Day82_Deep_Learning_YOLOv8_Object_Detection_Image_Video

September 9, 2025

1 YOLOv8 Object Detection – Complete Beginner Walkthrough

This notebook explains and implements **YOLOv8 (You Only Look Once)** for object detection.

We will cover:

- What is YOLO? Why use it?
- Difference between YOLO and traditional OpenCV Haar Cascades
- Pretrained models & datasets (COCO)
- Installing and setting up Ultralytics
- Running detection on images and videos
- Drawing bounding boxes with OpenCV

2 What is YOLO (You Only Look Once)?

- **YOLO** is a state-of-the-art, real-time object detection algorithm.
- It divides an image into grids and predicts bounding boxes & class probabilities in a **single forward pass**.
- That's why it's **fast and accurate** → suitable for real-time detection.

  Use cases: self-driving cars, surveillance, medical imaging, quality inspection in factories, etc.

3 YOLO vs Haar Cascade (OpenCV)

- **Haar Cascades**:
  - Old method in OpenCV for object detection (e.g., face detection).
  - Works by scanning the image with sliding windows & handcrafted features.

  - Limitations → only works well for simple tasks like faces; not robust for complex real-world detection.

- **YOLO (Deep Learning-based)**:
  - Learns automatically from huge datasets (like COCO).
  - Handles multiple objects, complex scenes, variations in lighting, orientation, etc.
  - Much more accurate & flexible than Haar Cascades.

  Conclusion: For modern applications → **YOLO is far better**.

4 Pretrained Models and Datasets

- Ultralytics provides **pretrained YOLO models** on the **COCO dataset** (80 object classes: person, car, dog, etc.).

- Different YOLOv8 model sizes:
  - `yolov8n.pt` → Nano (fastest, lightest)
  - `yolov8s.pt` → Small
  - `yolov8m.pt` → Medium
  - `yolov8l.pt` → Large
  - `yolov8x.pt` → Extra large (most accurate but heavy)

You can start with pretrained weights and fine-tune them on your custom dataset later.

5 Setup Environment

This checks:

- Python version
- Torch (PyTorch) version
- GPU/CPU availability

[ ]:

```
# Install YOLO library
!pip install ultralytics
```

[1]:

```
# Import and check environment
import ultralytics
ultralytics.checks()
```

```
Ultralytics 8.3.196  Python-3.13.5 torch-2.8.0+cpu CPU (Intel Core i3-5005U
2.00GHz)
Setup complete  (4 CPUs, 7.9 GB RAM, 139.2/237.8 GB disk)
```

6 Load a Pretrained YOLO Model

[2]:

```
from ultralytics import YOLO
import numpy as np

# Load a pretrained YOLOv8 Nano model
model = YOLO("yolov8n.pt", "v8")
```

7 Run Prediction on an Image

- `conf=0.25` → Minimum confidence threshold

- `save=True` → Saves results with bounding boxes to `runs/detect` folder

[3]:

```
detection_output = model.predict(
    source=r"C:\Users\Lenovo\OneDrive\Desktop\Python Everyday work\Github␣
  ↪work\Computer_Vision\YOLO\Test_Image_and_Video\Image.JPG",
    conf=0.25,
    save=True
)

# Display raw tensor output
print(detection_output)

# Convert first detection result to NumPy
print(detection_output[0].numpy())
```

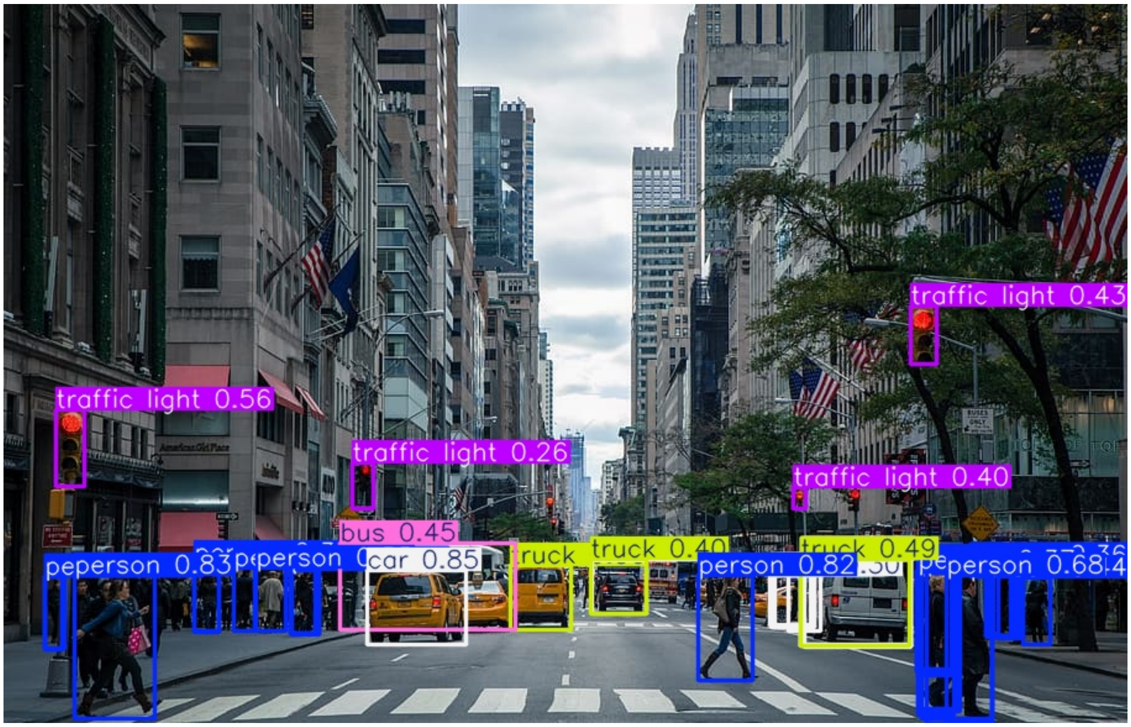
```
image 1/1 C:\Users\Lenovo\OneDrive\Desktop\Python Everyday work\Github
work\Computer_Vision\YOLO\Test_Image_and_Video\Image.JPG: 416x640 30 persons, 10
cars, 1 bus, 3 trucks, 4 traffic lights, 1061.3ms
Speed: 55.6ms preprocess, 1061.3ms inference, 26.7ms postprocess per image at
shape (1, 3, 416, 640)
Results saved to C:\Users\Lenovo\OneDrive\Desktop\Python Everyday

work\Github work\Computer_Vision\YOLO\runs\detect\predict
[ultralytics.engine.results.Results object with attributes:

boxes: ultralytics.engine.results.Boxes object
keypoints: None
masks: None
names: {0: 'person', 1: 'bicycle', 2: 'car', 3: 'motorcycle', 4: 'airplane', 5:
'bus', 6: 'train', 7: 'truck', 8: 'boat', 9: 'traffic light', 10: 'fire
hydrant', 11: 'stop sign', 12: 'parking meter', 13: 'bench', 14: 'bird', 15:
```

```
'cat', 16: 'dog', 17: 'horse', 18: 'sheep', 19: 'cow', 20: 'elephant', 21:
'bear', 22: 'zebra', 23: 'giraffe', 24: 'backpack', 25: 'umbrella', 26:
'handbag', 27: 'tie', 28: 'suitcase', 29: 'frisbee', 30: 'skis', 31:
'snowboard', 32: 'sports ball', 33: 'kite', 34: 'baseball bat', 35: 'baseball
glove', 36: 'skateboard', 37: 'surfboard', 38: 'tennis racket', 39: 'bottle',
40: 'wine glass', 41: 'cup', 42: 'fork', 43: 'knife', 44: 'spoon', 45: 'bowl',
46: 'banana', 47: 'apple', 48: 'sandwich', 49: 'orange', 50: 'broccoli', 51:
'carrot', 52: 'hot dog', 53: 'pizza', 54: 'donut', 55: 'cake', 56: 'chair', 57:
'couch', 58: 'potted plant', 59: 'bed', 60: 'dining table', 61: 'toilet', 62:
'tv', 63: 'laptop', 64: 'mouse', 65: 'remote', 66: 'keyboard', 67: 'cell phone',
68: 'microwave', 69: 'oven', 70: 'toaster', 71: 'sink', 72: 'refrigerator', 73:
'book', 74: 'clock', 75: 'vase', 76: 'scissors', 77: 'teddy bear', 78: 'hair
drier', 79: 'toothbrush'}
obb: None
orig_img: array([[[ 27,  28,  26],
        [ 25,  26,  24],
        [ 26,  27,  25],
        …,
        [ 63,  72,  69],
        [ 29,  37,  37],
        [ 18,  28,  28]],

       [[ 28,  29,  27],
        [ 25,  26,  24],
        [ 27,  28,  26],
        …,
        [ 43,  50,  47],
        [ 11,  19,  18],
        [ 44,  55,  53]],

       [[ 29,  30,  28],
        [ 27,  28,  26],
        [ 28,  29,  27],
        …,
        [ 31,  38,  33],
        [ 19,  29,  23],
        [ 83,  92,  89]],

       …,

       [[ 94,  83,  69],
        [ 94,  83,  69],
        [ 92,  81,  67],
        …,
        [113, 107, 100],
        [107, 101,  94],
        [103,  97,  90]],
        [[ 89,  78,  64],
         [ 88,  77,  63],
         [ 85,  74,  60],
         …,
         [129, 123, 116],
         [129, 123, 116],
         [133, 127, 120]],

        [[ 83,  72,  58],
         [ 81,  70,  56],
         [ 77,  66,  52],
         …,
         [132, 126, 119],
         [129, 123, 116],
         [138, 132, 125]]], dtype=uint8)
orig_shape: (582, 910)
path: 'C:\\Users\\Lenovo\\OneDrive\\Desktop\\Python Everyday work\\Github
work\\Computer_Vision\\YOLO\\Test_Image_and_Video\\Image.JPG'
probs: None
save_dir: 'C:\\Users\\Lenovo\\OneDrive\\Desktop\\Python Everyday work\\Github
work\\Computer_Vision\\YOLO\\runs\\detect\\predict'
speed: {'preprocess': 55.56060001254082, 'inference': 1061.301099951379,
'postprocess': 26.681900024414062}]
ultralytics.engine.results.Results object with attributes:

boxes: ultralytics.engine.results.Boxes object
keypoints: None
masks: None
names: {0: 'person', 1: 'bicycle', 2: 'car', 3: 'motorcycle', 4: 'airplane', 5:
'bus', 6: 'train', 7: 'truck', 8: 'boat', 9: 'traffic light', 10: 'fire
hydrant', 11: 'stop sign', 12: 'parking meter', 13: 'bench', 14: 'bird', 15:
'cat', 16: 'dog', 17: 'horse', 18: 'sheep', 19: 'cow', 20: 'elephant', 21:
'bear', 22: 'zebra', 23: 'giraffe', 24: 'backpack', 25: 'umbrella', 26:
'handbag', 27: 'tie', 28: 'suitcase', 29: 'frisbee', 30: 'skis', 31:
'snowboard', 32: 'sports ball', 33: 'kite', 34: 'baseball bat', 35: 'baseball
glove', 36: 'skateboard', 37: 'surfboard', 38: 'tennis racket', 39: 'bottle',
40: 'wine glass', 41: 'cup', 42: 'fork', 43: 'knife', 44: 'spoon', 45: 'bowl',
46: 'banana', 47: 'apple', 48: 'sandwich', 49: 'orange', 50: 'broccoli', 51:
'carrot', 52: 'hot dog', 53: 'pizza', 54: 'donut', 55: 'cake', 56: 'chair', 57:
'couch', 58: 'potted plant', 59: 'bed', 60: 'dining table', 61: 'toilet', 62:
'tv', 63: 'laptop', 64: 'mouse', 65: 'remote', 66: 'keyboard', 67: 'cell phone',
68: 'microwave', 69: 'oven', 70: 'toaster', 71: 'sink', 72: 'refrigerator', 73:
'book', 74: 'clock', 75: 'vase', 76: 'scissors', 77: 'teddy bear', 78: 'hair
drier', 79: 'toothbrush'}
obb: None
orig_img: array([[[ 27,  28,  26],
         [ 25,  26,  24],
         [ 26,  27,  25],
        …,
        [ 63,  72,  69],
        [ 29,  37,  37],
        [ 18,  28,  28]],

       [[ 28,  29,  27],
        [ 25,  26,  24],
        [ 27,  28,  26],
        …,
        [ 43,  50,  47],
        [ 11,  19,  18],
        [ 44,  55,  53]],

       [[ 29,  30,  28],
        [ 27,  28,  26],
        [ 28,  29,  27],
        …,
        [ 31,  38,  33],
        [ 19,  29,  23],
        [ 83,  92,  89]],

       …,

       [[ 94,  83,  69],
        [ 94,  83,  69],
        [ 92,  81,  67],
        …,
        [113, 107, 100],
        [107, 101,  94],
        [103,  97,  90]],

       [[ 89,  78,  64],
        [ 88,  77,  63],
        [ 85,  74,  60],
        …,
        [129, 123, 116],
        [129, 123, 116],
        [133, 127, 120]],

       [[ 83,  72,  58],
        [ 81,  70,  56],
        [ 77,  66,  52],
        …,
        [132, 126, 119],
        [129, 123, 116],
        [138, 132, 125]]], dtype=uint8)
orig_shape: (582, 910)
path: 'C:\\Users\\Lenovo\\OneDrive\\Desktop\\Python Everyday work\\Github
```

```
work\\Computer_Vision\\YOLO\\Test_Image_and_Video\\Image.JPG'
probs: None
save_dir: None
speed: {'preprocess': 55.56060001254082, 'inference': 1061.301099951379,
'postprocess': 26.681900024414062}
```

7.1 YOLOv8 Detection Output (Image Detection)

This is the detection result for Uploaded Image:



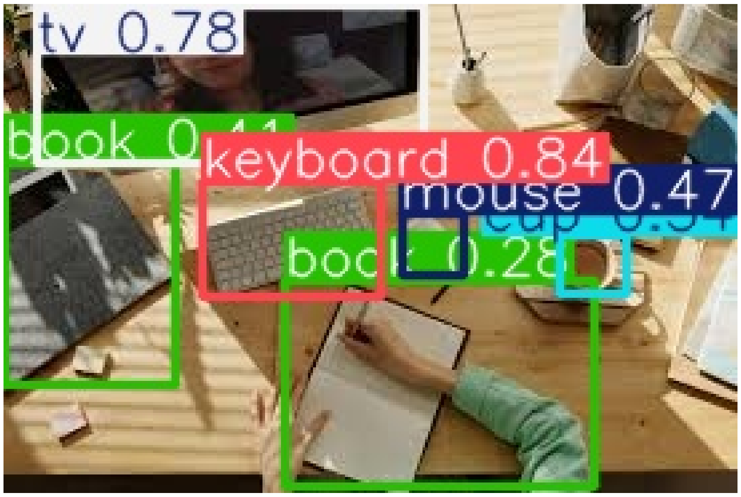
8 Load Class Labels and Colors

[4]:
```
import random

# Load COCO class names
with open(r"C:\Users\Lenovo\OneDrive\Desktop\Python Everyday work\Github␣
↪work\Computer_Vision\YOLO\coco.txt", "r") as f:
    class_list = f.read().split("\n")

# Assign random colors for bounding boxes
detection_colors = [(random.randint(0,255), random.randint(0,255), random.
↪randint(0,255)) for _ in class_list]

# Run YOLO detection on a different image
results = model.predict(
    source=r"C:\Users\Lenovo\OneDrive\Desktop\Python Everyday work\Github␣
↪work\Computer_Vision\YOLO\Test_Image_and_Video\Image_1.JPG",
    conf=0.25,
    save=True
)
```

```
image 1/1 C:\Users\Lenovo\OneDrive\Desktop\Python Everyday work\Github
work\Computer_Vision\YOLO\Test_Image_and_Video\Image_1.JPG: 448x640 1 cup, 1 tv,
1 mouse, 1 keyboard, 2 books, 681.0ms
Speed: 14.7ms preprocess, 681.0ms inference, 12.0ms postprocess per image at
shape (1, 3, 448, 640)
Results saved to C:\Users\Lenovo\OneDrive\Desktop\Python Everyday

work\Github work\Computer_Vision\YOLO\runs\detect\predict
```

8.1 YOLOv8 Detection Output (Image Detection)

This is the detection result for **Any Image** using COCO class labels:



9 Object Detection on Video

```
[11]: import cv2
import random
from ultralytics import YOLO

# Load YOLO model
model = YOLO("yolov8n.pt", "v8")

# Load COCO classes
with open(r"C:\Users\Lenovo\OneDrive\Desktop\Python Everyday work\Github␣
↪work\Computer_Vision\YOLO\coco.txt", "r", encoding="utf-8") as f:
    class_list = f.read().splitlines()
```

```python
# Assign random unique colors for each class
detection_colors = [(random.randint(0,255), random.randint(0,255), random.randint(0,255)) for _ in class_list]

# Open a video file
cap = cv2.VideoCapture(r"C:\Users\Lenovo\OneDrive\Desktop\Python Everyday work\Github work\Computer_Vision\YOLO\Test_Image_and_Video\Video.mp4")

if not cap.isOpened():
    print("Cannot open video")
    exit()

# Create a resizable window with minimize/maximize/close buttons
cv2.namedWindow("YOLOv8 Object Detection", cv2.WINDOW_NORMAL)

# Optionally maximize window (depends on OS support)
cv2.setWindowProperty("YOLOv8 Object Detection", cv2.WND_PROP_AUTOSIZE, cv2.WINDOW_NORMAL)

while True:
    ret, frame = cap.read()
    if not ret:
        print("End of stream")
        break

    # Run YOLO prediction
    results = model.predict(source=[frame], conf=0.45, save=False)
    detections = results[0].numpy()

    if len(detections) != 0:
        for i in range(len(results[0])):
            boxes = results[0].boxes
            box = boxes[i]
            clsID = int(box.cls.numpy()[0])
            conf = box.conf.numpy()[0]
            bb = box.xyxy.numpy()[0]

            # Different color for each object class
            color = detection_colors[clsID]

            # Draw bounding box
            cv2.rectangle(frame, (int(bb[0]), int(bb[1])), (int(bb[2]), int(bb[3])), color, 2)

            # Put class name + confidence
            cv2.putText(frame, f"{class_list[clsID]} {round(conf,2)}",
                        (int(bb[0]), int(bb[1]) - 10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.7, color, 2, cv2.LINE_AA)

    # Show video in a resizable window (with minimize/maximize/close)
    cv2.imshow("YOLOv8 Object Detection", frame)

    # Exit on ESC key
    if cv2.waitKey(1) == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

```
0: 384x640 5 persons, 2 chairs, 1 laptop, 588.6ms
Speed: 17.5ms preprocess, 588.6ms inference, 13.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 1 laptop, 511.7ms
Speed: 15.8ms preprocess, 511.7ms inference, 13.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 1 laptop, 535.4ms
Speed: 12.5ms preprocess, 535.4ms inference, 13.2ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 1 laptop, 611.9ms
Speed: 17.4ms preprocess, 611.9ms inference, 11.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 4 laptops, 585.3ms
Speed: 18.8ms preprocess, 585.3ms inference, 12.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 1 laptop, 461.9ms
Speed: 16.3ms preprocess, 461.9ms inference, 11.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 2 laptops, 526.2ms
Speed: 18.5ms preprocess, 526.2ms inference, 14.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 2 laptops, 529.5ms
Speed: 20.3ms preprocess, 529.5ms inference, 13.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 2 laptops, 602.9ms
Speed: 17.6ms preprocess, 602.9ms inference, 14.7ms postprocess per image at
```

```
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 2 laptops, 643.2ms
Speed: 31.6ms preprocess, 643.2ms inference, 14.0ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 2 laptops, 391.1ms
Speed: 18.1ms preprocess, 391.1ms inference, 11.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 2 laptops, 397.3ms
Speed: 17.3ms preprocess, 397.3ms inference, 12.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 452.6ms
Speed: 17.5ms preprocess, 452.6ms inference, 13.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 385.9ms
Speed: 17.8ms preprocess, 385.9ms inference, 41.4ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 413.7ms
Speed: 15.0ms preprocess, 413.7ms inference, 16.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 3 laptops, 861.1ms
Speed: 22.4ms preprocess, 861.1ms inference, 15.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 4 laptops, 687.8ms
Speed: 19.8ms preprocess, 687.8ms inference, 19.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 1607.5ms
Speed: 21.6ms preprocess, 1607.5ms inference, 69.8ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 725.1ms
Speed: 24.1ms preprocess, 725.1ms inference, 14.4ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 4 laptops, 737.5ms
Speed: 19.9ms preprocess, 737.5ms inference, 15.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 662.4ms
Speed: 21.4ms preprocess, 662.4ms inference, 19.4ms postprocess per image at
```

```
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 859.1ms
Speed: 23.0ms preprocess, 859.1ms inference, 16.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 877.1ms
Speed: 17.6ms preprocess, 877.1ms inference, 15.8ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 1069.3ms
Speed: 15.8ms preprocess, 1069.3ms inference, 74.0ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 652.0ms
Speed: 29.3ms preprocess, 652.0ms inference, 14.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 765.5ms
Speed: 42.4ms preprocess, 765.5ms inference, 15.5ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 450.8ms
Speed: 17.2ms preprocess, 450.8ms inference, 13.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 494.1ms
Speed: 19.7ms preprocess, 494.1ms inference, 15.5ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 471.0ms
Speed: 15.5ms preprocess, 471.0ms inference, 17.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 10 laptops, 1067.6ms
Speed: 21.2ms preprocess, 1067.6ms inference, 12.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 463.6ms
Speed: 15.7ms preprocess, 463.6ms inference, 15.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 7 laptops, 547.5ms
Speed: 15.5ms preprocess, 547.5ms inference, 13.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 4 laptops, 527.5ms
Speed: 16.9ms preprocess, 527.5ms inference, 13.2ms postprocess per image at
```

```
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 6 laptops, 655.8ms
Speed: 28.1ms preprocess, 655.8ms inference, 20.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 464.6ms
Speed: 19.0ms preprocess, 464.6ms inference, 12.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 2 laptops, 400.9ms
Speed: 16.4ms preprocess, 400.9ms inference, 12.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 4 laptops, 404.7ms
Speed: 15.9ms preprocess, 404.7ms inference, 12.4ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 416.9ms
Speed: 15.5ms preprocess, 416.9ms inference, 12.8ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 1 laptop, 411.1ms
Speed: 14.7ms preprocess, 411.1ms inference, 12.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 518.7ms
Speed: 14.6ms preprocess, 518.7ms inference, 17.0ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 1179.2ms
Speed: 15.2ms preprocess, 1179.2ms inference, 15.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 577.5ms
Speed: 21.1ms preprocess, 577.5ms inference, 15.0ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 5 laptops, 777.0ms
Speed: 17.6ms preprocess, 777.0ms inference, 13.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 6 laptops, 765.2ms
Speed: 29.7ms preprocess, 765.2ms inference, 12.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 4 laptops, 595.3ms
Speed: 18.1ms preprocess, 595.3ms inference, 12.4ms postprocess per image at
```

```
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 5 laptops, 766.1ms
Speed: 19.3ms preprocess, 766.1ms inference, 13.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 4 laptops, 616.9ms
Speed: 26.8ms preprocess, 616.9ms inference, 15.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 4 laptops, 651.6ms
Speed: 16.0ms preprocess, 651.6ms inference, 18.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 465.1ms
Speed: 15.9ms preprocess, 465.1ms inference, 18.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 684.0ms
Speed: 17.7ms preprocess, 684.0ms inference, 14.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 5 laptops, 467.4ms
Speed: 17.2ms preprocess, 467.4ms inference, 14.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 645.4ms
Speed: 17.5ms preprocess, 645.4ms inference, 19.2ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 715.2ms
Speed: 30.9ms preprocess, 715.2ms inference, 20.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 459.1ms
Speed: 18.4ms preprocess, 459.1ms inference, 22.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 511.7ms
Speed: 19.3ms preprocess, 511.7ms inference, 16.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 620.6ms
Speed: 19.0ms preprocess, 620.6ms inference, 15.5ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 450.2ms
Speed: 17.2ms preprocess, 450.2ms inference, 17.5ms postprocess per image at
```

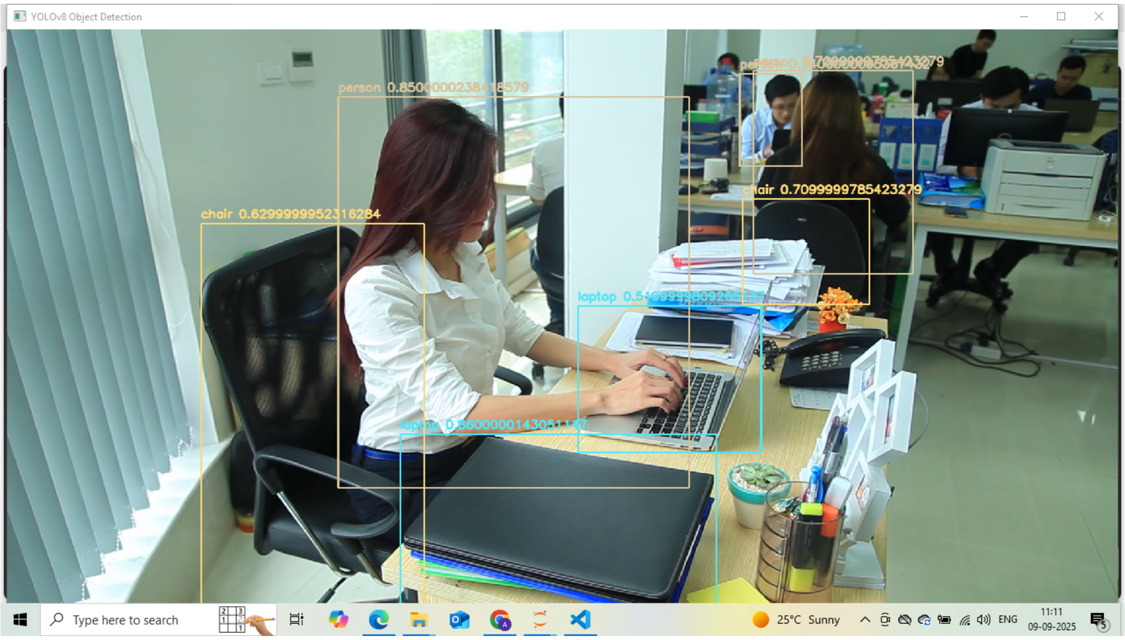
```
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 466.1ms
Speed: 15.4ms preprocess, 466.1ms inference, 19.5ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 560.2ms
Speed: 15.4ms preprocess, 560.2ms inference, 13.4ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 1 laptop, 423.8ms
Speed: 22.3ms preprocess, 423.8ms inference, 13.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 673.3ms
Speed: 15.3ms preprocess, 673.3ms inference, 12.8ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 1 laptop, 410.1ms
Speed: 17.8ms preprocess, 410.1ms inference, 11.8ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 489.0ms
Speed: 32.6ms preprocess, 489.0ms inference, 13.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 1 laptop, 411.0ms
Speed: 15.5ms preprocess, 411.0ms inference, 12.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 1 laptop, 407.0ms
Speed: 17.7ms preprocess, 407.0ms inference, 13.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 417.6ms
Speed: 18.4ms preprocess, 417.6ms inference, 12.0ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 475.4ms
Speed: 17.4ms preprocess, 475.4ms inference, 13.4ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 404.3ms
Speed: 17.1ms preprocess, 404.3ms inference, 13.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 4 laptops, 432.2ms
Speed: 15.7ms preprocess, 432.2ms inference, 11.4ms postprocess per image at
```

```
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 3 laptops, 401.1ms
Speed: 17.2ms preprocess, 401.1ms inference, 13.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 3 persons, 2 chairs, 3 laptops, 404.4ms
Speed: 15.5ms preprocess, 404.4ms inference, 13.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 1 cup, 2 chairs, 2 laptops, 449.3ms
Speed: 18.8ms preprocess, 449.3ms inference, 15.0ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 421.0ms
Speed: 17.4ms preprocess, 421.0ms inference, 14.5ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 435.4ms
Speed: 18.0ms preprocess, 435.4ms inference, 11.3ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 3 laptops, 424.9ms
Speed: 14.6ms preprocess, 424.9ms inference, 13.0ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 2 laptops, 610.1ms
Speed: 21.0ms preprocess, 610.1ms inference, 9.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 4 persons, 2 chairs, 3 laptops, 423.7ms
Speed: 18.0ms preprocess, 423.7ms inference, 14.6ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 8 persons, 2 chairs, 3 laptops, 427.4ms
Speed: 14.6ms preprocess, 427.4ms inference, 14.4ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 11 persons, 2 chairs, 3 laptops, 409.6ms
Speed: 17.1ms preprocess, 409.6ms inference, 13.9ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 8 persons, 2 chairs, 2 laptops, 405.1ms
Speed: 15.5ms preprocess, 405.1ms inference, 12.7ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 2 laptops, 555.5ms
Speed: 22.3ms preprocess, 555.5ms inference, 16.8ms postprocess per image at
```

```
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 2 laptops, 435.4ms
Speed: 17.5ms preprocess, 435.4ms inference, 13.5ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 2 laptops, 1388.2ms
Speed: 56.8ms preprocess, 1388.2ms inference, 14.1ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 2 laptops, 1372.2ms
Speed: 116.9ms preprocess, 1372.2ms inference, 16.4ms postprocess per image at
shape (1, 3, 384, 640)

0: 384x640 5 persons, 2 chairs, 2 laptops, 454.6ms
Speed: 20.7ms preprocess, 454.6ms inference, 14.5ms postprocess per image at
shape (1, 3, 384, 640)
```

9.1 YOLOv8 Detection Output (Video Detection)

This is the detection result for the video (screenshot saved as **Video.png**):



10 Conclusion – Day82 YOLOv8 Project

In this notebook, we successfully:

- Installed and set up **Ultralytics YOLOv8**

- Compared YOLO with Haar Cascades (OpenCV)
- Used **pretrained models on COCO dataset**
- Detected objects in **images** and **videos**
- Displayed results directly inside the notebook

10.1 Key Insights:

- YOLOv8 is **faster and more accurate** than traditional methods like Haar Cascades.
- Pretrained models on COCO allow instant use for 80+ classes.
- With simple code, YOLOv8 can handle both **images** and **video streams**.

10.2 Next Steps:

- Try YOLOv8 with **live webcam detection**
- Fine-tune YOLOv8 on a **custom dataset** for your own project
- Explore advanced tasks like **segmentation** and **pose estimation**

Thanks for following along!