

M S. RAMAIAH UNIVERSITY OF APPLIED SCIENCES



Python Mini Project

| Sl. No. | Reg. No. | Student Name |
|----------------|---------------------|------------------------------------|
| 1 | 19ETCS002083 | PASUPULETI ROHITH SAI DATTA |

B. Tech. in Computer Science and Engineering
Faculty of Engineering and technology
M.S. Ramaiah University of Applied Sciences
Bengaluru - 560058

April - 2022

A Mini Project Report on :
OBJECT DETECTION IN LIVE VIDEO
SURVEILLANCE

TABLE OF CONTENTS

| S.NO | CONTENTS |
|------|------------------|
| 1 | Introduction |
| 2 | Requirements |
| 3 | Implementation |
| 4 | Applications |
| 5 | Result Snapshots |

INTRODUCTION

Object detection is a phenomenon in computer vision that involves the detection of various objects in digital images or videos. Some of the objects detected include people, cars, chairs, stones, buildings, and animals.

Some of the generally used approaches of deep learning for object recognition are as follows:

ImageAI

Single Shot Detectors

YOLO (You Only Look Once)

Region-based Convolutional Neural Networks

What is YOLO exactly?

YOLO is an abbreviation for the term 'You Only Look Once'. This is an algorithm that detects and recognizes various objects in a picture (in real-time). Object detection in YOLO is done as a regression problem and provides the class probabilities of the detected images.

YOLO is an algorithm that uses neural networks to provide real-time object detection. This algorithm is popular because of its speed and accuracy. It has been used in various applications to detect traffic signals, people, parking meters, and animals.

Image Source

In the image above, there are many grid cells of equal dimension. Every grid cell will detect objects that appear within them. For example, if an object center appears within a certain grid cell, then this cell will be responsible for detecting it.

Bounding box regression

A bounding box is an outline that highlights an object in an image.

Every bounding box in the image consists of the following attributes:

Width (b_w)

Height (b_h)

Class (for example, person, car, traffic light, etc.)- This is represented by the letter c .

Bounding box center (b_x, b_y)

The following image shows an example of a bounding box. The bounding box has been represented by a yellow outline.

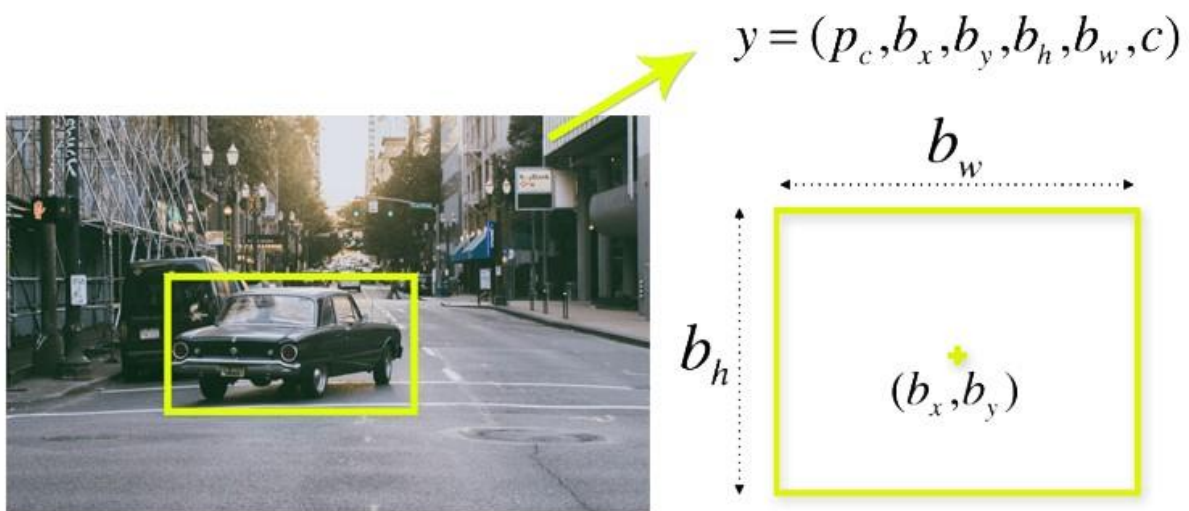


Image Source

YOLO uses a single bounding box regression to predict the height, width, center, and class of objects. In the image above, represents the probability of an object appearing in the bounding box.

Intersection over union (IOU)

Intersection over union (IOU) is a phenomenon in object detection that describes how boxes overlap. YOLO uses IOU to provide an output box that surrounds the objects perfectly.

Each grid cell is responsible for predicting the bounding boxes and their confidence scores. The IOU is equal to 1 if the predicted bounding box is the same as the real box. This mechanism eliminates bounding boxes that are not equal to the real box.

The following image provides a simple example of how IOU works.

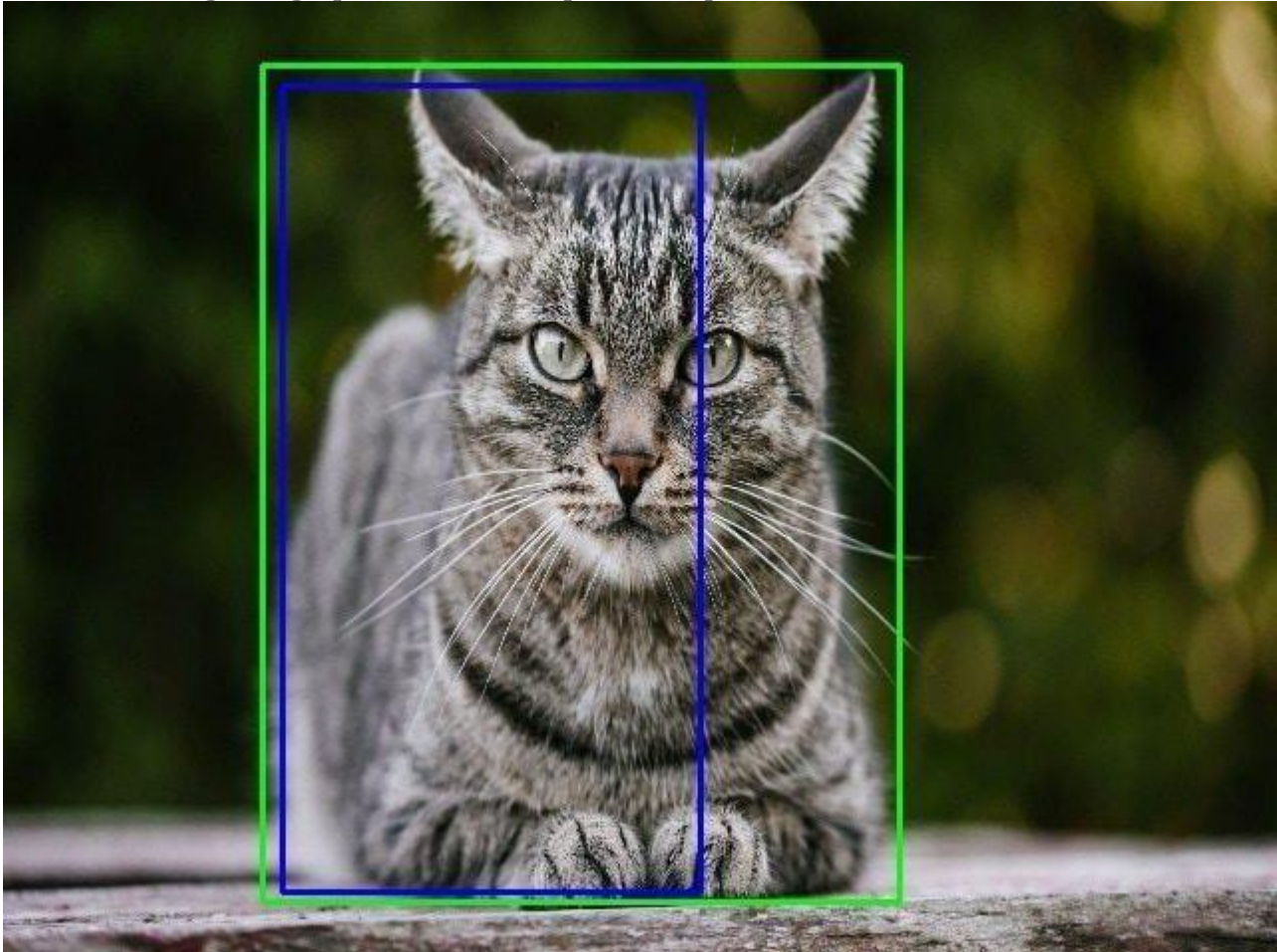


Image Source

In the image above, there are two bounding boxes, one in green and the other one in blue. The blue box is the predicted box while the green box is the real box. YOLO ensures that the two bounding boxes are equal.

REQUIREMENTS:

- Jupyter notebook
- Libraries need to install:
 - Numpy
 - OS
 - cv2
 - Time
 - Firebase_admin

- Credentials
- db

IMPLEMENTATION:

Code:

```
In [ ]: import numpy as np
import time
import cv2
import os
predict=0.5
minimum=0.5

import firebase_admin
from firebase_admin import credentials,db

cred = credentials.Certificate(r"C:\Users\pavan\OneDrive\Documents\object-c2597-firebase-adminsdk-suabj-51e632e83a.json")
firebase_admin.initialize_app(cred,{"databaseURL":"https://object-c2597-default-rtdb.firebaseio.com/"})

store = db.reference("/")

labelsPath=(r"C:\Users\pavan\OneDrive\Documents\yolo-coco\coco.names")
LABELS=open(labelsPath).read().strip().split("\n")
COLORS=np.random.randint(50,255,size=(len(LABELS),3),dtype="uint8")
weightspath=(r"C:\Users\pavan\OneDrive\Documents\yolo-coco\yolov3.weights")
configPath=(r"C:\Users\pavan\OneDrive\Documents\yolo-coco\yolov3.cfg")
net=cv2.dnn.readNetFromDarknet(configPath,weightspath)
#image=cv2.imread(r"C:\Users\ASUS\Desktop\SDP\download.jpg")
print("streaming started")
video_capture=cv2.VideoCapture(0)
while True:
    ret, image = video_capture.read()
    (H,W)=image.shape[:2]
    ln=net.getLayerNames()
    ln=net.getUnconnectedOutLayersNames()
    blob=cv2.dnn.blobFromImage(image,1/255.0,(416,416),swapRB=True,crop=False)
    net.setInput(blob)
    layerOutputs=net.forward(ln)
```

```
confidences=[]
classIDs=[]
for output in layerOutputs:
    for detection in output:
        scores=detection[5:]
        classID=np.argmax(scores)
        confidence=scores[classID]
        if confidence > predict:
            box=detection[0:4]*np.array([W,H,W,H])
            (centerX,centerY,width,height)=box.astype("int")
            x=int(centerX-(width/2))
            y=int(centerY-(height/2))
            boxes.append([x,y,int(width),int(height)])
            confidences.append(float(confidence))
            classIDs.append(classID)
idxs=cv2.dnn.NMSBoxes(boxes,confidences,predict,minimum)
if len(idxes)>0:
    for i in idxs.flatten():
        (x,y)=(boxes[i][0],boxes[i][1])
        (w, h) = (boxes[i][2], boxes[i][3])
        color=[int(c) for c in COLORS[classIDs[i]]]
        cv2.rectangle(image,(x,y),(x+w,y+h),confidences[i])
        test = "{}".format(LABELS[classIDs[i]])
        b = "{:.4f}".format(confidences[i])
        cv2.putText(image,test,(x,y-5),cv2.FONT_HERSHEY_SIMPLEX,1,color,2)
        store.update({test:b})
image1=cv2.resize(image,(900,600))
cv2.imshow("image",image1)
if cv2.waitKey(1) & 0xFF == ord("q"):
    break
video_capture.release()
cv2.destroyAllWindows()

streaming started
```

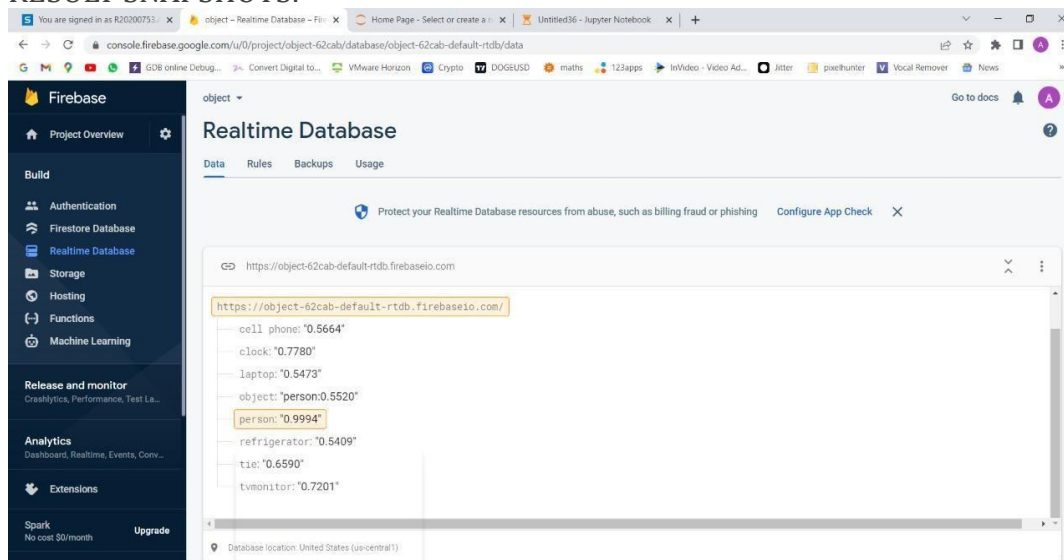
APPLICATIONS:

Autonomous driving: YOLO algorithm can be used in autonomous cars to detect objects around cars such as vehicles, people, and parking signals. Object detection in autonomous cars is done to avoid collision since no human driver is controlling the car.

Wildlife: This algorithm is used to detect various types of animals in forests. This type of detection is used by wildlife rangers and journalists to identify animals in videos (both recorded and real-time) and images. Some of the animals that can be detected include giraffes, elephants, and bears.

Security: YOLO can also be used in security systems to enforce security in an area. Let's assume that people have been restricted from passing through a certain area for security reasons. If someone passes through the restricted area, the YOLO algorithm will detect him/her, which will require the security personnel to take further action.

RESULT SNAPSHOTS:



The screenshot displays the Firebase Realtime Database interface. The left sidebar shows the 'Build' section with 'Realtime Database' selected. The main content area shows the 'Data' tab for a specific database. A JSON object is displayed in a code editor, with the following structure:

```
https://object-62cab-default-rtdb.firebaseio.com/  
{  
  "cell phone": "0.5664",  
  "clock": "0.7780",  
  "laptop": "0.5473",  
  "object": "person:0.5520",  
  "person": "0.9994",  
  "refrigerator": "0.5409",  
  "tie": "0.6590",  
  "tvmonitor": "0.7201"  
}
```

The database location is noted as 'United States (us-central1)'.

