ANALYSIS OF HUMAN DETECTION METHOD IN SOCIAL DISTANCING MONITORING

NUR AINA SYAFINAZ BINTI MUHAMAD ATFAN

B.ENG (HONS.) ELECTRICAL ENGINEERING (ELECTRONICS)

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name	:	NUR AINA SYAFINAZ BINTI MUHAMAD ATFAN
Date of Birth	:	13/11/1999
Title	:	ANALYSIS OF HUMAN DETECTION METHOD IN SOCIAL DISTANCING MONITORING
Academic Session	:	2021/2022

I declare that this thesis is classified as:

☐	CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐	RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☒	OPEN ACCESS	I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

____
(Student's Signature)

991113105352
____
New IC/Passport Number
Date:17/06/2022

____
(Supervisor's Signature)

DR. ROSDIYANA SAMAD
____
Name of Supervisor
Date: 27/06/2022

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.

THESIS DECLARATION LETTER (OPTIONAL)

Librarian,
Perpustakaan Universiti Malaysia Pahang,
Universiti Malaysia Pahang,
Lebuhraya Tun Razak,
26300, Gambang, Kuantan.

Dear Sir,

CLASSIFICATION OF THESIS AS RESTRICTED

Please be informed that the following thesis is classified as RESTRICTED for a period of three (3) years from the date of this letter. The reasons for this classification are as listed below.

Author's Name

Thesis Title

Reasons (i)

(ii)

(iii)

Thank you.

Yours faithfully,

____

(Supervisor's Signature)

Date: 27/June/2022

Stamp: DR. ROSDIYANA SAMAD
PENSYARAH KANAN
FAKULTI TEKNOLOGI KEJURUTERAAN ELEKTRIK DAN ELELTRONIK
UNIVERSITI MALAYSIA PAHANG
26600 PEKAN
PAHANG DARUL MAKMUR
TEL: 09-4246099 FAX: 09-4246111

Note: This letter should be written by the supervisor, addressed to the Librarian, *Perpustakaan Universiti Malaysia Pahang* with its copy attached to the thesis.

Universiti
Malaysia
PAHANG
Engineering • Technology • Creativity

SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and, in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of B.Eng (Hons.) Electrical Engineering (Electronics).

____

(Supervisor's Signature)

Full Name : Dr. Rosdiyana Bt. Samad

Position : Senior Lecturer

Date : 27/06/2022

____

(Co-supervisor's Signature)

Full Name :

Position :

Date :

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

____

(Student's Signature)

Full Name : NUR AINA SYAFINAZ BINTI MUHAMAD ATFAN

ID Number : EA18088

Date : 17/06/2022

ANALYSIS OF HUMAN DETECTION METHOD IN SOCIAL DISTANCING MONITORING

NUR AINA SYAFINAZ BINTI MUHAMAD ATFAN

Thesis submitted in fulfillment of the requirements
for the award of the degree of
B.Eng (Hons.) Electrical Engineering (Electronics)

College of Engineering

UNIVERSITI MALAYSIA PAHANG

JUNE 2022

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Allah SWT. because with His guidance, love and blessing. Without His blessing, I could not make it to complete this Final Year Project 2 (FYP2) report for my bachelor's degree level.

Next, I would like to acknowledge my special gratitude towards my supervisor, Dr. Rosdiyana Bt. Samad, because her endless positive comment and support towards me upon completion of FYP2 and FYP2 report. I will be eternally grateful and indebted to her for sharing her knowledge, and skills and guiding me to choose the correct method in this study of FYP2.

I would like to extend my sincere thanks to my beloved parents because of their lovely support and prayer for me even though they are not by my sides upon completing this report but they are always in my heart. Next, thank you to my siblings for giving me moral support to me keep going.

Lastly, I would like to thank my best friends Zahrah, Hidayah, Owie, and Farra for their physical encouragement and mental support all the way to completing this project.

ABSTRAK

Penjarakan sosial adalah amalan bukan perubatan yang membantu melambatkan penularan virus yang dicadangkan oleh Pertubuhan Kesihatan Sedunia (WHO). Penjarakan sosial bermaksud bahawa setiap kali orang ingin bersosial, mereka perlu mematuhi jarak sosial minimum 2 meter selain daripada orang di sekeliling mereka. Walau bagaimanapun, memandangkan setiap negara telah memerangi penyebaran virus selama hampir tiga tahun, amalan penjarakan sosial kini seolah-olah diabaikan oleh orang ramai kerana beberapa sebab seperti mereka tergesa-gesa. Kajian ini bertujuan untuk menganalisis pengesanan manusia menggunakan kaedah "*deep learning*" dalam pelbagai kedudukan dan untuk membangunkan pengesanan jarak sosial menggunakan kaedah yang dicadangkan. Pengesanan bermaksud mengenali kehadiran sesuatu seperti manusia atau objek. Dalam bidang pengesanan, beberapa kaedah sedang digunakan sama ada teknologi tradisional atau moden. Oleh itu, kaedah yang dicadangkan dalam pengesanan manusia dalam pemantauan jarak sosial adalah menggunakan algoritma "*deep learning"* iaitu *You Only Look Once (YOLO)* versi 3 dengan set data yang dipilih. Dalam kaedah ini, formula jarak Euclidean digunakan untuk mengukur jarak antara dua orang dengan pengiraan tambahan yang lain untuk menghasilkan output jarak terukur adalah tepat dengan pengukuran sebenar. Set data dikumpulkan dari sumber dalam talian yang berasal dari Kaggle, Unsplash, dan iStock. Terdapat kira-kira 403 imej yang digunakan untuk kajian ini. 282 imej digunakan untuk proses latihan dan 121 imej digunakan untuk proses ujian. Terdapat 300 set data tambahan sendiri digunakan untuk menguji pengesanan manusia dan pengesanan jarak sosial menggunakan model pengesanan yang sama. Google Colaboratory digunakan sebagai platform pengaturcaraan dengan penggunaan bahasa pengaturcaraan Python. Set data yang dipilih kemudiannya dilabel secara manual terlebih dahulu menggunakan perisian Labelimg sebelum melalui proses latihan. Hasilnya, set data semasa untuk setiap kedudukan seperti pandangan depan, pandangan belakang, pandangan sisi, dan orang ramai memberikan hasil pengesanan manusia masing-masing pada 94.44%, 91.67%, 97.50%, dan 88.89%. Oleh itu, ketepatan tertinggi pengesanan manusia adalah pada kedudukan pandangan sisi dengan peratusan ketepatan 97.50%. Selain itu, dengan menggunakan model yang sama keatas set data tersendiri memberikan ketepatan peratusan keseluruhan iaitu 98.00% dalam mengesan manusia. Seterusnya, pengesanan jarak sosial berjaya dibangunkan dengan menggunakan formula jarak Euclidean dengan mendarabkannya dengan mengira nisbah piksel hingga sentimeter iaitu 0.1978cm. Oleh itu, jarak yang diukur adalah dalam perwakilan sentimeter. Oleh itu, penjarakan sosial yang dikesan juga menghasilkan julat yang boleh diterima pada ±0.3cm. Untuk perancangan masa depan, set data perlu berada di sekitar seribu imej untuk proses latihan untuk membuat peningkatan ketepatan pengesanan. Kajian ini menyimpulkan bahawa penggunaan algoritma pengesanan objek yang dicadangkan semasa boleh digunakan untuk mengesan manusia dan kaedah jarak Euclidean berjaya dilaksanakan bersama-sama dengan algoritma pengesanan objek.

ABSTRACT

Social distancing is a non-medical practice that helps slow down the transmission of viruses which is suggested by the World Health Organization (WHO). Social distancing means that whenever people want to socialize, they need to adhere minimum suggested social distance of 2 meters apart from the people around them. However, as every country has battled the spread of the virus for almost three years, social distancing practice now seems ignored by public people due for some reason such as they are in a rush. This study aims to analyze human detection using the deep learning method in various positions and to develop social distancing detection using the proposed method. Detection means to recognize the presence of something such as human or object. In the area of detection, several methods are being used either traditional or modern technology. The proposed method for human detection in social distancing monitoring is by using a deep learning algorithm which is *You Only Look Once* (YOLO) version 3 with custom datasets. In this method, the Euclidean distance formula is used for measuring the distance between two people with another additional computation to produce the output measured distance true to the real-life measurement. The datasets are collected from online sources which are from Kaggle, Unsplash, and iStock. There are about 403 images is used for this study. 282 images are used for the training process and 121 images are used for the testing process. There are another additional 300 customed datasets are used to test the human detection and social distancing detection using the same detection model. Google Colaboratory is used as a programming platform with the use of Python programming language. The custom datasets are being labeled first using Labelimg software before going through the training process. As a result, the current datasets for each position such as front view, back view, side view, and the crowd gives the result of human detection at 94.44%, 91.67%, 97.50%, and 88.89% respectively. Hence, the highest accuracy of human detection goes to the side view position with a percentage accuracy of 97.50%. Besides, using the same model for customed dataset gives overall percentage accuracy which is 98.00% in detecting humans. Next, the social distance detection is successfully developed by using the Euclidean distance formula by multiplying it by a calculated ratio of a pixel to centimetre which is 0.1978cm. Therefore, the measured distance is in centimetres representation. Hence, the detected social distancing result is also in the range of acceptable at ±0.3cm. For future planning, datasets need to be around thousand images for the training process to make the detection accuracies increase. This study concludes that the use of current proposed object detection algorithms can be used to detect humans and the Euclidean distance method is successfully implemented together with the object detection algorithm.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

D	Total distance in pixel
Dc	Real-life total distance in centimetres

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Networks
COCO	Common Object in Context
COVID-19	Corona Virus Disease 2019
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
GPU	Graphic Processing Unit
IoU	Intersection over Union
mAP	Mean Average Precision
NMS	Non-Maximum Suppression
PNG	Portable Network Graphic
ReLU	Leaky Rectified Linear Unit
SOP	Standard Operating Procedures
WHO	World Health Organization
YOLO	You Only Look Once
YOLOV3	You Only Look Once version 3

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Introduction

According to the Cambridge dictionary, social distancing is defined as keeping a certain distance from other people. Early in March 2020, the world was announced as a global pandemic due to the existence of coronaviruses, COVID-19. During the pandemic, all countries are having lockdown for about three months to slow down the transmission of the viruses. Generally, COVID-19 is a newly found strain of the coronavirus group that is identified in humans. This virus can cause respiratory illness in humans as it will attack the human lungs and make humans uneasy to breathe. Coronaviruses are a broad group of viruses that can cause illnesses ranging from the common cold to more serious illnesses like Severe Acute Respiratory Syndrome (SARS) and Middle East Respiratory Syndrome (MERS). Research that is being done by the medical profession and scientists, finds out that there are symptoms that may appear to those who are infected by this virus. The symptoms are such as fever, cough, headache, loss of taste or smell, sore throat, shortness of breath, shaking with chills, and muscle pain. However, there are cases where the patient infected by COVID-19 are not having any of the symptoms and are classified as asymptomatic. Until October 20, 2021, about 241 million confirmed cases are being recorded according to the WHO website. This virus first being detected in November 2019 where a 55-years-old individual from Hubei Wuhan, China (Bryner J, 2020). The virus spread when one person who is infected by the virus itself breaths out droplets. The droplets contain the SARS Covid virus where the droplet may be breathed in by other people who have close contact with the infected person through the nose, mouth, or even eyes.

"Closed Contact" meaning that is defined by the CDC or Center for Disease Control and Prevention as being less than 6 feet from someone who is infected with the COVID-19 for at least 15 minutes. Therefore, to slow the spread of these viruses WHO encourage people to follow the Standard Operating Procedure (SOP) such as wearing a face mask, avoiding crowd places, and practicing social distancing. Practicing social distancing is one of the ways to prevent the spread of this coronavirus. People are suggested to keep individual distance from other people by at least one-meter distance. It is important to the higher-risk group from getting or experiencing severe diseases caused by the virus. This social distancing is being practiced in public areas such as restaurants, malls, and hypermarkets. But, nowadays after battling for almost 3 years with the viruses, people slowly ignore the social distance as suggested by the WHO just because the number of cases slowly decreases as the rate of vaccination increases. Also, people still cannot follow the new normal and adhere to the social distance because still cannot cope with the situation post-COVID-19. Figure 1.1 below shows the infographic of the social distance minimum set by the Ministry of Health (MOH) Malaysia. Whilst, Figure 1.2 shows the infographic of social distancing in the public area.

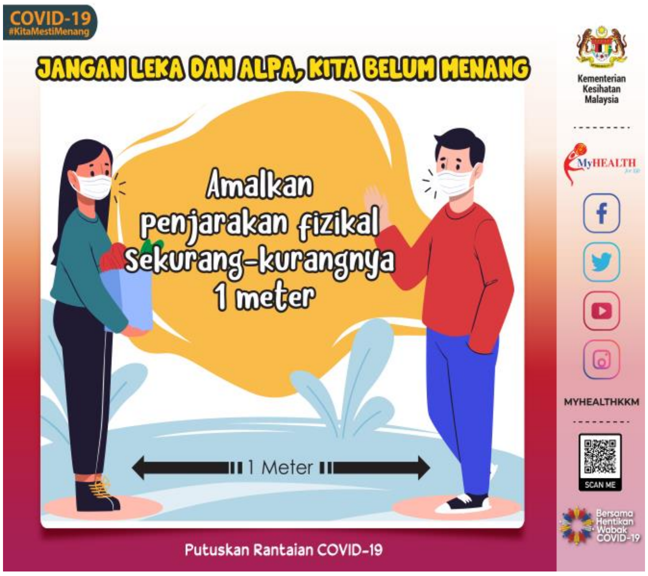

COVID-19
#KitaMestiMenang
JANGAN LEKA DAN ALPA, KITA BELUM MENANG
Amalkan penjarakan fizikal Sekurang-kurangnya 1 meter
1 Meter
Putuskan Rantaian COVID-19
Kementerian Kesihatan Malaysia
MyHEALTH
MYHEALTHKKM
SCAN ME
Bersama Hentikan Wabak COVID-19

Figure 1.1 Social Distance Infographic by Ministry of Health Malaysia (MOH)
Source: Ministry of Health Malaysia (n.d)

COVID-19
Restoran
Membeli makanan di restoran
STESEN MINYAK
Mengisi minyak di stesen minyak
Menggunakan kemudahan pengangkutan awam
JARAK SOSIAL 1 METER
ATM
Membuat urusan di ATM / bank
Farmasi
Membeli ubat di farmasi
Membeli barang keperluan di pasaraya
Kementerian Kesihatan Malaysia
myhealthkkm
MyHEALTH
SCAN ME

Figure 1.2 Infographic of Social Distancing in Public Area
Source: Ministry of Health Malaysia (n.d)

Recognition of something's presence, such as a human or an object is defined as detection. There are numerous techniques in the field of detection that make use of both classical and technologically advanced. Generally, detection in traditional ways required someone to observe the situation compared to the modern technologies detection which can be done with the help of a computerized system. For example, traditionally the detection of the human is made by an assigned person who needs to observe the human by looking at the human appearance itself. But, in modern detection technology, a human can be detected by the use of deep learning applications which requires more data in training to make the system work well in detecting humans. In this study, human detection and social distance detection between people will be the focus. Firstly, the detection process will detect the human first, and then it will continuously measure the distance between two peoples. Then, the output results are analyzed into a few parts which are the accuracy of human detection in various positions, the accuracy of human detection based on the number of humans per image in various positions, the accuracy of human detection of custom datasets, and social distancing detection of custom datasets.

1.2 Problem Statement

Social distancing means that whenever people want to socialize, they need to adhere minimum suggested social distance of 2 meters apart from the people around them. Unfortunately, nowadays social distancing compliance is less practice by the public, especially in the public environment such as libraries, lecture halls, supermarkets, and restaurants. The reason of non-compliances is because some people are in rush and still cannot cope with the new normal Standard Operating Procedures (SOP). For example, in a supermarket scenario, the buyers need to come near to the rack of their choice to take the food or their essentials at the same time, other people at that area also look up to the same things. Therefore, the social distance cannot be controlled during that time. Otherwise, the person itself needs to be more patient by waiting for the other buyers away from the rack.

Secondly, implementing the social distancing monitoring application might be new to Malaysia. Although many researchers developed the system in other countries, there is still has lacked in the detection of humans. The lack is covered in terms of the accuracy of the object detection. This is because of the obstacle of crowded areas that leads to difficulty in detecting people or not in a busy place. For example, of inaccuracy, the program itself is set to detect a tall object as a human. In this case, false detection can occur, since it focuses on high specification objects where a tree can also have those criteria.

1.3 Objectives

This study was done to analyze the human detection method for social distancing monitoring. The following objectives were set to achieve the aim of the study;

i. To analyze human detection using deep learning method in various positions.

ii. To develop social distancing using the proposed model.

1.4 Scope of Project

To achieve the objective of this study, a few scopes are being focused on such as volunteers for data collection, laptop requirements, measuring tools, source of datasets (online), and programming software. The participants of volunteers for this research are among UMP’s students themselves. To run the system, a suitable laptop is needed such as a laptop that has a processor by Intel Corp. version Intel Core i5. For recording, smart phone Samsung Galaxy A52 64MP Quad phone camera is used to record the volunteer standing and queuing at targeted areas for testing purposes. A measuring tool such as measuring tape is used to set up the distance between the volunteer. Besides, the selected programming platform is Google Colaboratory with the use of Python programming languages. The datasets are cropped using Lableimg software in order to generate text files of images for the training process. PhotoDirector by Acer application is also being used to extract the video into a few images.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter will review the relevant literature and research related to the method of object detection for social distancing monitoring systems that are implemented due to the pandemic Covid-19. This chapter first will discuss the Object Detection in Social Distancing Detection, then followed by the Object Detection Algorithms Evolutions. Then, the next part is focusing on the Social Distancing Detection method and most methods used in object detection methods in the social distancing monitoring system.

2.2 Object Detection in Social Distancing Detection

Object detection is one of the hotspots of study on the subject of computer vision (Qin & Xu, 2021). Object detection is the process of locating and detecting objects in a video stream (Melenli S. et al, 2020). In other words, object detection is to finding semantic object instances that belong to specific classes (Magoo et al., 2021; Yang et al., 2021). Furthermore, object detection is also defined as multiple items in a picture that can be recognized, detected, and localized using object detection. In the computer vision process of categorization and localization of its form in video footage, human detection may be regarded as object detection (Yew Cheong Hou et al, 2020).

(Ahmed et al., 2021; Srinivasan et al., 2021) proposed You Only Look Once (YOLOV3) object detection methods to recognize a person in a video surveillance dataset Whilst, (Hou et al., 2020; Magoo et al., 2021; Mercaldo et al., 2021; Puspita et al., 2021a; Sathyabama et al., 2020) also proposed the same method which is using YOLOV3 to detect the object in the video but this time for real-time detection. This show that the YOLOV3 object detection method is suitable to be used in detecting human either in real-time video or surveillance video dataset. This is because YOLOV3 is the fastest object detection method as it requires only one stage of detection. YOLOV3 is trained to do classification and box regression at the same time. In addition, this method gives results in terms of accuracy of around 92% (Ahmed et al., 2021). Other than that, the YOLOV3 algorithm has proven results in correctly detecting all humans (Mercaldo et al., 2021). Object detection in the YOLOV3 algorithm impressively can detect half of the human body as an object in pedestrian videos (Shalini et al., 2021).

Other than that, a new object detection model, Detection Transformer (DETR) is proposed by Nicholas Carion, Francisco Massa, Gabrield Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko (Carion et al., n.d.). This model is used to detect a human as it builds to recognize about 91 categories compared to YOLOV3 which is only capable to recognize 80 categories. This model works together with SocialNet which results in the accuracy of the bounding box. Besides, this model successfully developed a proposed bounding box to locate the predicted centre of bounding boxes. Figure 2.1 shows the overall architecture of SocialNet. Three different backbones are used to train the SocialNet. The backbones used are MobileNet, ResNet50, and VCG16. These backbones are called Auto-encoder backbones. DETR uses an encoder-decoder structure to force unique predictions. Between three backbones, the ResNet50 provides the best accuracies as it reaches 95.4% (Elbishlawi et al., 2021).

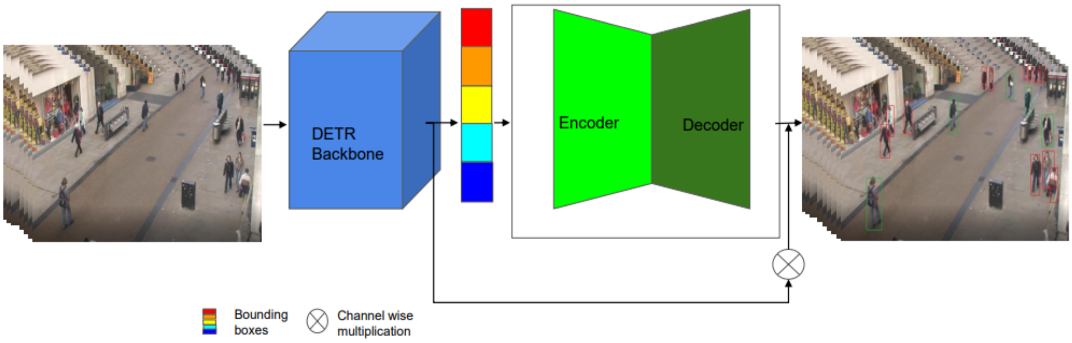


Figure 2.1 Overall Architecture of SocialNet
Source: (Elbishlawi et al., 2021)

Next, the object detection method is MobilenetSSD. Mobilenet is defined as a Convolutional Neural Network (CNN) that can be used to avoid the use of excessive computational resources (Puspita et al., 2021). The author compared the YOLOV3 with

MobilenetSSD performance in terms of average GPU runtime per second where MobilenetSSD 1.5 per seconds whilst YOLOV3 17 per seconds. SSD is Single Shot Multi-Box Detector where it is proven to have higher speed compared to YOLOV3 but in terms of accuracy YOLOV3 algorithm has higher accuracy compared to SSD (Ambika Neelopant et al., 2021).

Besides, (Saponara et al., 2021) used the second version of YOLO to detect the people. YOLOV2 is an object detection system that is designed to work in real-time (Saponara et al., 2021). YOLOV2 is the better version of YOLO compared to the first version. It also improves in terms of accuracy and speed during the process from the first version. MATLAB software is used to construct the neural network layers of YOLOV2 and being trained using Stochastic gradient decent. Stochastic gradient descent is the optimizer of the neural network (Glorot & Bengio, n.d.).

2.3 Object Detection Algorithms Evolutions

Initially, object detection is implemented without using deep learning. Hence, some researchers use traditional methods to detect an object. The traditional methods of object detection are Viola-Jones Detectors, HOG Detector, and Deformable Part-based Model (DPM)(Chinmoy Borah, 2020). This traditional method is being used in 2014 and backward. Since 2014, deep learning-based detection has existed. Deep learning-based detection is divided into two types which are “two-stage detection” and “one-stage detection” (Chinmoy Borah, 2020). The deep learning-based detection is started by a convolution neural network which is Region Convolution Neural Network (RCNN), followed by SPPNet which is a Spatial Pyramid Pooling Network, Fast RCNN, Faster RCNN, Feature Pyramid Networks (FPN), and the latest one is You Only Look Once (YOLO) and Single Hot Multi-Box Detector (SSD). Figure 2.2 below shows the Object Detection Milestone from traditional Detection Methods to Deep-Learning based Detection Methods.

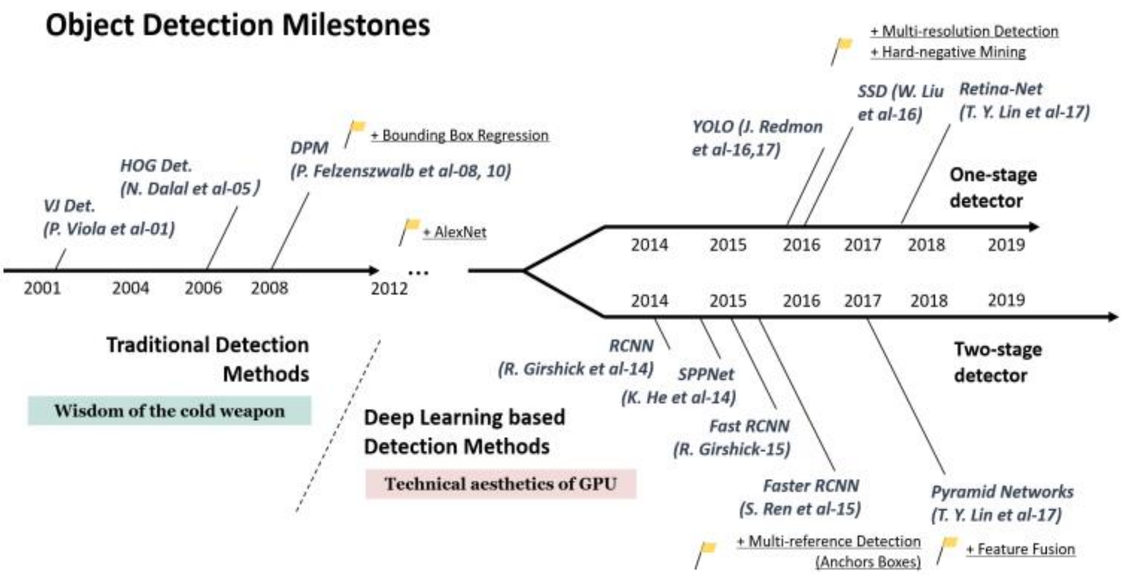

Object Detection Milestones
VJ Det.
(P. Viola et al-01)
HOG Det.
(N. Dalal et al-05)
DPM
(P. Felzenszwalb et al-08, 10)
+ Bounding Box Regression
+ AlexNet
2001
2004
2006
2008
2012
Traditional Detection Methods
Wisdom of the cold weapon
Deep Learning based Detection Methods
Technical aesthetics of GPU
YOLO (J. Redmon et al-16,17)
SSD (W. Liu et al-16)
Retina-Net (T. Y. Lin et al-17)
+ Multi-resolution Detection
+ Hard-negative Mining
One-stage detector
2014
2015
2016
2017
2018
2019
Two-stage detector
RCNN (R. Girshick et al-14)
SPPNet (K. He et al-14)
Fast RCNN (R. Girshick-15)
Faster RCNN (S. Ren et al-15)
Pyramid Networks (T. Y. Lin et al-17)
+ Multi-reference Detection (Anchors Boxes)
+ Feature Fusion

Figure 2.2 Object Detection Milestone from 2001 to 2019
Source: (Zou et al., 2019)

2.4 Social Distancing Detection

As COVID-19 has been announced as a pandemic starting in March 2020, few guidelines are being posted to the public in order to reduce the emission of the coronavirus. Practicing social distancing is one of succeed methods. The Ministry Health of Malaysia Government (MOH) also suggests to the public to follow at least one-meter distance during socializing. In social distancing detection, the distance between two people is measured to classify either violation in social distancing occurs or does not occurs. Traditionally, in a public area such as the supermarket. Customers need to line up with a certain distance between customers by standing at the marked tape on the floor provided by the supermarket management. The tapes are marked in Figure 2.3 below. Else, the security that on duty will monitor people's distance manually.

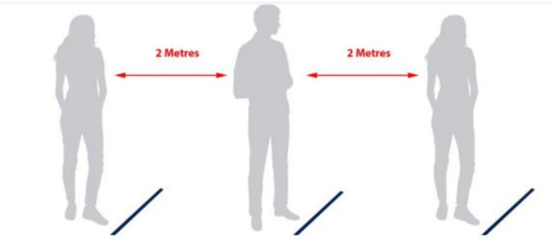


Figure 2.3 Marking Tape for Social Distancing
Source: (*Marking Tape for Social Distancing - Heskins*, 2020)

Thus, to ease the current ways, the existence of deep learning helps in promising a better solution for this problem. To measure the distance between people as Figure 2.4 below, the approaches of clustering and distance-based algorithms are used. Based on the author (Ramadass et al., 2020) the proposed approach to determine the social distancing violence is by being embedded the drone camera to the trained YOLOV3 custom data by using the software Microsoft Vott tool which requires the author to label the custom data into 4 classes which are people with mask, people without mask, social distance and no social distance. This method seems to lack in accuracies since it uses estimation where there is no mathematical calculation used in measuring the distance between two peoples. Thus, this method is not suggested since it will cause unbalanced in the detection of social distance.

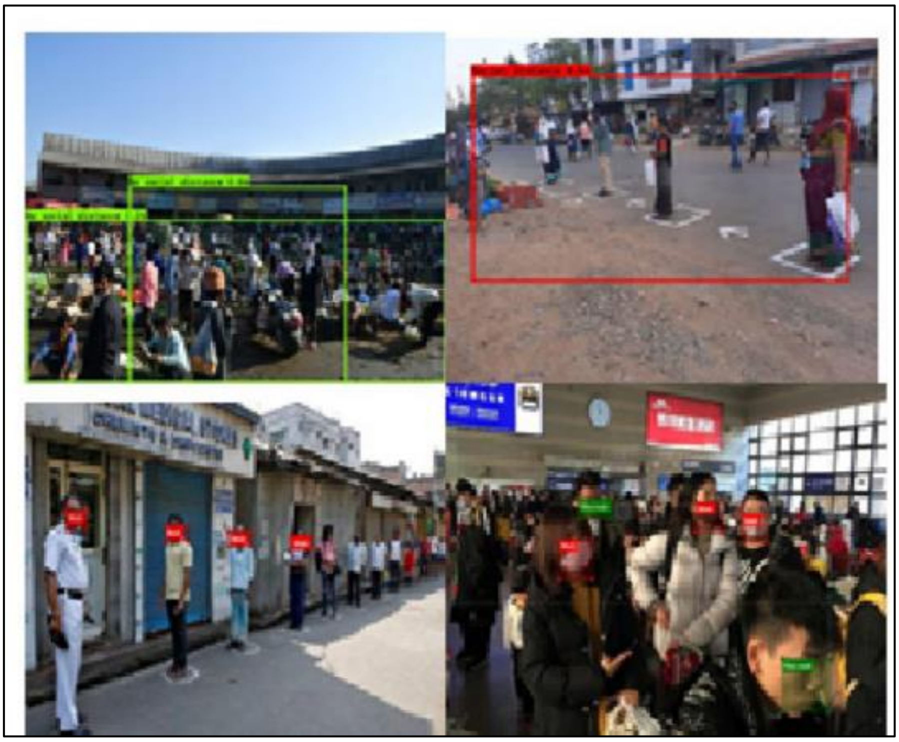


Figure 2.4 Social Distancing Detector
Source: (Ramadass et al., 2020)

Other authors proposed a mathematical approach to social distancing by the use of the Euclidean Distance approach (Ahmed et al., 2021; Hou et al., 2020; Karaman et al., 2021; Keniya & Mehendale, 2020; Magoo et al., 2021; Mercaldo et al., 2021; Puspita et al., 2021a; Qin & Xu, 2021; Sathyabama et al., 2020; Shalini et al., 2021; Yadav, 2020). This method used the mathematical expression of equation 2.1. The distance is measured by using the detected bounding box and the centroids value of the bounding box. By using pixel-to-distance assumptions, a predefined minimum social distance violation criterion is constructed. Therefore, to inspect whether the calculation of the distance is below or above the estimated values threshold, the green bounding box will appear if the estimated value of the threshold is similar to the calculated value. Otherwise, a red bounding box will appear which indicated there is a violation occurred (Ahmed et al., 2021).

$$euclidean\ distance(d) = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \qquad 2.1$$

Next, the Density-based spatial clustering of application with noise (DBSCAN) algorithm is also being proposed by (Srinivasan et al., 2021). DBSCAN is another method to measure the distance between two peoples and then perform clustering to recognize is there any violation of social distancing or not. The author also said this method is likely to be effective compared to other clustering methods. Besides, this method falls into the unsupervised method where it can class similar points together into a group. Therefore, when the system detects a human, it will class the midpoint of the bounding boxes. Figure 2.5 below shows how DBSCAN works in detecting social distancing. The blue circle represents the persons who are at an unsafe distance which is less than 2 meters. This method gives an accuracy result of social distance detection of around 89.79% which is left by 2.21% compared to using Euclidean distance method (Ahmed et al., 2021; Srinivasan et al., 2021).

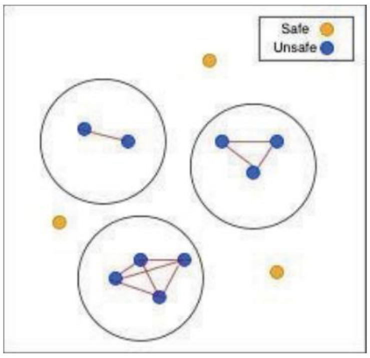


Figure 2.5 Clustering the Midpoint Using DBSCAN
Source: (Srinivasan et al., 2021)

2.5 Summary

To summarize, there are many methods for object detection in social distancing monitoring systems. The use of object detection can detect the presence of human or non-human in a frame as it will be the subject of the detection. To develop the human detection an algorithm as YOLO is required. From previous findings, it can be concluded that YOLOV3 algorithm gives fast and accurate detection because this algorithm improved to be able to detect even smaller objects. Besides, this algorithm runs in a faster manner compared to other algorithms. Next, the social distancing detection can be made by calculating the distance between two centroids as in the Euclidean distance formula that can be implemented in the python programming inside the YOLOV3 algorithm.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In this study, the preferable coding language used is Python language as the language is easy to understand. This programming language is used in the coding software which is Google Colaboratory. In Google Colaboratory, the use of Google Drive is used for calling the file path of the trained and tested files. Besides, this online software provides 12GB of free NVIDIA GPU up to certain hours. Thus, this chapter will explain deeply all the proposed methods used for human detection and social distancing detection.

3.2 Overall Proposed Framework

Figure 3.1 below shows the overall block diagram for the proposed method of social distancing monitoring. This method is separated into seven different processes which initially start by collecting datasets and creating the database, followed by training the human detection model, analyzing training parameters, selecting the final model, developing human detection by utilizing the trained model, testing the model with a new dataset, and testing the social distancing detection. In object detection, the testing and the training process are required to complete the system development. Before the training process, collected datasets need to separate into two categories which are training and testing images. Once the training process is done, the testing process is continued. Both processes used Google Colaboratory software in a python programming language with YOLO version 3 algorithm.

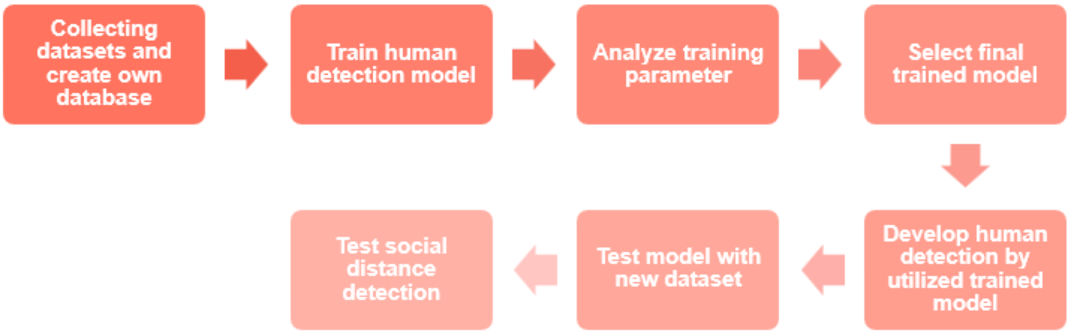

Collecting datasets and create own database
Train human detection model
Analyze training parameter
Select final trained model
Develop human detection by utilized trained model
Test model with new dataset
Test social distance detection

Figure 3.1 Overall Block Diagram for the Proposed Method

Figure 3.2 shows the overall flowchart of this project. Initially, the project is started by collecting datasets and creating a database. The dataset preparation is consisting of online source datasets and custom datasets. The online datasets are taken from the website such as Kaggle, Unsplash, and iStock. For custom datasets, there are a few videos of various positions of humans that are extracted as images. Next is the human detection process, which is divided into two parts, the training, and the testing part. First, the training process is started by calling the training image dataset that is labelled together with the text file and then creating a file path to the google drive to save the output training weight file. For the testing process, the weight file of the training process is called together with the configuration file which consists of the algorithm parameter details. Next, the testing process is started by calling the image file path of the testing image dataset to start the human detection in each of the images. The detector will label the found human with “human” and create the green bounding box. Then, whenever there are two bounding boxes or more, the system will start measuring the distance between the centre of the bounding box. If the distance is greater or equal to the safe distance that has been set, the output images will display green bounding boxes together with the distance value in centimetre. Otherwise, the output image will display the red bounding boxes.

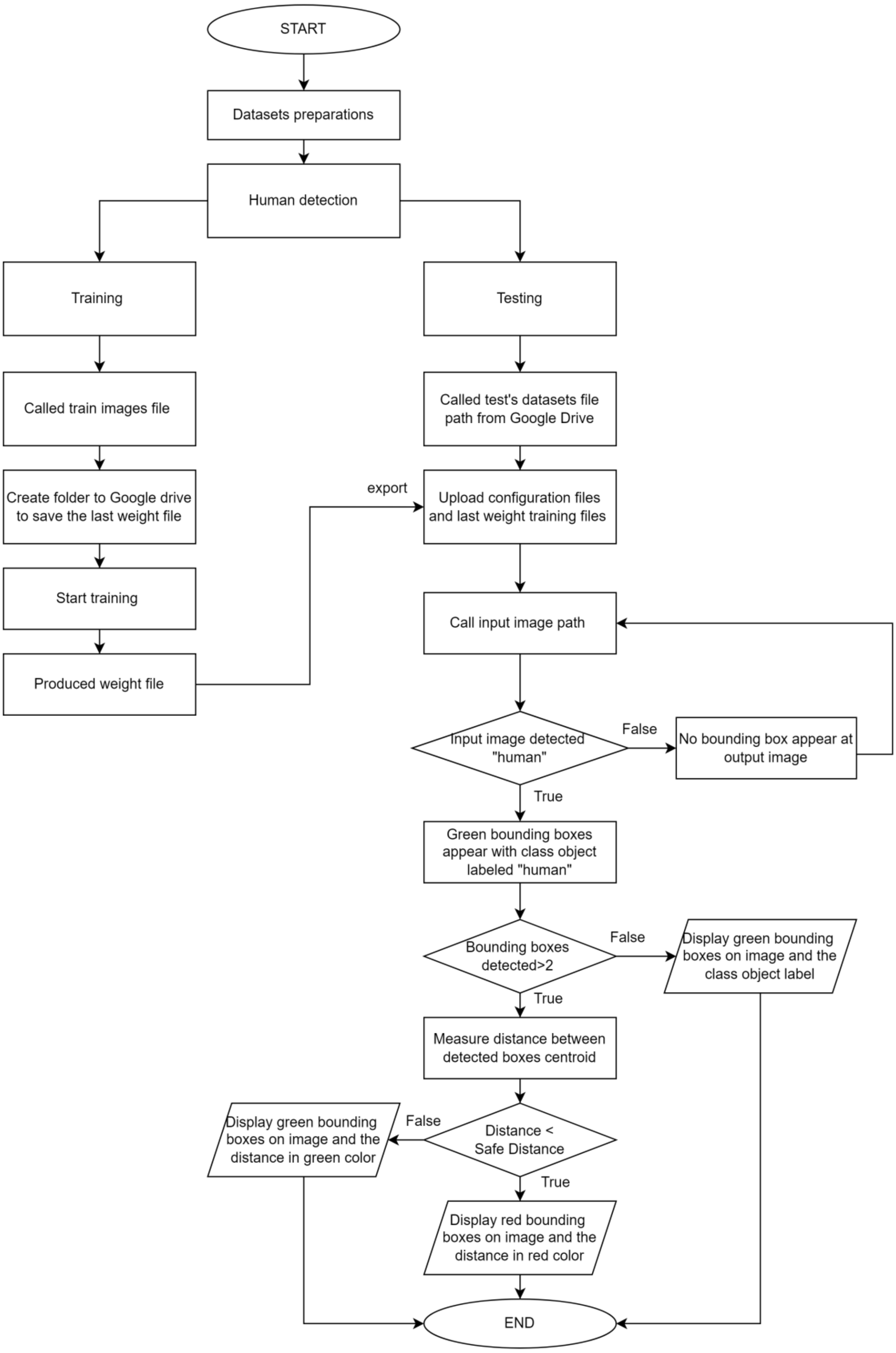

START
Datasets preparations
Human detection
Training
Testing
Called train images file
Called test's datasets file path from Google Drive
Create folder to Google drive to save the last weight file
export
Upload configuration files and last weight training files
Start training
Call input image path
Produced weight file
Input image detected "human"
False
No bounding box appear at output image
True
Green bounding boxes appear with class object labeled "human"
Bounding boxes detected>2
False
Display green bounding boxes on image and the class object label
True
Measure distance between detected boxes centroid
Display green bounding boxes on image and the distance in green color
False
Distance < Safe Distance
True
Display red bounding boxes on image and the distance in red color
END

Figure 3.2 Overall Flowchart

3.3 Data Collection

In this study, the datasets of humans are retrieved from online sources such as Unsplash, and iStock. The dataset is selected based on the various human position such as front view, back view, side view, and crowded. There are 77 images of front view, 70 images of back view, 72 images of side view, and 63 images of crowded. Various positions can make the model more accurate since the trained dataset will be fed with that such positions. Overall, there are about 282 images are selected for the training process. These numbers of the dataset are based on 70% of the total image. The total images for the training and testing using online datasets are about 403 images. The other 30% is for the testing process. Table 3.1 below shows the division of the dataset based on position and its source. Figure 3.3, Figure 3.4, Figure 3.5, and Figure 3.6 show some of the training datasets images based on their position.

Table 3.1 Training Datasets

Source	**Position**			
	Front View	**Back View**	**Side View**	**Crowded**
Unsplash	20	15	9	32
Kaggle	0	0	0	0
iStock	57	55	63	31
Total Image	77	70	72	63

Figure 3.3 Front View

Figure 3.4 Back View

Figure 3.5 Side View

Figure 3.6 Crowded

Next, several data which represent 30% of total datasets which around 121 images are retrieved from online sources as declared in Table 3.2. Figure 3.7, Figure 3.8, Figure 3.9, and Figure 3.10 shows some of the testing images that were used in testing the model.

Table 3.2 Testing Datasets

Source	**Position**			
	Front View	**Back View**	**Side View**	**Crowded**
Unsplash	14	6	15	3
Kaggle	4	15	11	3
iStock	18	15	14	3
Total Image	36	36	40	9

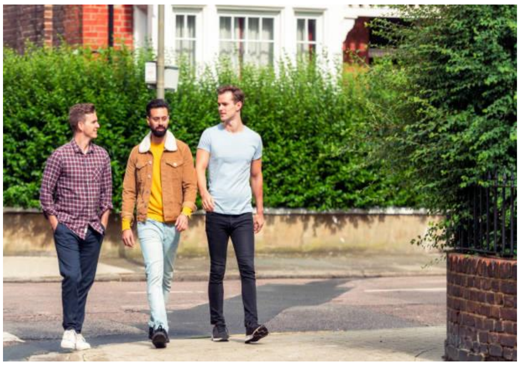


Figure 3.7 Front View

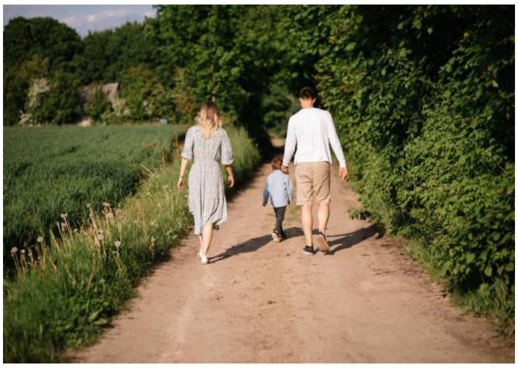


Figure 3.8 Back View

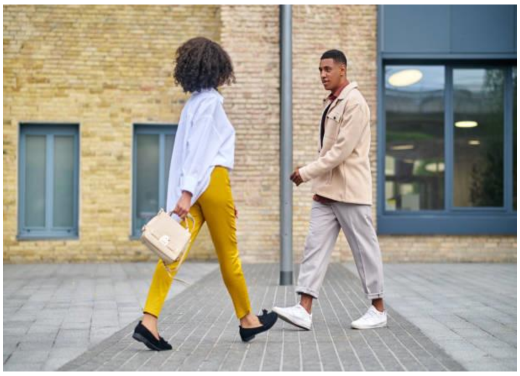


Figure 3.9 Side View

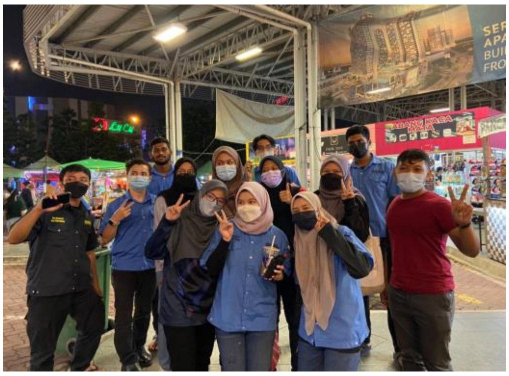
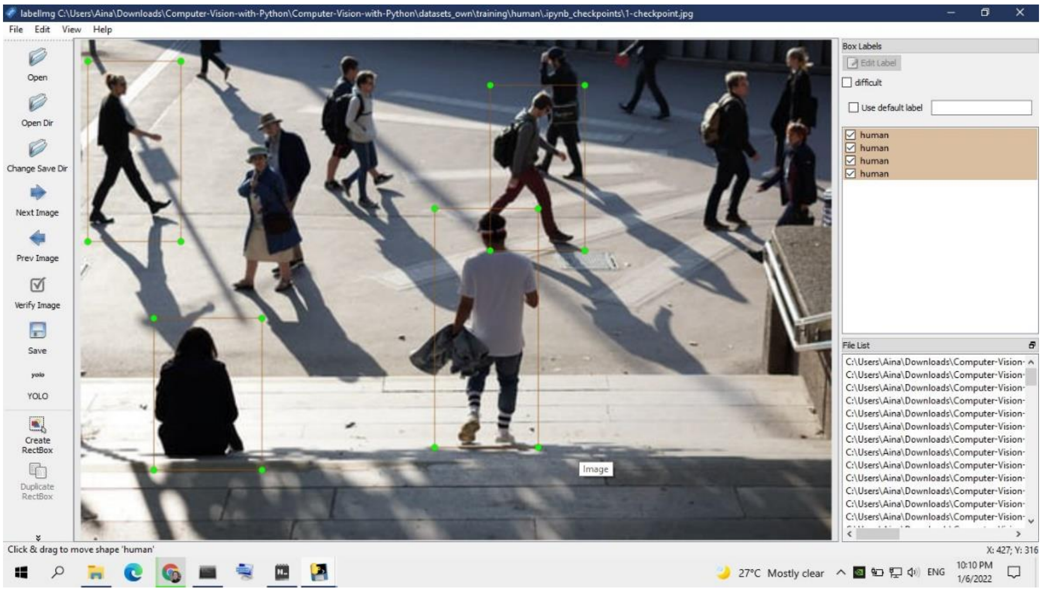


Figure 3.10 Crowded

3.4 Image Cropping and Labelling Using LabelImg Software

Generally, in YOLO version 3 the most used datasets are called “COCO” which is defined as “Common Object in Context”. COCO datasets provide labelled and segmented objects in images. These datasets consist of more than 200,000 images that are labelled according to their classes. These datasets consist of 80 classes such as people, umbrella, cat, bus, etcetera. In this study, custom datasets are used to replace the COCO datasets since this study focuses on human detection only. Thus, the software named “LabelImg” is used to manually crop the object which is human in an image. The datasets are initially divided into two different folders which are named training and testing accordingly. In the image cropping process, the training file is called in this software to manually crop the images which contain humans one by one as shown in Figure 3.11 below.

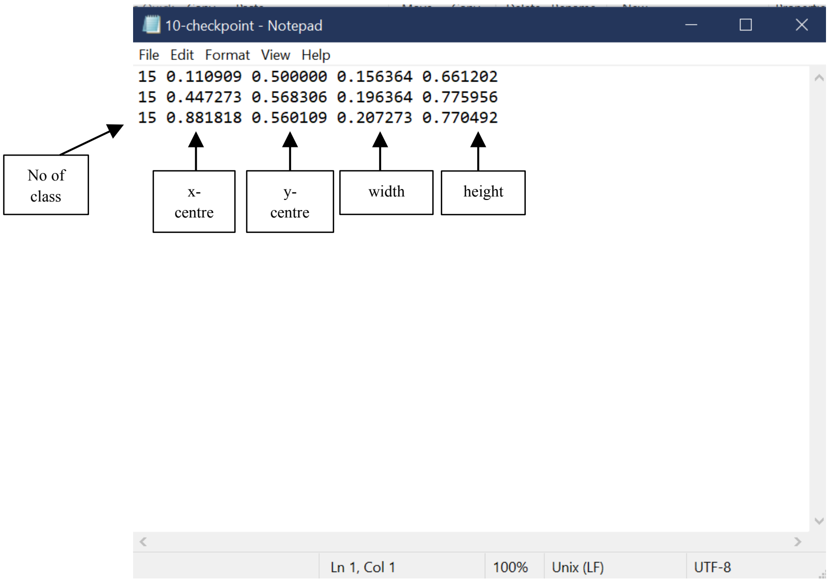


labelImg C:\Users\Aina\Downloads\Computer-Vision-with-Python\Computer-Vision-with-Python\datasets_own\training\human\.ipynb_checkpoints\1-checkpoint.jpg

Click & drag to move shape 'human'

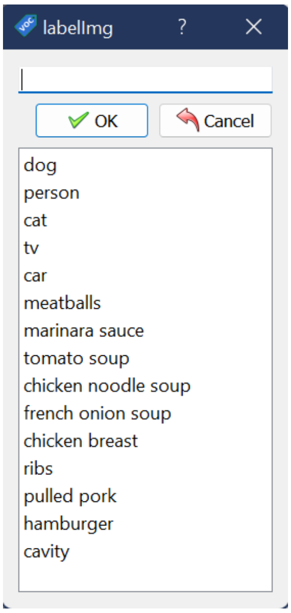
Figure 3.11 Cropping images for Human Class Using LabelImg

After cropping the human class, a text file is generated for each image. Thus, there are 282 text files are generated. The text file consists of five pieces of information such as the class number, *x*-centre, *y*-centre, width, and height of the labelled bounding boxes. From Figure 3.12 below 15 represent the class number of “human”, as in the software there are already 14 classes such as in Figure 3.13. Thus, in order to differentiate our dataset labelling for class, a “human” class is added. *X*-centre represents the drawn bounding box’s location of centre for *x* coordinate. The same goes for the *y*- centre which represents the location of centre for the *y* coordinate. The fourth and the fifth information are the width and the height of the bounding boxes. Figure 3.12 shows the information inside the text file.



```
15 0.110909 0.500000 0.156364 0.661202
15 0.447273 0.568306 0.196364 0.775956
15 0.881818 0.560109 0.207273 0.770492
```

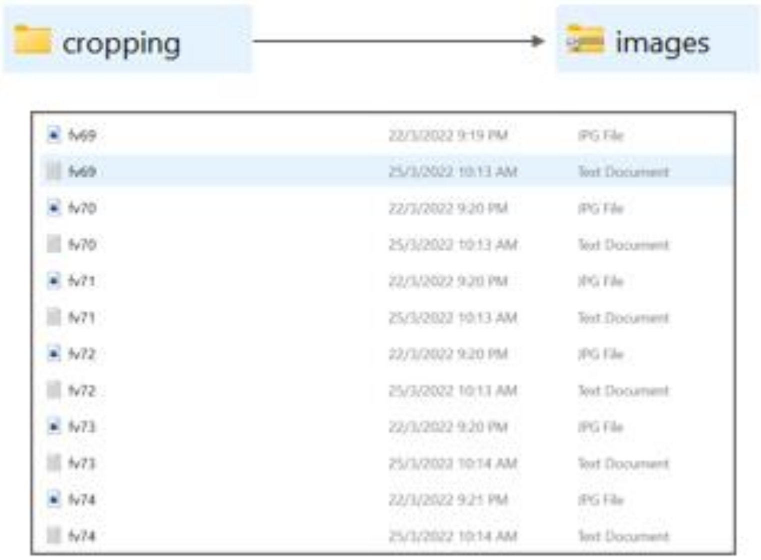
Figure 3.12 Text File After Cropping



dog
person
cat
tv
car
meatballs
marinara sauce
tomato soup
chicken noodle soup
french onion soup
chicken breast
ribs
pulled pork
hamburger
cavity

Figure 3.13 List of Existing Class in LabelImg

After the cropping process completed, all the text files and the train datasets are located in the same folder which renames as images and being zipped for the training process. Figure 3.14 shows the visualization of the images file and text file in a folder.



fv69	22/3/2022 9:19 PM	JPG File
fv69	25/3/2022 10:13 AM	Text Document
fv70	22/3/2022 9:20 PM	JPG File
fv70	25/3/2022 10:13 AM	Text Document
fv71	22/3/2022 9:20 PM	JPG File
fv71	25/3/2022 10:13 AM	Text Document
fv72	22/3/2022 9:20 PM	JPG File
fv72	25/3/2022 10:13 AM	Text Document
fv73	22/3/2022 9:20 PM	JPG File
fv73	25/3/2022 10:14 AM	Text Document
fv74	22/3/2022 9:21 PM	JPG File
fv74	25/3/2022 10:14 AM	Text Document

Figure 3.14 Files Inside the Cropping Folder to be Zipped

3.5 YOLO Version 3 Algorithms

YOLO is an abbreviation for “You Only Look Once”. YOLOV3 was released on 25th March 2018 by Joseph Redmon. This version is an upgraded version of YOLO and YOLOv2. YOLO version 3 can mainly be a real-time object detection algorithm that can detect a specific object from images or videos. To detect an object, YOLO employs features learned by a deep convolutional neural network. YOLO works as it only applies a single convolutional neural network to an image (Redmon & Farhadi, 2018). This algorithm will separate the image into a few regions and predicts its bounding boxes for each region. This algorithm used the Darknet-53 as its backbone with used for feature extraction.

3.5.1 YOLO Version 3 Architecture

Initially, YOLOV3 will divide the images into grids and start evaluating each grid to predict the bounding box based on the customized trained class probability. Then, it will pass images to the convolutional network to be flattened. After that, to avoid the overlapping of the bounding boxes in determining the object class probabilities, the Intersection over Union (IoU) and Non-Max Suppression (NMS) techniques are applied. Intersection over Union (IoU) is to specify the degree of overlap between two boxes. While Non-Max Suppression (NMS) technique is to choose the best bounding box from a group of overlapping boxes. Thus, the non-max suppression threshold is defined as 0.3. This is because, if the threshold value is below 0.1 it will not detect the overlapping of the boxes. The same goes if it sets to be very high which is a threshold value equal to 1, it will make the boxes overlap in multiple amounts. The network will resize the input image into 416 x 416. Then, the network will do multiscale detection at layers 82, layers 94, and layer 106. In layer 82, the kernel size 1 x 1 will detect at 13 x 13 input image which specifically detects a bigger object. While, at layer 94, the image is being detected at 26 x 26 to detect a medium object. Then, the last layers where detection occurs at 52 x 52 for the detection of a small object. This architecture in Figure 3.15 is explained more in the Darknet-53 which is the feature extractor of this algorithm in section 3.5.2.

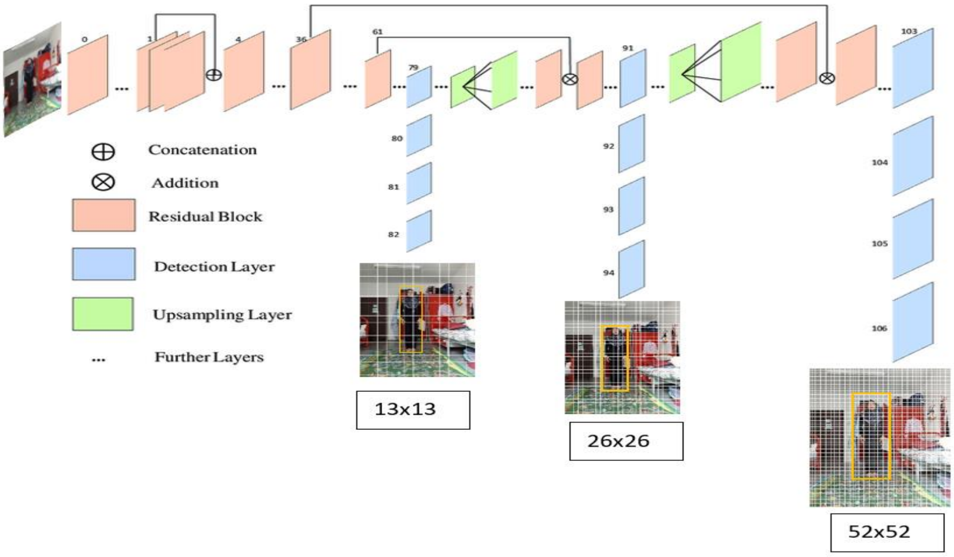
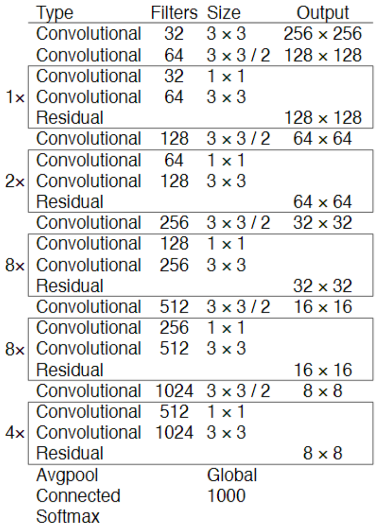


Figure 3.15 YOLO version 3 Network Architecture
Source:(Patel M, 2020)

3.5.2 Darknet-53

Darknet-53 is defined as the backbone of the YOLOV3. This backbone was introduced by YOLOV3 creators who is Joseph Redmon and Ali Farhadi. This darknet architecture uses a Convolution Neural Network consisting of 53-layer; thus, the name is Darknet-53 where 53 represents the number of layers being trained on ImageNet and composing the 3 x 3 and 1 x 1 filters size. Figure 3.16 below shows the Darknet-53 network (Redmon & Farhadi, 2018). Darknet-53 is more efficient compared to the previous Darknet-19. This network act as a feature extractor. Feature extractor works for the process of converting raw data into numerical features that can be handled while keeping the original data set's information. Besides, this network consists of 3 different prediction heads which process images in 3 different sizes of scale. The convolutional layers are followed by a few other layers which are batch normalization and ReLu activation. Also, numerous filters are convolved on the pictures using the convolution layer, resulting in multiple feature maps.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
	Convolutional	32	1 × 1	
1×	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
	Convolutional	64	1 × 1	
2×	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
	Convolutional	128	1 × 1	
8×	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
	Convolutional	256	1 × 1	
8×	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
	Convolutional	512	1 × 1	
4×	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 3.16 Darknet-53 Network Structure

Source: (Redmon & Farhadi, 2018)

There are a few parameters involved in this algorithm as shown in Table 3.3 and Table 3.4 below.

Table 3.3 Parameters in the Configuration Files of YOLOV3

Parameter	**Description**
Batch Size	Represent the number of samples that will be processed in a batch
Learning Rate	0.001
Channels	3
Momentum	0.9
Decay	0.005
Max Batches	1600 iterations
Policy	Activation function used is the Steps function
Convolution Layers	75 Convolutional layers used ReLu Leaky as activation function in each layer

Table 3.4 Data Augmentation Parameter

Parameter	Description
Width	416
Height	416
Angle	0
Saturation	1.5
Exposure	1.5
Hue	0.1

3.6 YOLO Version 3 Training

The training phase is the most important in this algorithm. The training phase will cause an improvement of the algorithm itself in terms of accuracy and performance during the testing processes. Thus, the quantity of data influenced the accuracy and the performance of the algorithm once it is tested. In this study, the trained data are about 282 images as it is only 70% of the total images whereas 403 images and the balanced images are set for the testing process. In the training process, the GPU is utilized, which will speed up the training process. The training process takes up from a few hours to several days, but with GPU it will speed up the process up to 2000 iterations, which is around 10 hours, but it is limited to the internet connection speed as well.

Figure 3.17 shows the block diagram of training the human detection. Initially, datasets of train images are collected. These datasets are downloaded from online sources such as Kaggle. Next, the images are trained using the YOLOV3 training algorithm to produce a newly trained model related to the fed images.

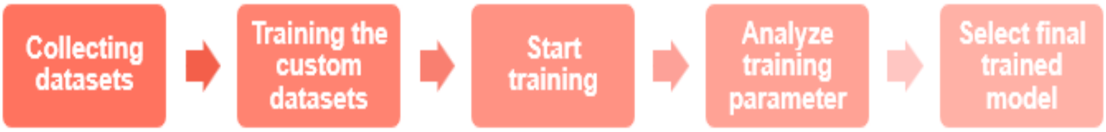

Collecting datasets
Training the custom datasets
Start training
Analyze training parameter
Select final trained model

Figure 3.17 Block Diagram of Training Human Detection

Referring to Figure 3.18, the flow of the training process is started by activation of GPU, followed by mounting the Google Drive by connecting Google Drive to the Google Colaboratory for importing the training dataset. Next, the neural network framework such as Darknet-53 is utilized in this project. Darknet is used to perform feature extraction (Redmon & Farhadi, 2018). Then, the network is compiled using Nvidia GPU as this network works together with GPU computation. Besides, a few configurations in training need to be set up such as the batch numbers which represent the iteration numbers, filters, and batch subdivisions. In short, filters are totalled multiplication of a number of bounding boxes per grid which is denoted by the value of 3, with the number of classes which is denoted by the value of 1 represented by human, and the number of 5. Next, by creating a new folder in Google Drive to store the training weight. After that, an *obj.names* file is created which consists of a class name which is "human". Finally, start the training based on the iteration number, and the weight file will keep updated for every 100 iterations in the folder created on Google Drive. It also creates another weight file whenever the iterations reached 1000 iterations. The weight file is a binary file, where the weight is stored in float data type. The function of this weight file is to represent the trained model.

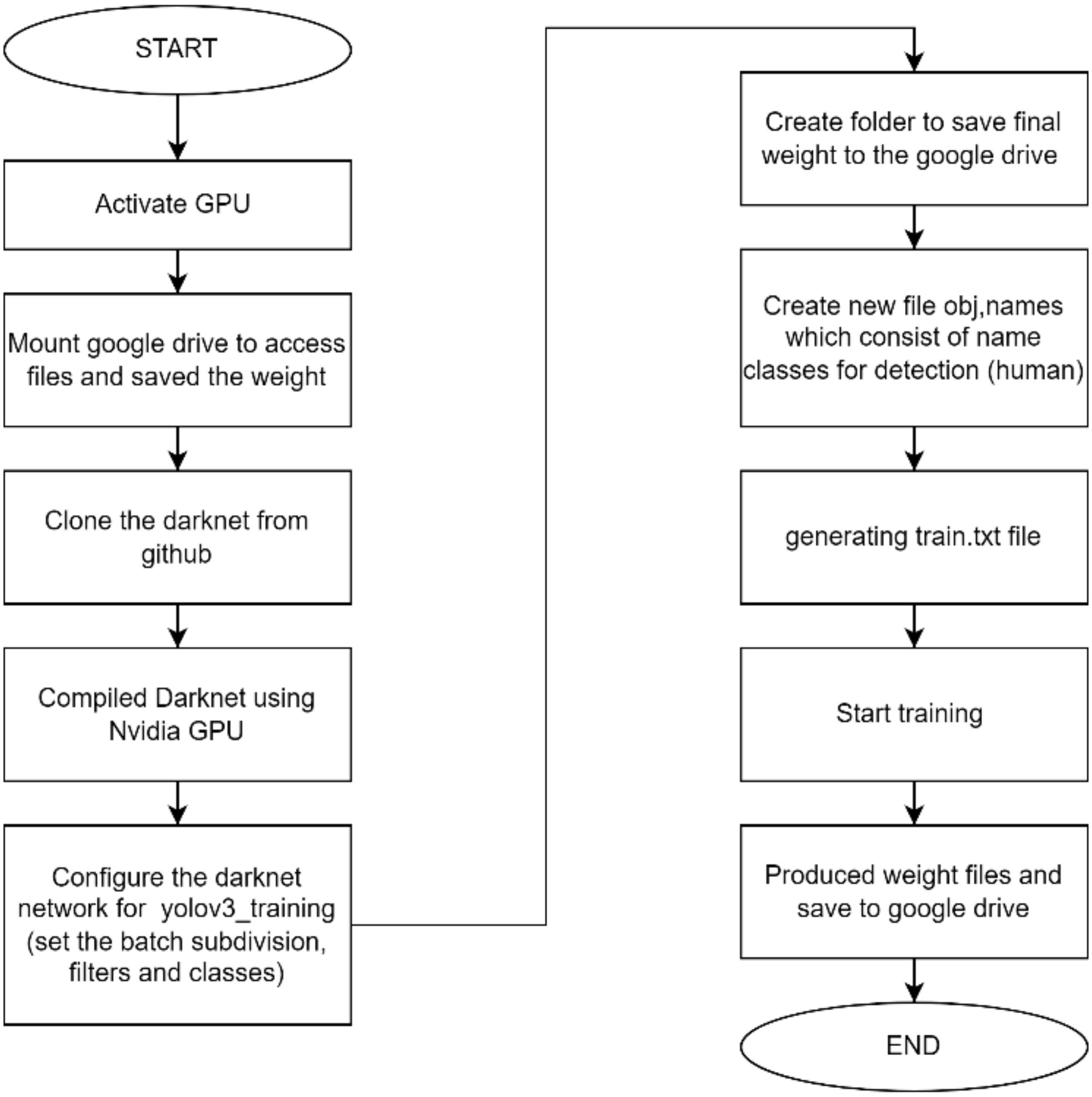

START
Activate GPU
Mount google drive to access files and saved the weight
Clone the darknet from github
Compiled Darknet using Nvidia GPU
Configure the darknet network for yolov3_training (set the batch subdivision, filters and classes)
Create folder to save final weight to the google drive
Create new file obj,names which consist of name classes for detection (human)
generating train.txt file
Start training
Produced weight files and save to google drive
END

Figure 3.18 Training Flowchart

3.7 YOLO Version 3 Testing

In the testing process, there are two testing is conducted which are human detection for various positions using online datasets and the other test is to test the reference image for measurement of social distance purposes. For the first testing, the block diagram in Figure 3.19 is the flow of the testing as the input image is called, then the system will detect a human in each image. After that, it will create green bounding boxes together with a “human” label when a human is detected. There are about 282 images are tested. For second testing another additional 300 images are also being tested, but only for human detection with social distancing detection.

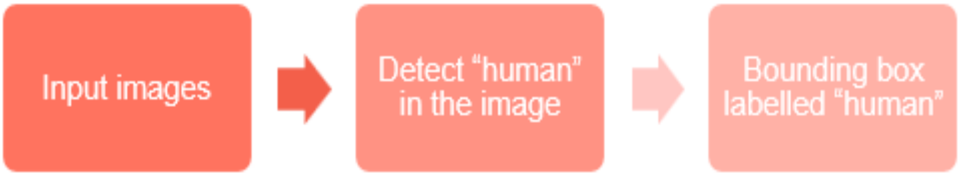


Figure 3.19 Testing Block Diagram

Next, the flowchart in Figure 3.20 shows the flow of the system throughout the testing process. Initially, Google Drive is mounted to access the related files such as generated weight file from training, testing configuration file, and test image file. The related library such as cv2, numpy, glob, random, and distance are imported. After that, the network is loaded into memory which consists of weight files and testing configuration files. This project is focusing on human detection. Thus, the class name is declared as a human so that when the detection is done, the output will display the word "human" on the output image. Next, the file path of the test images dataset is declared so that the detection system will continuously loop the test image after one image is tested. In every image that is being tested, there is some information such as the image's height, width, and its channel accessed and declared as *img.shape*. This testing process is followed by pre-processing images which are mean subtraction, resizing images, and channel swapping. Then, it will compute the output by running the forward path and will start displaying the class name on the screen as 0 since there is only one class in this detection. After that, the bounding box is extracted by checking the confidence value, if the confidence value is greater than 0.3, it will continue to calculate the value of x-centre, y-centre, width, height, x, and y. The calculated values are now being added to the list of boxes and continue calculating the indexes by applying non-maxima suppression with a threshold value and IoU threshold value. If the index, "I" is in the range of the boxes, it will plot the green box to represent there is a detected human in the image together with the class name label.

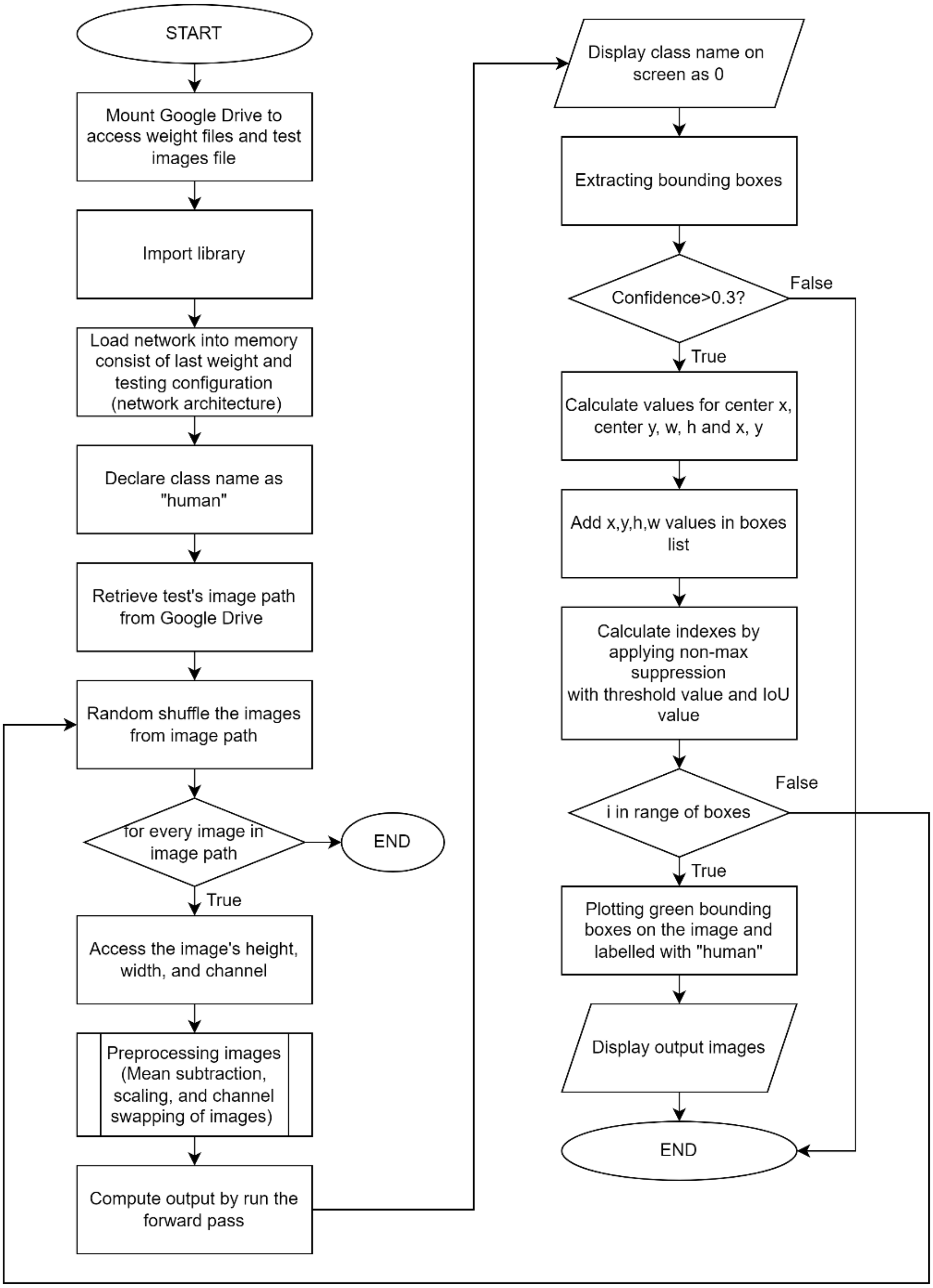

START
Mount Google Drive to access weight files and test images file
Import library
Load network into memory consist of last weight and testing configuration (network architecture)
Declare class name as "human"
Retrieve test's image path from Google Drive
Random shuffle the images from image path
for every image in image path
END
True
Access the image's height, width, and channel
Preprocessing images (Mean subtraction, scaling, and channel swapping of images)
Compute output by run the forward pass
Display class name on screen as 0
Extracting bounding boxes
Confidence>0.3?
False
True
Calculate values for center x, center y, w, h and x, y
Add x,y,h,w values in boxes list
Calculate indexes by applying non-max suppression with threshold value and IoU value
i in range of boxes
False
True
Plotting green bounding boxes on the image and labelled with "human"
Display output images
END

Figure 3.20 Testing Flowchart

3.7.1 Datasets Collection for Testing Images (Human Detection and Social Distancing Detection)

This dataset is specifically created for detection of human and social distancing detection, 5 videos are being recorded which consist of humans with different positions. The videos are extracted into 60 images for each video. The duration of the videos is about 3 minutes per video. The extracted images are taken every 3 seconds by using PhotoDirector for Acer software. Hence, Figure 3.21, Figure 3.22, Figure 3.23, Figure 3.24, and Figure 3.25 shows the extracted images based on the type of positions. Table 3.5 shows the details of the dataset.

Table 3.5 Dataset for Social Distancing Detection

Position	**Number of Image**
Front View	60
Side View 1	60
Side View 2	60
No SOP 1	60
No SOP 2	60
Total Image	300

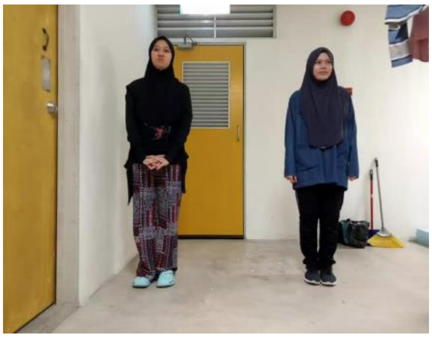


Figure 3.21 Front View

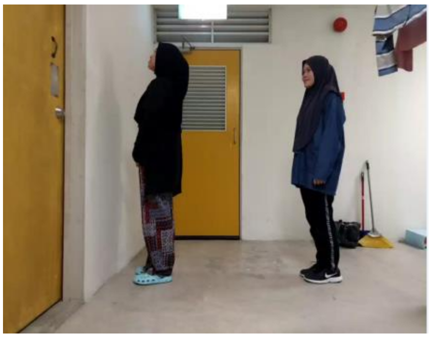


Figure 3.22 Side View 1

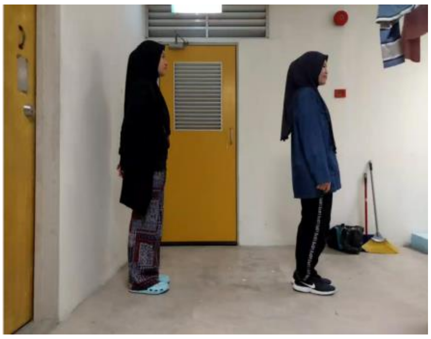


Figure 3.23 Side View 2

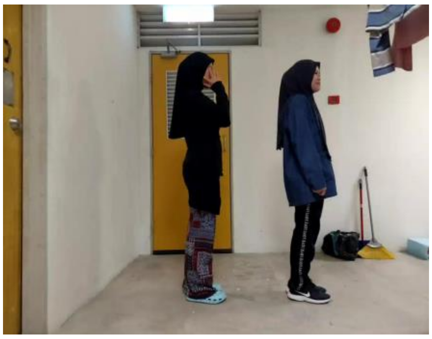


Figure 3.24 NO SOP 1

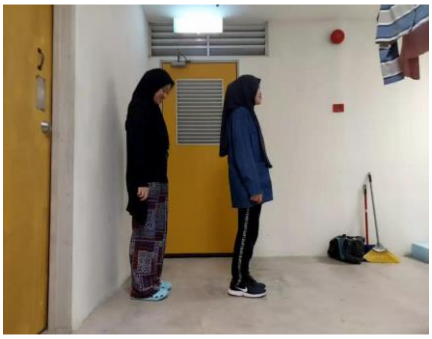
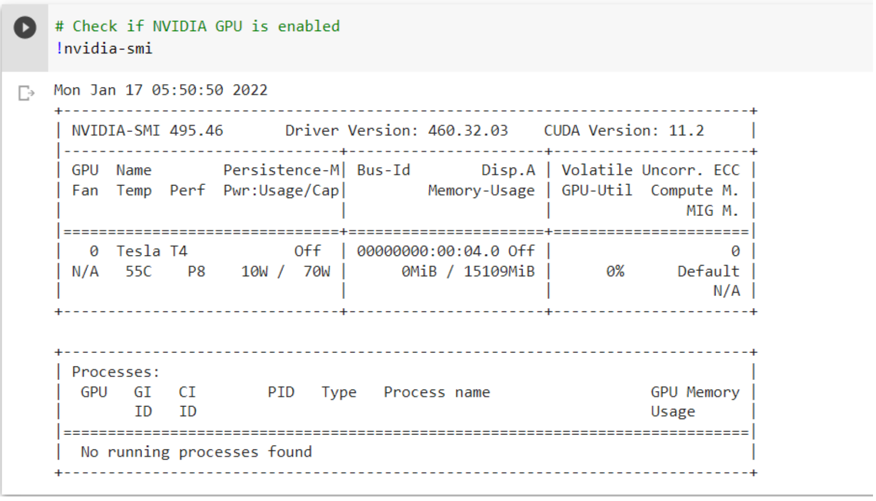


Figure 3.25 NO SOP 2

3.7.2 Testing Custom Datasets

The testing process is a process to test the generated file during the training process. Initially, the testing process is started by enabling the GPU NVIDIA as shown in Figure 3.26 below. The purpose of enabling the GPU is to increase the speed during the testing process. From the figure, the GPU used in Google Colaboratory is NVIDIA Tesla T4 which has a max execution time of around 12 hours of free access provided by Google Colaboratory.

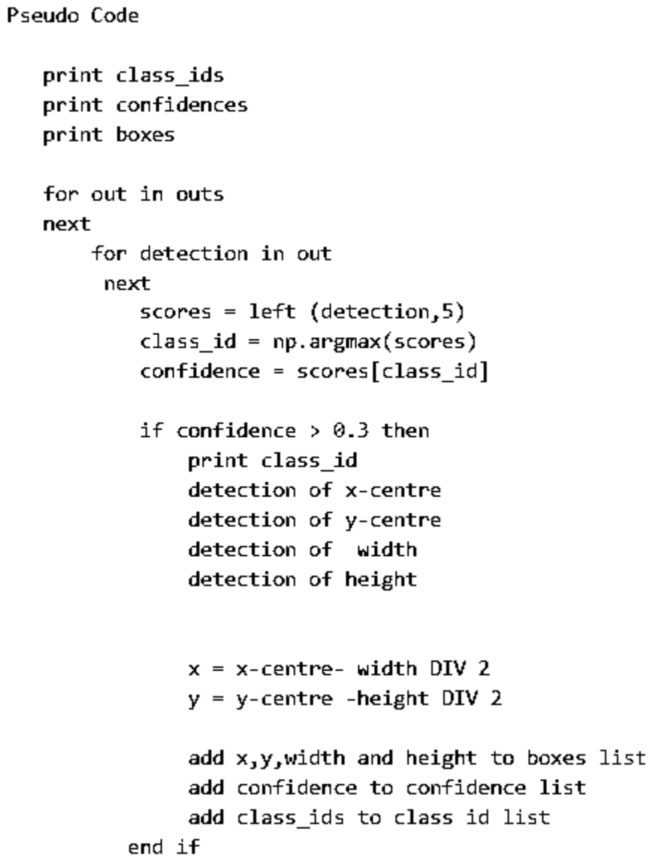


```
# Check if NVIDIA GPU is enabled
!nvidia-smi
```

```
Mon Jan 17 05:50:50 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 495.46       Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   55C    P8    10W /  70W |      0MiB / 15109MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

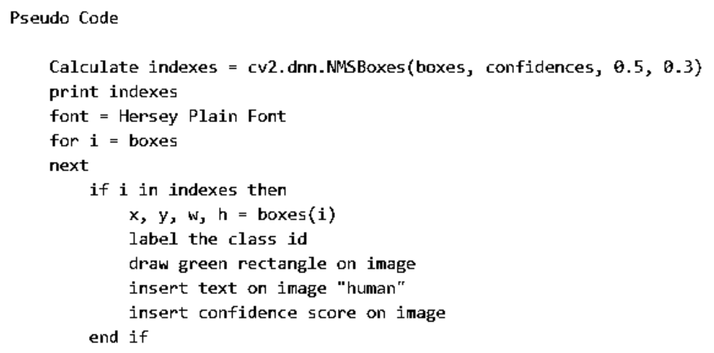
Figure 3.26 Enable GPU in Google Colaboratory

Google drive is used to import the weight file that is being generated during the training process and located in a single folder named “yoloV3”. In that folder, there are several files such as a configuration file, zipped labelled image training folder, online datasets test folder, reference image folder, and customed testing images folder. After that, the related libraries are imported to start the coding process. Next, the configuration file and weights file are loaded by using *cv* function *readNet* where it will read the deep learning network which only supports the configuration and model (weight) file. The class of the custom object is customized as “human” so that the output images that detect a human in the image will appear in that class together with the bounding box. After that, the coding continues to call the images file path from google drive where the file is located. In this project, the test image folder is named “online final dataset”. Then, the system will now detect the object which is human only and start creating the bounding boxes by scanning throughout all the bounding boxes output from the network by keeping the one with a high confidence score, and the box's class label is assigned as in Figure 3.27 and Figure 3.28. The *cv2.rectangle* is used to create the bounding boxes in green color with a border line thickness of 2, which represent by decimal codes 255 on the image with a starting point and end point. The border thickness value is represented in pixel values. The starting point is represented by the x-coordinates and y-coordinates. Whilst, the end point is represented by the total of x-coordinates of the detected object and its width and the total of y-coordinates of the detected image and its height. After that, the *cv2.putText* function is to put the text on the detected image. In this code, the detected object will be labelled as human, and its confidence values in the green color of the detected image with floating type number in 4 decimal places. The position of the text is being fixed with *x*-coordinates values of the detected object and *y*-coordinates values with subtraction of 10 which gives the position of the text above the drawn bounding box.



```
Pseudo Code

   print class_ids
   print confidences
   print boxes

   for out in outs
   next
       for detection in out
        next
           scores = left (detection,5)
           class_id = np.argmax(scores)
           confidence = scores[class_id]

           if confidence > 0.3 then
               print class_id
               detection of x-centre
               detection of y-centre
               detection of  width
               detection of height


               x = x-centre- width DIV 2
               y = y-centre -height DIV 2

               add x,y,width and height to boxes list
               add confidence to confidence list
               add class_ids to class id list
          end if
```

Figure 3.27 Pseudo Code of Human Detection



```
Pseudo Code

    Calculate indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.3)
    print indexes
    font = Hersey Plain Font
    for i = boxes
    next
        if i in indexes then
            x, y, w, h = boxes(i)
            label the class id
            draw green rectangle on image
            insert text on image "human"
            insert confidence score on image
        end if
```

Figure 3.28 Pseudo Code of Creating Bounding Boxes

3.8 Distance Measurement

In measuring the distance between the human, a reference image is needed for the measurement. The reference image is taken from the dataset. The image is consisting of a student in a front-standing position within a specific distance from a phone camera. The flowchart in Figure 3.29 shows the details step in setting up the reference image. Overall, the flowchart is just the same as the previous testing flowchart but with another additional process which is the output image will also display the width and height of the detected human on the image. Those width and height are in pixel unit representation. In order to make the distance measurement reliable in real-life situations, the pixel value of the width is presently being converted to the centimetre unit by using a ratio formulation. The human real-life width is measured beforehand with a value of 36 centimetres using a measuring tape.

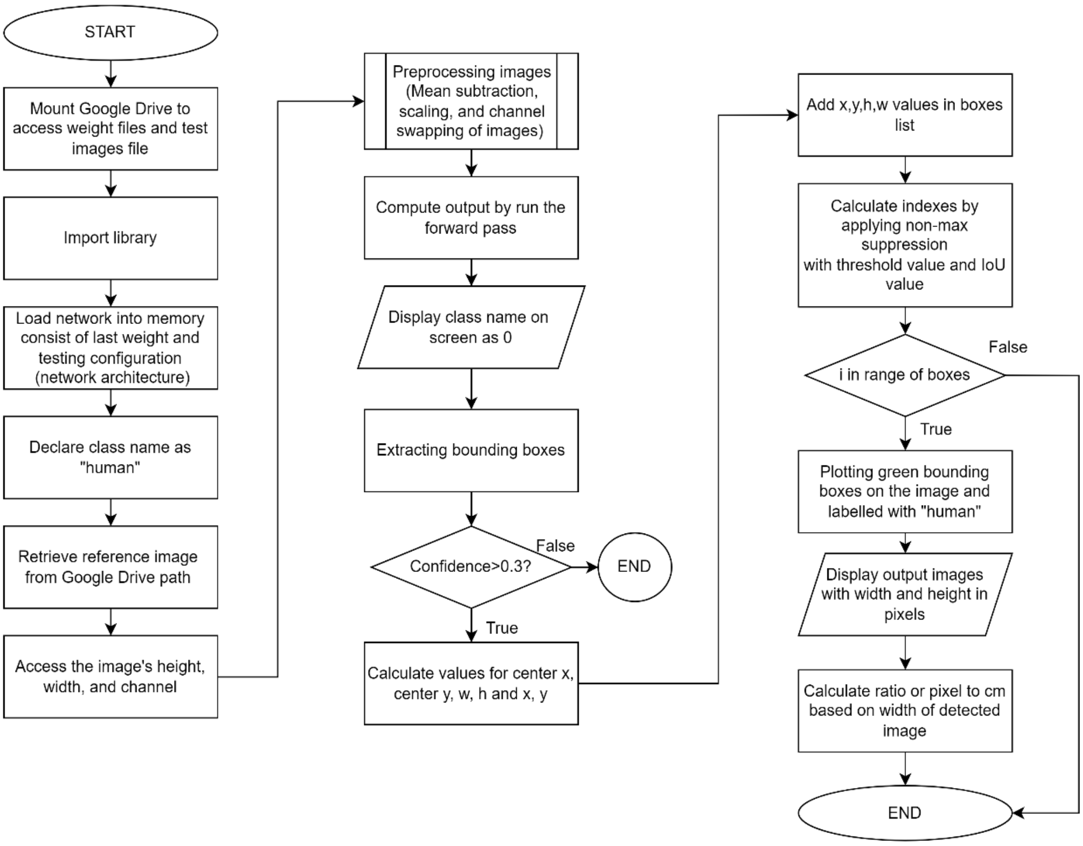

START
Mount Google Drive to access weight files and test images file
Import library
Load network into memory consist of last weight and testing configuration (network architecture)
Declare class name as "human"
Retrieve reference image from Google Drive path
Access the image's height, width, and channel
Preprocessing images (Mean subtraction, scaling, and channel swapping of images)
Compute output by run the forward pass
Display class name on screen as 0
Extracting bounding boxes
Confidence>0.3?
False
END
True
Calculate values for center x, center y, w, h and x, y
Add x,y,h,w values in boxes list
Calculate indexes by applying non-max suppression with threshold value and IoU value
i in range of boxes
False
True
Plotting green bounding boxes on the image and labelled with "human"
Display output images with width and height in pixels
Calculate ratio or pixel to cm based on width of detected image
END

Figure 3.29 Flowchart of Set Up Reference Image

3.8.1 Set Up Reference Image tools

In this subsection, two tools are being used to set up the reference image dataset which is measuring tape, and a phone camera as Figure 3.30 and Figure 3.31 below.

Figure 3.30 Measuring Tape

Figure 3.31 Samsung Galaxy A53 Phone Camera

To make the detection functional well, the position of the model must similar to the train image such as the front view position. Thus, Figure 3.32 and Figure 3.33 below are the sketches. Figure 3.34 shows the real-life scenario of the set-up process. The distance between the camera and the human is about 202cm.

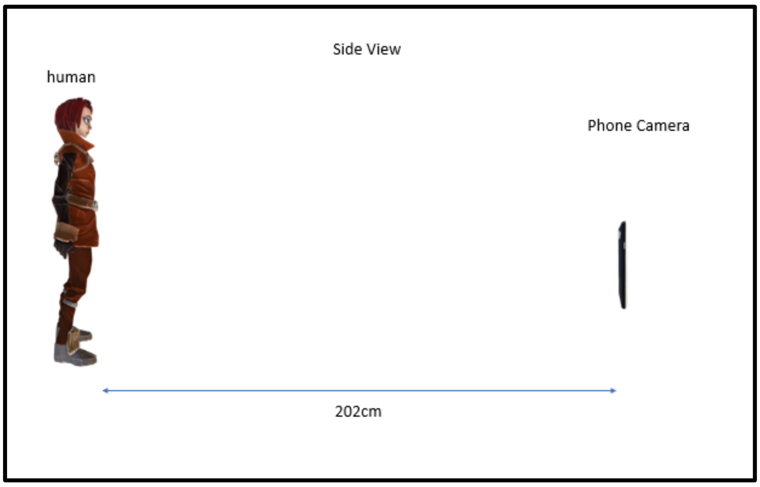


Figure 3.32 Sketches of Side View

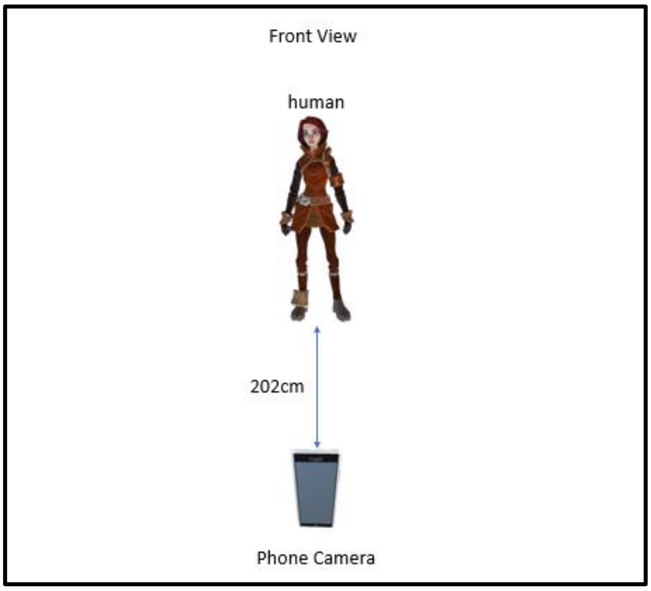


Figure 3.33 Sketches of Front View

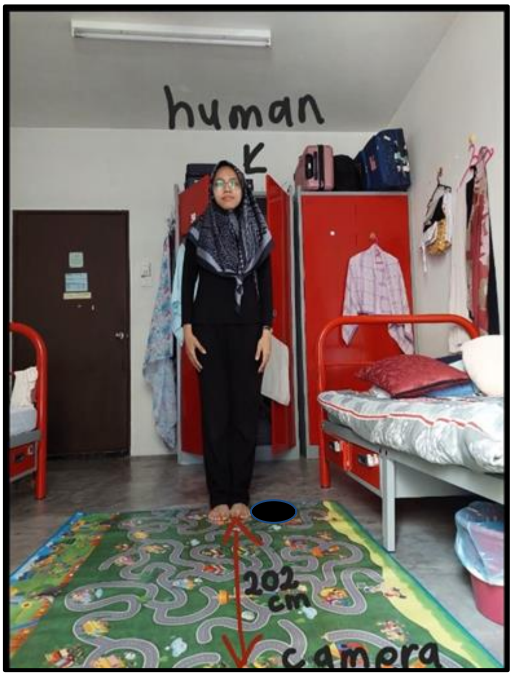


Figure 3.34 Real Scenario Image

To match the real-life measurement with the image measurement, the ratio approach is used. As in the flowchart in Figure 3.29, there are the width and height of the detected object based on the bounding box that appeared in the output image, thus the width is used to calculate the ratio between pixels and centimetres.

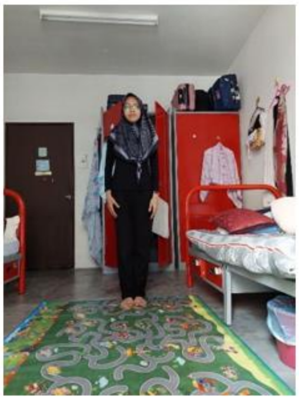
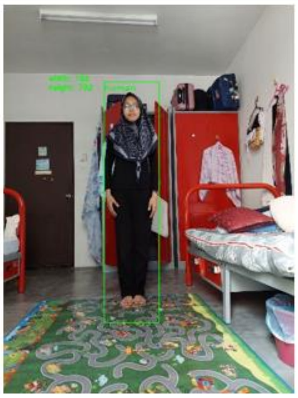
Ref image before detection	Ref image after detection
	

Figure 3.35 Before and After Detection Output

The width or total distance in the pixel is annotated as *D*, while real-life width or total distance in centimetre is annotated as *Dc*. Equation 3.1 is generated by a simple ratio calculation which produces a pixel value in centimetre unit. Thus, 1 pixel is equivalent to 0.1978cm.

$$1\,pixel \;=\frac{Dc\ (cm)}{D\ (pixel)}=\frac{36\ cm}{182\ pixel} \tag{3.1}$$

Concerning computing the distance between two people, the Euclidean distance formula as shown in Equation 3.2 is used. This expression measures the distance between the pairwise bounding boxes between two people. For any of two people, the distance between them is less than “Safe Distance” (i.e., the set value of safe social distance which is 100cm) which is considered to contribute to non-adherence of social distancing. The equation in Equation 3.2 being multiplied with 0.1978 is because to make Euclidean distance value is measured in pixel converted into centimetre as calculating using ratio approach in section 3.8.1.

$$d = [\sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}]\ (0.1978) \qquad 3.2$$

3.8.2 Social Distancing Detection

Figure 3.36 shows the block diagram of the social distancing approximation detection. The method will initially detect the model through the custom-trained model then compute the distance between two people and will classify if there is social distancing between two people or not based on a set safe distance.

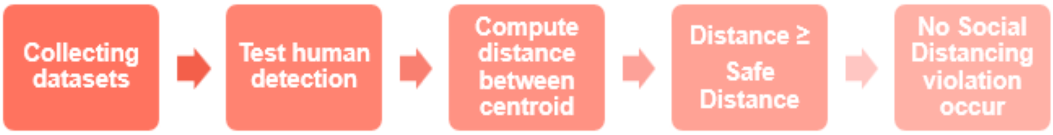

Collecting datasets
Test human detection
Compute distance between centroid
Distance ≥ Safe Distance
No Social Distancing violation occur

Figure 3.36 Block Diagram for Social Distancing Detection

The flowchart of social distancing detection is illustrated in

Figure 3.37. There is an additional function that is being added after the human detection process. The system checked if the box indexes are greater than 2 or not. If the indexes give the true values, then it will proceed in plotting the bounding boxes and a line from the centroid of the first box to another box. The lines represent the calculated distance between two humans. Hence, the Euclidean distance formula with additional multiplication with the calculated ratio value which is 0.1978 is used to determine the total distance between two people in centimetres. However, the violation is detected once the measured distance is less than “Safe Distance” which is 100cm. Thus, the bounding boxes that are created at the output will be in red representation together with the line that is being plotted from the centroid of the first box to the second box. The measured distance is displayed on the top left of the image frame with red color also. However, if the measured distance is greater or equal to the “Safe Distance” values, the output of

detection and the distance value will be displayed in green color. This represents that human comply with the social distancing safe distance.

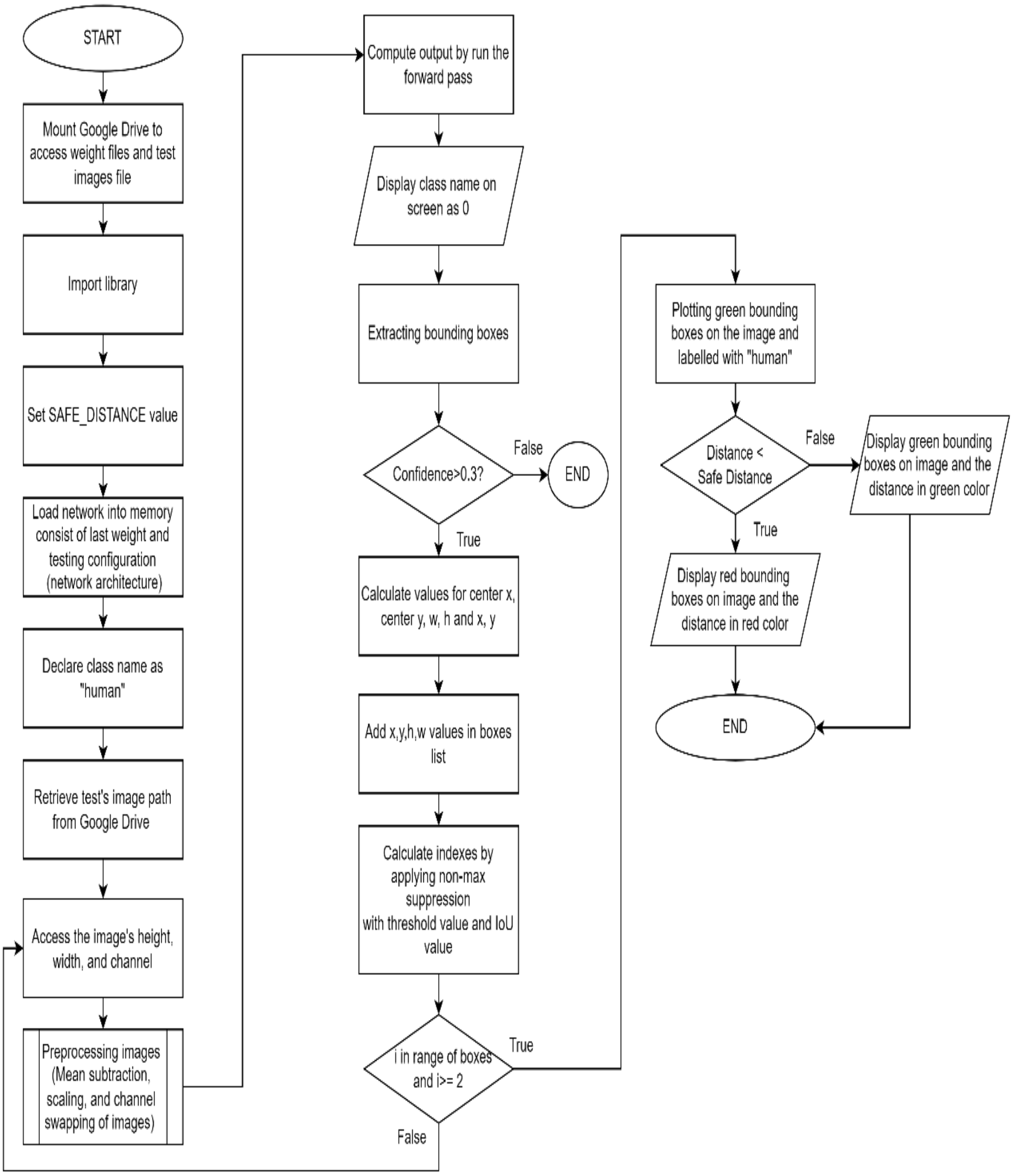

START
Mount Google Drive to access weight files and test images file
Import library
Set SAFE_DISTANCE value
Load network into memory consist of last weight and testing configuration (network architecture)
Declare class name as "human"
Retrieve test's image path from Google Drive
Access the image's height, width, and channel
Preprocessing images (Mean subtraction, scaling, and channel swapping of images)
Compute output by run the forward pass
Display class name on screen as 0
Extracting bounding boxes
Confidence>0.3?
False
END
True
Calculate values for center x, center y, w, h and x, y
Add x,y,h,w values in boxes list
Calculate indexes by applying non-max suppression with threshold value and IoU value
i in range of boxes and i>= 2
True
False
Plotting green bounding boxes on the image and labelled with "human"
Distance < Safe Distance
False
Display green bounding boxes on image and the distance in green color
True
Display red bounding boxes on image and the distance in red color
END

Figure 3.37 Flowchart of Testing Social Distancing

CHAPTER 4

RESULT AND DISCUSSION

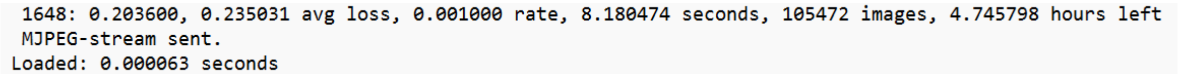
4.1 Introduction

This chapter will discuss the overall results of the proposed method. The proposed method for human detection as mentioned in chapter 3 utilized You Only Look Once version 3 (YOLOV3). Next, for distance detection, the Euclidean distance formula is used and multiplied by 0.1978. This human detection and social distancing monitoring method is tested on the custom datasets. The datasets were retrieved from the online source which was from Kaggle, Unsplash, and iStock websites. Besides, there is another custom dataset of around 300 images that are being extracted from videos to test the overall system. The detection accuracies are calculated based on the position for human detection. Thus, this chapter is divided into two sections. First section 4.2, will be focusing on the performance of the produced trained model. Secondly, section 4.3 will be discussing the analysis of human detection and social distancing detection with the use of the different datasets.

4.2 Training YOLOV3 Results

In the training process, it is expected a weight file will be generated at the end of the training process. Along with the weight files, there is a list of output lines of the training process which calculates the number of iterations, total losses, average losses, and Mean Average Precision (mAP). Thus, from the output lines, the information such as average losses and Mean Average Precision (mAP) are being analyzed to see the performance of the trained model using custom datasets.

In the YOLOV3 training phase with a customed dataset, it produced output as shown in Figure 4.1. The figure shows the information of the batch output after 1648 iterations which consists of many iterations, total losses, average losses, and learning rate. 1648 is defined as the number of last iterations. 0.2036 indicates the overall loss and 0.2350 indicates the average loss. Next, the 0.001 rate represents the current learning rate of the training defined in the configuration files. Next, *burn_in* is the number of batches that will ramp the learning rate from 0 to the set learning rate. As the iterations have passed the *burn_in* iterations which are 1000 iterations, thus the learning rate is maintained until the end of iterations. Meanwhile, 8.1804 seconds indicates the total time that is being spent for the current iteration which is at 1648 iterations. In this training phase, the overall training time for this project already reached 12 hours of continuous training. Since the use of free online GPU provided by Google Colaboratory is only 12 hours. On the other hand, the internet speed also affects the training duration.



```
 1648: 0.203600, 0.235031 avg loss, 0.001000 rate, 8.180474 seconds, 105472 images, 4.745798 hours left
 MJPEG-stream sent.
Loaded: 0.000063 seconds
```

Figure 4.1 Training Result After 1648 Iterations

The training can be stopped as shown in Figure 4.2, the average losses value decreasing into small values. It shows the average losses are decreasing from 2045.1033 to 0.2350. Losses is showing how bad the model is in predictions. Therefore, all the average values are being analyzed using excel as in below Figure 4.3 to see the trend of the average losses of the trained model.

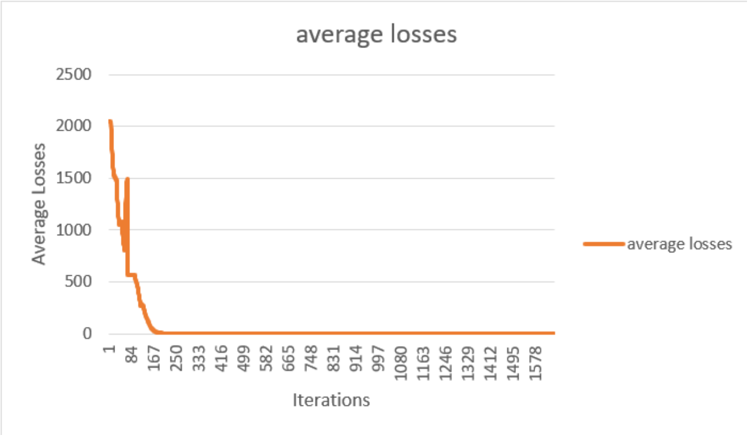
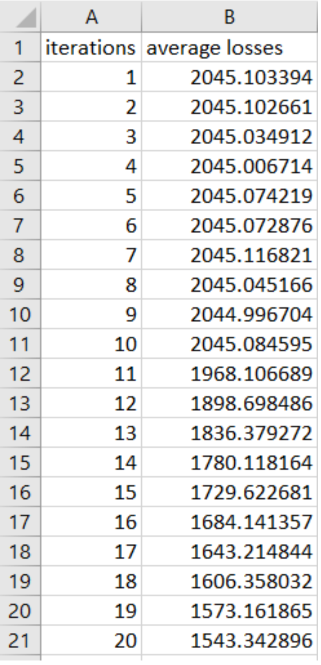


Figure 4.2 Average Losses vs. Iterations



	A	B
1	iterations	average losses
2	1	2045.103394
3	2	2045.102661
4	3	2045.034912
5	4	2045.006714
6	5	2045.074219
7	6	2045.072876
8	7	2045.116821
9	8	2045.045166
10	9	2044.996704
11	10	2045.084595
12	11	1968.106689
13	12	1898.698486
14	13	1836.379272
15	14	1780.118164
16	15	1729.622681
17	16	1684.141357
18	17	1643.214844
19	18	1606.358032
20	19	1573.161865
21	20	1543.342896

Figure 4.3 List of Average Losses of 1st Iteration Until 20th Iterations

The object detection model commonly used Mean Average Precision (mAP) to evaluate the performance of the trained model. The indicator of the best Mean Average Precision (mAP) is determined by the value itself. The higher the Mean Average Precision (mAP) means that the model is good enough for the detection process. In this project, the calculated Mean Average Precision (mAP) during training is 99.68% as shown in Figure 4.4. Thus, this model is suitable to use for in testing in human detection.

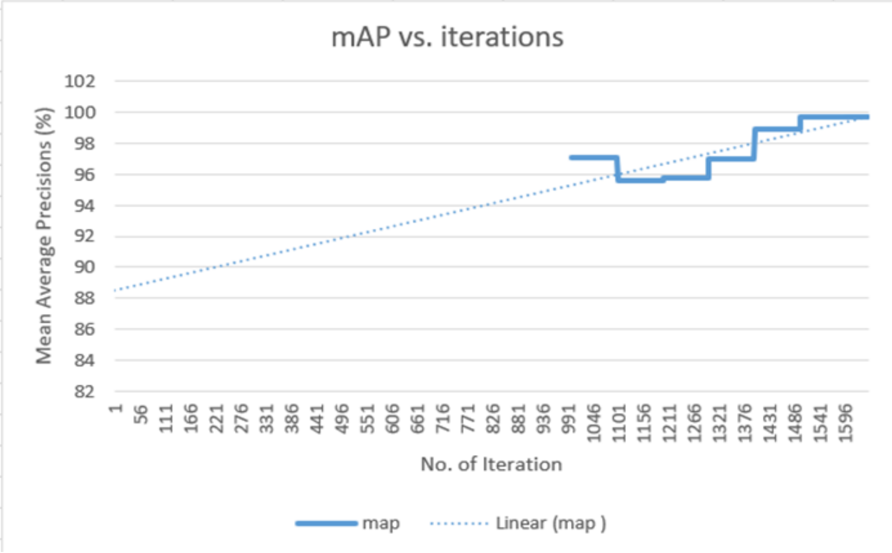


Figure 4.4 mAP vs. Iterations

4.3 Testing YOLOV3 on Human Detection and Social Distancing Detection

After the training process, the testing process will take place by feeding the input images into the algorithm to test the model in terms of its object detection. After 1648 iterations of training it is expected that the detection accuracy over 121 images is higher as the training accuracy model represented by mAP is 99.68%. The results are shown in sections 4.3.1 and 4.3.2.

4.3.1 Human Detection Using Online Datasets

This subsection will discuss the results of human detection in various positions. This YOLOV3 detects the human based on the set object threshold which is 0.5 and the non-maxima suppression threshold at 0.3. Thus, if the object scores are in the range of 0.5 and above, then the detection occurs. Figure 4.5 shows the first example output of the front view position of humans where there are two green bounding boxes representing the detection of both humans in the images are successfully detected with object score around 0.9973 for the right bounding box and 0.9366 for the left bounding box. While Figure 4.6 shows the detected human with an object score of 0.9955. The object scores represent how close the detected object is to the real object. The second example of front view output is also detected as human even though the person is holding an umbrella. Furthermore, this position gives overall accuracy of 94.44% which was calculated using equation 4.1 with a total of 34 images out of 36 images are successfully detected humans in the images.

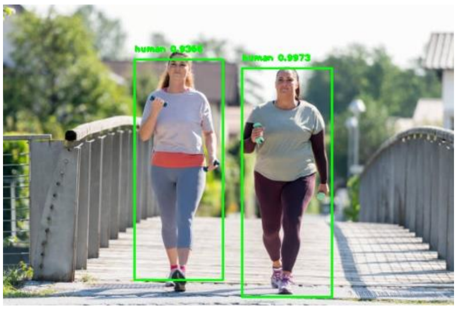


Figure 4.5 Example 1 Front View

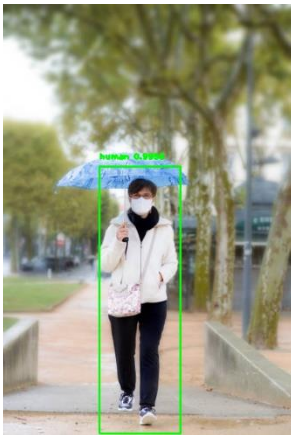


Figure 4.6 Example 2 Front View

For the back view image sample, the testing model is capable to detect the human with object scores of 0.9225 even if the individual is in a dark situation, where there is a shadow of the tree on the human body. Thus, the data augmentation during training helps in detecting humans in the image even in a dark situation. The data augmentation works when there is an adjustment of the image *Exposure* as in Table 3.4. Therefore, this Figure 4.7 image exposure will become brighter compared to the input image. Next, in the second example as in Figure 4.8, the model also successfully detects the human as the object score is 0.9807 even though the man is holding a watering can. Thus, the number of images for this position is classified successfully with 91.67% of accuracy (33 images out of 36 images).

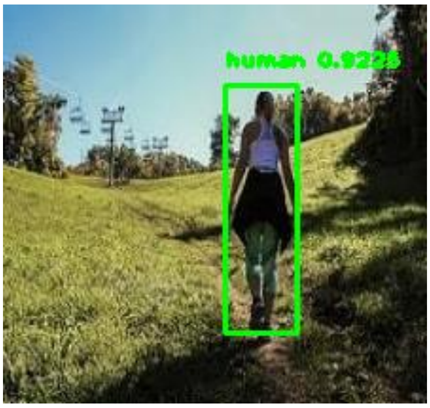


Figure 4.7 Example 1 Back View

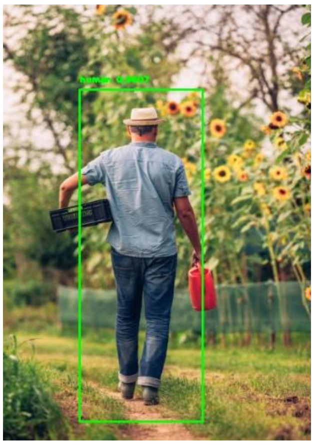


Figure 4.8 Example 2 Back View

For the side view image sample, the model also successfully detects the human in the image as the humans in Figure 4.9 has an object score of 0.9422 for the left bounding box, and 0.9784 for the right bounding box. Next, in the second example of side view as shown in Figure 4.10, a standing woman also successfully detected with object scores of left bounding boxes is 0.9960 and the other bounding box with object score of 0.9865. In this figure, the model also successfully detects the human position in the side view even though both of them are opposite to each other. Overall, this position gives higher accuracy compared to other positions since there are about 39 out of 40 images successfully detects human in the images which give a percentage accuracy of 97.50%.

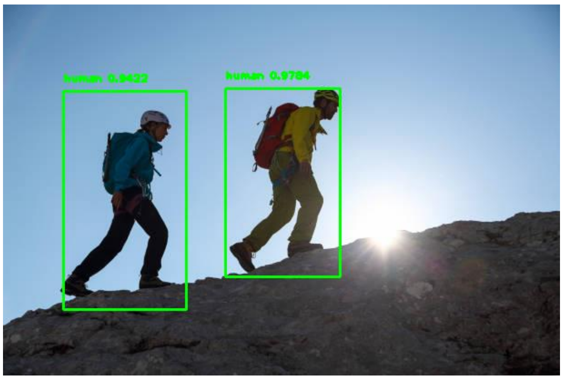


Figure 4.9 Example 1 Side View

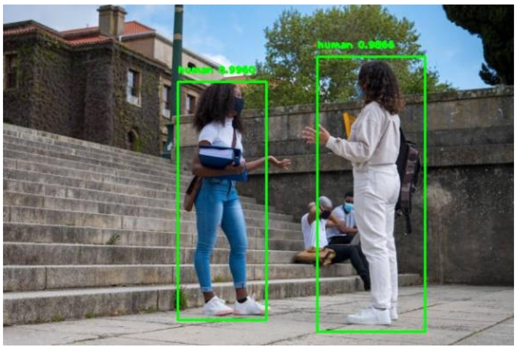


Figure 4.10 Example 2 Side View

Lastly, for the crowded position the first example in Figure 4.11, the model still can detect the human in the figure as the body of the human is half of the frame, this is due to the object scores which are all above 0.5 where from the left bounding boxes to the right bounding boxes the object score is 0.9391, 0.9865, and 0.9556 respectively. In Figure 4.12, the human is also being detected by the model. Overall, the percentage of accuracy for this position is 88.89% and 8 out of 9 images are detected humans in it.

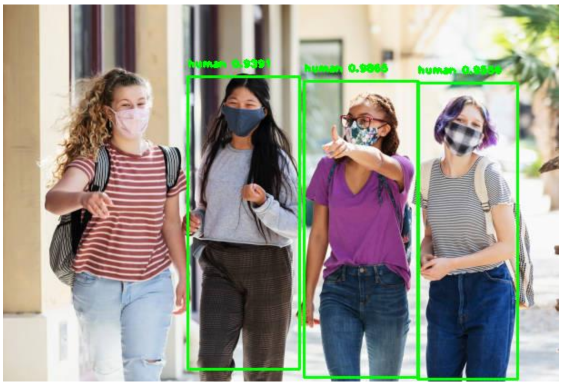


Figure 4.11 Example 1 Crowded

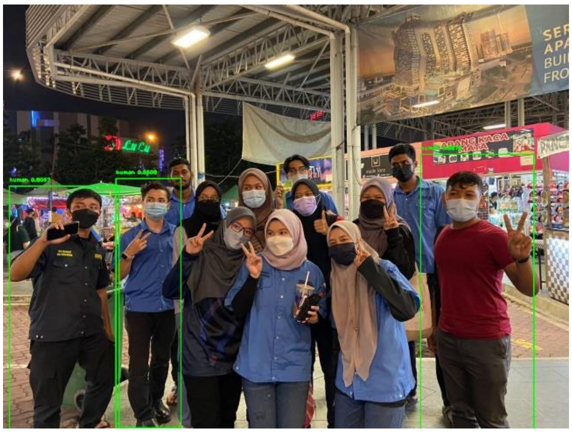


Figure 4.12 Example 2 Crowded

Overall, for the online dataset for testing, there are 114 out of 121 images are come out with the result of human detection by calculating accuracy percentages as equation 4.1. The percentage accuracies of human detection are 94.21%. For each position, the summary of percentage accuracies is calculated as in Table 4.1.

$$accuracy(\%) = \frac{total\ images\ that\ detect\ human}{total\ input\ images} \times 100\% \qquad 4.1$$

Table 4.1 Testing Accuracy Based on Human Positions

No	**Positions**	**Total Image**	**Detected Human**	**Accuracy (%)**
1.	Front view	36	34	94.44
2.	Back view	36	33	91.67
3.	Side view	40	39	97.50
4.	Crowded	9	8	88.89
	Total	121	114	94.21

The number of humans per image also calculated which to determine what is the maximum number of humans that can be detected by this method. Generally, each human in each position is being calculated manually to take the actual number of humans detected per image. Table 4.2 shows the summary of the percentage accuracies of the detected number of humans per image in every position with an overall accuracy value of 80.62%. The highest detection number of humans per position goes to the front view sample image with a percentage accuracy of 91.25%. Overall, the maximum number of humans detected in an image is about seven humans as in Figure 4.13 below.

Table 4.2 Testing Accuracy Based on Number of Human Per Image (in each position)

No	**Positions**	**Total Image**	**Detected Human (Per image)**	**Accuracy (%)**
1.	Front view	36	32.85	91.25
2.	Back view	36	29.03	80.64
3.	Side view	40	31.17	77.93
4.	Crowded	9	4.5	50.00
	Total	121	97.55	80.62

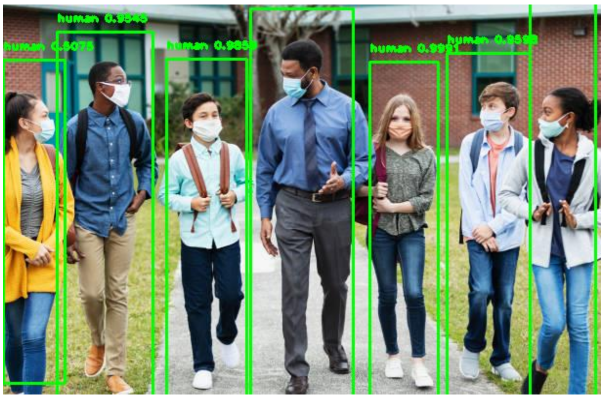


Figure 4.13 Maximum Human Detected

4.3.2 Human Detection on Custom Datasets

For the human testing of custom datasets by using the same model, the results of accuracies are represented in Table 4.3. Only in a position which is NO SOP 1, the model cannot detect the human in the image because the object score of the human is less than 0.5. However, for the front view, side view 1, side view 2, and no sop 2 the model successfully detected both humans in the images. Figure 4.14 below shows the result for the front view position image.

Table 4.3 Testing Accuracy of Human Detection Using Custome Datasets

No	Positions	Total Image	Detected Human	Accuracy (%)
1.	Front view	60	60	100.00
2.	Side view 1	60	60	100.00
3.	Side view 2	60	60	100.00
4.	NO SOP 1	60	54	90.00
5.	NO SOP 2	60	60	100.00
	Total	300	294	98.00

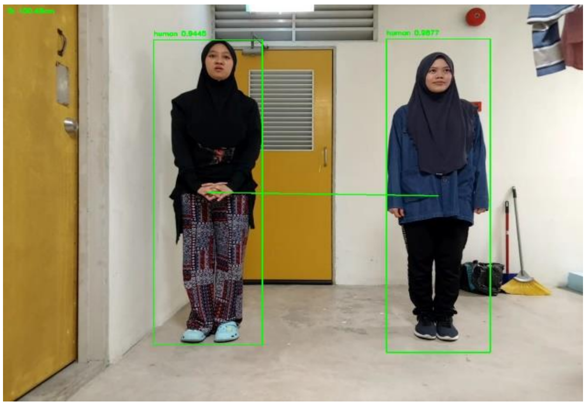


Figure 4.14 Example of Both Humans are Detected

4.3.3 Social Distancing Detection

In social distancing detection, the detection model calculates the distance whenever there is more than one bounding box is detected. Then, the model calculates the distance between the centroids of bounding boxes by using the Euclidean distance formula. The distance between two people is categorized as safe if the distance between two people is larger or equal to the “Safe Distance” value which is 100cm. Figure 4.15 shows the result of social distancing detection between people of the tested image in the front view position. From this image, the displayed distance on the top-left of the image shows the distance is 100.10cm, which is closed to the real-life distance between two people. However, for the second example of front view social distancing detection, as shown in Figure 4.16, the measured distance is 100.29cm. These two images have different distances as this is the custom dataset, which is from a recorded video that is being extracted into 60 images. Thus, based on the bounding boxes observed, the values of object score for both humans in Figure 4.15 and Figure 4.16 are different whereas for the human on the left the object score is 0.9705 in Figure 4.15 and 0.9619 in Figure 4.16. Meanwhile, the score for humans on the left in Figure 4.15 is 0.9903 and in Figure 4.16 is 0.9880. This happened due to the bounding box is not fit for the detected human. The detection occurred as the threshold score matched the value in predicted bounding boxes during training. Moreover, during the training, there are various sizes of humans. Hence, the precise number for detection at the detected distance is accepted at ±0.3cm.

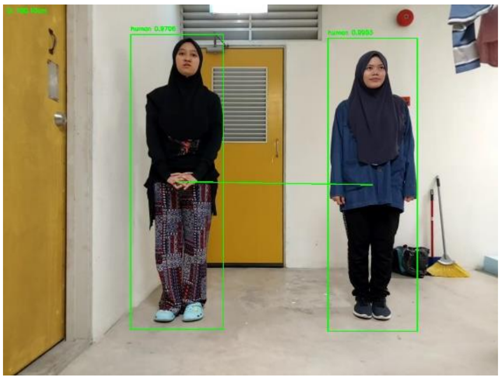


Figure 4.15 Example 1 Front View Safe Distance

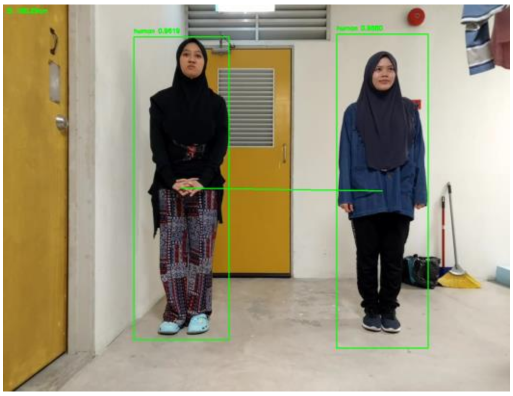


Figure 4.16 Example 2 Front View Safe Distance

Next, red bounding boxes indicate the social distancing violation occurs where the distance between two people is below the “Safe Distance” value which is 100cm. Therefore, for the positions NO SOP 1 and NO SOP 2 the setup distance between two people during datasets are taken is about 60cm. Figure 4.17 show the reading of the distance between two humans is 60.33cm. Since the distance measured is below the “Safe Distance” value, therefore the bounding boxes are red-colored to indicate the violation in social distancing occurred.

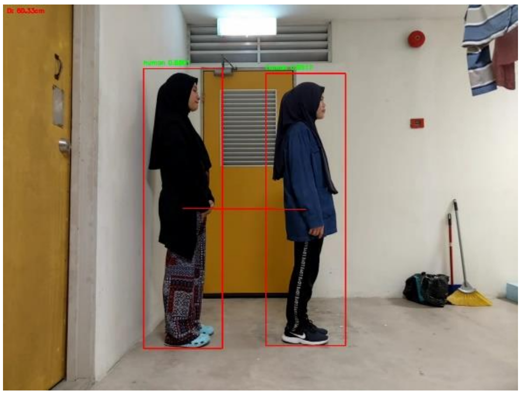


Figure 4.17 Unsafe Distance Between 2 People (NO SOP 2)

4.4 Misdetection of human

After 121 images from the online dataset are tested, there is a misdetection image of a human. This faulty detection occurs when it detects the stacked pallet as human as shown in Figure 4.18. This misdetection occurs because the object score of the stacked pallet is about 0.5117. As discussed in the testing section 4.3.1. The object score is dropped to 0.5 and the stacked pallet looks like a man in a white suit.

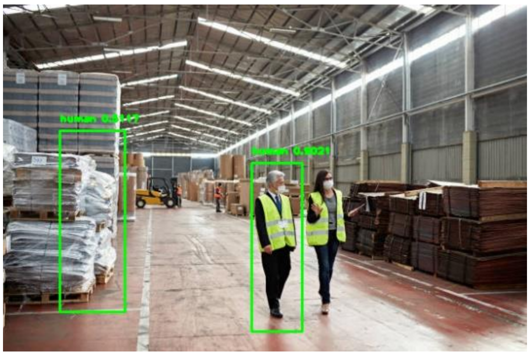


Figure 4.18 Misdetection Pallet

In a detection system, there must be imperfections such as the system being unable to detect a particular object. From the results from a customed dataset, there are a few misdetections that occur in detection the of humans as shown in Figure 4.19. This image is extracted from a video. The system directly detects the hanging shirt as human. This can be clarified because the background of the hanging shirt is similar to the detected human. Hence, the human with yellow-door background is not detected as a human. From this image sample, the background of the object also influenced the detection results.

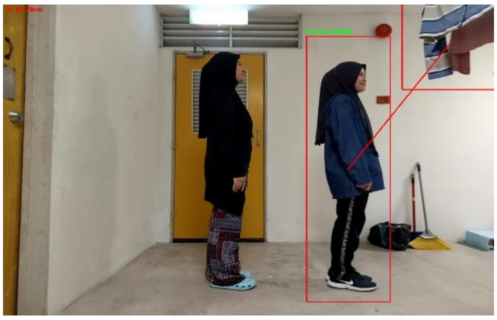


Figure 4.19 Misdetection of Human

CHAPTER 5

CONCLUSION

5.1 Introduction

This chapter will summarize the method used and the overall result of this study. Future work recommendations are also discussed in this chapter to improve the current performance and the application of the system. Next, the project's impact on society is also being discussed in this chapter.

5.2 Conclusion

In this study, the deep learning approach is used for human detection. The deep learning method, YOLOV3 is chosen because of the output performance produced after the training process. The trained model gives 99.68% of Mean Average Precision (mAP) for human class prediction. To conclude, the current datasets collected from online sources which are about 403 produced overall accuracy of 94.21% of human detection. Besides, each position such as front view, back view, side view, and the crowd gives a result of human detection at 94.44%, 91.67%, 97.50%, and 88.89% respectively. Hence, the highest accuracy of human detection goes to the side view position with a percentage accuracy of 97.50%. Besides, using the same model for its dataset gives overall percentage accuracy which is 98.00% in detecting humans. Next, the social distance detection is successfully developed by using the Euclidean distance formula by multiplying it by a calculated ratio of a pixel to centimetre which is 0.1978cm. Hence, the detected social distancing result is also in the range of acceptable at ±0.3cm. For future work, it is suggested to increase the number of images to achieve the targeted percent accuracy. Next, this method can be implemented in real-time since the speed of detection is fast and not only for social distancing applications but can be used for other applications such as virtual reality games. Impact on the society, this project might help in making the society becomes more discipline in practicing the new norm. This is due to the society realizing that whenever they are not adhering to the social distancing the possibility for them to be infected by the viruses is high since the viruses are now likely easy to spread. In addition, when people are infected by the virus it is difficult for them to go through their daily life.

REFERENCES

1st known case of coronavirus traced back to November in China | Live Science. (n.d.). Retrieved January 25, 2022, from https://www.livescience.com/first-case-coronavirus-found.html

Ahmed, I., Ahmad, M., Rodrigues, J. J. P. C., Jeon, G., & Din, S. (2021). A deep learning-based social distance monitoring framework for COVID-19. *Sustainable Cities and Society*, *65*. https://doi.org/10.1016/j.scs.2020.102571

Ambika Neelopant, Dr. S. V. Viraktamath, & Pratiksha Navalgi. (2021, February 16). *Comparison of YOLOv3 and SSD Algorithms – IJERT*. https://www.ijert.org/comparison-of-yolov3-and-ssd-algorithms

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (n.d.). *End-to-End Object Detection with Transformers*. https://github.com/facebookresearch/detr.

Elbishlawi, S., Abdelpakey, M. H., & Shehata, M. S. (2021). SocialNet: Detecting Social Distancing Violations in Crowd Scene on IoT devices. *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*, 801–806. https://doi.org/10.1109/WF-IoT51360.2021.9595383

Evolution of Object Detection. Object detection is still a relatively… | by Chinmoy Borah | Analytics Vidhya | Medium. (n.d.). Retrieved February 1, 2022, from https://medium.com/analytics-vidhya/evolution-of-object-detection-582259d2aa9b

Glorot, X., & Bengio, Y. (n.d.). *Understanding the difficulty of training deep feedforward neural networks*. http://www.iro.umontreal.

Hou, Y. C., Baharuddin, M. Z., Yussof, S., & Dzulkifly, S. (2020). Social Distancing Detection with Deep Learning Model. *2020 8th International Conference on Information Technology and Multimedia, ICIMU 2020*. https://doi.org/10.1109/ICIMU49871.2020.9243478

Karaman, O., Alhudhaif, A., & Polat, K. (2021). Development of smart camera systems based on artificial intelligence network for social distance detection to fight against COVID-19. *Applied Soft Computing*, *110*. https://doi.org/10.1016/j.asoc.2021.107610

Keniya, R., & Mehendale, N. (2020). Real-Time Social Distancing Detector Using Socialdistancingnet-19 Deep Learning Network. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3669311

Magoo, R., Singh, H., Jindal, N., Hooda, N., & Rana, P. S. (2021). Deep learning-based bird eye view social distancing monitoring using surveillance video for curbing the COVID-19 spread. *Neural Computing and Applications*. https://doi.org/10.1007/s00521-021-06201-5

Marking Tape for Social Distancing - Heskins. (2020). https://www.heskins.com/marking-tape-for-social-distancing

Mercaldo, F., Martinelli, F., & Santone, A. (2021). A Proposal to Ensure Social Distancing with Deep Learning-based Object Detection. *Proceedings of the International Joint Conference on Neural Networks*, *2021-July*. https://doi.org/10.1109/IJCNN52387.2021.9534231

Puspita, I. A., Akbar, R., & Irwansyah, A. (2021a). Monitoring Violations of Social Distancing COVID-19 On Standing Queues With Euclidean Distance Method. *2021 International Electronics Symposium (IES)*, 511–516. https://doi.org/10.1109/IES53407.2021.9594006

Puspita, I. A., Akbar, R., & Irwansyah, A. (2021b). Monitoring Violations of Social Distancing COVID-19 On Standing Queues With Euclidean Distance Method. *2021 International Electronics Symposium (IES)*, 511–516. https://doi.org/10.1109/IES53407.2021.9594006

Qin, J., & Xu, N. (2021). Reaserch and implementation of social distancing monitoring technology based on SSD. *Procedia Computer Science*, *183*, 768–775. https://doi.org/10.1016/j.procs.2021.02.127

Ramadass, L., Arunachalam, S., & Z, S. (2020). Applying deep learning algorithm to maintain social distance in public place through drone technology. *International Journal of Pervasive Computing and Communications*, *16*(3). https://doi.org/10.1108/IJPCC-05-2020-0046

Real-Time Maintaining of Social Distance in Covid-19 Environment using Image Processing and Big Data. (2020).

Redmon, J., & Farhadi, A. (n.d.). *YOLOv3: An Incremental Improvement*. https://pjreddie.com/yolo/.

Redmon, J., & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*. https://pjreddie.com/yolo/.

Saponara, S., Elhanashi, A., & Gagliardi, A. (2021). Implementing a real-time, AI-based, people detection and social distancing measuring system for Covid-19. *Journal of Real-Time Image Processing*. https://doi.org/10.1007/s11554-021-01070-6

Sathyabama, B., Devpura, A., Maroti, M., & Rajput, R. S. (2020). Monitoring pandemic precautionary protocols using real-time surveillance and artificial intelligence. *Proceedings of the 3rd International Conference on Intelligent Sustainable Systems, ICISS 2020*, 1036–1041. https://doi.org/10.1109/ICISS49785.2020.9315934

Shalini, G. v., Margret, M. K., Niraimathi, M. J. S., & Subashree, S. (2021). Social Distancing Analyzer Using Computer Vision and Deep Learning. *Journal of Physics: Conference Series*, *1916*(1). https://doi.org/10.1088/1742-6596/1916/1/012039

Srinivasan, S., Rujula Singh, R., Biradar, R. R., & Revathi, S. A. (2021). COVID-19 monitoring system using social distancing and face mask detection on surveillance video datasets. *2021 International Conference on Emerging Smart Computing and*

Informatics, ESCI 2021, 449–455. https://doi.org/10.1109/ESCI50559.2021.9396783

Yadav, S. (2020). Deep Learning based Safe Social Distancing and Face Mask Detection in Public Areas for COVID-19 Safety Guidelines Adherence. *International Journal for Research in Applied Science and Engineering Technology*, *8*(7). https://doi.org/10.22214/ijraset.2020.30560

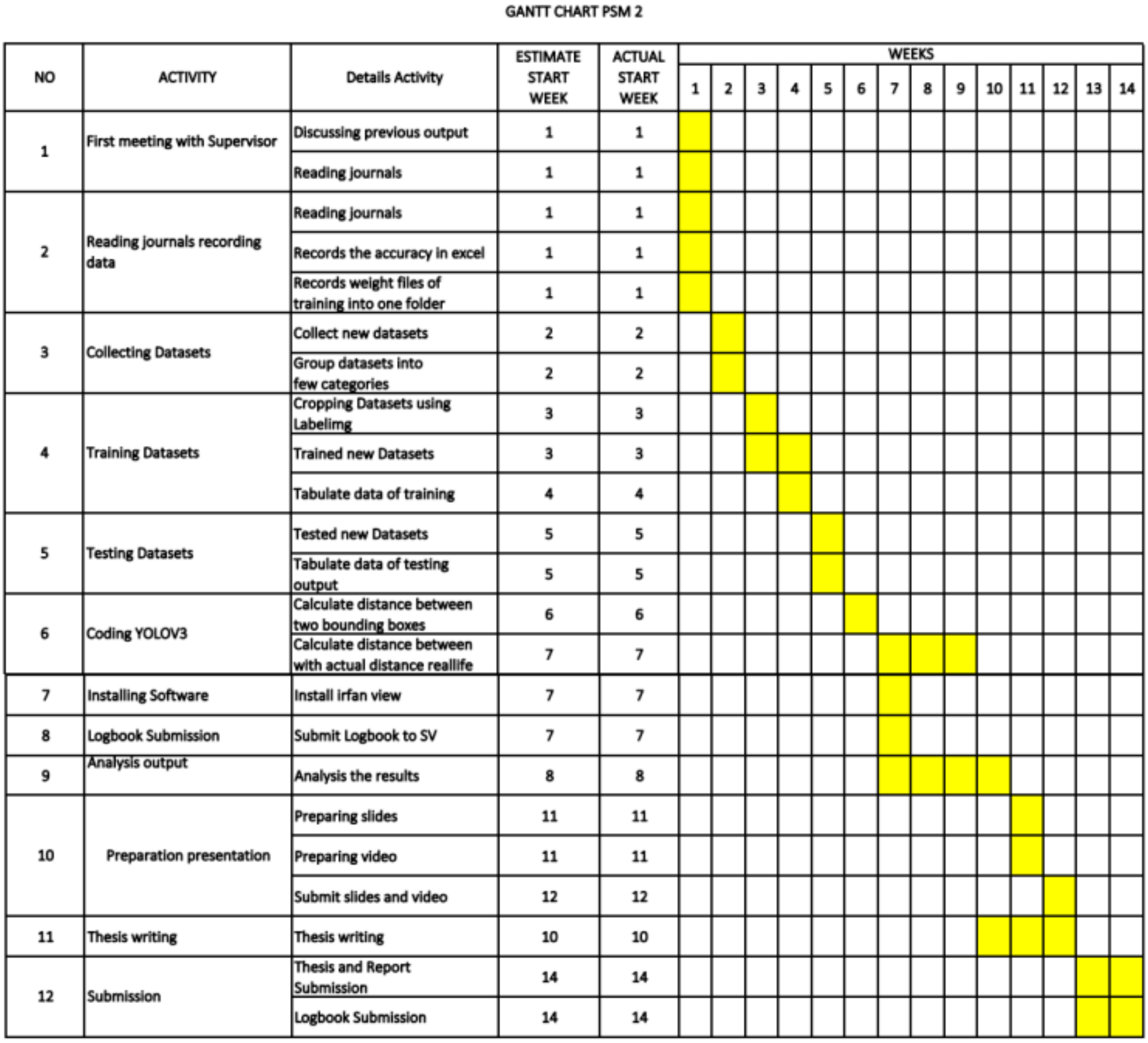
Yang, D., Yurtsever, E., Renganathan, V., Redmill, K. A., & Özgüner, Ü. (2021). A Vision-Based Social Distancing and Critical Density Detection System for COVID-19. *Sensors (Basel, Switzerland)*, *21*(13). https://doi.org/10.3390/s21134608

YOLO: Engineering Challenges for Training and Deploying YOLO on Edge Device | by Mitesh Patel | Analytics Vidhya | Medium. (n.d.). Retrieved February 5, 2022, from https://medium.com/analytics-vidhya/yolo-engineering-challenges-for-training-and-deploying-yolo-on-edge-device-8d753748d243

Zou, Z., Shi, Z., Guo, Y., Ye, J., & Member, S. (n.d.). *Object Detection in 20 Years: A Survey*.

APPENDICES

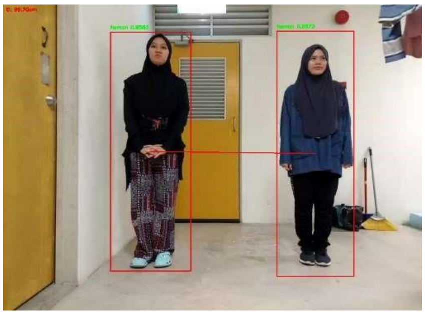
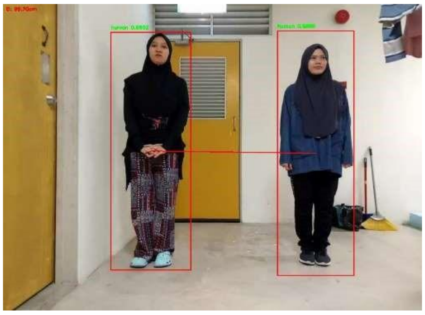
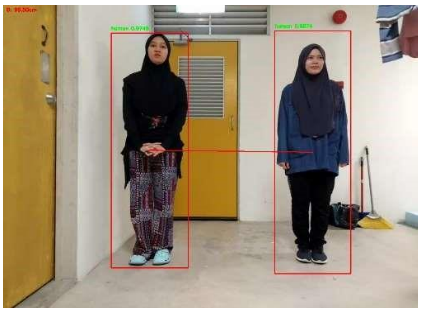
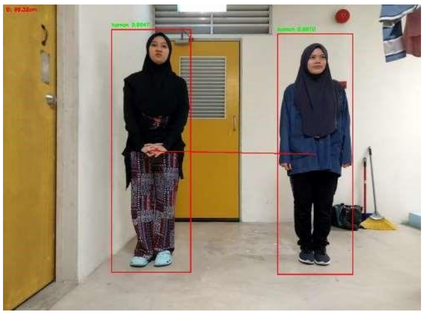
Appendix A: Gantt Chart



GANTT CHART PSM 2

NO	ACTIVITY	Details Activity	ESTIMATE START WEEK	ACTUAL START WEEK	WEEKS													
					1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	First meeting with Supervisor	Discussing previous output	1	1	🟨													
		Reading journals	1	1	🟨													
2	Reading journals recording data	Reading journals	1	1	🟨													
		Records the accuracy in excel	1	1	🟨													
		Records weight files of training into one folder	1	1	🟨													
3	Collecting Datasets	Collect new datasets	2	2		🟨												
		Group datasets into few categories	2	2		🟨												
4	Training Datasets	Cropping Datasets using Labelimg	3	3			🟨											
		Trained new Datasets	3	3			🟨	🟨										
		Tabulate data of training	4	4				🟨										
5	Testing Datasets	Tested new Datasets	5	5					🟨									
		Tabulate data of testing output	5	5					🟨									
6	Coding YOLOV3	Calculate distance between two bounding boxes	6	6						🟨								
		Calculate distance between with actual distance reallife	7	7							🟨	🟨	🟨					
7	Installing Software	Install irfan view	7	7							🟨							
8	Logbook Submission	Submit Logbook to SV	7	7							🟨							
9	Analysis output	Analysis the results	8	8							🟨	🟨	🟨	🟨				
10	Preparation presentation	Preparing slides	11	11											🟨			
		Preparing video	11	11											🟨			
		Submit slides and video	12	12												🟨		
11	Thesis writing	Thesis writing	10	10										🟨	🟨	🟨		
12	Submission	Thesis and Report Submission	14	14													🟨	🟨
		Logbook Submission	14	14													🟨	🟨

Appendix B: Analysis on Social Distancing Detection

video	front	measured	100cm
No	**Output**	**Distance(cm)**	**Accuracy** **Human Detection**
1		99.7	1
2		99.7	1
3		99.5	1
4		99.32	1