

# DETECT SOCIAL DISTANCE AND FACE MASK USING YOLOv5

Md. Rashidul Islam

Omar Faruq

Md. Fazle Rabbi

A thesis in fulfilment of the requirements for the degree  
of  
Bachelor of Science in Computer Science and Engineering



Department of Computer Science and Engineering  
Dhaka University of Engineering & Technology, Gazipur  
Gazipur, Bangladesh

June, 2022

# DETECT SOCIAL DISTANCE AND FACE MASK USING YOLOv5

MD. Rashidul Islam

Student no: 164088

Reg: 9745, Session: 2019-2020

Md. Fazle Rabbi

Student no: 164108

Reg: 9765, Session: 2019-2020

Omar Faruq

Student no: 164116

Reg: 9773, Session: 2019-2020

Supervisor: Dr. Fazlul Hasan Siddiqui

Professor

A thesis submitted for the degree of

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND  
ENGINEERING

Department of Computer Science and Engineering

Dhaka University of Engineering & Technology, Gazipur

June, 2022

# Declaration

It is thereby declared that the work presented in this thesis or any part of this thesis has not been submitted elsewhere for the award of any degree or diploma.

Signatures:

.....

(Md. Rashidul Islam)

.....

(Omar Faruq)

.....

(Md. Fazle Rabbi)

.....

Dr. Fazlul Hasan Siddiqui

Professor

Department of Computer Science & Engineering

Dhaka University of Engineering & Technology, Gazipur.

# Acknowledgements

All praises and thanks to Almighty Allah, who has enabled us with blessings to complete this thesis.

We are really grateful to our supervisor **Dr. Fazlul Hasan Siddiqui**, professor, Department of Computer Science Engineering, DUET, Gazipur-1700. Without his close supervision, valuable advice, constant encouragement, cooperation, and guidance throughout the course of the thesis it may be impossible to overcome the difficulties and make the thesis successful.

Also thanks to **Dr. Mohammad Abul Kashem**, professor, Department of Computer Science Engineering, DUET, Gazipur-1700.

Beside our supervisor, we are also thankful to **Dr. Md. Obaidur Rahman**, Head of the Department of Computer Science Engineering, DUET, Gazipur-1700 for his every step to success our Bachelor of Science Degree and managed our thesis constructively. Finally, we would like to express thankful to all faculty members and staffs of the department of CSE and all of our class mates.

# Abstract

Since 31st December, 2019 COVID-19 began spreading and has become a pandemic in a few months. Alleviation of such a pandemic can be solved by following social distancing and wearing masks. The Covid-19 has a huge impact on different sectors in many countries and such impact caused problems to many people around the world. A small step of wearing a face mask as well as following social distancing could help mitigate the spread of the virus. But some unconscious people don't follow these rules. It is quite difficult to detect those who violate these rules. This task can easily be done with the help of object detection. Our main motive is to detect face masks and calculate the social distance among people in a crowded place in real-time. For this purpose, we use the YOLOv5 object detection and distance measurement technique. Object detection stands for detecting multiple objects in a single image or in a single frame (in case of video or real-time observation) and localizing them with a bounding box. Here model will detect whether a person wears a mask or not. Distance is measured by calculating euclidean distance between two objects. A lot of work is done based on object detection using different methods, Such as R-CNN (Regional-based Convolution Neural Networks), which give the higher accuracy, but this model is not able to achieve real-time speed even with fast R-CNN, and faster R-CNN. Because all of these use a two-shot method. The problem of real-time object detection is solved

with the help of YOLOv5. The distance measurement is also done with the help of YOLOv5.

# Contents

<b>Declaration</b>	<b>i</b>
<b>Acknowledgements</b>	<b>ii</b>
<b>Abstract</b>	<b>iii</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Abbreviations</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Definition . . . . .	4
1.2.1 Mean average precision (mAP): . . . . .	4
1.2.2 Confusion Matrix: . . . . .	4
1.2.3 Accuracy (A): . . . . .	5
1.2.4 Precision (P): . . . . .	6
1.2.5 Recall (R): . . . . .	6
1.2.6 Intersection over Union (IoU): . . . . .	6
1.3 Objectives and Possible Outcomes . . . . .	7
1.3.1 Objectives: . . . . .	7

1.3.2	Possible Outcomes . . . . .	7
1.4	Application . . . . .	8
1.5	Contribution . . . . .	8
1.6	Thesis Organization . . . . .	9
<b>2</b>	<b>Literature Review</b>	<b>10</b>
2.1	Face mask detection . . . . .	10
2.2	Social distance measuring . . . . .	11
2.2.1	Facemask dataset . . . . .	12
2.2.2	YOLOv5 Model . . . . .	13
2.3	Related Work . . . . .	14
<b>3</b>	<b>System Details</b>	<b>15</b>
3.1	YOLOv5 Architecture . . . . .	15
3.2	Face Mask detection . . . . .	17
3.3	Distance measure . . . . .	20
<b>4</b>	<b>Result and Discussion</b>	<b>23</b>
4.1	Environmental Equipment . . . . .	23
4.1.1	How to create Dataset? . . . . .	23
4.1.2	Required tools . . . . .	24
4.1.3	For our model, we need the below configuration: . . . . .	25
4.2	Result . . . . .	27
<b>5</b>	<b>Conclusions</b>	<b>31</b>
5.1	Conclusions . . . . .	31
5.2	Future Work . . . . .	31
5.3	The code is given below . . . . .	32





# List of Figures

1.1	Gaussian curve that illustrates the distribution virus spread rate among the individuals, with and without applying the social distancing and wearing masks. . . . .	2
1.2	Images showing two different cases of violation . . . . .	3
1.3	Confusion matrix . . . . .	5
1.4	Confusion matrix . . . . .	7
2.1	Comparison among different object detection model . . . . .	11
3.1	General architecture of YOLOv5 network . . . . .	16
3.2	Overview of YoloV5 architecture . . . . .	17
3.3	Select output bounding box of an object from multiple bounding boxes . . . . .	18
3.4	Output of face mask detectin . . . . .	19
3.5	The above flow chart describes the steps involved for people detection and social distancing classification. . . . .	20
3.6	Output of the social distance measurement . . . . .	21

3.7	Output of the social distance measurement . . . . .	22
4.1	Our datasets . . . . .	24
4.2	A Picture can have one or more classes . . . . .	24
4.3	Image information that we get after labelling . . . . .	25
4.4	Type of classes . . . . .	27
4.5	A Picture have no classes . . . . .	28
4.6	Results from our dataset . . . . .	29
4.7	Results curve . . . . .	30

# List of Abbreviations

<b>YOLO</b>	You Only Look Once
<b>COVID-19</b>	Coronavirus Disease of 2019
<b>WHO</b>	World Health Organization
<b>mAP</b>	Mean average precision
<b>R-CNN</b>	Region-based Convolutional Neural Network
<b>IoU</b>	Intersection over Union
<b>TP</b>	True positive
<b>FN</b>	False Negative
<b>FP</b>	False Positive
<b>TN</b>	True Negative

# Chapter 1

## Introduction

### 1.1 Motivation

Covid-19 [1] is an acute respiratory infectious disease due to coronavirus infection. Till now COVID-19 has caused more than 6 million deaths and more than 510 million confirmed cases in many countries around the world. To prevent the spread of the virus, necessary measures have been taken by governments all around the world. Gaussian curves illustrate a small spike in the effectiveness of the health care system, making it easier for patients to prevent the virus by following persistent advice from health authorities. Any unforeseen sharp spikes and rapid increases in infection rates (like the red curve in figure 1.1 [2], will lead to health service failure and, consequently, an increase in the number of infections). number of deaths Figure 1 illustrates the importance of following social distancing guidelines to minimize the spread of viruses in individuals [3] [4]

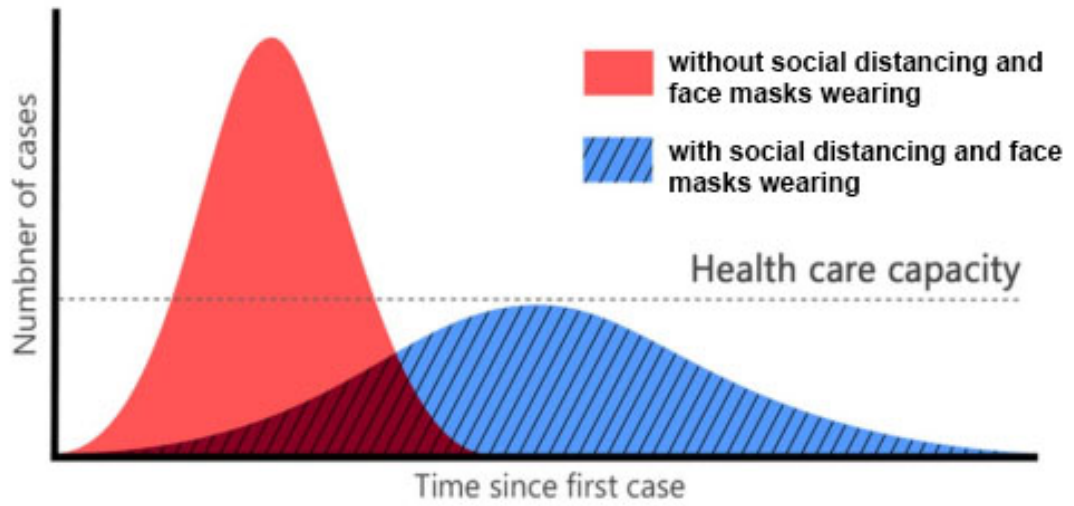


Figure 1.1 Gaussian curve that illustrates the distribution virus spread rate among the individuals, with and without applying the social distancing and wearing masks.

From the figure, we can see that it is possible to reduce COVID-19 infection by adhering to social distance. Although a few vaccines [5] have been advanced to suppress the spread of the virus, according to WHO (World Health Organization) maintaining a safe social distance among people is the most effective way. Social distance is the maintenance of a minimum distance (i.e, 3 feet) from one another. In the case of isolation and quarantine, social distance is different. Figure 1.2 shows two test cases of the violation detector. Both figures include people's ratings have different heights and stand at different angles.

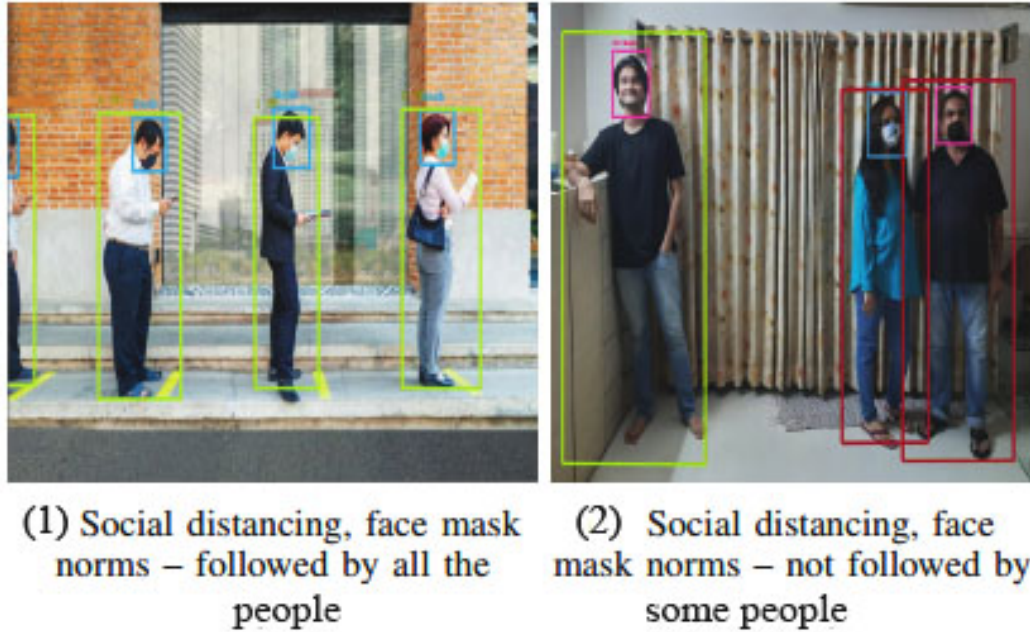


Figure 1.2 Images showing two different cases of violation

Figure 1.2 shows two test cases left image in which people follow the social distancing and face mask norms and the right picture in which some people do not follow the social distancing and face mask norms. It is a little bit difficult for people to determine whether all the people in a certain place are following the social distancing and face mask norms or not. To address this issue an AI( Artificial Intelligence) based model has been proposed that can detect whether a person wearing a mask or not and maintaining the social distance or not. In this thesis we use YOLOv5. Because YOLOv5 can detect real-time objects faster than the rest of the method. The model will detect who wearing a mask or not and who maintaining social distance or not. If anyone violates one of these two norms then it gives a signal that someone violates the norms.

## 1.2 Definition

### 1.2.1 Mean average precision (mAP):

Mean Average Precision (mAP) is a metric used to evaluate object detection models such as Fast R-CNN, Faster R-CNN, YOLO, Mask R-CNN, etc. The mean of average precision (AP) values are calculated over recall values within the range 0 to 1.

**mAP formula is calculated on the following sub metrics:**

- a) Confusion Matrix,
- b) Intersection over Union(IoU),
- c) Recall,
- d) Precision

### 1.2.2 Confusion Matrix:

A confusion matrix is an  $N \times N$  matrix used to evaluate the performance of a classification model. Where  $N$  is the number of target classes. Matrix compares actual target values with those predicted by the model shown in figure 1.4



		<b>Predicted Class</b>	
		<b>Yes</b>	<b>No</b>
<b>Actual class</b>	<b>Yes</b>	<b>True Positive (TP)</b>	<b>False Negative (FN)</b>
	<b>No</b>	<b>False Positive (FP)</b>	<b>True Negative (TN)</b>

Figure 1.3 Confusion matrix

### 1.2.3 Accuracy (A):

Accuracy is the proportion of correct results out of the total number of cases tested.

$$A = \frac{TP + TN}{TP + FN + FP + TN}$$

Accuracy is a good evaluation choice for classification problems that are well balanced and unbiased or have no class imbalance

### 1.2.4 Precision (P):

Precision measures the proportion of truly positive out of predicted positives. Measured from a column of the confusion matrix.

$$P = \frac{TP}{TP + FP}$$

### 1.2.5 Recall (R):

Recall measures what proportion of actual positives is correctly classified. It is measured from the row of the confusion matrix.

### 1.2.6 Intersection over Union (IoU):

Intersection over Union is an assessment metric used to degree the accuracy of an item detector on a specific dataset. Now Consider two areas A and B, then the IoU is calculated in figure ??

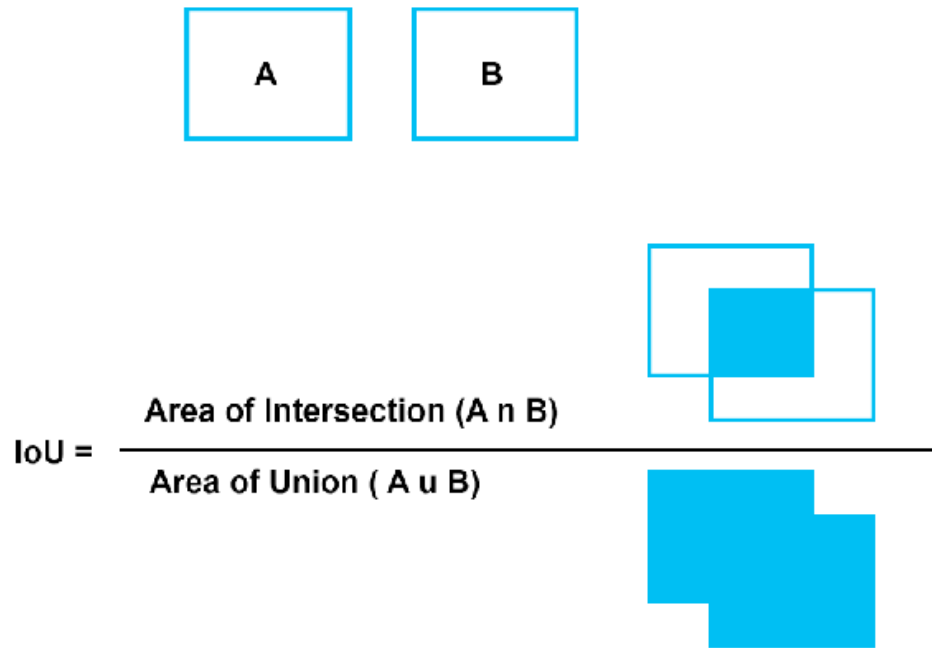


Figure 1.4 Confusion matrix

## 1.3 Objectives and Possible Outcomes

### 1.3.1 Objectives:

The following objective

- a) To create our own custom dataset
- b) To train our proposed model with the custom dataset

### 1.3.2 Possible Outcomes

- a) To detect the person whether they wear a mask or not

- b) To detect the social distance

## 1.4 Application

- a) Assist in alleviating the effect of COVID-19
- b) Help in reducing Infectious diseases
- c) Reduce the manual effort in detecting the violation

## 1.5 Contribution

In this thesis, we address the problem of slow FPS( Frame per second). Here we propose to use YOLOv5 to detect face masks and calculate social distance among the people in real-time. YOLOv5 is the latest version of the YOLO (You Only Look Once) method with a detection speed of up to 140 Frames Per Second (FPS) and 90 percent smaller than the previous version [6]. So it reduces the recognition time. In this proposed approach here, we train the existing model using custom data set rather than using an existing data set.

- a) Study of existing algorithm
- b) Create Custom data set
- c) Combining two existing models
- b) Train the model using custom data set
- c) Find out right epochs for custom data set

## 1.6 Thesis Organization

In chapter 1 we have discussed about the background and applications about the detect face mask and social distance

In chapter 2 we have focused on literature review

In chapter 3 we have disclosed on methodology of detection of face mask and social distance

# Chapter 2

## Literature Review

Nowadays, Several countries have adopted plenty of safety precautions to prevent the spread of Covid-19. Even though vaccines are already available, using a face mask and maintaining social distance play the most important role to prevent this pandemic. Maintaining social distance also assists to prevent any type of Infectious disease. Detecting the social distance and wearing mask norms is done through object detection. Object detection involvement in the combination of object classification as well as object localization makes it one of the most challenging topics in the field of computer vision.

### 2.1 Face mask detection

Wearing a face mask in public could minimize the number of people infected with COVID-19. Detection, There are many object detection networks including R-CNN family [7], SSD [8], YOLOv5 [9],MMDetection [10] etc.

In [11],author described why we used YOLOv5. Using an object detection

model such as YOLOv5 is most likely the simplest and most reasonable approach to this problem. This is because we're limiting the computer vision pipeline to a single step since object detectors are trained to detect a:

- a) Bounding box
- b) Corresponding label

All versions of YOLO can detect objects in real-time but R-CNN can not detect objects in real time. Here we use YOLOv5 rather than another version of YOLO. Because YOLOv5 is a faster, comparatively smaller model. A comparison table is given in the figure 2.1 to understand why we have used YOLOv5.

Model	Train	Test	mAP	FLOPS	FPS
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45
YOLOv4-416	COCO trainval	test-dev	43.5		65
YOLOv5-416	COCO trainval	test-dev	<b>50 - 95</b>		<b>256</b>
SSD321	COCO trainval	test-dev	45.4		16
DSSD321	COCO trainval	test-dev	46.1		12
R-FCN	COCO trainval	test-dev	51.9		12
SSD513	COCO trainval	test-dev	50.4		8
DSSD513	COCO trainval	test-dev	53.3		6
FPN FRCN	COCO trainval	test-dev	59.1		6
Retinanet-50-500	COCO trainval	test-dev	50.9		14
Retinanet-101-500	COCO trainval	test-dev	53.1		11
Retinanet-101-800	COCO trainval	test-dev	57.5		5

Figure 2.1 Comparison among different object detection model

## 2.2 Social distance measuring

Social distance means adhering to a certain distance. According to World Health Organization (WHO), this distance is 2 meters. Coronavirus outbreaks can be

reduced by adhering to social distances. In [12], a system was proposed to monitor social distancing in a real-time environment with the help of a deeply convoluted network using the concept of the sliding window as a region proposal. It measures the distance between two different people and points them in a different circle or rectangle around them in real-time. Which is too difficult to do manually in public places. Using Convolutional Neural Networks and Image Transformation methods, our proposed system intends to provide an easy and effective solution to quantify social distance and detect those in danger. This framework centers around an answer for determining social distancing utilizing YOLOv5 object discovery on video film and pictures continuously. The system utilizes the YOLOv5 object detection to distinguish people in a video [13]. Rahim A et al [14] proposed a framework which utilizes the YOLOv4 model for real-time object detection. They also proposed the social distance measuring approach in their YOLOv4 model framework. Our model is more suitable for the field application environment to realize fast mask detection and recognition with high accuracy. Because we use YOLOv5 that is a faster model that we observe in the figure 2.1.

### 2.2.1 Facemask dataset

Dataset defines a bunch of images that we used during our project to train our model. Our facemask dataset consisted of 2000+ images with three labels, including "Mask", "No Mask" and "Person". Each image had various sizes, labels, and bounding boxes in the YOLO Darknet TXT. One text file per image (containing the annotations and a numeric representation of the label) and a labelmap (which translates the numeric IDs to human readable strings) are included in this format. In, [11] author mentioned that The images from the face mask dataset



have to be divided into three groups: images for model training, images for result validation, and images for model testing. In this thesis, the YoloV5 was used to execute the CNN in order to determine whether a person is wearing a facemask or not. To execute our experiment on object detection with diverse datasets, we collected and manually labeled numerous images available in public sources. The experimental findings for facemask detection using deep learning models with different epochs, like 35, 50, and 100. Many other research studies propose different datasets to address the problem of crowded detections more successfully.

### 2.2.2 YOLOv5 Model

Our key objective is to implement a working YOLOv5 object detector to detect humans in each frame. In [15], “YOLO”, referring to “You Only Look Once”, is a family of object detection models introduced by Joseph Redmon in a 2016 publication “You Only Look Once: Unified, Real-Time Object Detection. Since then, several newer versions have been released, of which, the first three were released by Joseph Redmon. On June 29, Glenn Jocher released the latest version YOLOv5, claiming significant improvements concerning its predecessor. For now, the YOLO series includes YOLO-v1 [15], YOLO-v2 [16], YOLO-v3 [17], YOLO-v4 [18] and YOLO-v5 [19]. YOLO-v5 is the most recent version of the YOLO project. YOLOv5 has three layers: Backbone, Neck, and Head. The backbone is a CNN that collects and forms image features at various levels of detail. In [20] author described that, Yolov5’s backbone is divided into four types: Yolov5s, Yolov5m, Yolov5l, and Yolov5x. The criterion for dividing this is the difference between depth multiple (model depth multiple) and width multiple (layer width multiple). Yolo5Vs is the fastest, but with less accuracy, and Yolov5x is the

slowest, but with improved accuracy. In addition, Yolov5's backbone is divided into four types: Yolov5s, Yolov5m, Yolov5l, and Yolov5x [20]. Compared with the existing YOLOv5 model, the proposed method gives promising results. Detecting the mask from the point of view of evaluating the accuracy and precision of the detector. A separate data set was used to train the model.

## 2.3 Related Work

When the world began to take precautions against the coronavirus, a number of facemask detection systems were presented. Imran Ahmed et al. [21] proposed a deep learning platform based on YOLOv3 model in detecting humans and correspondingly track their social distance using an overhead perspective. Seemanthini and Manjunath deployed the human detection technique for an action recognition system [22]. F-RVO and YOLO were used to create a quadruped robot that promotes social distancing in urban environments.

# Chapter 3

## System Details

In this method proposed here uses the YOLOv5 algorithm to detect face mask and social distance. Here we also train our model on the custom dataset to detect whether a person wears a mask or not. Here we discussed how can we detect a face mask? How can we measure distance among people? And finally, how can we decide the result?

### 3.1 YOLOv5 Architecture

YOLOv5 is a one-shot multi-stage object detection network. The object detection speed of YOLOv5 is so high compared to other versions of YOLO. Recently, YOLOv5 has been used for object detection in real-time [23] and also used to measure distance among different objects in real-time. YOLOv5 [24] makes use of convolutional layers, which makes it a totally convolutional neural network. YOLOv5 is divided into three main parts: feature extractor, detector, and classifier.

An input image is taken from the input resource. It then goes through the object extractor to extract the object maps at different scales from the image. The feature maps are then fed into the detector to get information about the location of the bounding boxes and the membership of the object found for one of the defined classes defined by the classifier [25]. This process is described in Fig. 3.1

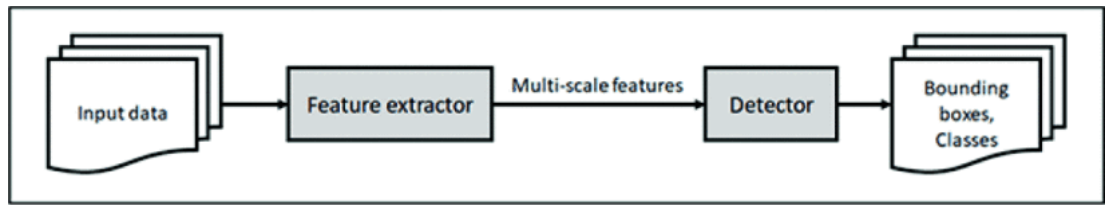


Figure 3.1 General architecture of YOLOv5 network

In YOLOv5 architecture, the CSPDarknet [18] is used to feature extraction, and PANet [26] is used to detect. The detailed network architecture of the YOLOv5 is shown in the figure 3.2 [27] It detects the object in three-stage: the backbone, the head, and the detection figure 3.2 [27]. The backbone is a CNN that collects and forms image features at various levels of detail [27].

Here we have used the YOLOv5 framework. This framework consists of three main parts

- a) **Backbone:** A convolutional neural network synthesizes and shapes image features at varying levels of detail.
- b) **Neck:** A series of layers to mix and match image features to pass them forward for predictions.
- c) **Head:** Collect features from the neck and takes bounding box and class prediction steps.

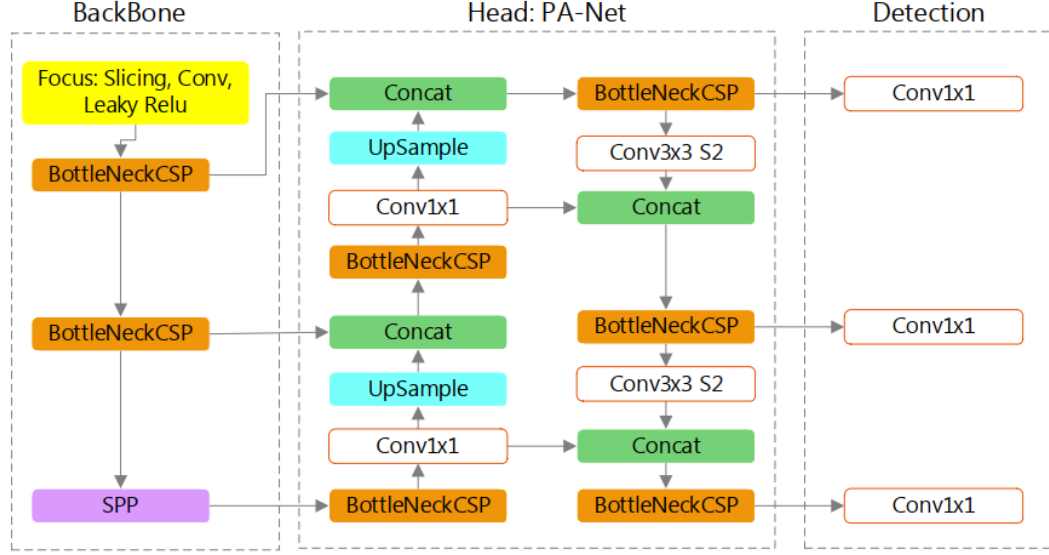


Figure 3.2 Overview of YoloV5 architecture

## 3.2 Face Mask detection

The model categorizes the face into two classes one is a mask and another is unmasked. For the classification task, the model first divided the image into  $S \times S$  grids ( i.e,  $3 \times 3$  or  $4 \times 4$  or  $19 \times 19$  etc.) shown in figure 3.3. Then it generates a matrix like

$$\begin{bmatrix} P_c & B_x & B_y & B_w & B_h & C_1 & C_2 \end{bmatrix}^T$$

Here,

$P_c$  = Probability of a class

$B_x$  = value of x-axis of the center of the bounding box

$B_y$  = value of y-axis of the center of the bounding box

Bw = width of the bounding box

Bh = height of the bounding box

C = class ( or label

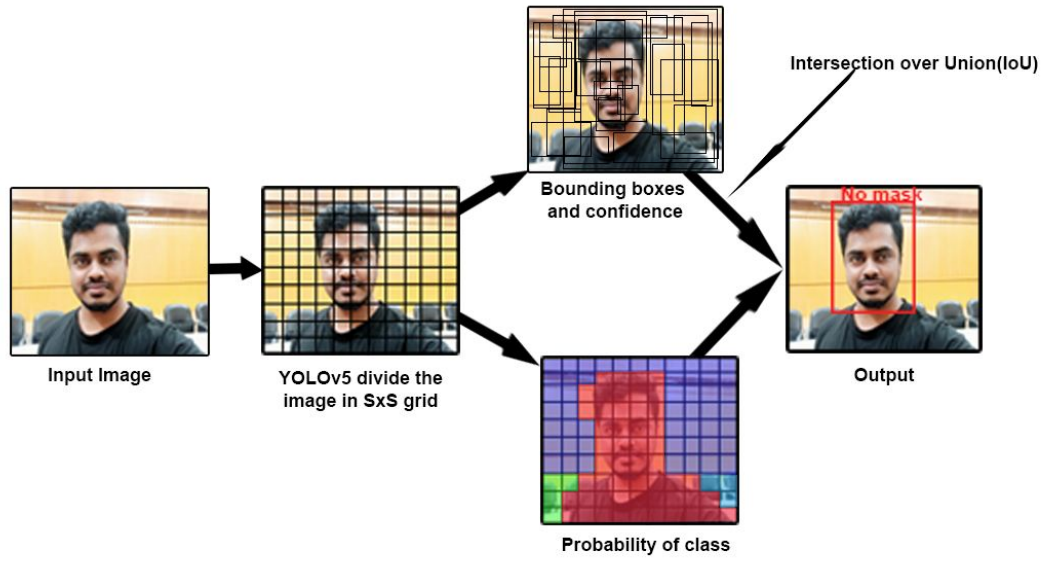


Figure 3.3 Select output bounding box of an object from multiple bounding boxes

For each grid, the method calculates the above matrix. The value of  $P_c$  is either 1 or 0. If the value of  $P_c$  is 0, it means that there is no object in that grid, so no need to calculate the other element of the matrix. When the value of  $P_c$  is 1, it means that the object consists in that grid, and calculates the rest of the elements of the matrix. Here  $C$  represents the class or label number ( i.e if there exist two objects cat and the person then the value of  $C$  becomes 1 and 2 for two classes. The rest element of the matrix is explained above. In this way calculate the matrix for all grids of an input. There can exist multiple objects in one image. Many bounding boxes are created for an object, then IoU (Intersection

of Union) is performed among that bounding boxes to select the object with one bounding box (shown in figure 3.3). The threshold value of IoU is greater than or equal to 0.5 and less than 1.0. Finally, select the frame whose IoU value is greater. For better performance, it needs a GPU-based environment. YOLO also works in non-GPU environments but its performance is comparatively lower than GPU-based [28] [29].

Here we use google colab and an online GPU to run the code. The result of our experiment on the face mask detection part is given in the figure 3.4. The person who wears a mask is labeled as Mask and the person who does not wear a mask is labeled as No Mask with precision(mAP).



Figure 3.4 Output of face mask detectin

### 3.3 Distance measure

To calculate distance, first detect the person (object). After detecting person we get the center point of person, then calculate the distance between or among person by calculating the euclidian distance using center values. let  $(x_1, y_1)$  be the center value of one person and  $(x_2, y_2)$  is the center value of another person, then the distance between two person is

$$socialdistance, d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Now if the distance is less than 2m then that two person violate the socila distance. If the distance is equal or geter than 2m then that two person maintainance the social distance. The flow diyagram of the distance measurement shown in figure 3.5 [30].

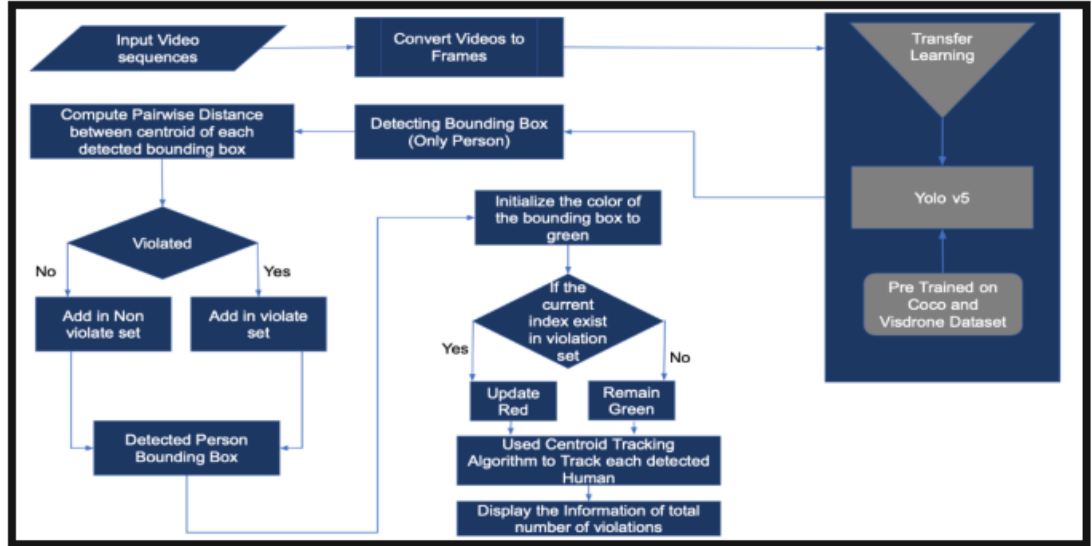


Figure 3.5 The above flow chart describes the steps involved for people detection and social distancing classification.



Here we use google colab and an online GPU to run the code. The result of our experiment on the distance measure part is given in the figure ?? and figure 3.7. The person who does not violate the social distance norms detect with the green rectangle box and the person who violate the social distance norms is detect with a red rectangle box with a line to indicate who violates the norms.

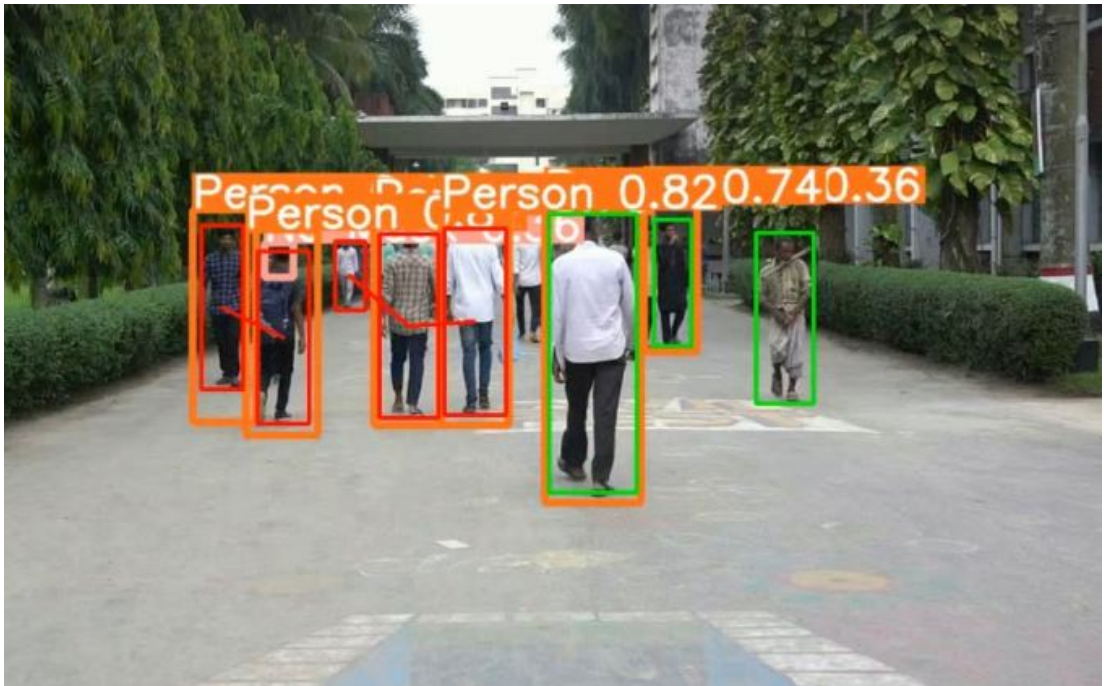


Figure 3.6 Output of the social distance measurement



Figure 3.7 Output of the social distance measurement

# Chapter 4

## Result and Discussion

In this chapter, we describe the environment of the experiment, the required tools, and the experimental result.

### 4.1 Environmental Equipment

#### 4.1.1 How to create Dataset?

- First try to make the dataset balanced. It is not good if we use 300 image for mask class and 100 image for No mask class. We should make it balance.
- Label every instance of every class.if we do not give a bounding box for an instance, For this type miss detector will perform very poorly.
- Annotation should be consistence
- Annotation should be consistence

	Train	Validation
Dataset - 1	457	122
Dataset - 2	982	120
Dataset - 3	1533	615

Figure 4.1 Our datasets

Here, in Figure 4.1 there is a table showing three datasets that we used to train our model. Dataset-1 we downloaded 457 different images and labelled them with our own class with validation 122. In dataset-2 there are 982 images with 120 validation and 1533 images with 615 validation in dataset-3.

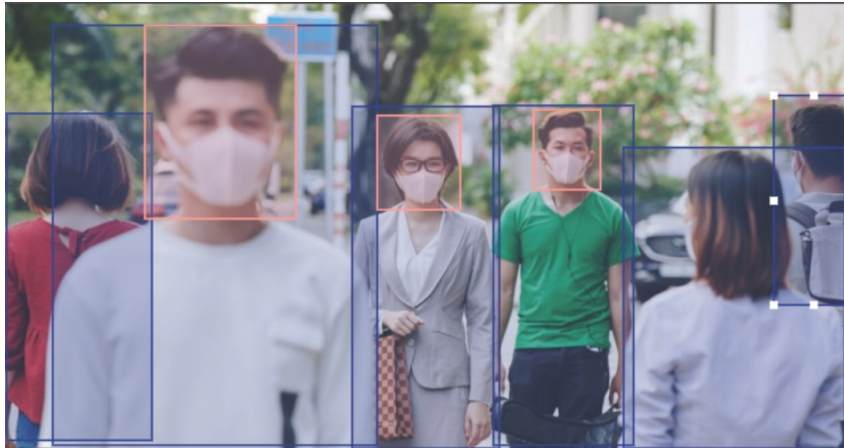


Figure 4.2 A Picture can have one or more classes

In figure 4.2 there are different people with different movements. For every instant we had to select them with different classes. That is why there could be multiple number of classes in one image.

#### 4.1.2 Required tools


There are many tools for labelling images


- <https://roboflow.com/>
- <https://cvat.org/>
- <https://labelbox.com/>
- <https://www.makesense.ai/>


For image labelling we use maksense.ai

After labelling it gives a text file for every images.

0	0.478536	0.247059	0.111107	0.231933
1	0.787869	0.216807	0.138884	0.225210
1	0.198243	0.275630	0.128783	0.221849
2	0.179304	0.576471	0.320696	0.847058
2	0.472223	0.558824	0.308070	0.868908
2	0.801757	0.552941	0.330797	0.894118

  
 classes

  
 Object Center

  
 Width


  
 Height

Figure 4.3 Image information that we get after labelling

### 4.1.3 For our model, we need the below configuration:

#### Base:

matplotlib 3.2.2 or latest version

numpy 1.18.5 or latest version

opencv-python or latest version

Pillow 7.1.2 or latest version

PyYAML 5.3.1 or latest version

requests 2.23.0 or latest version

scipy 1.4.1 (Google Colab version) or latest version

torch 1.7.0 or latest version

torchvision 0.8.1 or latest version

tqdm 4.41.0 or latest version

protobuf 3.20.1 or lower version

<https://github.com/ultralytics/yolov5/issues/8012>

**Logging:**

tensorboard 2.4.1 or latest version and wandb

**Plotting:**

pandas 1.1.4 or latest version

seaborn 0.11.0 or latest version

**Export:**

coremltools 4.1 (CoreML export) or latest version

onnx 1.9.0 (ONNX export) or latest version

onnx-simplifier 0.3.6 ( ONNX simplifier) or latest version

scikit-learn 0.19.2 (CoreML quantization)

tensorflow 2.4.1 (TFLite export) or latest version

tensorflowjs 3.9.0 (TF.js export) or latest version

openvino-dev (OpenVINO export)

**Extras:**

ipython interactive notebook

psutil system utilization

thop FLOPs computation

albumations 1.0.3 or latest version

roboflow

## 4.2 Result

Object detection can detect any type of object depending on the train data set. Here we train our model with data set that contains only three labels they are a mask, no mask, and person shown in figure 4.4. So our result identifies whether a person wears a mask or not and maintains the social distance or not. This means the model can detect a person who violates the social distance and wear mask norms.

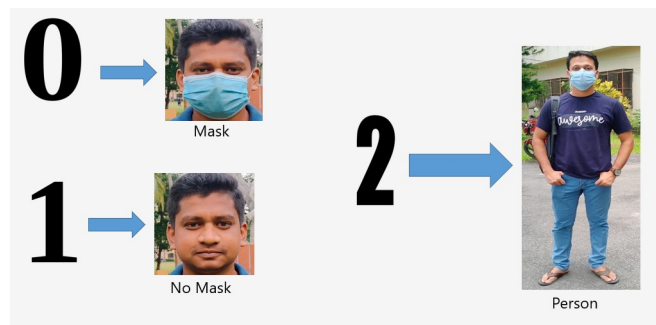


Figure 4.4 Type of classes

Every image can include one or more classes. We can use an image that includes none of these classes. It can help reduce false positives to less than 10 percent that shown in the figure 4.5.



Figure 4.5 A Picture have no classes

When we train the YOLOv5 model using different data sets we get below performance,

**Data set-1 in which contain 457 train and 122 validation picture and 3 labels ( mask, unmask, person )**

- a) 30 epochs completed in 0.133 hours with mAP 0.176.
- b) 50 epochs completed in 0.221 hours with mAP 0.184.
- c) 100 epochs completed in 0.440 hours with mAP 0.17.

**Data set-2 in which contain 982 train and 120 validation picture and 3 labels ( mask, unmask, person )**

- a) 50 epochs completed in 0.296 hours with mAP 0.591.
- b) 100 epochs completed in 0.589 hours with mAP 0.59.



**Data set-3 in which contain 1533 train and 616 validation picture and 3 labels ( mask, unmask, person )**

a) 50 epochs completed in 0.397 hours with mAP 0.86.

Here we see that if we just increase the epoch it will overfit. But if we increase the data set size and choose the correct epoch then we will get the maximum mAP. Precision, recall, and mAP is shown in the figure 4.6, and curve of our final result is shown in the figure 4.7.

	Precision	Recall	Mean average Precision
Dataset - 1	0.312	0.454	0.239
Dataset - 2	0.493	0.513	0.478
Dataset - 3	0.852	0.821	0.857

Figure 4.6 Results from our dataset

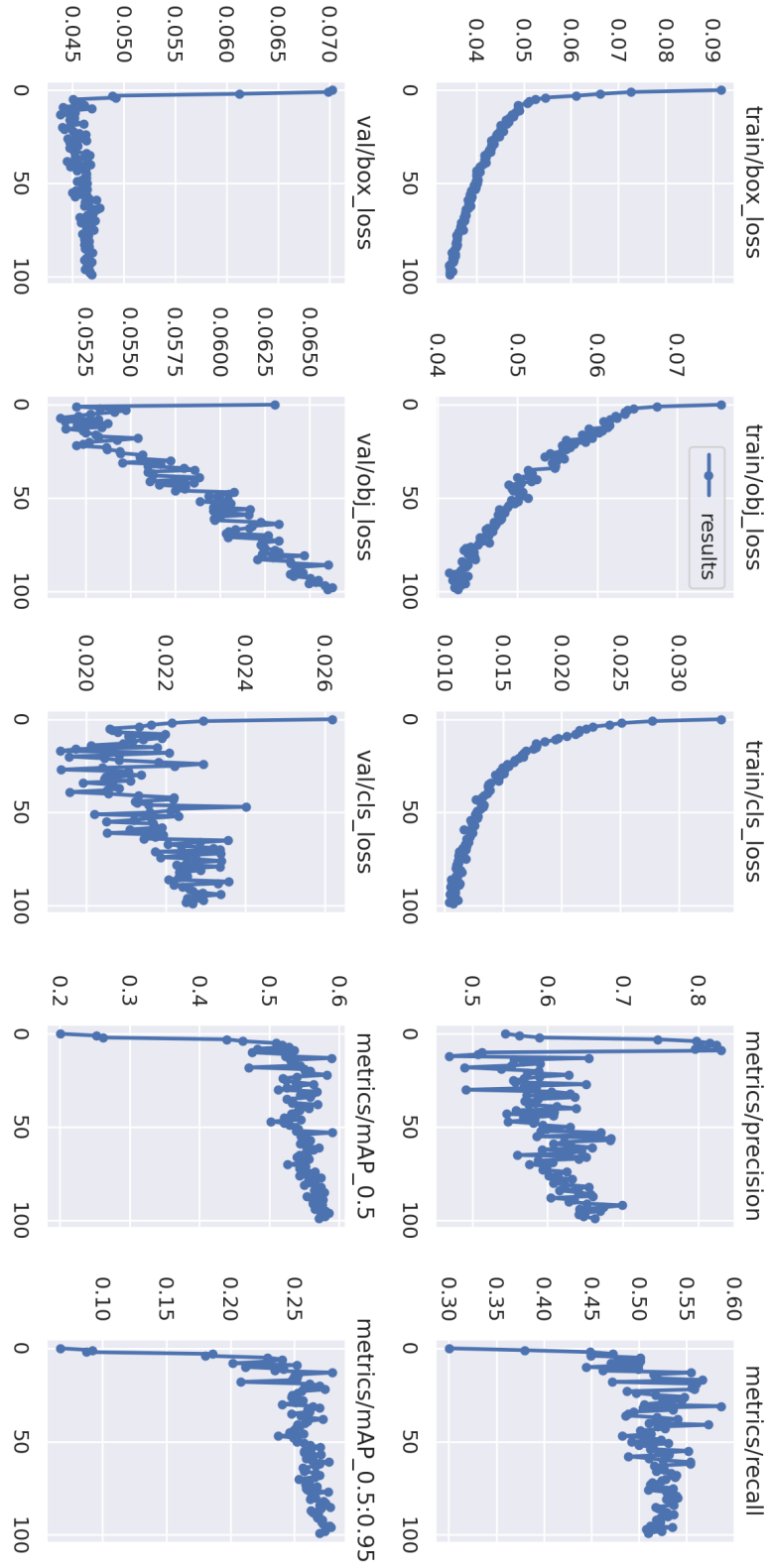


Figure 4.7 Results curve

# Chapter 5

## Conclusions

### 5.1 Conclusions

In our thesis, we have presented a model that can detect face masks and social distance. It is useful for preventing infectious diseases(i.e, COVID-19, Influenza (flu), Chickenpox, Monkeypox, Infectious mononucleosis etc). This model can cover a particular region that can cover the camera. If any person does not wear a mask or violate the social distance norms, then the model can detect that person who violates the norms.

### 5.2 Future Work

In the future, its efficiency should be improved, we can add a sound system to give an alert when a person violates the face mask and social distance norms. So it could be used in any type of detection-based system just need to make the data set based on the system where we want to apply this model.

# Appendix

## 5.3 The code is given below

```
+ Code + Text      Connect | Editing | ^
↑ ↓ ↻ ⌨ ✎ 📄 🗑 ⋮
▼ Clone repository, install dependencies and check PyTorch and GPU.

[ ] !git clone https://github.com/ultralytics/yolov5 # clone
    %cd yolov5
    %pip install -qr requirements.txt # install

    import torch
    import utils
    display = utils.notebook_init() # checks

YOLOv5 v6.1-242-ga80dd66 Python-3.7.13 torch-1.11.0+cu113 CUDA:0 (Tesla T4, 15110MiB)
Setup complete (2 CPUs, 12.7 GB RAM, 38.7/78.2 GB disk)
```

### ▼ Mount Google Drive

```
[ ] from google.colab import drive
    drive.mount('/content/drive')

Mounted at /content/drive
```

### ▼ Train YOLOv5s on Custom Dataset for 50 epochs

```
[ ] !python train.py --img 640 --batch 16 --epochs 50 --data Custom_faruq.yaml --weights yolov5s.pt --cache
```

detect.py detect custom classes by last.pt file on images and saving results to runs/detect

```
!python detect.py --weights '/content/yolov5/runs/train/exp/weights/last.pt' --img 640 --conf 0.25 --source '/content/drive/MyDrive/2nd.jpg'
```

#### ▼ Show Image After Detect

```
[ ] display.Image(filename='/content/yolov5/runs/detect/exp/thesismate.jpg', width=600)

!python detect.py --weights '/content/yolov5/runs/train/exp/weights/last.pt' --img 640 --conf 0.25 --source '/content/drive/MyDrive/File_for_test/189.jpg'

[ ] display.Image(filename='/content/yolov5/runs/detect/exp5/189.jpg', width=600)
```

detect.py detect custom classes by last.pt file on video and saving results to runs/detect

```
!python detect.py --weights '/content/yolov5/runs/train/exp/weights/last.pt' --img 640 --conf 0.25 --source '/content/drive/MyDrive/file_for_detect/Face_Mask_distance.mp4'
```

We import various libraries.

```
[ ] from base64 import b64encode
    from google.colab import files
    from google.colab.patches import cv2_imshow
    from IPython.display import HTML
    from PIL import Image
    from tqdm.notebook import tqdm
    import cv2
    import numpy as np
    import os
    import torch
```

#### ▼ Model

The model used is YOLOv5x, the best YOLOv5 model. We import it with Torch Hub.

When we pass an image to this model, it returns to us where objects are in the image, which objects they are, and what is the model's confidence about this.

```
[ ] model = torch.hub.load('ultralytics/yolov5', 'yolov5x',
    pretrained=True, verbose=False)
    model.cuda('cuda:0');
```

**Function for Video Display**

+ Code + Text

```
[ ] def display_video(path):
    # Input video path
    save_path = "/content/yolov5/runs/detect/exp/ove1.mp4"

    # Compressed video path
    compressed_path = "/content/yolov5/runs/detect/exp/ove1.mp4"

    os.system(f"ffmpeg -i {save_path} -vcodec libx264 {compressed_path}")

    # Show video
    mp4 = open(compressed_path, 'rb').read()
    data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
    HTML("""
    <video width=400 controls>
      <source src="%s" type="video/mp4">
    </video>
    """ % data_url)
```

save file path in filename variable

```
[ ] filename = '/content/yolov5/runs/detect/exp/ove1.mp4'
```

**• People detection**

The first way to calculate distance among people is just calculate the distance among the rectangles (boxes) returned by the model, more precisely the distance among their centers.

```
[ ] def center_distance(xyxy1, xyxy2):
    '''Calculate the distance of the centers of the boxes.'''
    a, b, c, d = xyxy1
    x1 = int(np.mean([a, c]))
    y1 = int(np.mean([b, d]))

    e, f, g, h = xyxy2
    x2 = int(np.mean([e, g]))
    y2 = int(np.mean([f, h]))

    dist = np.linalg.norm([x1 - x2, y1 - y2])
    return dist, x1, y1, x2, y2
```

When we have a frame of a video, we can detect the people on the frame using YOLOv5x and draw the rectangles. The color of the rectangle indicates if the person is too close to another person.

```
[ ] def detect_people_on_frame(img, confidence, distance):
    '''Detect people on a frame and draw the rectangles and lines.'''
    results = model([img[:, :, :-1]]) # Pass the frame through the model and get the boxes

    xyxy = results.xyxy[0].cpu().numpy() # xyxy are the box coordinates
    #      x1 (pixels) y1 (pixels) x2 (pixels) y2 (pixels) confidence      class
    # tensor([[7.47613e+02, 4.01168e+01, 1.14978e+03, 7.12016e+02, 8.71210e-01, 0.00000e+00],
    #         [1.17464e+02, 1.96875e+02, 1.00145e+03, 7.11802e+02, 8.08795e-01, 0.00000e+00],
    #         [4.23969e+02, 4.30401e+02, 5.16833e+02, 7.20000e+02, 7.77376e-01, 2.70000e+01],
    #         [9.81310e+02, 3.10712e+02, 1.03111e+03, 4.19273e+02, 2.86850e-01, 2.70000e+01]])

    xyxy = xyxy[xyxy[:, 4] >= confidence] # Filter desired confidence
    xyxy = xyxy[xyxy[:, 5] == 0] # Consider only people
    xyxy = xyxy[:, :4]

    colors = ['green']*len(xyxy)
    for i in range(len(xyxy)):
        for j in range(i+1, len(xyxy)):
            # Calculate distance of the centers
            dist, x1, y1, x2, y2 = center_distance(xyxy[i], xyxy[j])
            if dist < distance:
                # If dist < distance, boxes are red and a line is drawn
                colors[i] = 'red'
                colors[j] = 'red'
                img = cv2.line(img, (x1, y1), (x2, y2), (0, 0, 255), 2)
    for i, (x1, y1, x2, y2) in enumerate(xyxy):
        # Draw the boxes
        if colors[i] == 'green':
            color = (0, 255, 0)
        else:
            color = (0, 0, 255)
        img = cv2.rectangle(img, (x1, y1), (x2, y2), color, 2)
    return img
```

To detect people in a video, we iterate through all frames of the video, and save a new video with the rectangles drawn.

```
[ ] def detect_people_on_video(filename, confidence=0.9, distance=60):
    '''Detect people on a video and draw the rectangles and lines.'''
    # Capture video
    cap = cv2.VideoCapture(filename)

    # Get video properties
    fps = cap.get(cv2.CAP_PROP_FPS)
    width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

    # Define the codec and create VideoWriter object
    fourcc = cv2.VideoWriter_fourcc(*'XVID')
    if os.path.exists('output.avi'):
        os.remove('output.avi')
    out = cv2.VideoWriter('output.avi', fourcc, fps, (width, height))
```

```
# Iterate through frames and detect people
vidlen = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
with tqdm(total=vidlen) as pbar:
    while cap.isOpened():
        # Read a frame
        ret, frame = cap.read()
        # If it's ok
        if ret == True:
            frame = detect_people_on_frame(frame, confidence, distance)
            # Write new video
            out.write(frame)
            pbar.update(1)
        else:
            break

# Release everything if job is finished
cap.release()
out.release()
cv2.destroyAllWindows()
```

apply the detection tool.

```
[ ] detect_people_on_video(filename, confidence=0.5)
```

100%  1785/1785 [01:42<00:00, 17.84it/s]



# References

- [1] TV Team. Dj: Coronavirus: a visual guide to the outbreak. 6 mar, 2020.
- [2] Sergio Saponara, Abdussalam Elhanashi, and Qinghe Zheng. Developing a real-time social distancing detection system based on yolov4-tiny and bird-eye view for covid-19. *Journal of Real-Time Image Processing*, pages 1–13, 2022.
- [3] Probir Ghosh and Mohammad Manir Mollah. The risk of public mobility from hotspots of covid-19 during travel restriction in bangladesh. *The Journal of Infection in Developing Countries*, 14(07):732–736, 2020.
- [4] Faruque Ahmed, Nicole Zviedrite, and Amra Uzicanin. Effectiveness of workplace social distancing measures in reducing influenza transmission: a systematic review. *BMC public health*, 18(1):1–13, 2018.
- [5] Peter J Hotez. Covid-19 and the antipoverty vaccines. *Molecular Frontiers Journal*, 4(01n02):58–61, 2020.
- [6] Imam Husni Al Amin, Falah Hikamudin Arby, Edy Winarno, Budi Hartono, and Wiwien Hadikurniawati. Real-time social distance detection using yolov5 with bird-eye view perspective to suppress the spread of covid-19. In

- 2022 2nd International Conference on Information Technology and Education (ICIT&E)*, pages 269–274. IEEE, 2022.
- [7] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.
- [8] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.
- [9] Tien Dung Le. taka at the finsbd-3 task: Tables and figures extraction using object detection techniques. In *Companion Proceedings of the Web Conference 2021*, pages 280–282, 2021.
- [10] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.
- [11] Vernika Sapra, Rohan Gupta, Parikshit Sharma, Rashika Grover, and Urvasi Sapra. Eye4u-multifold protection monitor. In *Innovations in Computational Intelligence and Computer Vision*, pages 323–341. Springer, 2022.
- [12] Mohd Ansari, Dushyant Kumar Singh, et al. Monitoring social distancing through human detection for preventing/reducing covid spread. *International Journal of Information Technology*, 13(3):1255–1264, 2021.

- 
- [13] RITIK Shukla, ARPAN KUMAR Mahapatra, and J Selvin Paul Peter. Social distancing tracker using yolo v5. *Turkish Journal of Physiotherapy and Rehabilitation*, pages 1785–1793, 2021.
  - [14] Adina Rahim, Ayesha Maqbool, and Tauseef Rana. Monitoring social distancing under various low light conditions with deep learning and a single motionless time of flight camera. *Plos one*, 16(2):e0247440, 2021.
  - [15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
  - [16] Rui Li and Jun Yang. Improved yolov2 object detection model. In *2018 6th international conference on multimedia computing and systems (ICMCS)*, pages 1–6. IEEE, 2018.
  - [17] Liquan Zhao and Shuaiyang Li. Object detection algorithm based on improved yolov3. *Electronics*, 9(3):537, 2020.
  - [18] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.
  - [19] Jiangtao Qi, Xiangnan Liu, Kai Liu, Farong Xu, Hui Guo, Xinliang Tian, Mao Li, Zhiyuan Bao, and Yang Li. An improved yolov5 model based on visual attention mechanism: Application to recognition of tomato virus disease. *Computers and Electronics in Agriculture*, 194:106780, 2022.

- 
- [20] Hyun-Tae Kim and Sang-Hyun Lee. A study on object distance measurement using opencv-based yolov5. *International Journal of Advanced Culture Technology*, 9(3):298–304, 2021.
- [21] Imran Ahmed, Misbah Ahmad, Joel JPC Rodrigues, Gwanggil Jeon, and Sadia Din. A deep learning-based social distance monitoring framework for covid-19. *Sustainable Cities and Society*, 65:102571, 2021.
- [22] K Seemanthini and SS Manjunath. Human detection and tracking using hog for action recognition. *Procedia computer science*, 132:1317–1326, 2018.
- [23] Ye Li, Kangning Yin, Jie Liang, Chunyu Wang, and Guangqiang Yin. A multi-task joint framework for real-time person search. *arXiv preprint arXiv:2012.06418*, 2020.
- [24] Fangbo Zhou, Huailin Zhao, and Zhen Nie. Safety helmet detection based on yolov5. In *2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA)*, pages 6–11. IEEE, 2021.
- [25] Jiacong Fang, Qiong Liu, and Jingzheng Li. A deployment scheme of yolov5 with inference optimizations based on the triton inference server. In *2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, pages 441–445. IEEE, 2021.
- [26] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9197–9206, 2019.

- 
- [27] Jirarat Ieamsaard, Surapon Nathanael Charoensook, and Suchart Yammen. Deep learning-based face mask detection using yolov5. In *2021 9th International Electrical Engineering Congress (iEECON)*, pages 428–431. IEEE, 2021.
- [28] Krisha Bhambani, Tanmay Jain, and Kavita A Sultanpure. Real-time face mask and social distancing violation detection system using yolo. In *2020 IEEE Bangalore Humanitarian Technology Conference (B-HTC)*, pages 1–6. IEEE, 2020.
- [29] Aleksa Ćorović, Velibor Ilić, Siniša urić, Mališa Marijan, and Bogdan Pavković. The real-time detection of traffic participants using yolo algorithm. In *2018 26th Telecommunications Forum (TELFOR)*, pages 1–4. IEEE, 2018.
- [30] Narayana Darapaneni, Shrawan Kumar, Selvarangan Krishnan, Arunkumar Rajagopal, Anwesh Reddy Paduri, et al. Implementing a real-time, yolov5 based social distancing measuring system for covid-19. *arXiv preprint arXiv:2204.03350*, 2022.