**Machine Learning-> CAP781->CA3**

Name: Md Aquib Name: Vivek Kumar Gupta

Regn: 12215525 Regn: 12215379

Roll no: 21 Roll no: 18.

Topic: Object Detection Using Python, OpenCV and YOLOv3

import cv2  
import numpy as np  
  
# Load YOLO model  
yolo = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")  
  
# Check if YOLO model is loaded successfully  
if yolo.empty():  
 print("Error: Failed to load YOLO model.")  
 exit()  
  
# Load classes  
classes = []  
with open("coco.names", "r") as file:  
 classes = [line.strip() for line in file.readlines()]  
  
# Get layer names  
layer\_names = yolo.getLayerNames()  
  
# Check if layer names are loaded successfully  
if not layer\_names:  
 print("Error: Failed to get layer names.")  
 exit()  
  
try:  
 # Get output layers  
 # output\_layers = [layer\_names[i[0] - 1] for i in yolo.getUnconnectedOutLayers()]  
 output\_layers = [layer\_names[i - 1] for i in yolo.getUnconnectedOutLayers()]  
except IndexError as e:  
 print("Error: Invalid index to scalar variable.")  
 print(e)  
 exit()  
  
# Define colors  
colorRed = (0, 0, 255)  
colorGreen = (0, 255, 0)  
  
# Load image  
name = "dog\_bike.jpg"  
img = cv2.imread(name)  
  
# Check if image is loaded successfully  
if img is None:  
 print("Error: Failed to load image.")  
 exit()  
  
height, width, channels = img.shape  
  
# Detect objects  
blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True, crop=False)  
yolo.setInput(blob)  
outputs = yolo.forward(output\_layers)  
  
class\_ids = []  
confidences = []  
boxes = []  
  
# Process detections  
for output in outputs:  
 for detection in output:  
 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)  
  
 # Check for potential division by zero  
 if w != 0 and h != 0:  
 boxes.append([x, y, w, h])  
 confidences.append(float(confidence))  
 class\_ids.append(class\_id)  
  
# Apply non-maximum suppression  
indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)  
  
# Draw bounding boxes and labels  
for i in range(len(boxes)):  
 if i in indexes:  
 x, y, w, h = boxes[i]  
 label = str(classes[class\_ids[i]])  
 cv2.rectangle(img, (x, y), (x + w, y + h), colorGreen, 3)  
 cv2.putText(img, label, (x, y + 10), cv2.FONT\_HERSHEY\_PLAIN, 8, colorRed, 8)  
  
# Display and save the image  
#cv2.imshow("Image", img)  
cv2.imwrite("output3.jpg", img)  
#cv2.waitKey(0)  
#cv2.destroyAllWindows()

Output:

A phone and a bottle of water

Description automatically generated



