YOLOv8 Object Detection Project

# 1. Introduction

This document provides detailed steps and information about the YOLOv8 Object Detection project. The project demonstrates real-time object detection using the YOLOv8 model, leveraging Python and OpenCV. The YOLOv8 model is pre-trained on the COCO dataset, enabling it to detect various objects in images and video streams.

# 2. Project Description

The YOLOv8 Object Detection project is designed to detect objects in real-time from a video feed, whether it's from a webcam or a pre-recorded video file. The project uses the YOLOv8n variant, a lightweight version of the model, to achieve fast and accurate object detection. Detected objects are highlighted with bounding boxes and labeled with their respective class names and confidence scores. Additionally, the project generates random colors for different classes to enhance the visual distinction between detected objects.

# 3. Prerequisites

Before starting, ensure that your system meets the following requirements:  
- Python 3.x  
- OpenCV  
- Ultralytics YOLO  
- Numpy

# 4. Setup Instructions

## Cloning the Repository

1. Clone the repository using the following command:

```bash  
git clone https://github.com/your-username/YOLOv8-Object-Detection.git  
```  
2. Navigate to the project directory:  
```bash  
cd YOLOv8-Object-Detection  
```

## Installing Dependencies

3. Install the required dependencies:

```bash  
pip install -r requirements.txt  
```

## Downloading YOLOv8 Model Weights

4. Download the YOLOv8 model weights and place them in the `weights/` directory. You can obtain the weights from the official Ultralytics repository or website.

## Preparing Class Labels

5. Ensure that the class labels file (e.g., `coco.txt`) is present in the appropriate directory. This file should contain the labels used by the model, such as the 80 common object categories from the COCO dataset.

# 5. Running the Project

1. Open the script (e.g., `object\_detection.py`) in your preferred code editor.  
2. Modify the paths in the script to point to your video file or webcam.  
3. Run the script using the following command:  
```bash  
python object\_detection.py  
```  
4. The video feed will display with detected objects highlighted and labeled.

# 6. Customization

## Modifying Confidence Thresholds

You can adjust the confidence threshold for object detection by modifying the `conf` parameter in the script. This parameter controls the minimum confidence level required for an object to be detected and displayed.

## Changing Input Sources

The input source for object detection can be switched between a video file and a webcam. To use a webcam, set `cap = cv2.VideoCapture(0)`. To use a video file, specify the path to the file.

## Adjusting Detection Settings

Various detection settings can be adjusted to optimize performance, such as resizing the video frames or changing the model's input size. These settings can be modified in the script according to your needs.

# 7. Conclusion

The YOLOv8 Object Detection project is a powerful tool for real-time object detection, capable of identifying multiple objects in a video feed with high accuracy. The project's flexibility allows for easy customization, making it suitable for a wide range of applications. Potential improvements include integrating the model with APIs for real-time deployment or experimenting with other YOLO variants for different use cases.

# 8. Appendix

For additional information and resources, refer to the following links:  
- [Ultralytics YOLO GitHub Repository](https://github.com/ultralytics/ultralytics)  
- [COCO Dataset](http://cocodataset.org/)  
- [OpenCV Documentation](https://docs.opencv.org/)