in images. The system loads a pre-trained YOLOv8 model, performs inference on an input image, and visualizes bounding boxes with labels.

## Objectives

* To understand the working of YOLOv8 for object detection.
* To perform inference using a pre-trained YOLOv8 model.
* To visualize predictions with bounding boxes and labels.
* To analyze YOLO’s performance on sample images.

## Requirements

* Operating System: Windows/Linux/MacOS
* Python Version: 3.x
* Tools: Jupyter Notebook / Google Colab
* Libraries Used: ultralytics (YOLOv8), OpenCV, Matplotlib

## Theory

YOLO (You Only Look Once) is a state-of-the-art real-time object detection model. YOLOv8 is the latest version developed by Ultralytics with improvements in accuracy and speed.  
  
Input: An image.  
YOLO Model: Detects objects in a single forward pass using convolutional neural networks.  
Output: Bounding boxes, class labels, and confidence scores for each detected object.

## Methodology

1. Install dependencies (ultralytics, opencv-python, matplotlib).
2. Load pre-trained YOLOv8 model (yolov8s.pt).
3. Read input image using OpenCV.
4. Run inference on the image with model(img\_path).
5. Visualize results by plotting bounding boxes and labels.

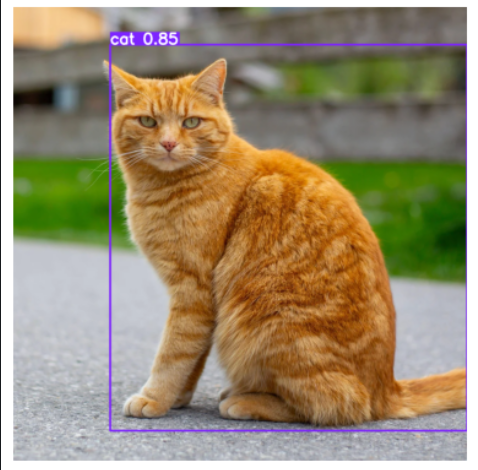
## Code Explanation (Snippet)

from ultralytics import YOLO  
import cv2  
import matplotlib.pyplot as plt  
  
# Load YOLOv8 model  
model = YOLO("yolov8s.pt")  
  
# Load image  
img\_path = "image.png"  
results = model(img\_path)  
  
# Plot annotated results  
res\_img = results[0].plot()  
plt.imshow(cv2.cvtColor(res\_img, cv2.COLOR\_BGR2RGB))  
plt.axis("off")  
plt.show()

## Graphs and Visualizations

• Input Image

  
• Annotated Output Image with bounding boxes, labels, and confidence scores.



## Advantages

* Real-time object detection with high accuracy.
* Supports multiple object detection in a single image.
* Pre-trained models available for quick implementation.

## Limitations

* Requires GPU for real-time performance on large datasets.
* May detect incorrectly if objects are too small or occluded.
* Pre-trained model limited to COCO dataset classes unless fine-tuned.

## Applications

* Autonomous driving (pedestrian and vehicle detection).
* Security and surveillance systems.
* Retail analytics (customer behavior, product detection).
* Healthcare (medical image analysis).

## Working / Algorithm

1. Install and import required libraries.
2. Load pre-trained YOLOv8 model.
3. Read the input image.
4. Run inference to detect objects.
5. Plot results with bounding boxes and labels.

## Conclusion

An Object Detection system was implemented using YOLOv8. The model successfully detected multiple objects in images and displayed them with bounding boxes and labels. YOLOv8 proved to be an efficient and accurate real-time detection system, widely applicable across industries.