International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 04 | April - 2023 Impact Factor: 8.176 ISSN: 2582-3930

OBJECT DETECTION AND ESTIMATING THE DISTANCE BETWEEN DETECTED OBJECTS USING DEEP LEARNING ALGORITHMS

VUNNAM ESWAR CHANDU[1],CHAGARLAMUDI HARSHITHA[2],VENNA NAVYA SREE[3],ELURI GANESH[4]

[1]Vunnam Eswar Chandu, Department of Information Technology & Dhanekula Institute of Engineering and Technology

[2]Chagarlamudi Harshitha,Department of Information Technology & Dhanekula Institute of Engineering and Technology

[3]Venna Navya Sree, Department of Information Technology & Dhanekula Institute of Engineering and Technology

[4]Eluri Ganesh, Department of Information Technology & Dhanekula Institute of Engineering and Technology

[5]Ch Ramesh Kumar, Associate Profesor & Dhanekula Institute of Engineering and Technology

---

Abstract - Object detection is a computer vision technique that allows us to identify and locate objects in an image. Computer vision is a field of artificial intelligence that trains computers to interpret and understand the visual world. Object detection involves a bounding box around each object in the image and distance should be estimated between objects. It is used to identify the objects in a given image like person, bicycle, dog etc. With this kind of identification and localization, object detection can be used to count objects in a scene and determine and track their precise locations, all while accurately labeling them. In our project we are using the COCO dataset and YOLOv3 using a deep learning method.

***Key Words*:** Deep learning-YOLO, a monocular camera, distance calculation, and object identification

1. INTRODUCTION

Object detection is one of the most important research directions for computer vision. It can be challenging for beginners to distinguish between different related computer vision tasks. For example, image classification is straightforward, but the differences between object localization and object detection can be confusing, especially when all three tasks may be just as equally referred to as object recognition. Image classification involves assigning a class label to an image,

Whereas object localization involves drawing a bounding box around one or more objects in an image. With this kind of identification and localization, object detection can be used to count objects in a scene and determine and track their precise locations, all while accurately labeling them. Imagine, for example, that contains two cats and a person.

Object detection allows us to at once classify the types of things found while also locating instances of them within the image. The YOLOv3 method is used to detect objects. In advanced mathematics, the concept of distance has been generalized to abstract metric spaces, and other distances than Euclidean have been studied. In some applications in statistics and optimization, the square of the Euclidean distance is used instead of the distance itself.

The main objectives of our project are to build a capable framework capable of detecting objects present in an image and find the distance present between the objects in the image.YOLOv3 is a real-time object detection algorithm, which uses a deep convolution neural network to classify and locate objects. In short, the motivation behind the application of YOLOv3 lies in the algorithm's high frame rate. Exhibiting the best performance compared to other object detection models such as SSD (Single Shot Multibox Detector) or Faster R-CNN, YOLOv3 yields the capability to most accurately detect small objects and fine details in images.

2. METHODOLOGY

2.1 DATASET

A good dataset will contribute to a model with good precision and recall. The Common Objects in Context (COCO) dataset is one of the most popular open source object recognition databases used to train deep learning programs. COCO dataset is a large-scale object detection, segmentation, key-point detection, and captioning dataset. The dataset consists of 328K images. This database includes hundreds of thousands of images with millions of already labeled objects for training.

2.2 DATA-PREPROCESSING

Step:1 Download the model

Downloading yolov3.weights, yolov3.cfg file and coco.names file. This coco.names file contains the list of 80 classes on which yolov3 model is pretrained

YOLO v3 weights:

This contains the weights of YOLO v3 network and it is a binary file and the weights are stored in the float data type.

YOLO v3 configuration:

YOLO v3 configuration file contains all information related to the YOLOv3 architecture and its parameters.

Step:2 Initialize the parameter

Confidence Threshold=0.5This will ignore the bounding boxes which are less probable to detect the objects.nmos threshold=0.3

This is a non-max suppression threshold which helps to ignore the multiple bounding boxes for the same object.input width=416

Fixing the width of input image.input height=416

Fixing the height of input image.

Step:3 Load the classes and models.

Load of list of classes in coco.names file Open() is used

to open the file.and read() is used to read the data from

file.labelsPath = 'coco.names'

LABELS = open(labelsPath).read().strip().split("\n")

```
['person', 'bicycle', 'car', 'motorbike', 'aeroplane', 'bus', 'train', 'truck', 'boat', 'traffic light', 'fire hydrant', 'stop
sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear', 'zebra', 'giraffe', 'backpa
ck', 'umbrella', 'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball
glove', 'skateboard', 'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana',
'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza', 'donut', 'cake', 'chair', 'sofa', 'pottedplant', 'be
d', 'diningtable', 'toilet', 'tvmonitor', 'laptop', 'mouse', 'remote', 'keyboard', 'cell phone', 'microwave', 'oven', 'toaste
r', 'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors', 'teddy bear', 'hair drier', 'toothbrush']
```

Step:4 Loading yolo v3.weights and yolov3.cfg using cv2.dnn.readNetFromDarknet()

net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)

Step:5 Read the input image

Reading the input image using imread(). It is the method present in open –CV

library.

The coco.names file contains the names of the different objects that our model has been

trained to identify.

Syntax:cv2.imread (path)

step:6 Processing the input image using blobFromImage function

cv2.dnn.blobFromImage() returns a blob which is our input image after resizing,

normalizing and channel swapping.

Syntax:cv2.dnn.blobFromImage(image,scale_factor,size,swapRB=True,crop=False)

The image is scaled to 0 or 1 using a scale factor of 1/255.

The image size is resized to (416,416)

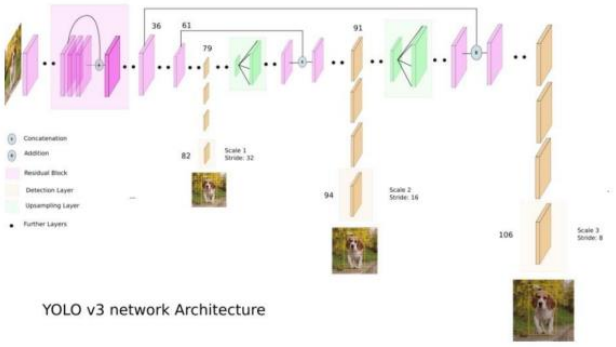
SwapRB is used to get color images only .(Swap Red,Blue)

```
[[[[0.227451   0.23529413 0.24705884 ... 0.7490196  0.6117647
    0.30588236]
   [0.23137257 0.23137257 0.2392157  ... 0.77647066 0.58980395
    0.24705884]
   [0.23529413 0.23137257 0.2392157  ... 0.7960785  0.6039922
    0.26784315]
   ...
   [0.60784316 0.6509804  0.61960787 ... 0.49411768 0.3921569
    0.25098044]
   [0.6313726  0.61960787 0.60784316 ... 0.49411768 0.35686275
    0.27451   ]
   [0.6117647  0.6039216  0.60784316 ... 0.4784314  0.3529412
    0.26784315]]

  [[0.23137257 0.2392157  0.2509804  ... 0.94117653 0.49411768
    0.2392157 ]
   [0.23529413 0.2392157  0.24313727 ... 0.9294118  0.39607847
    0.2392157 ]
   [0.2392157  0.23529413 0.24313727 ... 0.8941177  0.27058825
    0.25882354]
   ...
   [0.6392157  0.68235296 0.654902   ... 0.48627454 0.34117648
    0.26784315]
   [0.6666667  0.654902   0.6431373  ... 0.48235297 0.29803923
    0.18039216]
   [0.6509804  0.6431373  0.64705884 ... 0.4666667  0.2901961
    0.15686275]]

  [[0.20392159 0.20784315 0.21960786 ... 0.5568628  0.227451
    0.13725491]
   [0.20392159 0.20784315 0.21176472 ... 0.5647059  0.20392159
    0.14901961]
   [0.21176472 0.20392159 0.21176472 ... 0.54509807 0.16862746
    0.19215688]
   ...
   [0.6901961  0.7294118  0.69411767 ... 0.4666667  0.32941177
    0.19215688]
   [0.7137255  0.7019608  0.6862745  ... 0.46274513 0.2901961
    0.16470589]
   [0.69803923 0.6901961  0.6862745  ... 0.44705886 0.28235295
    0.14509805]]]]
```

Preprocessing output

2.3 ALGORITHMS

The algorithms we used – **YOLOv3**



You Only Look Once (YOLO) is a one-stage, multi-scale, anchor-based object detector that makes use of DarkNet-53's fully convolutional architecture.Based on YOLOv3, YOLOX [30] alters the algorithm to become an anchor-free one that permits the usage of decoupled heads and hence results in a somewhat greater detection accuracy. Moreover, YOLO has been enhanced in a number of ways to increase its capabilities and accuracy.A prediction vector p of a cell and anchor at a specific scale is given as p=(b,c,o), where b=(x,y,w,h) are coordinates created by a bounding box regressor, c=(c1,c2,,cn) addresses the confidence of having a certain class up to n classes, and o=(b,c,o,n) addresses the scale of the cell and anchor.

where o conveys objectness, i.e., assurance that p catches a genuine object. We have 33r prediction vectors, where r is the sum of all the cells in the three output layers, and each prediction vector has a total of 5+n values. Usually, r = 14,157.

YOLOv3 stands for YOU ONLY LOOK ONCE. YOLOv3 is single shot object detection. YOLOv3 is computationally fast, and can be used in real time environment.

YOLOv3 HAS MAINLY TWO MAJOR COMPONENTS:

Feature extractor Feature detector Feature extractor will extract only a few features that contain almost the same amount of information as the original set of features. Feature detector will help you to identify the objects. In YOLOv3 Darknet-53 acts as a feature extractor, the 53 layers of darknet are further stacked with 53 more layers for the detection, making yolov3 as a total of 106 layers.

UNSAMPLING LAYER

By performing the zoom operation on the image, the quality of the image will be affected, so yolo uses unsampling layer which is used to enlarge the original image so it can be displayed in higher resolution which helps in detecting small objects.

DETECTION LAYER

There are 3 detection layers in yolov3. It detects the feature maps of three different sizes at the 82nd, 94th, 106th layer. At 81st layer, it has a stride of 32. So the size is reduced to 13X13.Then 82 nd layer helps you to detect large objects. At 93rd layer, it has a stride of 16, so the size is reduced to 26X26. Then 94 th layer helps you to detect medium sized objects. At 105th layer, it has a stride of 8, so the size is reduced to 52X52. Then 106 th layer helps you to detect small objects.

RESIDUAL BLOCK

Residual block is a stack of convolution layers with skip connections. These convolution layers are used to extract the various features from input image.

In convolution layer, the mathematical operation of convolution is performed between input image and filter. By sliding the filter over the input image, the dot product is taken between the filter and parts of the input image. The output of the block is feature map which gives the information about the image such as corner and edges.

Convolution layer performs convolution on the initial input image. Convolution basically passes a small ―filter‖ or ―weights‖ box over the initial image matrix and performs matrix multiplication on the image pixel values. Consider input image is of size 3*3 and filter size is 2*2

Input image:

V1	V2	V3
V4	V5	V6
V7	V8	V9

filter:

F1	F2
F3	F4

Matrix multiplication

v1*f1+v2*f2+v4*f3+v5*f4=X1
v2*f1+v3*f2+v5*f3+v6*f4=X2
v4*f1+v5*f2+v7*f3+v8*f4=X3
v5*f1+v6*f2+v8*f3+v9*f4=X4

Convolution layer output:

X1	X2
X3	X4

Bounding box predictions:

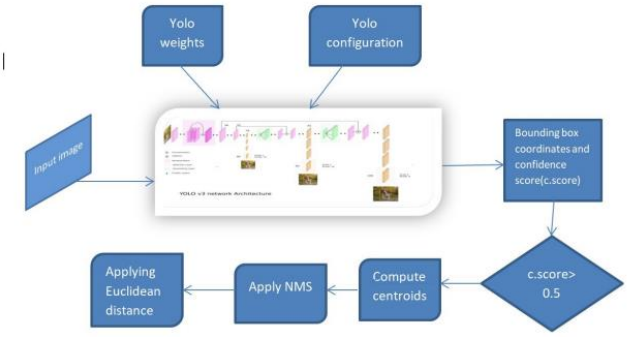
YOLO algorithm is used for predicting the accurate bounding boxes from the image.The image divides into 3x3 grids by predicting the bounding boxes for each grid and class probabilities. Both image classification and object localization techniques are applied for each grid of the image and each grid is assigned with a label.Then the algorithm checks each grid separately and marks the label which has an object in it and also marks its bounding boxes.The labels of the grid without object are marked as zero.

SYSTEM ARCHITECTURE

A system architecture is the conceptual model that defines the structure , behaviour, and more views of a system. An architecture description is a formal description and representation of a system organized in a way that supports reasoning about the structures and behaviour of the system.

A System architecture can consist of system components and the sub-system developed , that will work together to implement the overall system. There have been efforts to formalize language to describe system architecture.

A representation of system, including a mapping of functionality onto hardware and software components, a mapping of the software architecture onto hardware architecture, and human interaction with the components.



PSEUDO CODE

#Importing required Libraries
import numpy as np
import cv2
from scipy.spatial import distance as dist

#Initializingtheparameters
Confidencethreshold=0.5
nmsthreshold=0.4
inputwidth=416
inputheight=416

#Loading the classes and model

Open the text file and performing read operation using read().

Read YOLOv3 weights as weightsPath

Read YOLOv3 configuration as configPath

Loading YOLOv3 network using opencv and readNetFromDarknet

#Reading an input image

Reading the input image using imread() function. The image Path should be given as an argument to imread().

#Preprocessing the input image and sending to the network

cv2.dnn.blobFromImage() returns a blob which is our input image after resizing, normalizing and channel swapping.

The image is scaled to 0 or 1 using scale factor of 1/255.

The image size is resized to (416,416) SwapRB is used to get color images only .(Swap Red,Blue)

#Passing blob to network

By using setInput() function pass the blob to network

 # Processing the network's output

Comparing obtained confidence with confidencethreshold

Applying non max suppression

#Getting the centroid of detected object

Bounding box coordinates returns the center of detected object. In centroids list we append the center of each object. The append() function in python adds a single data item to the existing list. It will modify the original list by adding the item to the end of the list.

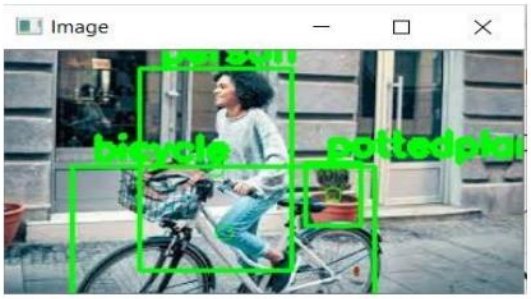
#Compute the distance between centroids using Euclidean distance.

Using dist.cdist() gives the pairwise distances between objects using Euclidean metric.

 # Draw the bounding box around objects and printing distance

Draw the bounding box using cv2.rectangle() startX, startY, endX, endY are the coordinates of rectangle.

RESULT



DISTANCE ESTIMATION

Output:

```
distance between person and bicycle is 44.28317965096906
distance between person and pottedplant is 70.09279563550022
distance between bicycle and pottedplant is 68.54195795277518
```

CONCLUSION

This project proposes an efficient real-time deep learning based framework to automate the process of monitoring the social distance via object detection and tracking approaches, where each individual is identified in the real-time with the help of bounding boxes. The general bounding boxes aid in identifying the clusters or groups of people satisfying the closeness property computed with the help of pairwise vectorized approach. The number of violations is confirmed by computing the number of groups formed and violation index term composed as the ratio of the number of people to the number of groups. The extensive trails were conducted. YOLOv3 gives an efficient approach for object detection and Euclidean distance is an approach for object detection.

We have built an extensible model for detecting objects and finding distance between them when provided with an image as an input. This model can be used as a general framework and with slight alteration in the provided parameters to have various real-life applications.

One such application we have tested is for social distance monitoring. If the person is in safe-state it shows a green bounding box otherwise red bounding box depending upon the distance between the persons.

Since this application is intended to be used in any working environment; accuracy and precision are highly desired to serve the purpose. Higher number of false positive may raise discomfort and panic situation among people being observed. There may also be genuinely raised concerns about privacy and individual rights which can be addressed with some additional measures such as prior consents for such working environments, hiding a person's identity in general.

REFERENCES

[1]W. H. Organization, ―WHO corona-viruses (COVID-19),‖ https://www.who.int/emergencies/diseases/novel-corona-virus-2019, 2020, [Online; accessed May 02, 2020].

[2] WHO, ―Who director-general's opening remarks at the media briefing on covid-19-11 march 2020.‖ https://www.who.int/dg/speeches/detail/, 2020, [Online; accessed March 12, 2020].

[3] L. Hensley, ―Social distancing is out, physical distancing is in—here's how to do it,‖ Global News–Canada (27 March 2020), 2020.

[4] ECDPC, ―Considerations relating to social distancing measures in response to COVID-19 – second update,‖ https://www.ecdc.europa.eu/ en/publicationsdata/considerations, 2020, [Online; accessed March 23, 2020].

[5] M. W. Fong, H. Gao, J. Y. Wong, J. Xiao, E. Y. Shiu, S. Ryu, and B. J. Cowling, ―Nonpharmaceutical measures for pandemic influenza in nonhealthcare settings—social distancing measures,‖ 2020.

[6] F. Ahmed, N. Zviedrite, and A. Uzicanin, ―Effectiveness of workplace social distancing measures in reducing influenza transmission: a systematic review,‖ BMC public health, vol. 18, no. 1, p. 518, 2018.

[7] W. O. Kermack and A. G. McKendrick, ―Contributions to the math