**Python Code for YOLO Object Detection**

Import cv2

Import numpy as np

# Load YOLO

Net = cv2.dnn.readNet(“yolov3.weights”, “yolov3.cfg”)

With open(“coco.names”, “r”) as f:

Classes = [line.strip() for line in f.readlines()]

Layer\_names = net.getLayerNames()

Output\_layers = [layer\_names[I – 1] for I in net.getUnconnectedOutLayers()]

# Load Image

Image = cv2.imread(“test.jpg”)

Height, width, channels = image.shape

# Prepare Image for YOLO

Blob = cv2.dnn.blobFromImage(image, 0.00392, (416, 416), (0, 0, 0), True, crop=False)

Net.setInput(blob)

Outs = net.forward(output\_layers)

# Show Information on Screen

Class\_ids = []

Confidences = []

Boxes = []

For out in outs:

For detection in out:

Scores = detection[5:]

Class\_id = np.argmax(scores)

Confidence = scores[class\_id]

If confidence > 0.5: # Confidence threshold

Center\_x = int(detection[0] \* width)

Center\_y = int(detection[1] \* height)

W = int(detection[2] \* width)

H = int(detection[3] \* height)

X = int(center\_x – w / 2)

Y = int(center\_y – h / 2)

Boxes.append([x, y, w, h])

Confidences.append(float(confidence))

Class\_ids.append(class\_id)

# Non-Maximum Suppression

Indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

# Draw Bounding Boxes

For I in range(len(boxes)):

If I in indexes:

X, y, w, h = boxes[i]

Label = str(classes[class\_ids[i]])

Confidence = confidences[i]

Color = (0, 255, 0) # Green color for bounding box

Cv2.rectangle(image, (x, y), (x + w, y + h), color, 2)

Cv2.putText(image, f”{label} {confidence:.2f}”, (x, y – 10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, color, 2)

# Display Image

Cv2.imshow(“YOLO Object Detection”, image)

Cv2.waitKey(0)

Cv2.destroyAllWindows()

**Using YOLO with Video or Webcam**

Replace the image loading section with:

Cap = cv2.VideoCapture(0) # For webcam; replace 0 with video file path for video

While True:

Ret, frame = cap.read()

If not ret:

Break

Height, width, channels = frame.shape

Blob = cv2.dnn.blobFromImage(frame, 0.00392, (416, 416), (0, 0, 0), True, crop=False)

Net.setInput(blob)

Outs = net.forward(output\_layers)

Class\_ids = []

Confidences = []

Boxes = []

For out in outs:

For detection in out:

Scores = detection[5:]

Class\_id = np.argmax(scores)

Confidence = scores[class\_id]

If confidence > 0.5:

Center\_x = int(detection[0] \* width)

Center\_y = int(detection[1] \* height)

W = int(detection[2] \* width)

H = int(detection[3] \* height)

X = int(center\_x – w / 2)

Y = int(center\_y – h / 2)

Boxes.append([x, y, w, h])

Confidences.append(float(confidence))

Class\_ids.append(class\_id)

Indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

For I in range(len(boxes)):

If I in indexes:

X, y, w, h = boxes[i]

Label = str(classes[class\_ids[i]])

Confidence = confidences[i]

Color = (0, 255, 0)

Cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)

Cv2.putText(frame, f”{label} {confidence:.2f}”, (x, y – 10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, color, 2)

Cv2.imshow(“YOLO Object Detection”, frame)

If cv2.waitKey(1) & 0xFF == ord(‘q’):

Break

Cap.release()

Cv2.destroyAllWindows()