OBJECT DETECTION FOR GENERAL OBJECT TYPE USING MOBILENET-SSD NEURAL NETWORK DATABASE

LAI JIUN HONG

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2021

OBJECT DETECTION FOR GENERAL OBJECT TYPE USING MOBILENET-SSD NEURAL NETWORK DATABASE

Submitted in accordance with requirement of the Universiti Teknikal Malaysia Melaka (UTeM) for the Bachelor Degree of Manufacturing Engineering (Hons.)

by

LAI JIUN HONG

FACULTY OF MANUFACTURING ENGINEERING

2021

DECLARATION

I hereby, declared this report entitled "Object Detection For General Object Type Using MobileNet-SSD Neural Network Database" is the result of my own research except as cited in references.

Signature : ……

Author's Name : LAI JIUN HONG

Date : 2nd JULY 2021

APPROVAL

This report is submitted to the Faculty of Manufacturing Engineering of Universiti Teknikal Malaysia Melaka as a partial fulfilment of the requirement for Degree of Manufacturing Engineering (Hons). The member of the supervisory committee is as follow:

.....

(Dr. Shahriman Bin Abdullah)

ABSTRAK

Pada era globalisasi ini, teknik pengesanan objek terhadap komputer telah berkembang dengan secara mendadak di bidang eletrik dan elektronik industri. Selain itu, tugasan utama bagi teknik pengesanan objek adalah untuk mengklasifikasikan kategori objek dan mengenalpasti kedudukan objek dalam gambar atau video tertentu. Baru-baru ini, kemajuan dalam pembelajaran mendalam telah meningkatkan kecekapan teknik pengesanan objek dalam sektor kelajuan dan ketepatan. Tambahan pula, kemajuan dalam rangkaian neural yang mendalam bagi meningkatkan prestasi yang tinggi dalam seni bina supaya sistem tertanam dapat ditingkatkan dengan lebih pesat. Projek ini bertujuan untuk mengkaji kesesuaian sistem pengesanan objek beroperasi di Raspberry Pi 4 Model B. Seterusnya, Raspberry Pi juga dipanggil sebagai papan komputer tertanam yang terkenal. Demi membandingkan prestasi algoritma SSD dan algoritma MobileNet-SSD, terutamanya dari segi ketepatan. Justeru itu, eksperimen yang bersesuaian hendaklah dilakukan dengan menggunakan parameter yang berbeza, iaitu ukuran gambar input (280 x 280 piksel, 320 x 320 piksel dan 360 x 360 piksel) dan menggunakan penjarakan antara kamera dan objek yang berbeza (0.6 meter hingga 1 meter). Selanjutnya, terdapat lima jenis objek yang berkenaan akan menjalani eksperimen tersebut iaitu tetikus, telefon bimbit, alat kawalan jauh, komputer riba dan papan kekunci. Berdasarkan hasil keseluruhan yang diperoleh, hal ini dapat disimpulkan bahawa algoritma MobileNet-SSD mempunyai prestasi yang lebih baik, dengan mencapai purata kadar pengesanan hampir 76% dibandingkan dengan algoritma SSD, yang hanya memiliki 65% purata kadar pengesanan dalam sistem pengesanan. Jadi, algoritma MobileNet-SSD lebih sesuai untuk menjalankan sistem pengesanan objek pada masa nyata di Raspberry Pi dengan ketepatan yang tinggi. Oleh itu, Raspberry Pi sesuai digunakan sebagai peralatan dalam sistem pengesanan objek masa nyata. Bukan itu sahaja, ia juga dapat menggantikan dengan komputer tradisional demi mengurangkan kos penyelenggaraan dan secara tidak langsung meningkatkan kecekapan kerja.

ABSTRACT

Object detection has been experienced a dramatically technological improvement in the computer vision area. Besides that, the object detection technique is the combination of object classification and object localisation. The main task for object detection is to classify the object categories and identify the object's position in a certain image or video. Recently, advances in deep learning have significantly enhanced the efficiency of object detection techniques in speed and precision. This has allowed modern desktop computer systems to conduct extremely effective object detection at the real-time level. Other than that, the advancement in creating the high performance of deep neural network architectures for the embedded system has been increasing rapidly. Besides that, this project aims to investigate the appropriateness of operating object detection on the Raspberry Pi 4 Model B. Next, Raspberry Pi also is known as a popular embedded computer board. By comparing the performance of the SSD algorithm and MobileNet-SSD algorithm, especially on the accuracy, there is one experimental framework that should be carried out by using the different parameters, which is the size of input images (280 x 280 pixels, 320 x 320 pixels and 360 x 360 pixels) and the range distance between camera and object (0.6 meters until 1 meter). Furthermore, five types of objects will undergo in the experiment: a computer mouse, cell phone, remote, laptop, and keyboard. Based on the overall result obtained, it can be concluded that the MobileNet-SSD algorithm has better performance, which has an almost 76% average detection rate compared with the SSD algorithm, which only has a 65% of average detection rate in the detection system. Hence, the MobileNet-SSD algorithm is more suitable for running real-time object detection on Raspberry Pi with high accuracy and precision than the SSD algorithm. Apart from that, Raspberry Pi is suitable to use as hardware in a real-time object detection system. It can replace the traditional desktop system to reduce the maintenance fee and indirectly improve work efficiency.

DEDICATION

Only

my beloved father, Lai Fuh Chyan

my appreciated mother, Tan Hong Leng

my adored sister and brother, Lai Pei Shan and Lai Jiun Liang

for giving me moral support, money, cooperation, encouragement and also understandings

Thank You So Much & Love You All Forever

ACKNOWLEDGEMENT

I should like to acknowledge and thank all those who gave me the time and the opportunity to complete this study. A special thanks to my supervisor, Dr Shariman Bin Abdullah, whose help, giving suggestions and encouragement, helped me have a clear mindset on how to efficiently carry out my work progress and finish my project on time. Without his kind direction and proper guidance, this project would have been a never success.

Furthermore, I want to give special thanks to my coursemates who offered me a lot of inspiration and mental cooperation to complete this project. With the support and encouragement given by my friends and my family, I overcame the tensions and problems I faced when I was completing this report.

In conclusion, I would like to thank everybody who was important to this Final Year Project report; I would like to express the apology that I could not mention each of you personally.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST ABBREVIATIONS, SYMBOLS AND NOMENCLATURES

KKM	-	Kementerian Kesihatan Malaysia
CCTV	-	Closed-Circuit Television
CV	-	Computer Vision
AI	-	Artificial Intelligence
ML	-	Machine Learning
IDE	-	Integrated Development Environment
DNN	-	Deep Neural Network
ANN	-	Artificial Neural Network
CNN	-	Convolutional Neural Network
FPN	-	Feature Pyramid Network
TPUs	-	Tensor Processing Units
CPUs	-	Central Processing Units
GPUs	-	Graphics Processing Units
SSD	-	Single Shot Detector
R-CNN	-	Region-based Convolutional Neural Networks
R-FCN	-	Region-based Fully Convolutional Network
PASCAL VOC	-	PASCAL Visual Object Classes
CIFAR	-	Canadian Institute For Advanced Research
MS-COCO	-	Microsoft Common Objects in Context
SUN	-	Scene UNderstanding

Caltech	-	California Institute of Technology
ILSVRC	-	ImageNet Large Scale Visual Recognition
MNIST	-	Modified National Institute of Standards and Technology
ReLU	-	Rectified Linear Units
BLOB	-	Binary Large Object
NMS	-	Non-maximum Suppression
USB	-	Universal Serial Bus
HDMI	-	High Definition Multimedia Interface
LED	-	Light Emitting Diode
OpenCV	-	Open Source Computer Vision Library
LabVIEW	-	Laboratory Virtual Instrumentation Engineering Workbench
MATLAB	-	Matrix Laboratory
SD	-	Secure Digital
RAM	-	Random Access Memory
mAP	-	mean Average Precision
FPS	-	Frames Per Second
MB	-	Megabyte
GB	-	Gigabyte
%	-	Percentage
m	-	meter
ms	-	millisecond
s	-	second
V	-	Volt
W	-	Watt

VAC	-	Volt of Alternating Current
P	-	Pixel
A	-	Ampere
Lm	-	Lumen
SoC	-	System On a Chip
GHz	-	Gigahertz
LPDDR	-	Low-Power Double Data Rate
GPIO	-	General-Purpose Input/Output
OS	-	Operating System
HD	-	High Definition

CHAPTER 1
INTRODUCTION

1.1 Background of Study

In an era of globalization, the "pandemic" word is always heard everywhere. Almost the whole world on Earth are concerned about this global issue. This pandemic disease will affect the body health of a person and cause death if the person is in a critical situation. From the statistic posted by Kementerian Kesihatan Malaysia (KKM), since 10 December 2020, there are almost more than 1000 cases in one day. According to KKM, the total active cases in Malaysia until 24 December 2020 in 100,318 cases and 468 disease clusters. Besides that, the pandemic has not only caused the global economic downturn, but even some companies will face the fate of bankruptcy.

Furthermore, the global unemployment rate is increasing day by day due to the pandemic issue. Indirectly the crime will also increase due to the current situation. According to the crime index presented by the Department of Statistics Malaysia, the crime rate in 2019 is rising to 256,6 compared to 273,8 in 2018, per 100,000 population. There are a few methods that can be implemented to prevent or even reduce the risk of crime—for example, using a surveillance system like CCTV or simple camera drones for aerial inspection. As additional information, object detection is a computer vision technology that can apply to the surveillance system to recognize and locate certain objects like animals, signboard, vehicles, furniture, person, and so on in an image or video.

The evolution of deep learning has been introduced convolutional neural networks as a subset of deep neural networks which able to transfer large image datasets to computers. In addition, it will teach the computer to identify or track the object that has been seen in the videos or images with high accuracy, indirectly improve the performance and efficiency. By the way, a few challenges that the manufacturing industry needs to overcome, such as visual tasks, need a huge amount of computing power to run, the limitation of hardware resource on the device, and running object detection in real-time.

In addition, Raspberry Pi is a small single-board computer that the object detector can implement to maintain a real-time frame rate performance with high precision. On the other hand, the manufacturing industry's issue is that the traditional computer system provides high computing power than the single-board computer. Still, it is too expensive, and the size is big. Consequently, the single-board computer is chosen as project hardware due to its size, performance and cost-effectiveness.

1.2 Problem Statement

The recent years have seen many exciting developments in the field of the automation industry. Real-time object detection is such a more challenging task in computer vision applications. Previously, Single Shot Detector (SSD) algorithm is used to perform real-time object detection. However, the setup for the detection system is a huge expense for the industry due to its need for high computing power to support and run the program. Other than that, the performance of the detection system is also low accuracy and lower speed. Therefore, the MobileNet-SSD algorithm has been used in this project to increase the accuracy and speed of the object detection system when performing real-time tasks. In addition, the single-board computer will replace the traditional computer system to develop a low costs detection system with high accuracy and high speed, indirectly improving the performance of the object detection system.

1.3 Objectives

The objectives of the project are important in order to ensure that the research is carried out in order to resolve the issue which has been investigated. Objectives are shown as below:

1. To implement MobileNet-SSD algorithm into a low-cost vision detection system using OpenCV in order to create high accuracy and precision cost-effectiveness detection system.
2. To improve detection system with more cost-effectiveness detection system using a single board computer.

1.4 Scope

The scope of this project will be fixed according to the requirements from the objectives. By referring to the first objective, this project focused on creating high accuracy and precision cost-effectiveness detection system in the field of electric and electronic industry. However, there are too many types of electronic devices are currently available in the surroundings. Following are the scopes and limitations of the project:

- Five types of electronic devices: computer mouse, cell phone, remote, laptop and keyboard
- Size of an input image: 280 x 280 pixels, 320 x 320 pixels and 360 x 360 pixels.
- Range of distance between the camera and the object: 0.6m, 0.7m, 0.8m, 0.9m, and 1m.
- Network architecture: MobileNet-SSD algorithm

1.5 Importance Of The Study

Based on the statistic of the crime index in Malaysia, the crime rate is increasing day by day to a critical level. This issue should be carried out as fast as possible in order to reduce the crime rate. In this way, this study is important for people who have the capability to create its own home security system by using the Raspberry Pi with an object detector. It not only will reduce the cost of installing the CCTV surveillance system, but it also can increase the safeness of the members in order to reduce the risk of crime.

Besides that, the detection system can also be applied in small industries to trace the working environment of employees. For example, a production line delay due to the operator having an accident in line. In this way, the data collected in the detection system supports the operator and uses an effective way to solve the issue.

In addition, this study also important for the visually impaired. By using the object detection system to help a person who has a low vision problem to recognize the appearance and shape of the object accurately. Other than that, Google Lookout is an application on Google Pixel smartphone using the camera to detect the object and give the information to the user. This kind of application is very useful for the visually impaired.

1.6 Organization Of Report

Chapter 1 introduces the background of this project with respect to the title of this project. The problem statements are established through a range of research methods and literature reviews in books, the Internet, newspapers and the current global question. After the problem had been identified, there are a few objectives that should be carried out to solve the problem. This project's scope is limited so that the data can be obtained and analyzed accurately when conducting the experiment research for this project. Chapter 1 also

highlighted the importance of accurate use of engineering history information while running this project.

Chapter 2 have covered the basic theories, engineering knowledge and the details of information based on the title of this project. This information can be obtained from the journal, articles, the Internet, books and so on. There are a lot of data that can gain from different researchers in order to summarize the information.

Chapter 3 essentially addresses the approach to be used for the execution of this project. After analysis of different material from Chapter 2, the information was compiled, and an idea for this project came out. The way to incorporate the object detector implementation on Raspberry Pi for the object detection system is designed and built in this chapter. The appropriate components for this project were also compared with the information obtained in Chapter 2. This chapter will describe the experimental test in this project that should be carried out and the technique of collecting and analyzing the data to achieve this project's objective.

Chapter 4, this chapter is showing the results obtained from applying the methodology from Chapter 3. The result obtained will be the detection accuracy of different electronic devices after the image processing algorithm is applied. The distance between object and camera and the size of input images are also recorded and discussed in this chapter.

Chapter 5 effectively addresses the conclusion of all steps taken to implement this project, and the proposals for further implementation on this subject are made in this area. The recommendations and suggestions for this subject are offered from various perspectives.

1.7 Summary

Overall of this chapter is to summarize the purposes, the reasons and the initiatives of the project. The project title has been explained and addressed explicitly to clarify the definition used by this project in order to develop a real-time object detecting system that can recognize and locate the object in an image or video for home security purposes. This project aims to develop high accuracy and cost-effective detection system application that is affordable and efficient.

CHAPTER 2
LITERATURE REVIEW

This chapter deals with the research subject and the previous studies of other researchers based on their research regarding articles and internet sources. It included the issue of computer vision, object detection, deep neural network, algorithms, datasets, and image processing.

2.1 Object detection in Computer Vision

Computer vision is an interdisciplinary area of research that focuses on how to be able to obtain high-level expertise in computer graphics and video in system environments. From the engineering viewpoint, it attempts to simplify functions only the visual system can do. Besides, computer vision helps to ensure immediate observations, analyses, and comprehension of viable knowledge on a particular picture or a series of pictures. It requires establishing the theoretical and algorithmic basis for automated visual comprehension. Computer vision involves the philosophy underlying automated devices used to derive data from images as a research discipline. Image data could include different elements, such as video clips, views from various camera, and multi-dimensional imagery obtained from diagnostic files or scanners. Other than that, computer vision tends as a technical field to discuss its principles and models by incorporating them into the architecture for computer vision systems. According to Wu et al. (2020), object detection is known as visual recognition. Apart from that, the purpose of object detection is to classify objects in some

particular placed class on one given image, and assign the adequate class identifier to each object case. There are numerous main visual recognition issues in the area of computer vision which are image classification, object detection, semantic segmentation and instance segmentation.

Image classification is a kind of primary skill of the visual perception system. It mainly focuses on identifying the images among predefined categories by automatically (Feng et al., 2019). The main purpose of image classification is to identify semantic object categories in a given image. In the new era globalization, researchers developed innovative techniques to improve classification accuracy. Besides that, classification models typically only function well on small datasets like CIFAR-10 and MNIST. As extra information, Deng et al. (2009) has been created a large-scale image of ImageNet dataset in 2009. It was about the same time that well-known deep learning technology began exhibiting fantastic classification success and reached the computer vision stage. Example of image classification is shown in Figure 2.1.

Figure 2.1: Example of image classification (Wu et al., 2020).

Object detection is the primary skill most computers and robot vision systems need. Besides, object detection using in deep learning and computer vision to interact with data

feeds and video archives allow functionality to distinguish multiple types of subjects. In addition, object detection is a crucial visual perception activity and one of the main fields for computer vision applications. It mainly addresses the locating and position of particular objects in an image. As extra information, object detection is an image processing system that identifies and distinguishes objects like people, vehicles and animals from digital photographs and videos. This technology has the ability to at once identify one or more objects inside a digital image or video. Although object detection has been around for years, in a variety of sectors it is now more noticeable than ever before. Object detection using deep learning techniques provides more precision for different object types (Vahab et al., 2008).

Another important and difficult task in computer vision is to identify and locate the object instances either from a vast range of predefined categories in natural images or for a specific object, example, blurred image field and human face. As additional information, object detection and image classification have the same problems that many highly variable objects tend to handle. Apart from that, the object detection task is more challenging than image classification task. This is because object detection is compulsory to identify the precise position of the object of interest where an image classification task only identifies semantic object categories in a given image (Feng et al., 2019). Example of object detection is shown in Figure 2.2.

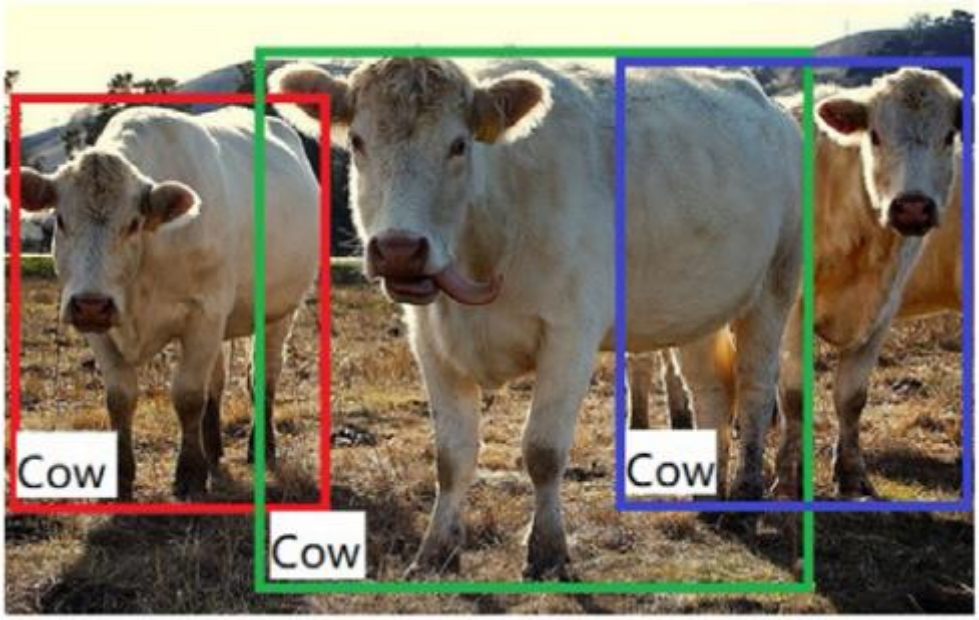


Figure 2.2: Example of object detection (Wu et al., 2020).

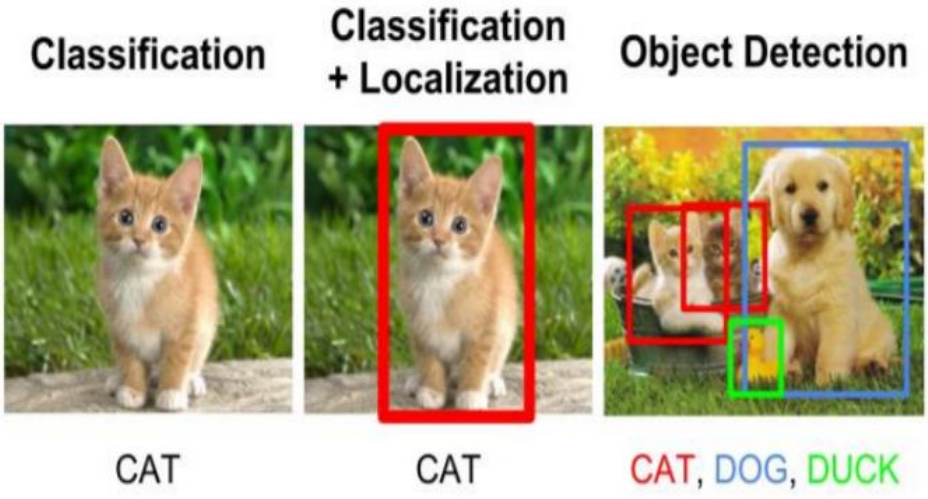


Figure 2.3: Process of object detection (Vahab et al., 2008).

Figure 2.3 illustrates the process of object detection. A classification describes as decide which of the several potential groups in that patch is present. Detection and localization establish whether a particular item of interest is situated somewhere in the picture and provide detailed details about the object.

Image segmentation is known to be a pixel-level classification that separates an image by classifying each pixel as a particular object into meaningful regions. Besides, the principle of an unattended local area fusing and separating has been widely studied in conventional image segmentation based on clustering, optimization of global parameters or user engagement. The thriving deep learning technologies have encouraged a broad scale supervised classification, from classifying images of objects to finding objects at box depths, and to segmenting objects in a pixel context (Feng et al., 2019). In addition, image segmentation consists of two categories which are semantic segmentation and instance segmentation. Furthermore, the goal of semantic segmentation is to anticipate pixel-wise classifiers to give each pixel a particular category mark, thereby offering an even more detailed picture. Instance segmentation is proposed to distinguish distinct entities and assign a separate pixel level mask to each. In reality, instance segmentation can be interpreted as a special object identification environment where pixel-level localization is required instead of a bounding box locating an object (Wu et al., 2020). Figure 2.4 and Figure 2.5 show the examples of image segmentation which is semantic segmentation and instance segmentation.

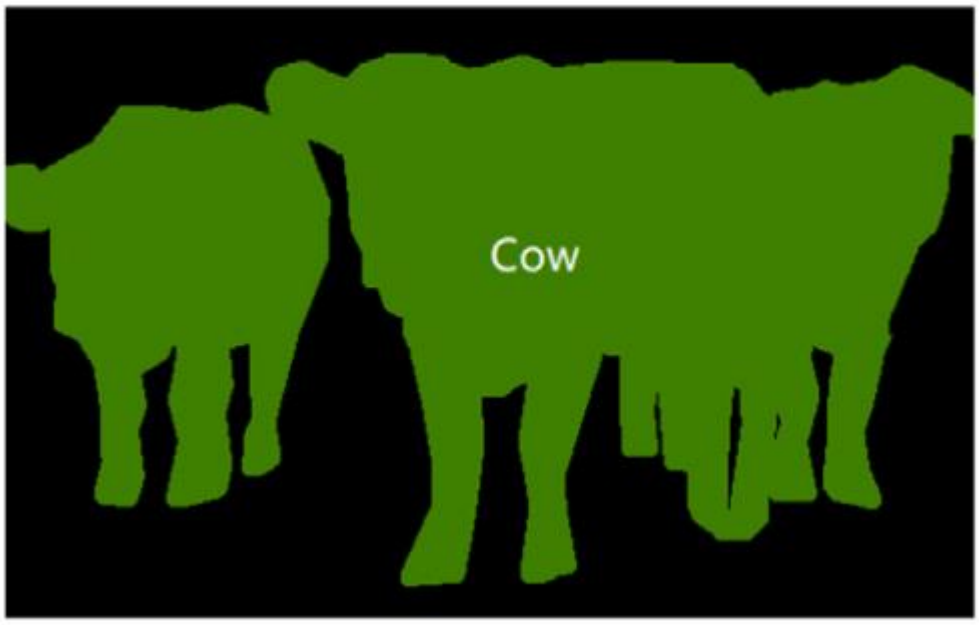


Figure 2.4: Example of semantic segmentation (Wu et al., 2020).

Figure 2.5: Example of instance segmentation (Wu et al., 2020).

2.2 Deep Neural Network In Deep Learning

Deep learning is a function of the use of the degree to synchronize artificial intelligence (AI), which emulates the capacity of the human brain to process information and it generates ways to be used in decision-making. Deep learning is an area of artificial intelligence that creates devices with a network able to learn from unelaborated or unlabeled outcomes. It is also known as a deep neural network. With the advancement of deep learning, big data has been created as the data that can easily access and shared through cloud computing to the user.

According to Shah & Kapdi (2017), deep neural network (DNN) defined as a multi-layered artificial neural network (ANN) between input and output layers. Although the neural network has too many types, the components are always the same such as functions, weight, synapses, neurons and biases. Besides that, the components can work like a human brain and it can be learned like other machine learning algorithm. Deep networks are those with depths exceeding 3 or more than 3 layers. The key drawback of using these methods is that non-linearity can still be used. As extra information, non-linearity can be used by applying Rectified Linear Units in the hidden layers. Besides, the total parameter in the deep neural network is $(n+1) \times k$ where n and k represented as input and output respectively. Figure 2.6 shows the operation of a deep neural network. Based on Figure 2.6, nodes represented as neurons of the human brain and it also is such a small component in the system. The mechanism will take place when the signal enters in nodes. As extra information, nodes are typically are categories into the layer.

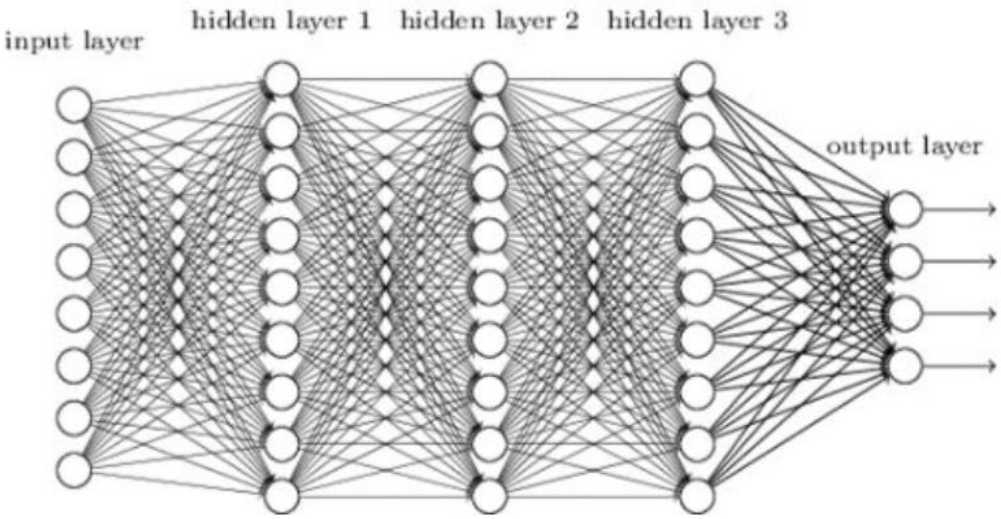


Figure 2.6: Deep neural network.

The advantages of using a linear function are better than using a non-linear function. These can be proved that the linear function differentiation is constant. Besides that, GPU matric multiplication is faster in linear function although non-linear. Linear functions are very constant, minor variations influence performance. A linear function can be used when integrating a neural network with more layers. Besides, the function of Rectified Linear Units (ReLU) is to convert the linear equation into a linear equation. Furthermore, the advantage of Rectified Linear Units is that function derivative is a fixed function. The function can be activated when adding a hidden layer in a neural network. Rectified Linear Units function is defined as $y = \begin{cases} 0 \; if \; x < 0 \\ x \; if \; x \geq 0 \end{cases}$.

Based on Shah & Kapdi (2017), regularization is a technique indirectly decreasing the number of parameters and it using wide network raises the risk of overfitting. Besides, implement regularization with the function of minimizing the overfitting when is completed. The time taken to complete the training depends on the results and the curve is almost linear. After some time overfitting begins and the testing should be finished at that stage. In addition, there are two approaches that regularization should take action which is L2 regularization and dropout. L2 Regularization applies some function to the loss function, decreasing the total number of iterations. Dropout loses several units from the neural network, but the network is not completely based on those units. Temporarily replaces the device. It avoids overfitting and offers an effective way to integrate several different neural networks.

2.2.1 Convolutional neural network (CNN) in Deep Neural Network

A convolutional neural network (CNN) is a subset of deep neural networks. Besides, CNN is most used to evaluate graphic imagery. The feed-forward artificial neural network commonly used to deliver accurate results in computer vision tasks. On the other side, convolutional neural networks described as a traditional neural network. It also has deeper layers, biases, performances and weights by a non-linear simulation. Furthermore, convolutional neural networks neurons are organized volumetrically such as height, distance, and depth (Galvez et al., 2019).

Figure 2.7 display the architecture of the convolutional neural network. Convolutional layer, pooling layer and a fully connected layer are shown in the CNN architecture. Normally, the convolutional layer and pooling layer will change constantly. Depth of each filter increases as the output size such as height and distance decreases. Besides, the last layer of the convolutional neural network is the same as the last stage for the fully connected layer.

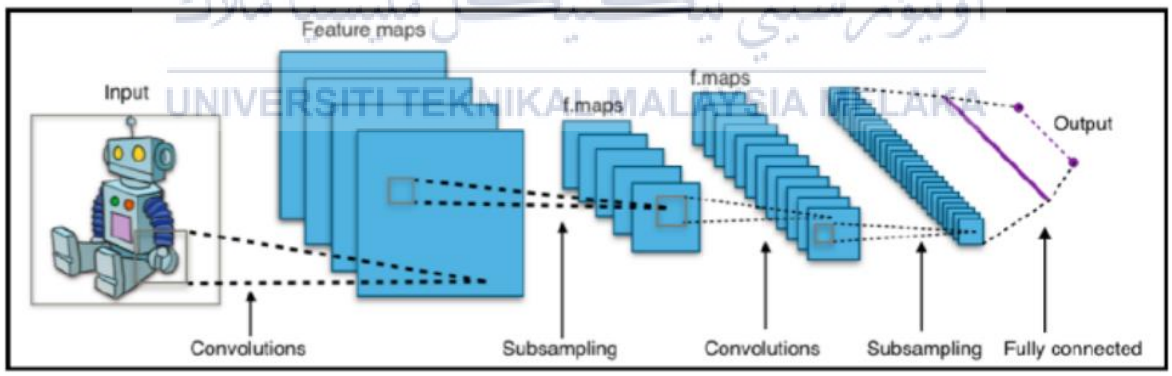


Figure 2.7: Architecture of the convolutional neural network (Galvez et al., 2019).

According to Galvez et al. (2019), the researcher states that input is an image containing pixel values and it has three dimensions. The convolution layer measures the output of neurons attached to input local regions. The parameters of layer consist of a series of learning filters that converged over the height and the width of the input volume, spanning across its depth, to measure the dot product between the input entries and the filter. This

creates a 2-dimensional filter activation map. As the result, the network learns filters to activate when a specific form of operation is observed in a certain spatial location in the input. The Rectified Linear Unit (ReLU) layer function performs an elementary activation function. The function of the Rectified Linear Unit is $f(x) = \max(0, x)$. For a negative value, the function is zero and it rises linearly for positive values. The researcher proved that the volume size is not affected even the function is zero. The outputs of the pooling area have maximum activation in an area. It samples spatial measurements such as distance and height. The output layer is the fully-connected layer that is equivalent to the neural network final layer. Commonly, the layer uses periodic softmax activation to spread the likelihood over the number of output groups (Galvez et al., 2019).

2.2.1.1 MobileNet architecture used for object detection

The architecture of MobileNet design uses depth-divisible convolutions that radically decrease the sum of parameters as opposed to ordinary network convolutions with equivalent network depth. They are lightweight, low-power, low-latency models parameterized to satisfy the use cases asset constraints like ongoing execution. Besides, there is a lack of precision for low-complexity deep neural networks when using strongly divisible convolutions.MobileNets architecture can be used for image classification, object detection, image segmentation and embeddings. In addition, inception model as large scale models is similar than MobileNets architecture.

According to Pehlivan et al. (2020), the researcher states that MobileNet is known as the base neural network. It selected as the based model convolutional neural network for classification of objects. Besides that, MobileNet requires fewer processing resources. This provides good result in embedded systems, This provides strong outcomes in embedded systems, mobile devices and systems without losing precision. This results well in embedded systems, mobile devices and low-computing performance systems. As extra information, it will not affect the precision of the system. MobileNets are built on a streamlined architecture that uses deep-separable convolutions to construct lightweight deep neural networks. The

two hyperparameters of the model essentially show a trade-off between latency and accuracy. Other than that, these hyperparameters allow the model maker to select the correct model for implementation depending on the problem. Figure 2.8 display the framework of MobileNet architecture.

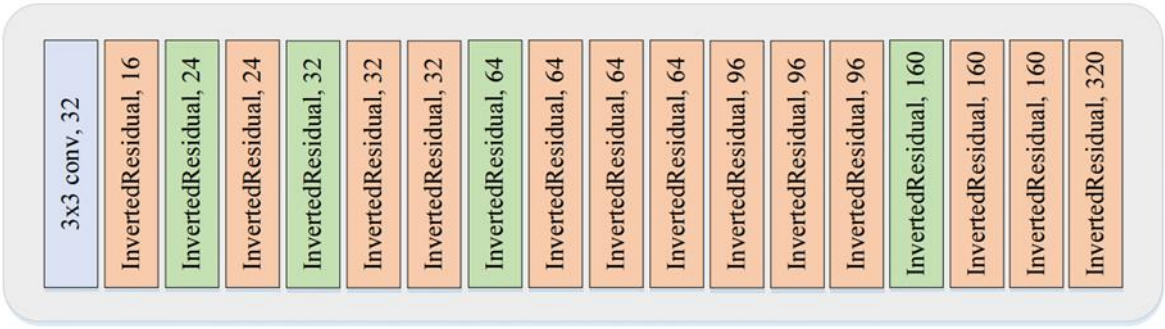


Figure 2.8: Framework of MobileNet architecture(Chiu et al., 2020).

2.2.1.2 Single Shot Detector (SSD) architecture used for object detection

According to Ali et al. (2019), single shot detector algorithm is easy compared with techniques which call for object proposals because it completely excludes the development of proposals and highlights re-sampling phases and encapsulates all estimates within a single network which makes SSD simple to train and easily integrated into frameworks that involve a segment for discovery. It produces bounding boxes and provides detection scores for the object with the help of non-maximum suppression point. This progression progresses to the last detections. Figure 2.9 shows the framework of the Single Shot Detector.

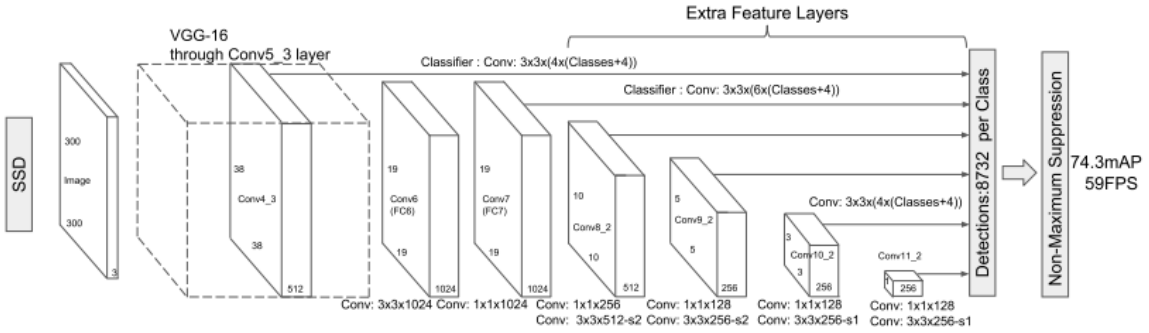


Figure 2.9: Framework of Single Shot Detector (SSD) (W. Liu et al., 2015).

Furthermore, SSD is described as a detection network. There are various default boxes applied and used to the feature maps over varying aspect sizes and ratios in the architecture of the Single Shot Detector. By using image classification network to determine the feature maps and the bounding box extraction function can be extracted in one stage. Each of the bounding boxes generates the score for each object category.

2.2.1.2.1 MobileNet-SSD architecture used for object detection

According to Pehlivan et al. (2020), MobileNet-SSD architecture is the combination of MobileNet architecture and Single Shot Detector (SSD) architecture where MobileNet architecture is used for prediction and SSD architecture is to identify the result of classification. In addition, MobileNet-SSD network architecture is shown in Figure 2.10 and Mobilenet-SSD backbone networks derived from multi-layer feature maps is shown in Table 2.1. Besides, the model of MobileNet-SSD will adequately minimize parameter amounts and achieve higher precision under minimal equipment conditions. Other than that, the model involves four parts which are the Single Shot Detector architecture used for classification and bounding box regressions, the MobileNet architecture is used for features extraction of the image, the input layer is for the objective image and the output layer is for getting the final result of the detection. This model supports fast and precise object identification provided that the MobileNet framework derived the overall computational complexity (Ali et al., 2019). Figure 2.11 shows the pyramid structure of MobileNet-SSD object detector.

Based on Ali et al. (2019), the researcher claimed that MobileNet-SSD has some drawbacks such as SSD implements extra layers as feature extraction layers. The improvement in these extra layers' size is apparent enough that multi-scale objects are observed, but the identification of small objects is not quite satisfactory. The researcher also suggests an improved Single Shot Detector system, called I-SSD, to solve these problems. These system is inspiring by GoogLeNets Inception network architecture and deep residual network.

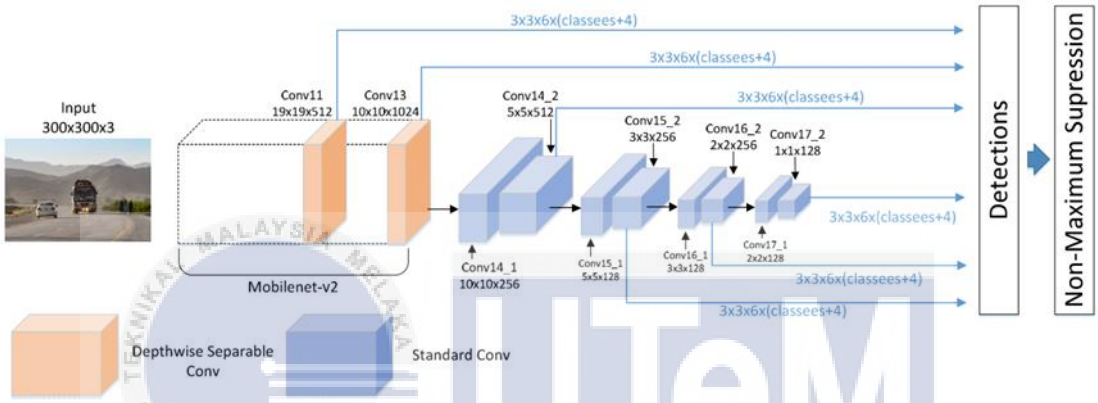


Figure 2.10: MobileNet-SSD network architecture (Chiu et al., 2020).

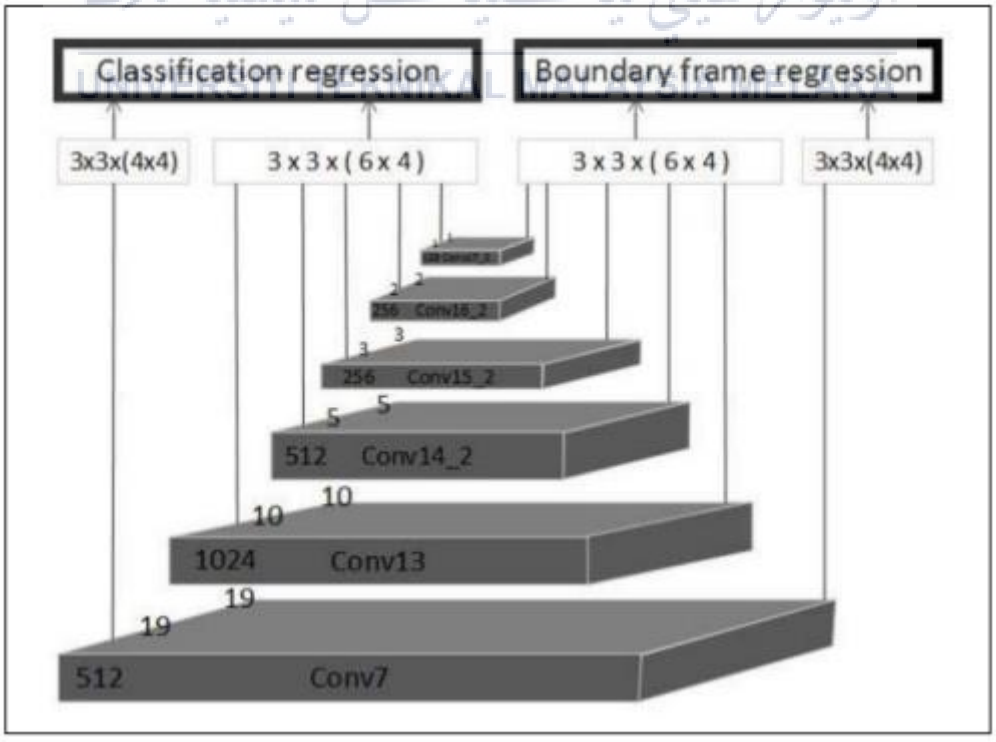


Figure 2.11: Pyramid structure of Mobile-SSD detector (Ali et al., 2019)

Table 2.1: Dimensional of the multi-layer feature maps (MobileNet-SSD)(Chiu et al., 2020)

Detector Model	MobileNet-SSD
Layer 1	19x19x96
Layer 2	10x10x1280
Layer 3	5x5x512
Layer 4	3x3x256
Layer 5	2x2x256
Layer 6	1x1x128

Besides that, MobileNet-SSD architecture is also known as the pre-trained model for object detection tasks. In addition, MobileNet-SSD architecture trained with PASCAL VOC dataset has up to 72.7% mAP. It means that almost 20 object category can be identified. MobileNet-SSD achieves best-precision trade-off for real-time implementations inside the fastest detectors. Furthermore, MobileNet-SSD architecture trained with MS-COCO dataset is used for real-time applications. Figure 2.12 shows the way for learning system task to get full data after the operation is completed.

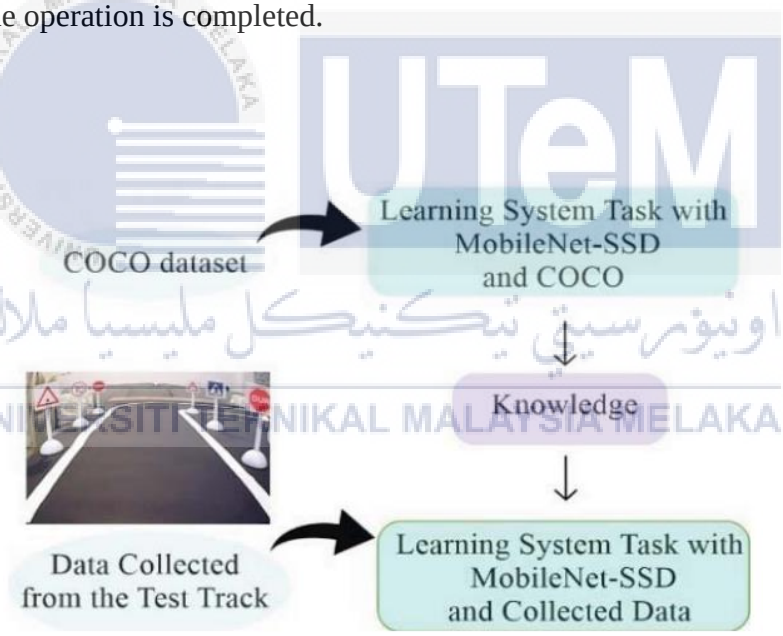


Figure 2.12: Learning system task (Pehlivan et al., 2020).

The training phase is observed on TensorBoard during the models training section. Besides, The mean-average accuracy and speed of the pre-trained model are the TensorFlow model zoo's values in Table 2.2. Pre-trained model's speed indicates processing time per image. The inference test time will be obtained after the code is running in order to get the images with bounding boxes, and it also measured with the same categories of images. As

extra information, the input image's specific hardware and size will affect the speed of the network.

In contrast, the transfer learning time data should be recorded with the inconsistent result due to specific tools and size of the input image. By evidence, the model's speed ranking must be the same as after and before transfer learning. In addition, the value of mean-average accuracy as the way to compare with different models. In conclusion, based on traffic sign detection, MobileNet-SSD as the pre-trained model of transfer learning is the suitable choice due to the fastest speed and comparable mAP with other pre-trained models. This can be proved that MobileNet-SSD trained with transfer learning has 71,58% of mean-average accuracy(mAP).

Table 2.2: Performance of detection networks (Pehlivan et al., 2020).

		Pre-trained Model		Model after Transfer Learning	
Detection Network	**Base Network**	**mAP (%)**	**Speed (ms)**	**mAP (%)**	**Inference Test Time (s)**
SSD	MobileNet	22	31	71,58	0,96
	Inception V2	24	42	73,49	1,41
	ResNet 50 fpn	35	76	78,21	1,84
Faster R-CNN	ResNet 101	32	106	71,51	2,16
	ResNet 50	30	89	73,1	1,87
	Inception V2 2	28	58	60,24	1,31
RFCN	ResNet 101	30	92	71,51	3,91

According to Chiu et al. (2020), the comparison result between the performance of MobileNet-SSD architecture and MobileNet-SSDv2 architecture is obtained. Firstly, the impact of modifying the backbone network model to MobileNet-v2. The findings reveal that the updated approach increases detection precision by around 0.6% mAP relative to the initial method. Secondly, the FPN module has been introduced to fuse multi-scale function maps that help to improve detection accuracy by 2.6%. Consequently, the proposed detector Mobilenet-SSDv2 will increase the Pascal VOC dataset's detection accuracy to 75% mAP. Based on the result, the performance of MobileNet-SSDv2 detector is 3.2% better than the current Mobilenet-SSD detector.

By comparing the speed of computing, without using the Feature Pyramid Network module the average frame rate when the proposed network detection model processes a 720p video stream on Nvidia Jetson AGX Xavier is around 23 FPS. Once the Feature Pyramid Network module has been added, the frame rate will only hit around 21 FPS on average. Other than that, MobileNet-SSDv2 detector has a major boost in processing speed relative to the original MobileNet-SSD detector, which has an overall processing speed of about 17 FPS. Besides that, MobileNet-SSDv2 network model's memory bandwidth is 32 MB which is it has 7 MB more than the current Mobilenet-SSD. However, the proposed detector will dramatically increase the pace and precision of processing (Chiu et al., 2020).

Table 2.3: Comparison between MobileNet-SSDv2 network and MobileNet-SSD network (Chiu et al., 2020).

Detector Model	**MobileNet-SSDv2**		**MobileNet-SSD**	
Input Image Resolution	512x512		320x320	512x512
+Mobilenet-v2	√	√		
+FPN		√		
mAP (%)	73.0	75.6	68.8	72.4
Frame Per Second(FPS)	23	21	20	17
Model Size (MB)	32	32	25	25

2.3 Tensorflow In Object Detection

TensorFlow is such a popular framework, and it is used for natural language analysis, text description and overview, voice recognition and translation. Besides that, TensorFlow is versatile and has an extensive collection of libraries and resources to build and execute Machine Learning applications. TensorFlow primarily utilizes deep learning with Python in designing strategies. In contrast, there are many hidden layers in deep learning compare with conventional machine learning networks. Besides, Deep Learning TensorFlow is one of the strongest libraries globally because much of the data is unstructured and unmarked.

According to Abadi et al. (2016), TensorFlow is a machine learning system that works in large-scale, heterogeneous settings. Besides, it uses dataflow graphs to reflect computation, mutual position, and mutating operations. TensorFlow maps dataflow graph nodes over several computers in a cluster and across many computing devices inside a computer. Example of computational devices is the multi-core central processing unit (CPUs), Tensor Processing Units (TPUs) and general-purpose computing on graphics processing units (GPUs). Figure 2.13 illustrates TensorFlow's dataflow graph, which provides input data, reader, preprocessing, parameter, training, and periodic checkpoint.

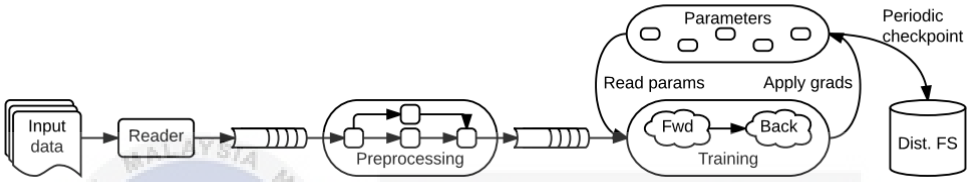


Figure 2.13: Dataflow graph of TensorFlow

In addition, both large-scale training and inference are supported by TensorFlow. TensorFlow effectively uses hundreds of powerful servers for rapid testing and operates qualified inferencing models on different platforms, varying from broad clustered clusters in a datacenter to inferencing locally on mobile devices. Simultaneously, promoting innovation and analysis into experimental machine learning frameworks and system-level optimizations are scalable and general enough (Abadi et al., 2016).

2.4 Dataset Related To Object Detection

Dataset defined as a data array. These datasets are used for machine-learning analysis and have cited in scholarly articles reviewed by peer reviews. Besides that, datasets are part and parcel of the machine learning industry. Significant innovations in this domain will result from advancement in learning algorithms (such as deep learning), computer hardware, and the availability of high-quality training datasets. Besides, supervised and semi-monitored

machine learning algorithms are typically hard to generate with highly-qualified labelled training data because there is a great deal of time to mark the data.

According to Liu et al. (2020), the researcher states that throughout the background of object recognition science, datasets have played an important part not only as a standard base for calculating and comparing the output of successful algorithms but also moving the field into more challenging and difficult issues. Recently, deep learning approaches have brought immense progress for many visual recognition challenges, and they play an essential role in the success of many annotated data. Access to vast quantities of photos on the Internet enables extensive datasets to catch a great richness and variety of objects, allowing unparalleled object recognition efficiency.

Furthermore, these are the four datasets usually used for generic object detection such as PASCAL VOC dataset, ImageNet dataset, Open Images dataset and MS-COCO dataset. Besides, Table 2.4 and 2.5 are shown summarized for the properties and characteristic of these datasets. Other than that, there are three phases to construct large-scale annotated datasets that define target item categories, gather several candidate images to reflect selected categories on the Internet, and annotate the collected images, usually by designing crowdsourcing techniques (Liu et al., 2020).

Table 2.4: Four datasets for object detection (Liu et al., 2020).

Dataset Name	Total Images	Categories	Images Per Category	Object Per Image	Images Size	Started Year
PASCAL VOC (Everingham et al., 2015)	11,540	20	303 ~ 4087	2.4	470 x 380	2005
ImageNet (Russakovsky et al., 2015)	14 million+	21,841	-	1.5	500 x 400	2009
Open Images (Kuznetsova et al., 2017)	9 million+	6000+	-	8.3	Varied	2017
MS-COCO (Lin et al., 2014)	328,000+	91	-	7.3	640 x 480	2014

Table 2.5: Characteristic of four datasets (Liu et al., 2020).

Dataset Name	Characteristics
PASCAL VOC (Everingham et al., 2015)	• Covers only 20 categories that are common in everyday life. • A large number of training images. • Close to real-world applications. • Significantly larger intraclass variations. • Objects in scene context • Multiple objects in one image. • Contains many difficult samples.
ImageNet (Russakovsky et al., 2015)	• A large number of object categories. • More instances and more categories of objects per image. • More challenging than PASCAL VOC. • The backbone of the ILSVRC challenge. • Images are object-centric.
Open Images (Kuznetsova et al., 2017)	• Annotated with image-level labels, object bounding boxes and visual relationships • Open Images V5 supports large scale object detection, object instance segmentation and visual relationship detection.
MS-COCO (Lin et al., 2014)	• Even closer to real-world scenarios • Each image contains more instances of objects and richer object annotation information. • Contains object segmentation notation data that is not available in the ImageNet dataset.

For extra information, these four datasets have formed the foundation of its respective detection challenges. Moreover, each challenge involves a freely accessible dataset of images, a ground-truth annotation and structured appraisal software, an annual competition and accompanying workshop. Table 2.6 demonstrations the performance of four datasets such as PASCAL VOC, ImageNet, Open Images and MS-COCO in each type of challenge.

Table 2.6: Summary of the performance of object detection datasets in each challenge (Liu et al., 2020).

Challenge	Object Classes	Number of Images			Number of Annotated Objects		Summary (Training +Validation)		
		Training	Validation	Testing	Training	Validation	Images	Boxes	Boxes/Image
PASCAL VOC Object Detection Challenge									
VOC07	20	2,501	2,510	4,952	6,301(7,844)	6,307(7,818)	5,011	12,608	2.5
VOC08	20	2,111	2,221	4,133	5,082(6,337)	5,281(6,347)	4,332	10,364	2.4
VOC09	20	3,473	3,581	6,650	8,505(9,760)	8,713(9,779)	7,054	17,218	2.3
VOC10	20	4,998	5,105	9,637	11,577(13,339)	11,797(13,352)	10,103	23,374	2.4
VOC11	20	5,717	5,823	10,994	13,609(15,774)	13,841(15,787)	11,540	27,450	2.4
VOC12	20	5,717	5,823	10,991	13,609(15,774)	13,841(15,787)	11,540	27,450	2.4
ILSVRC Object Detection Challenge									
ILSVRC13	200	395,909	20,121	40,152	345,854	55,502	416,030	401,356	1.0
ILSVRC14	200	456,567	20,121	40,152	478,807	55,502	476,668	534,309	1.1
ILSVRC15	200	456,567	20,121	51,294	478,807	55,502	476,668	534,309	1.1
ILSVRC16	200	456,567	20,121	60,000	478,807	55,502	476,668	534,309	1.1
ILSVRC17	200	456,567	20,121	65,500	478,807	55,502	476,668	534,309	1.1
MS COCO Object Detection Challenge									
MS COCO15	80	82,783	40,504	81,434	604,907	291,875	123,287	896,782	7.3
MS COCO16	80	82,783	40,504	81,434	604,907	291,875	123,287	896,782	7.3
MS COCO17	80	118,287	5,000	40,670	860,001	36,781	123,287	896,782	7.3
MS COCO18	80	118,287	5,000	40,670	860,001	36,781	123,287	896,782	7.3
Open Images Challenge Object Detection (OICOD)									
OICOD18	500	1, 643, 04	100, 000	99, 999	11, 498, 734	696, 410	1, 743, 042	12, 195, 144	7.0

2.4.1 PASCAL Visual Object Classes (VOC) dataset

According to Everingham et al. (2010), the challenge of PASCAL VOC is a benchmark in the identification and detection of visual object types, provides vision and machine learning communities with stock photos and annotation details and standard validation procedures. Besides that, these challenge involves two parts which is a public picture dataset, annotation of the field and structured assessment software; and an annual competition and workshop. Besides, five challenge tasks commonly involve in PASCAL VOC dataset, which are classification, detection, segmentation, action classification, and person layout. From these five challenge tasks that separate into two groups which are principal challenges tasks and subsidiary challenge tasks. Classification, detection, and segmentation represented as the principal challenge tasks, whereas action classification and person layout are the subsidiary challenge tasks. Next, the overview of principal challenge tasks and subsidiary challenge tasks are shown respectively in Table 2.7 and Table 2.8.

Table 2.7: Tasks for each type of principal challenge (Everingham et al., 2010, 2015).

Principal Challenge Tasks	Tasks
Classification	For each of twenty object classes predict the presence or absence of at least one object of that class in a test image. Participants are required to provide real-valued confidence of the object's presence for each test image so that a precision-recall curve can be drawn. Participants may choose to tackle all, or any subset of object classes, for example 'cars only' or 'motorbikes and cars'.
Detection	For each of the twenty classes, predict the bounding boxes of each object of that class in a test image (if any), with associated real-valued confidence. Participants may choose to tackle all or any subset of object classes. Two competitions are defined similarly to the classification challenge.
Segmentation	For each test image, predict the object class of each pixel, or give it 'background' status if the object does not belong to one of the twenty specified classes. There are no confidence values associated with this prediction. Two competitions are defined similarly to the classification and detection challenges.

Table 2.8: Tasks for each type of subsidiary challenge (Everingham et al., 2010, 2015).

Subsidiary Challenge Tasks	Tasks
Action classification	For each of ten action classes predict if a specified person (indicated by a bounding box) in a test image is performing the corresponding action. The output is real-valued confidence that the action is being performed so that a precision-recall curve can be drawn. The action classes are 'jumping', 'phoning', 'playing an instrument', 'reading', 'riding a bike', 'riding horse', 'running', 'taking photo', 'using computer', 'walking', and participants may choose to tackle all, or any subset of action classes, for example walking only' or 'walking and running'. Note, the action classes are not exclusive; for example, a person can be both 'riding a bicycle' and 'phoning'.
Person layout	For each person in a test image (their bounding box is provided) predict the presence or absence of parts (head, hands and feet), and the bounding boxes of those parts. The prediction of a person layout should be output with associated real-valued confidence of the layout so that a precision-recall curve can be generated for each person. The success of the layout prediction depends both on (i) a correct prediction of parts present/absent (e.g. are the hands visible or occluded); (ii) a correct prediction of bounding boxes for the visible parts. Two competitions are defined similarly to the classification challenge.

Based on Everingham et al. (2010), the purpose of the PASCAL VOC challenge is to analyze the efficiency of recognition methods on a broad range of natural pictures. There are two significant subsets data which are trainval data and test data. In addition, PASCAL VOC dataset has twenty classes of annotation images. For instance, for each of the 20 classes, all photos are described with bounding boxes for classification and detection (Everingham et al., 2015). Besides, bounding boxes represent all situations in the related class in the picture that are labelled as non-difficult. Figure 2.14 shows the 20 object classes in PASCAL VOC dataset.

Figure 2.14: PASCAL VOC dataset (20 Classes) (Liu et al., 2020).

Figure 2.15 are showing the samples of annotation images which consists of five challenge tasks are classification, detection, segmentation, action classification and person layout in PASCAL VOC dataset. In the sample images of classification and detection, each image has an annotation file which gives each object in one of the twenty classes on the image a bounding box and object class name. For additional information, the image will contain several objects from the multiple classes. A subset of photos is often annotated with pixel-wise segmentation of each present entity to support segmentation competition. Other than that, images for the action classification challenge task have been partially annotated with people, bounding boxes, reference points and actions. Next, images for the person

layout taster have been annotated with parts of people such as the head, hands and feet (Everingham et al., 2015).

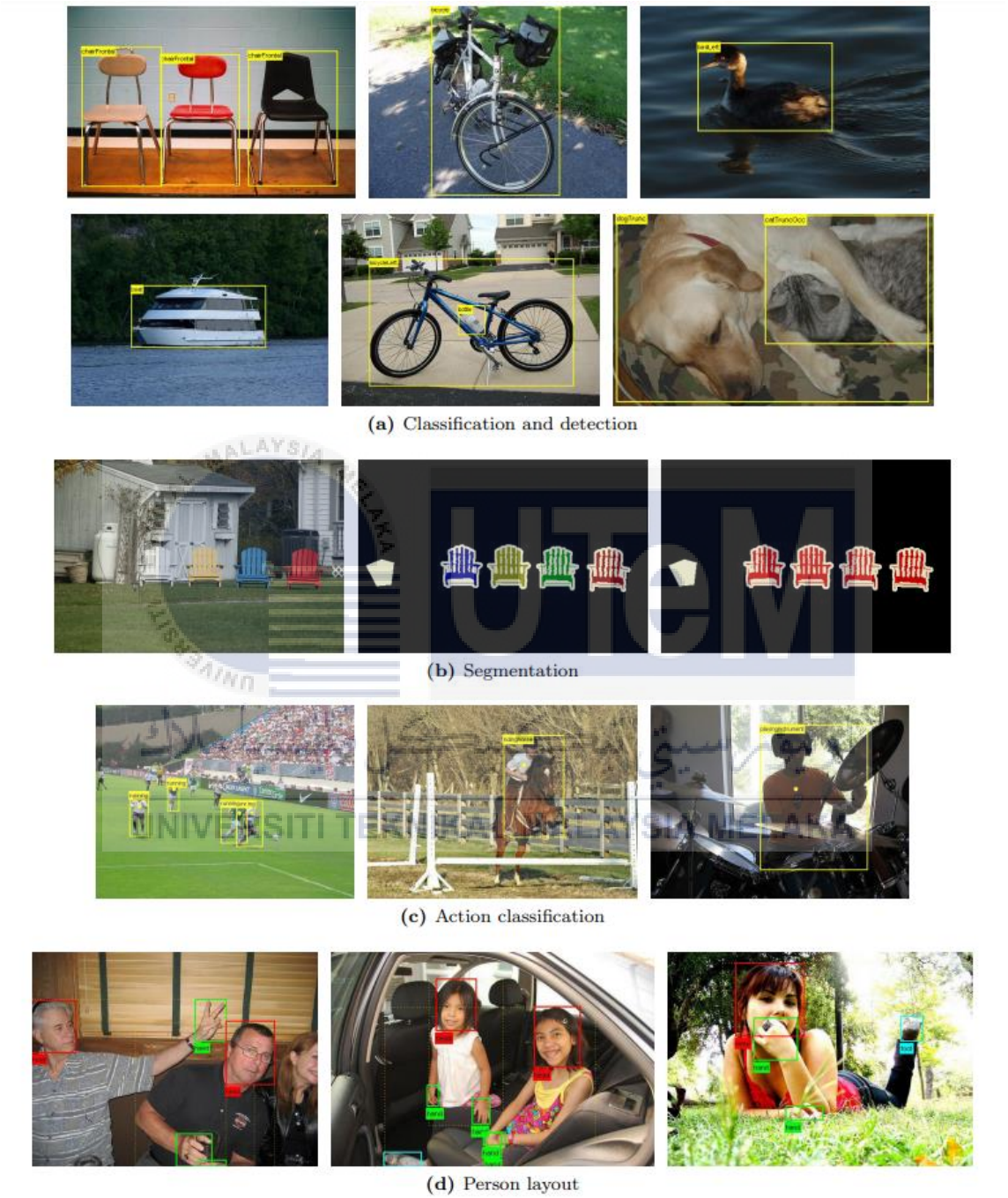


Figure 2.15: Sample of annotation images in PASCAL VOC dataset (Everingham et al., 2015).

2.4.2 ImageNet dataset

ImageNet is a WordNet Hierarchy image database in which each node of the hierarchy represents hundreds and thousands of pictures. Besides that, the ImageNet project is a broad graphic library for use in applications for visual object recognition. About 14 million photographs have been hand-annotated by the initiative to display which items are captured, and bounding boxes are also visible in at least one million images. Other than that, ImageNet comprises more than 20,000 common category types, for example, "balloon" or "strawberry" consisting of hundreds of photos. According to Russakovsky et al., (2015), the researcher states that ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a reference in classification and detection of objects on hundreds of categories and millions of images. Figure 2.16 shows the 200 object classes in ILSVRC dataset.

Figure 2.16: ILSVRC dataset (200 Classes) (Liu et al., 2020).

Table 2.9: Challenge tasks in ImageNet dataset (Russakovsky et al., 2015).

Challenge Tasks (years in parentheses)	Tasks
Image Classification (2010 – 2014)	Algorithms produce a list of object categories present in the image.
Single-object localization (2011-2014)	Algorithms produce a list of object categories present in the image, along with an axis-aligned bounding box indicating the position and scale of one instance of each object category.
Object detection (2013-2014)	Algorithms produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each object category.

2.4.2.1 Image Classification Task

According to Russakovsky et al. (2015), the image classification task data is made up of images obtained from Flick and other search engines that are manually labelled with the inclusion of 1000 categories of items. Besides that, each picture includes a mark of the ground truth. Algorithms generate a set of object categories for each image. Furthermore, the accuracy of a product is measured using the label closest representing the picture label ground truth.

Based on the challenge tasks in the ImageNet Large Scale Visual Recognition Challenge, each of the challenge tasks consists of two types of object categories which are the easiest and hardest classes. From image classification tasks, there are 121 out of 1000 types of objects have 100% image recognition precision. It is categorized in different forms, such as mammals and creatures with distinctive structures. Besides, the precision of the hardest classes in the image classification is lower than 59%, including metallic and human-made objects, substances and a widely diversified class of scenes. Figure 2.17 shows the easiest and hardest classes in the image classification task.

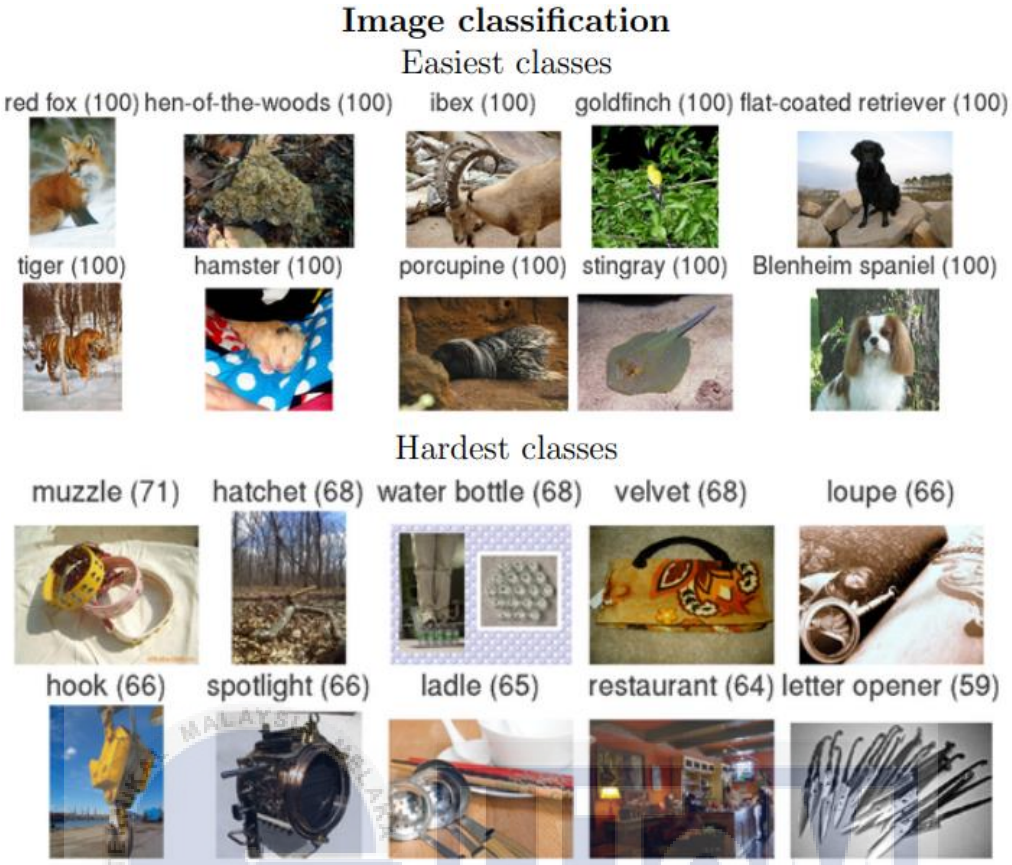


Figure 2.17: Easiest and hardest classes in the image classification task (Russakovsky et al., 2015).

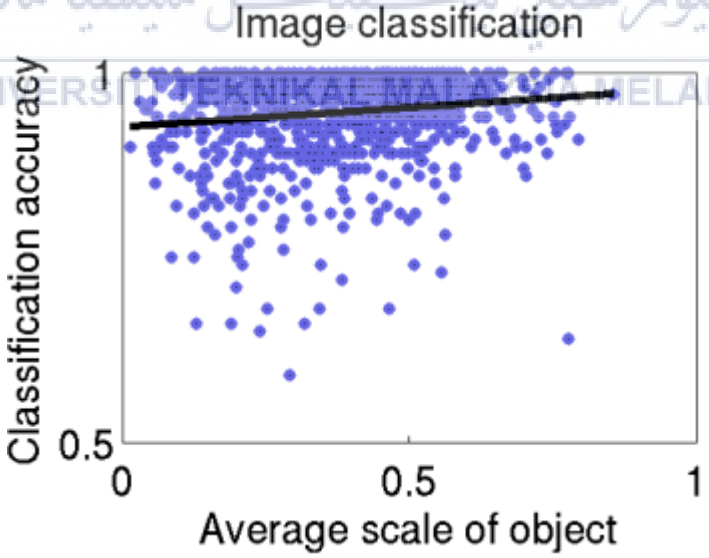


Figure 2.18: Classification accuracy versus the average scale of an object.

2.4.2.2 Single-object localization task

Based on Russakovsky et al. (2015) claimed that the data for the single object localization task consists of the same pictures obtained for the image classification task, hand labelled with the inclusion of one of 1,000 types of objects. Besides that, each picture includes a mark of the ground truth. Furthermore, any case in this group is noted with an axis-aligned bounding box. For each photo, algorithms list the categories of objects present in a picture and a bounding box showing the location and size for a single instance of each category of item. Apart from that, the accuracy of the label is measured based on the entity type mark that better correlates to the simple truth mark, and the additional condition is that the expected instance is located correctly as well.

From single-object localization task, both mammals and birds are the ten easiest classes with a range of 99.0% to 100% precision. Besides that, metallic products, thin framework and widely diverse groups are among the hardest classes. As extra information, spacebar as a thin framework which is the most challenging class and only have 23.0% of accuracy. Figure 2.19 shows the easiest and hardest classes in the single-object localization task.

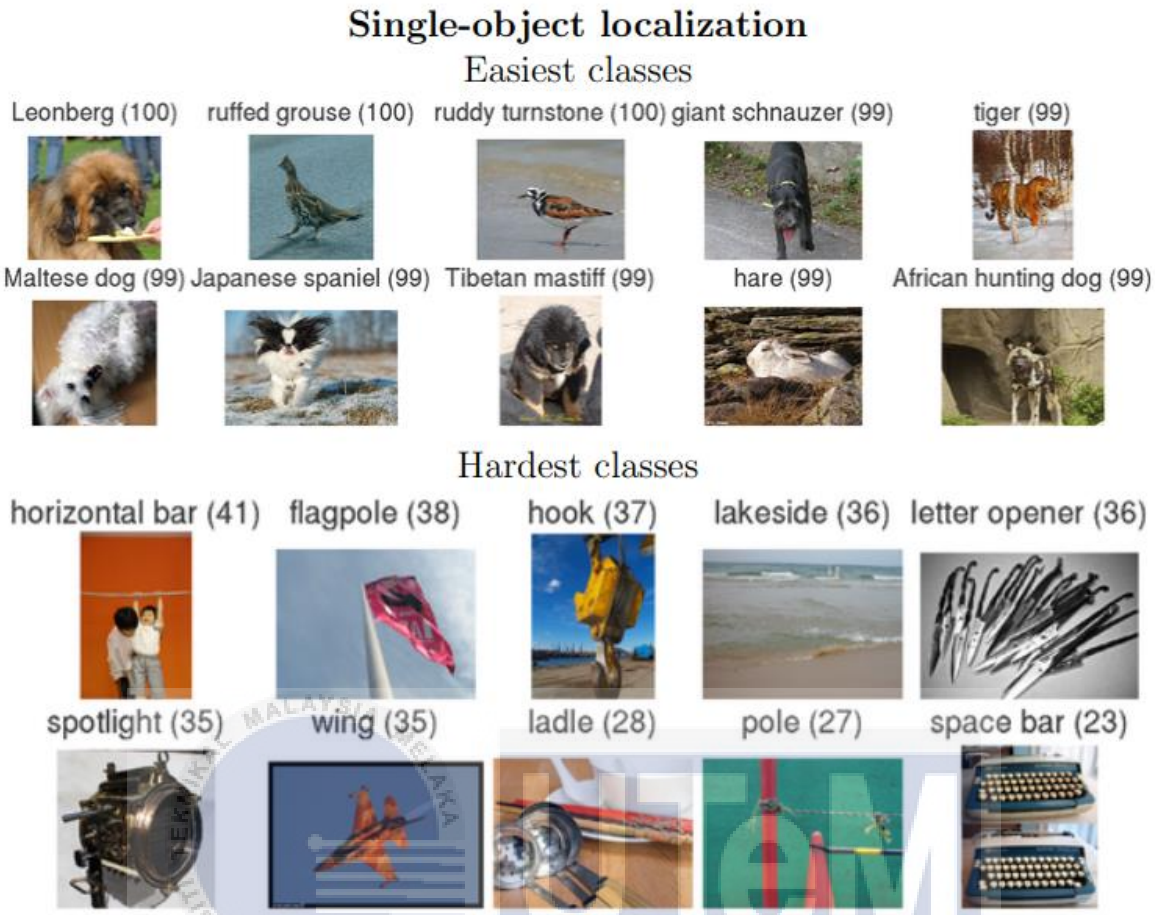


Figure 2.19: Easiest and hardest classes in the single-object localization task (Russakovsky et al., 2015).

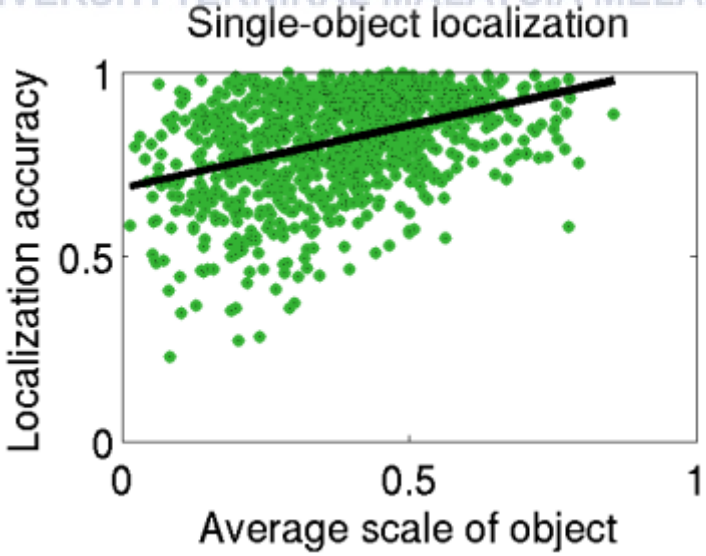


Figure 2.20: Localization accuracy versus the average scale of an object.

2.4.2.3 Object detection tasks

According to Russakovsky et al. (2015), data for object detection tasks involves fresh images obtained from Flickr utilizing scene-level queries. Besides, algorithms generate bounding boxes for each picture, showing the location and scale of all instances of all target categories. Other than that, the performance of labelling is measured by recording and accuracy. Besides, the object detection task has the ability of an algorithm to label and identify all instances of all target objects in an image. The following are three main obstacles in gathering the data for object detection (Russakovsky et al., 2015):

1) Choose the collection of specific objects that are suitable for the benchmarking of object detection efficiency in cluttered images.
2) Prepare the more complex collection of scenario images.
3) The dataset should be fully annotated for all objects.

For object detection task, the easiest classes are living creatures, plus distinctive form and colour, and quite impressive. Besides, the butterfly is categorized in the easiest classes, which are not yet wholly detected but is almost with 92.7% average precision. For the hardest classes, the average precision of tiny thin objects and extremely diverse classes is below 8.0%. Figure 2.21 shows the easiest and hardest classes in the object detection task.

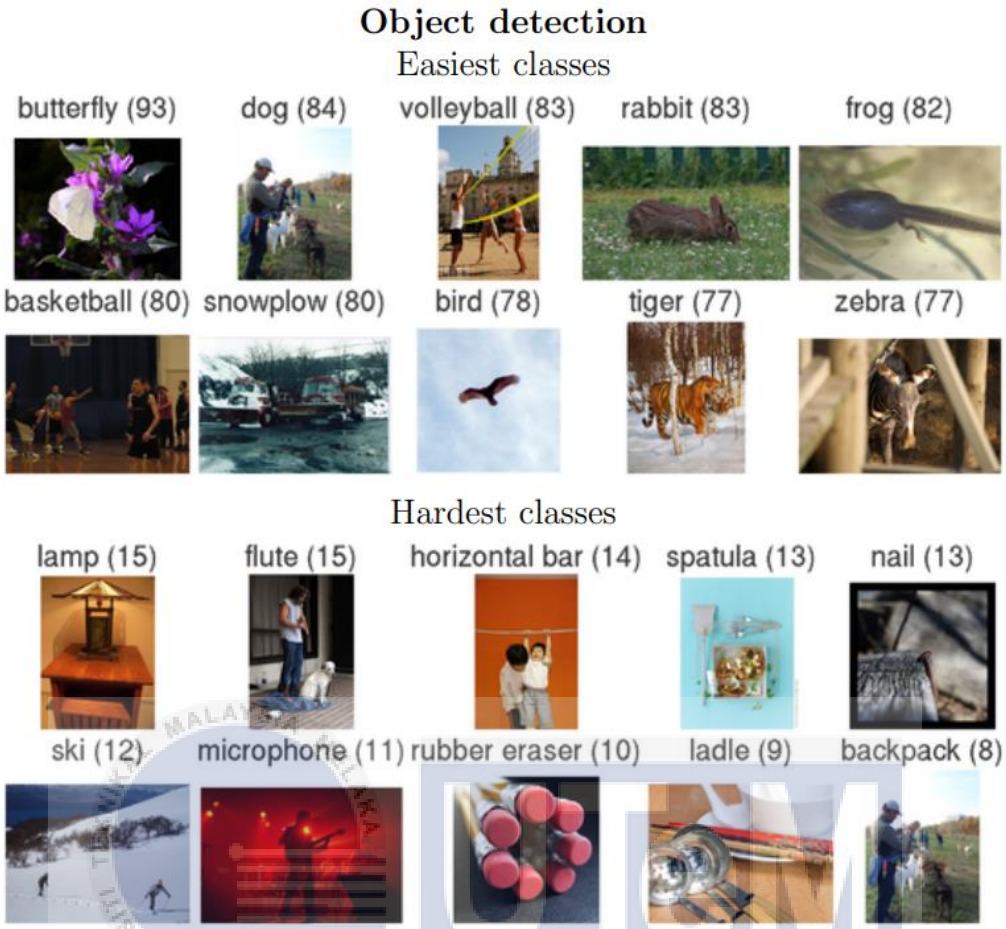


Figure 2.21: Easiest and hardest classes in the object detection task (Russakovsky et al., 2015).

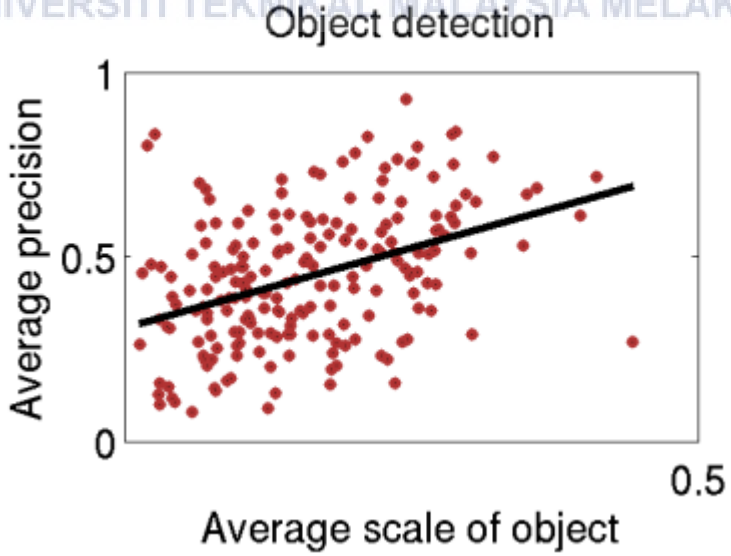


Figure 2.22:Average precision versus the average scale of an object.

2.4.3 Open Images dataset

Open Images is a dataset with almost 9 million picture URLs. Besides, these photographs have been annotated with image-level marks covering thousands of groups. The dataset comprises a training collection of 9,011,219 photos, 41,260 pictures, and 125,436 pictures. According to Kuznetsova et al. (2017), the researcher mentioned that Open Images V4 is a 9.2 million image dataset with unified image classification, object detection, and visual relationship detection annotations. Besides, the photos have a "Creative Commons Attribution" license, enabling the uploading and adapting of the content, gathered from Flickr without a pre-defined set of class names or identifiers, resulting in natural grade statistics and preventing initial style distortions. Figure 2.23 shows the sample of annotations images in Open Images dataset. Figure 2.24 illustrations the 500 object classes in Open Images Detection Challenge dataset.

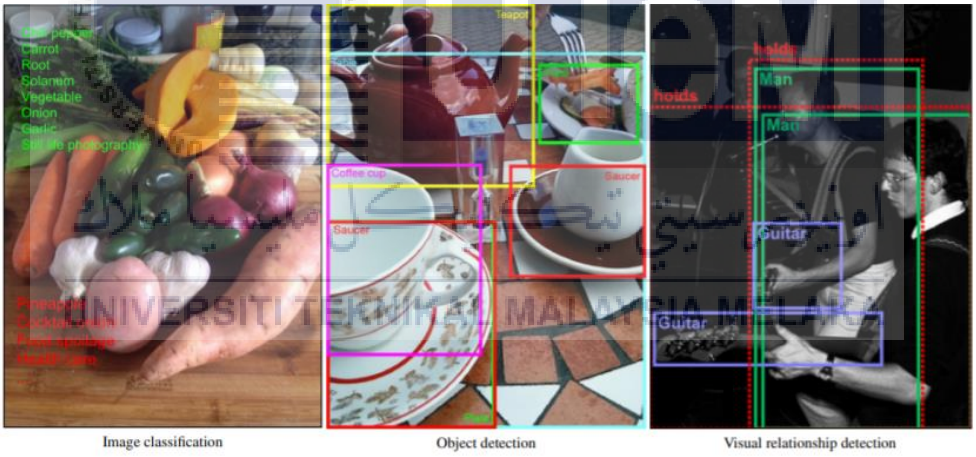


Figure 2.23: Sample of annotations images for each task (Kuznetsova et al., 2017).

Figure 2.24: Open Images Detection Challenge dataset (500 classes) (Liu et al., 2020).

Following are the method for collecting the images in Open Images dataset (Kuznetsova et al., 2017):

<table>
<tr><th>No.</th><th>Procedure</th></tr>
<tr><td>1.</td><td>Identify all Flickr images with CC-BY license. This was done in November 2015.</td></tr>
<tr><td>2.</td><td>Download the original version of these images and generate a copy at two resolutions:
1600HQ: Images have at most 1,600 pixels on their longest side and 1,200 pixels on their shortest. JPEG quality of 90.
300K: Images have roughly 300,000 pixels. JPEG quality of 72.</td></tr>
<tr><td>3.</td><td>Extract relevant metadata of all images to give proper attribution:
OriginalURL: Flickr direct original image url.
OriginalLandingURL: Flickr image landing page.
License: Image license, a subtype of CC-BY.
Author: Flickr name of the author of the photo.
Title: Title given by the author in Flickr.
AuthorProfileURL: Link to the Flickr author profile.
OriginalMD5: MD5 hash of the original JPEG-encoded image.</td></tr>
<tr><td>4.</td><td>Remove images containing inappropriate content (porn, medical, violence, memes, etc.) using the safety filters on Flickr and Google SafeSearch.</td></tr>
<tr><td>5.</td><td>Remove near-duplicate images, based on low-level visual similarity.</td></tr>
<tr><td>6.</td><td>Remove images that appear elsewhere on the internet. This was done for two reasons: to prevent invalid CCBY attribution and to reduce bias towards web image search engines.</td></tr>
<tr><td>7.</td><td>Recover the user-intended image orientation by comparing each original downloaded image to one of the Flickr resizes.</td></tr>
<tr><td>8.</td><td>Partition the images into train (9,011,219 images), validation (41,620) and test (125,436) splits.<table><tr><th></th><th>Train</th><th>Validation</th><th>Test</th></tr><tr><td>Images</td><td>9,011,219</td><td>41,620</td><td>125,436</td></tr></table></td></tr>
</table>

2.4.3.1 Analysis of Open Images dataset

Table 2.10 shows the comparison between four datasets for object detection includes the number of classes, images and bounding boxes. Based on the table below, Open Images dataset is significantly bigger than prior datasets and has 17 times more bounding object boxes than the MS-COCO dataset. It also involves complicated pictures with annotated multiple items.

Table 2.10: Comparison between four datasets for object detection

	PASCAL VOC	ILSVRC		Open Images	MS-COCO
		All	Dense		
Classes	20	200	200	600	80
Images	11,540	476,688	80,462	1,910,098	123,287
Boxes	27,450	534,309	186,463	15,440,132	886,284
Boxes/image	2.4	1.1	2.3	8.1	7.2

Figure 2.25 shows the relationship between the percentage of images versus the number of object per images. From Figure 2.25 can be proved that MS-COCO dataset and Open Images dataset are considerably less prejudicial to single object images.

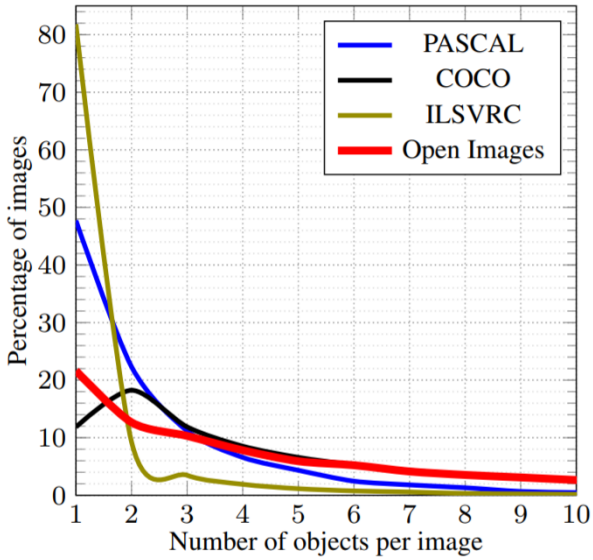


Figure 2.25: Graph of the percentage of images versus the number of object per images (Kuznetsova et al., 2017).

Figure 2.26 displays the relationship between the number of images versus the number of object per images. Based on the Figure 2.13, the researcher proved that Open Images contains considerably more images than other datasets throughout the entire spectrum of boxes per picture, especially with high values, where it covers several of the previously undiscovered regimes.

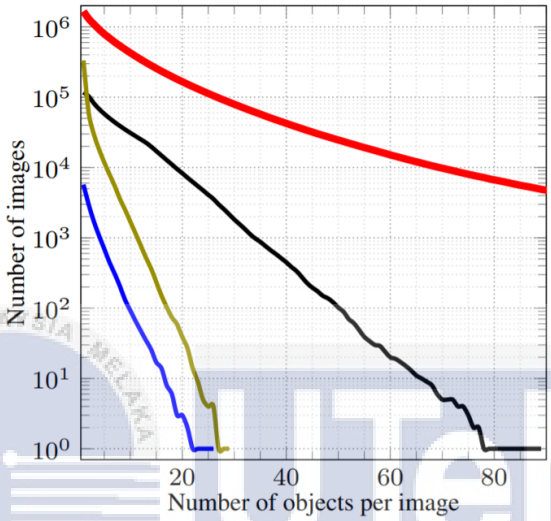


Figure 2.26: Graph of the number of images versus the number of object per images (Kuznetsova et al., 2017).

Besides that, Figure 2.27 shows the relationship between the number of boxes and class sorted position. The horizontal axis is displayed on a logarithmic scale to compare with the four datasets with fewer classes visually. Based on the graph below show that Open Images dataset is usually more significant than the other datasets in magnitude. In addition, Open Images dataset consists of 11 classes which have more sample images than the largest class in MS-COCO dataset. As a specific illustration, there are 257,253 instances in the MS-COCO dataset. In comparison, Open Images dataset contains 3,505,362 instances of a grouping of classes corresponding to a male person, woman, child, girl and kid. On the other side, Open Images contains 517 classes with more instances than MS-COCO dataset, which has 198 instances and ILVSRC, which contain 417 classes with 502 instances.

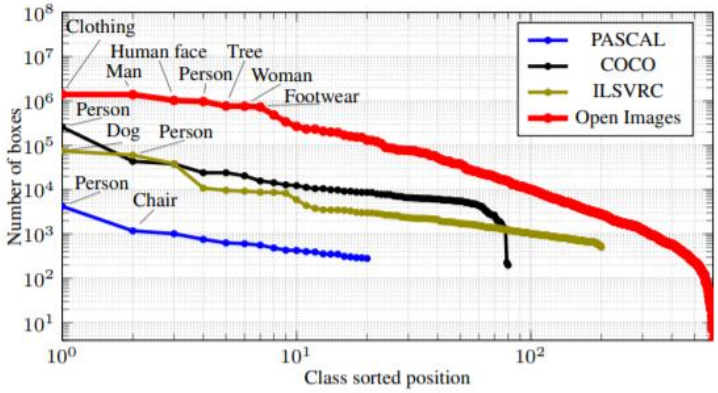


Figure 2.27: Graph of number of boxes versus class sorted position (Kuznetsova et al., 2017)

Figure 2.28 shows the relationship between the percentage of images and number of district classes per images for each dataset. Based on the Figure 2.15, the researcher proved that Open image dataset and MS-COCO dataset provide a much wealthier distribution of pictures in co-occurring groups compared with ILSVRC dataset and PASCAL VOC dataset, which are more inclined towards a single class of the scene.

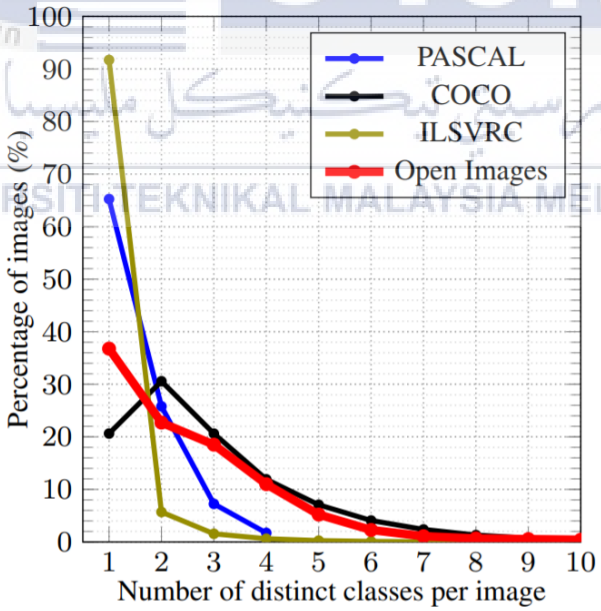


Figure 2.28: Graph of the percentage of images versus the number of distinct classes per image (Kuznetsova et al., 2017).

Figure 2.29 displays the relationship between the number of images and number of district classes per images for each dataset. It indicates that the Open Images dataset has at every

stage in the curve at least one more magnitude order than the MS-COCO dataset. For instance, the dataset for Open Images has around 1000 images with 14 distinct classes, while MS-COCO dataset consists of 1000 images with 20 distinct classes, ILVSRC dataset has no image with more than 11 distinct classes, and PASCAL VOC has no image with no more than four distinct classes.

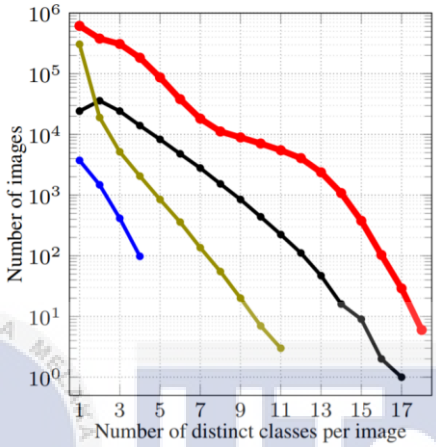


Figure 2.29: Graph for the number of images and number of distinct classes per image (Kuznetsova et al., 2017).

2.4.4 Microsoft Common Objects in Context (MS-COCO) dataset

The Microsoft Common Objects in Context (MS-COCO) dataset is the gold standard benchmark for measuring computer vision model performance. The MS-COCO dataset is labelled with data to train supervised computer vision models that can recognize familiar objects in the dataset. MS-COCO dataset comprises 91 common object types, 82 of which have over 5,000 classified instances. The dataset has 2,500,000 named instances in 328,000 images. Unlike the common ImageNet dataset, MS-COCO dataset has fewer categories, but more per category. In addition, the dataset is considerably higher in the number of instances per category than the dataset of PASCAL VOC and SUN. Furthermore, a crucial difference between MS-COCO dataset and others is the number of labelled instances per image, which may help to learn contextual knowledge. MS- COCO dataset contains more object instances per image (7.7) than ImageNet dataset (3.0) and PASCAL VOC dataset (2.3). In comparison,

the SUN dataset, which contains essential contextual information, has more than 17 objects and "stuff" per image, but fewer total object instances. Figure 2.17 and Figure 2.18 show the 80 object classes for MS-COCO dataset and 11 supercategories in MS-COCO dataset with 91 categories icons respectively. Besides, Figure 2.30 shows the sample of annotation images in MS-COCO dataset.

Figure 2.30: MS-COCO dataset (80 Classes) (Liu et al., 2020).

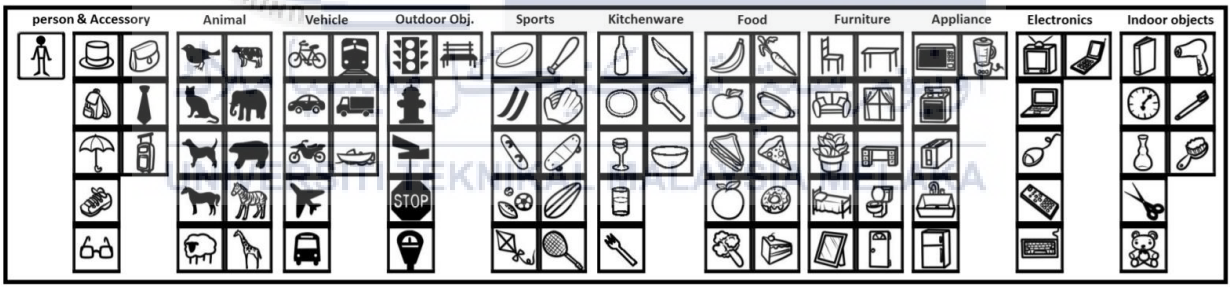


Figure 2.31: 91 categories icons in MS-COCO dataset (Lin et al., 2014).

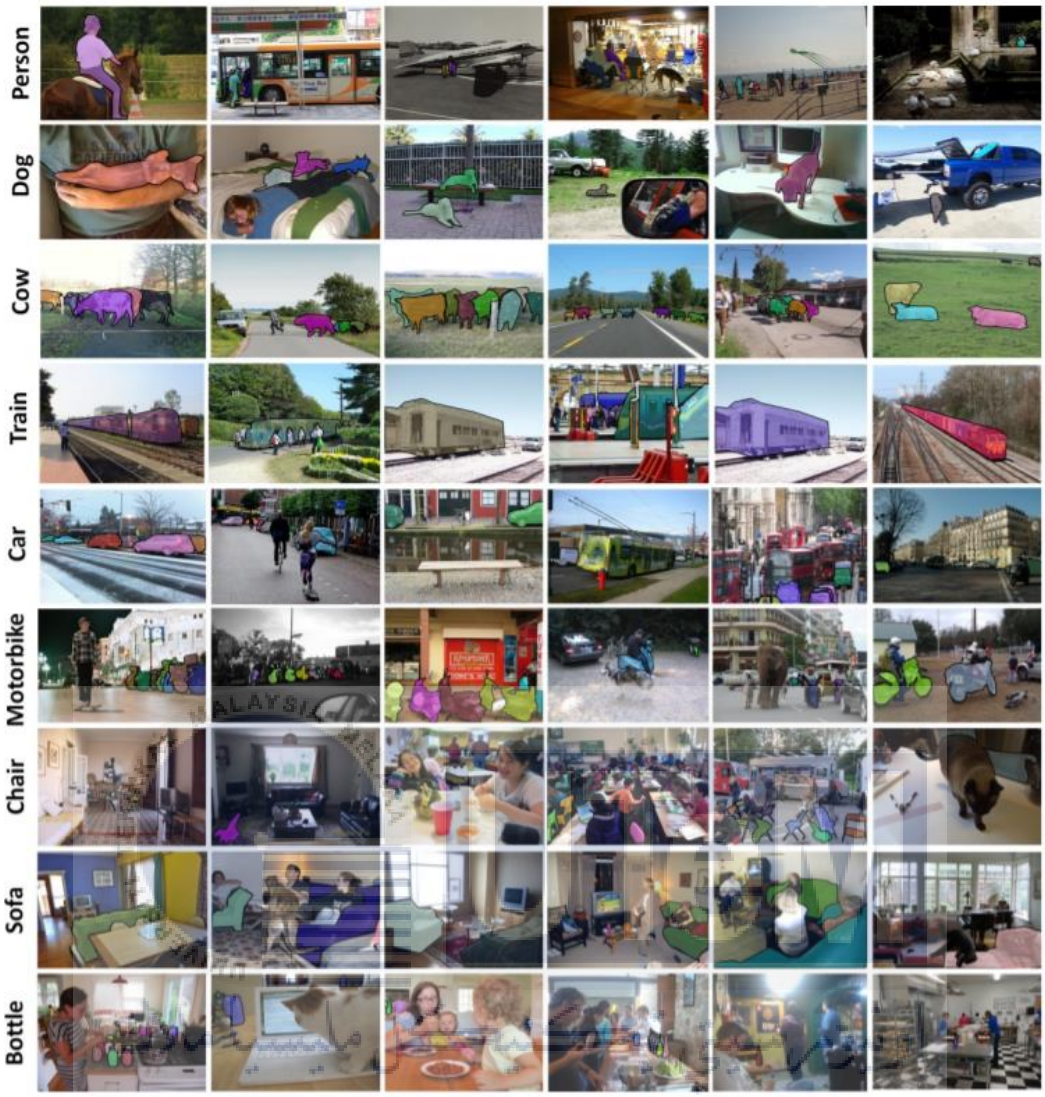


Figure 2.32: Sample of annotation images in MS-COCO dataset (Lin et al., 2014).

2.4.4.1 Image classification task

Object classification challenge includes binary labels indicating whether objects are in an image. Besides, this type of early datasets contained photographs featuring a single entity with blank backgrounds, such as MNIST handwriting digits or COIL household objects. Caltech 101 and Caltech 256 marked the transition to more accurate images from the Internet, thus increasing the number of object types to 101 and 256, respectively. Because of the more significant number of training samples, CIFAR-10 and CIFAR-100 gave 10 and 100 categories from a dataset of tiny 32x32 images. While these datasets included 60,000

photographs and hundreds of classes, they only captured just a small fraction of our visual environment (Lin et al., 2014). Figure 2.33 shows the image classification on object recognition datasets.

Figure 2.33: Image classification

2.4.4.2 Object detection task

According to Lin et al. (2014), the researcher mentioned that object detection requires both specifying that an object belonging to a given class is present and localizing it in the image. Bounding boxes is commonly used to display the location of an object. For detecting simple object types, a multi-year project from 2005 to 2012 was committed to developing and preserving a collection of broadly accepted benchmark datasets. Based on Everingham et al. (2010), the researcher states that PASCAL VOC dataset consists of 20 of object classes spread over 11,000 pictures and about 27,000 object bounding boxes were labelled, of which approximately 7,000 had comprehensive segmentations. Moreover, ImageNet dataset provides 200 of object classes by using a subset of 400,000 images and impressive 350,000 objects were labelled using bounding boxes. In addition, detection datasets play an essential role to detect the object in natural environments which consists of highly dependent on contextual information.The function of bounding boxes often restricts the precision with

which detection algorithms can be measured.Figure 2.34 shows the sample image of object localization.

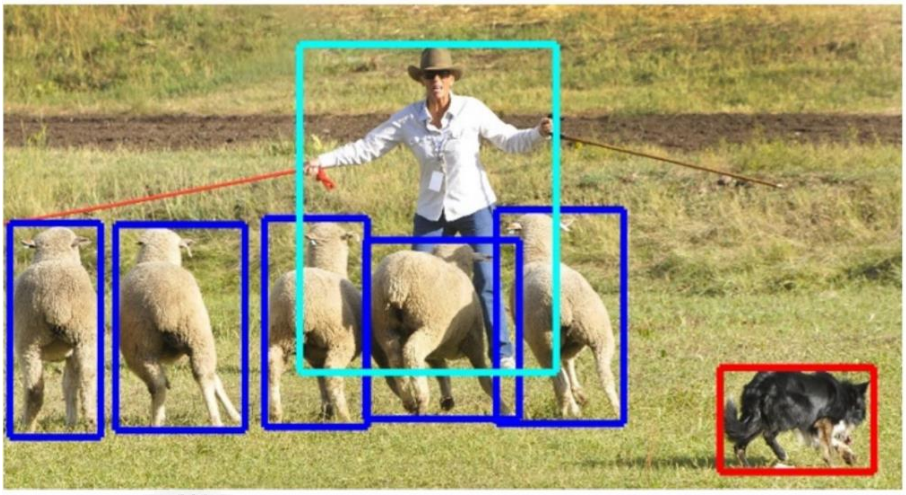


Figure 2.34: Object localization

2.4.4.3 Semantic scene labelling task

Role of labelling semantic objects in a scene involves that each pixel of an image is identified as belonging to a group, such as a sky, chair, floor and lane. Unlike detecting mission, individual object instances should not be segmented. This makes the labelling of objects on which specific examples, such as grass, streets or walls, are difficult to identify. Other than that, datasets are including the indoor and outdoor scene, and it also provides the depth information for the scene (Lin et al., 2014). Figure 2.35 shows the sample images of semantic pixel-level segmentation.

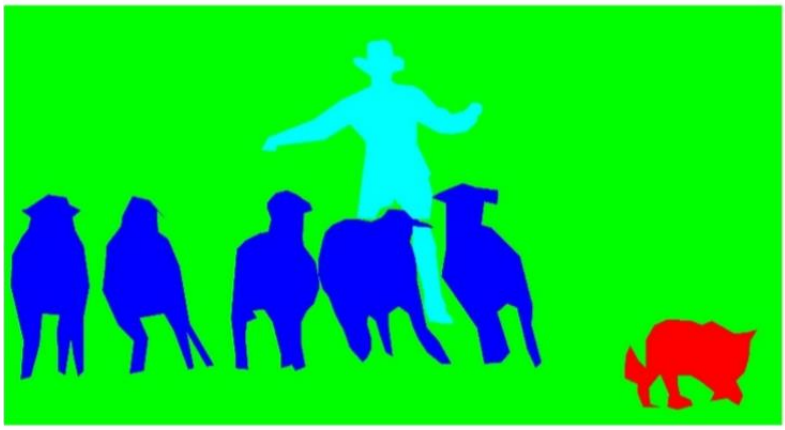


Figure 2.35: Semantic pixel-level segmentation.

2.4.4.4 Analysis of the statistic of ms-coco dataset

According to Lin et al. (2014), the researcher claimed that the analysis is provided with some bar chart and graph to compare datasets based on the properties. The datasets include are ImageNet dataset, PASCAL VOC dataset and SUN dataset. As extra information, each type of datasets has a different size, a different type of images and different in the list of labelled categories. In deep, ImageNet dataset was developed to collect several object types and many of which are fine-grained while SUN dataset is more focusing on labelling scene categories and common object occur in the scene. Other than that, the main application of PASCAL VOC dataset is object recognition in natural images while MS-COCO dataset is intended to identify and segment objects in a natural context. Figure 2.36 displays the chart for comparing the MS-COCO dataset and PASCAL VOC dataset based on the number of instances per category.

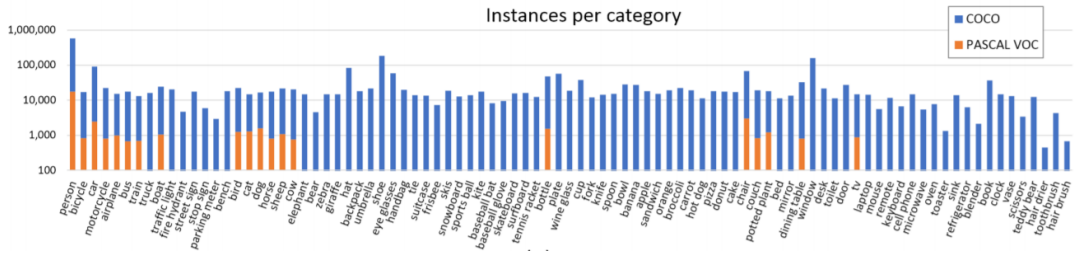


Figure 2.36: The number of instances per category for MS-COCO dataset and PASCAL VOC dataset (Lin et al., 2014).

Figure 2.37 displays a review of the several object recognition datasets indicating the number of object categories and the number of instances per category. Based on the graph, the researcher has proved that the MS-COCO dataset has fewer categories but more instances per category than the ImageNet dataset and SUN dataset. Besides, it helps to study dynamic models capable of accurate localization. By comparing with PASCAL VOC dataset, MS-COCO dataset performs more categories and more instances per category.

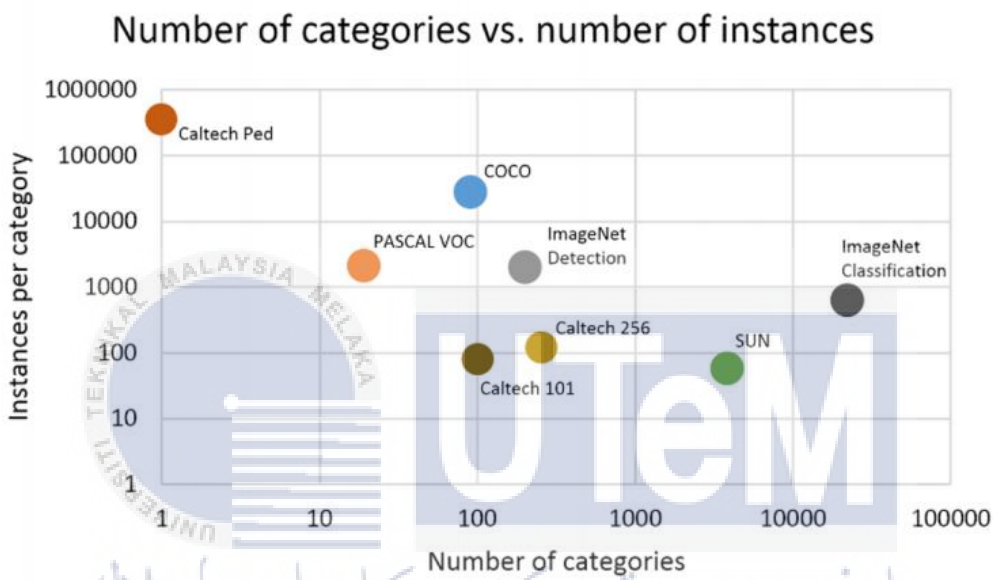


Figure 2.37: Review of the several object recognition datasets indicating the number of object categories and the number of instances per category (Lin et al., 2014).

Figure 2.38 and Figure 2.39 show the graph of categories per images and instances per image respectively for MS-COCO dataset, PASCAL VOC dataset, ImageNet dataset and SUN dataset. From both graphs, the researcher proved that the average of MS-COCO dataset contains 3.5 categories per image and 7.7 instances per image while comparing with PASCAL VOC dataset and ImageNet dataset which both averages of datasets have less than two categories per image and three instances per image. Besides that, MS-COCO dataset contains only 10% of the images and each image only have one category. By the way, more than 60% of the images contain a single object in both PASCAL VOC dataset and ImageNet dataset. In addition, SUN dataset provides the most contextual knowledge since it uses a variety of definitions depending on a scene.

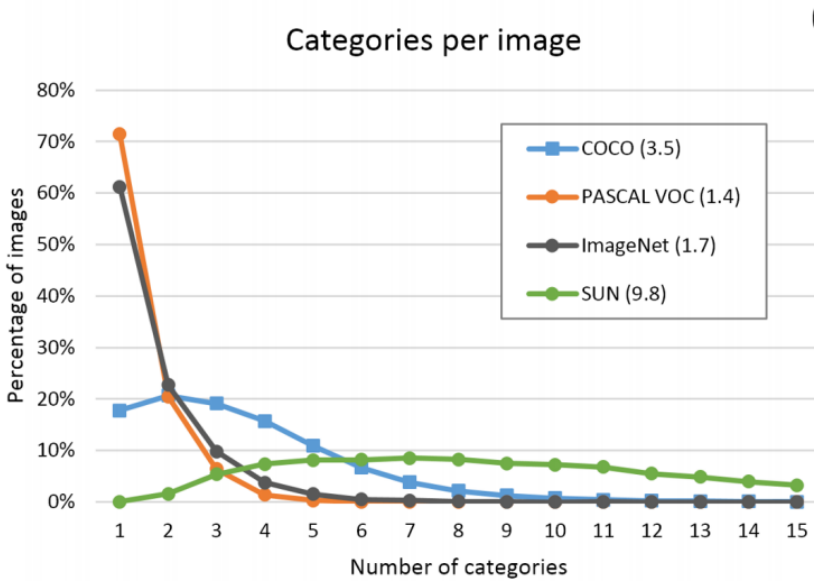


Figure 2.38: Graph of the number of categories per images for each dataset (Lin et al., 2014).

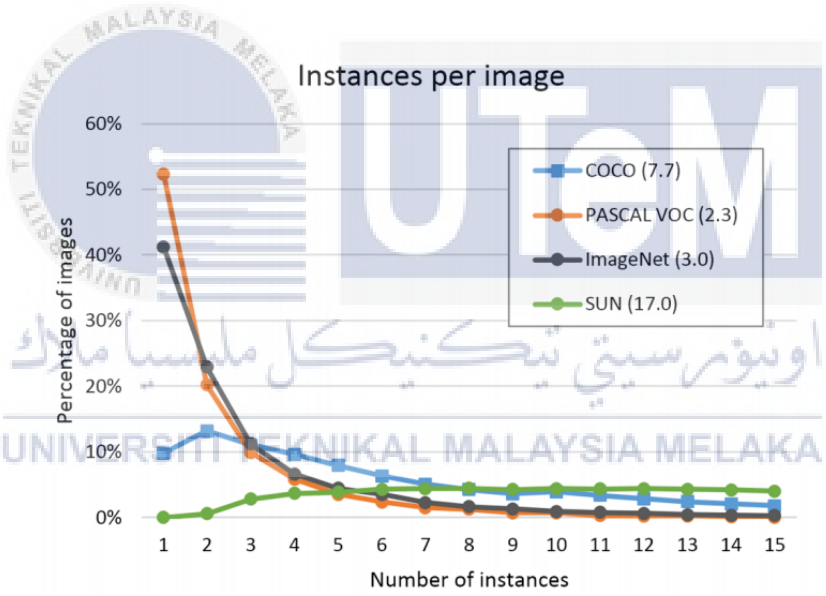


Figure 2.39: Graph of the number of instances per images for each dataset (Lin et al., 2014).

Figure 2.40 displays the average size of objects in MS-COCO dataset, PASCAL VOC dataset, ImageNet dataset and SUN dataset. According to Lin et al. (2014), the researcher mentioned that based on the graph it could be concluded that the smaller size of objects is difficult to recognize compare with bigger size of objects because more contextual reasoning is required to consider smaller objects.

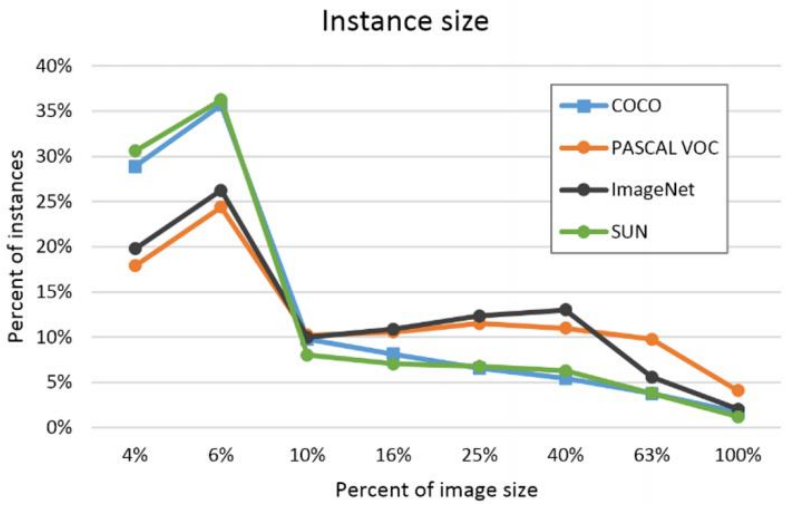


Figure 2.40: Range of instance sizes for each dataset (Lin et al., 2014).

2.5 Components Of Object Detection System

Various components are needed to implement the object detection system method, the following topic will be addressed on the necessary components and the software required. These necessary components for creating an effective object detection system are studied based on different literature reviews.

2.5.1 Raspberry Pi

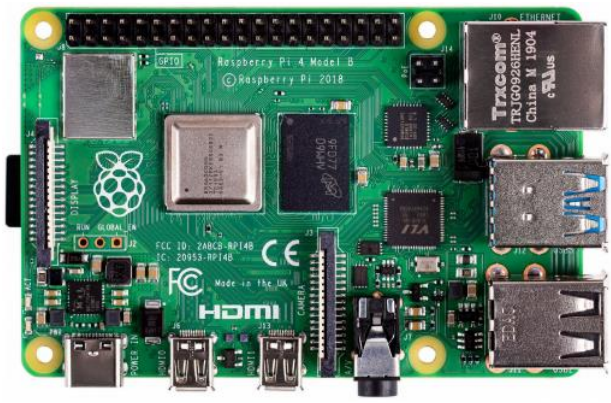


Figure 2.41: Raspberry Pi 4 Model B (Pi, 2020).

The Raspberry Pi Foundation created raspberry Pi in partnership with Broadcom in the UK as a collection of compact single-board computers. Based on Askar et al. (2015) states that Raspberry Pi is a lightweight, low-cost device with low weight and size like a credit card. Essentially, it's a system-on-a-chip (SoC) with port connections. It has a 32-bit ARM processor that provides 700 to 1000 MHz processing speed and 4 GPU core footage. Raspberry Pi Model B is fitted with a 512 Mb SDRAM and two USB ports. It is possible to link to an Ethernet network. Additionally, Raspberry Pi supports innovative technology like OpenGL ES2 and accelerated audio or video encoding through hardware. These features render the Raspberry Pi an outstanding computer vision tool. As extra information, Raspberry Pi's processor operates default to 700MHz (Askar et al., 2015). Figure 2.41 shows the specification of the Raspberry Pi 4 Model B.

Processor	Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
Memory	2GB, 4GB or 8GB LPDDR4 (depending on model)
Connectivity	2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth 5.0, BLE Gigabit Ethernet 2 × USB 3.0 ports 2 × USB 2.0 ports.
GPIO	Standard 40-pin GPIO header (fully backwards-compatible with previous boards)
Video & sound	2 × micro HDMI ports (up to 4Kp60 supported) 2-lane MIPI DSI display port 2-lane MIPI CSI camera port 4-pole stereo audio and composite video port
Multimedia	H.265 (4Kp60 decode); H.264 (1080p60 decode, 1080p30 encode); OpenGL ES, 3.0 graphics
SD card support	Micro SD card slot for loading operating system and data storage
Input power	5V DC via USB-C connector (minimum 3A1) 5V DC via GPIO header (minimum 3A1) Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment	Operating temperature 0–50°C
Production lifetime	The Raspberry Pi 4 Model B will remain in production until at least January 2026.

Figure 2.42: Specification of Raspberry Pi 4 Model B (Pi, 2020).

2.5.2 Raspberry Pi Camera Module

Figure 2.43: Raspberry Pi Camera Module v2.

Raspberry Pi Camera Module v2 is a high-quality 8-megapixel Sony IMX219 specially built Raspberry Pi image sensor with a fixed focus lens. Besides that, It can hold 3280 x 2464 pixel static images and also supports 640x480p60/90,1080p30 and 720p60 footage. It links to Pi through one of the little sockets on the board's high surface and uses the specially built CSi framework for cameras. Other than that, the Raspberry Pi camera module can be used for high-definition video and even images. It is ideal for the lapse in time and slow motion. Besides that, the human being also can build effects using libraries which are connected to the camera. It connects to the CSI port on the Raspberry Pi with a 15cm ribbon cord. Raspberry Pi can be read through the PiCamera Python library in order to get the information (Sunitha et al., 2019).

Name	Camera Module v1	Camera Module v2	HQ Camera
Net price	$25	$25	$50
Size	Around 25 × 24 × 9 mm		38 x 38 x 18.4mm (excluding lens)
Weight	3g	3g	
Still resolution	5 Megapixels	8 Megapixels	12.3 Megapixels
Video models	1080p30, 720p60 and 640 × 480p60/90	1080p30, 720p60 and 640 × 480p60/90	1080p30, 720p60 and 640 × 480p60/90
Linux integration	V4L2 driver available	V4L2 driver available	V4L2 driver available
C programming API	OpenMAX IL and others available	OpenMAX IL and others available	
Sensor	OmniVision OV5647	Sony IMX219	Sony IMX477
Sensor resolution	2592 × 1944 pixels	3280 × 2464 pixels	4056 x 3040 pixels
Sensor image area	3.76 × 2.74 mm	3.68 x 2.76 mm (4.6 mm diagonal)	6.287mm x 4.712 mm (7.9mm diagonal)
Pixel size	1.4 µm × 1.4 µm	1.12 µm x 1.12 µm	1.55 µm x 1.55 µm
Optical size	1/4"	1/4"	
Full-frame SLR lens equivalent	35 mm		
S/N ratio	36 dB		
Dynamic range	67 dB @ 8x gain		
Sensitivity	680 mV/lux-sec		
Dark current	16 mV/sec @ 60 C		
Well capacity	4.3 Ke-		
Fixed focus	1 m to infinity		N/A
Focal length	3.60 mm +/- 0.01	3.04 mm	Depends on lens
Horizontal field of view	53.50 +/- 0.13 degrees	62.2 degrees	Depends on lens
Vertical field of view	41.41 +/- 0.11 degrees	48.8 degrees	Depends on lens
Focal ratio (F-Stop)	2.9	2.0	Depends on lens

Figure 2.44: Camera Module v1 vs Camera Module v2 vs HQ camera.

2.6 Image Processing Software Of Computer Vision System

Photo analysis is very useful and important in many different fields, such as forestry, remote sensing and telecommunications. According to Sharma (2017), MATLAB, OpenCV, and LabVIEW provide several methods for extracting the raw data from a picture obtained through pixel manipulation. The various pixel manipulation techniques used are available.

Besides that, photo processing is applied in many diverse areas of life including agriculture, remote sensing, telecommunications, and medicine. A picture is composed of innumerable pixels. Image processing algorithms are employed for the processing of these

pixels. LabVIEW, OpenCV, MATLAB, and similar tools will yield many options for obtaining the information contained in the image captured (S.Sathiyamoorthy, 2014). The processor must determine whether a given image is acceptable or unacceptable on a case-by-case basis.

2.6.1 MATLAB

Figure 2.45: Icon of MATLAB

MATLAB is a widely used computer graphics computer programming. This programming environment comes with MATLAB toolboxes that provide functions for integrating MATLAB functions with external languages like C, C++, Java, .NET and Microsoft Excel. MATLAB language provides support to numerical computations on vector and matrix analysis. With the MATLAB language, programs and algorithms can be developed much faster than in traditional languages. With vector and matrix operations, the need for lengthy for-loops is eliminated. A single line of MATLAB code can often replace several lines of programming code. MATLAB also incorporates traditional programming languages such as flow control, error handling, and object-oriented (OOP). Besides the fundamental data types, MATLAB allows user-defined data types too. While executing several commands, results are achieved one at a time. This strategy helps to brainstorm many possibilities and test them with different approaches (Epiphany, 2014).

Besides that, the Graphical User Interface Development Environment in MATLAB is the graphic user interface design tool that enables non-programmers to easily and quickly design and build custom graphical user interfaces (GUI) in MATLAB. Graphical user

interfaces can also be created by code. The immersive GUI allows the instructor to inspire the learner with the idea behind the image processing techniques (Epiphany, 2014).

2.6.1.1 Image processing algorithms using MATLAB

According to Sharma (2017) claimed that algorithms for image processing are designed using programming scripts. Any code must be translated into Simulink. It is important to test that the quality of the image is correct after using the program. Therefore, analysis of images and statistical analysis is carried out to check the quality of images. We presented a classification algorithm for image processing and validated the model with simulation. Draft image processing algorithms using the Simulink toolbox, and checking the efficiency of built algorithms with statistical parameters. Besides that, Simulink is used to assist the development of computer vision algorithms. The proposed methods for image processing using MATLAB will be indicated in the following Figure 2.46.

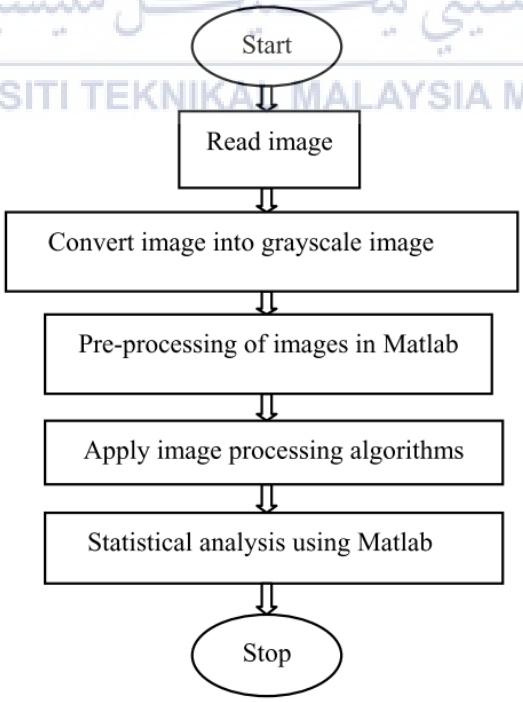


Figure 2.46: Design flow in image processing using MATLAB (Sharma, 2017).

Firstly, images are processed into grayscale images and then pre-processing is performed to eliminate noise, and then algorithms are applied to image processing. By using MATLAB, the effects of image processing algorithms can evaluate through both qualitatively and quantitatively. Statistical analysis is accomplished to quantitatively test the image's performance (Sharma, 2017).

2.6.2 LabVIEW

Figure 2.47: Icon of LabVIEW

LabVIEW named as Laboratory Virtual Instrument Engineering Workbench. LabVIEW is a system-design tool and development framework for a visual programming language. National Instruments established LabVIEW as a standard for monitoring research instrumentation. LabVIEW is a graphical programming language that allows various elements to be joined together to create the desired application. Generally, LabVIEW can be used for instrument management and data processing. LabVIEW simplifies hardware integration, allowing data sets to be obtained and visualized from virtually any I/O computer, from scratch or third parties. Combined with a graphical syntax, which increases code speed.

2.6.3 OpenCV

Figure 2.48: Icon of OpenCV

According to Yongxiang (2009), OpenCV is a well-known Open Source Computer Vision Library created by Russia's Intel development centre. It is an open-source programming library that primarily works on computer vision in real-time. OpenCV comprises over 500 optimized image and video processing algorithms, including factory product inspection, medical imaging, surveillance, user experience, camera calibration and stereo vision and robotics (Matuska et al., 2012). Besides that, it consists of numerous C feature libraries and C++ groups. Other than that, OpenCV usually used in image processing and computer vision algorithms. OpenCV is compatible with the Intel Image Processing Library (IPL) which has been built by Intel, as well as another image processing library. As extra information, IPL is used for low-level digital image processing, while OpenCV is primarily used in specialized image processing, including object detection, object tracking, movement analysis, object segmentation and recognition, and 3D reconstruction. Following are the advantages of using OpenCV as the computer vision library (Yongxiang, 2009):

1. OpenCV is an application interface of more than 300 C programming features. It does not rely on external libraries, which means it can operate independently and with other external libraries.
2. All OpenCV algorithms focus on the complex IPL data structures that are highly versatile and more than half of which are optimized for Intel processors when configured and complied, rendering them highly competitive.
3. The interfaces for some other languages or platforms, such as MATLAB, EiC and Ch will be built under the OpenCV directory.

4. The OpenCV source code is open, and users can change or introduce additional classes to the library. Other developers can use the code as long as it follows design requirements. All these characteristics have provided OpenCV great vitality and capacity for growth.

2.6.3.1 Image processing algorithms using OpenCV

According to Deepthi & Sankaraiah (2011), the researcher claimed that OpenCV is optimized for high device performance, emphasising real-time applications. Besides, OpenCV has been written with Optimized C such that multi-core processors will gain from it. The library has been improved with C language according to the Windows image processing framework. Figure 2.49 demonstrates the planned mobile framework implementation for image processing.

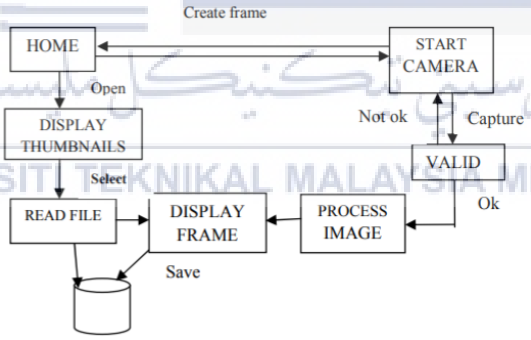


Figure 2.49: Implementation of image processing on a mobile platform (Deepthi & Sankaraiah, 2011).

Table 2.11: Comparison Table of MATLAB, LabVIEW and OpenCV(Anaz, 2020; Sharma, 2017)(Retrieved from https://opencv.org/platforms/).

SOFTWARE	MATLAB	LABVIEW	OpenCV
CPU Time Consumption	Normal	Longest	Shortest
Simulations	Execute speed fast	Execute speed low	Execute speed very fast
Functions	Limited functions in MATLAB	Functions optimized in LabVIEW	More functions in OpenCV
Graphical User Interface	More painful and limited construction	Fast and simple construction	Fast and simple construction
Platform	Supports Windows, Linux, macOS and cross-platform	Supports Windows, Linux and macOS.	Supports Windows, Linux, Android, macOS, FreeBSD, OpenBSD and cross-platform
Open Source	No	No	Yes
Cost	Costly	Normal	Free

2.7 Summary

Based on the literature review, the most issue in computer vision is image classification, object detection, semantic segmentation, and instance segmentation. According to Feng et al. (2019), image classification is a primary skill of the visual perception system usually used to identify semantic object categories in a given image. In addition, Feng et al. (2019) also claimed that image segmentation is known to be a pixel-level classification that separates an image by classifying each pixel as a particular object into meaningful regions. Other than, image segmentation consists of two categories which are semantic segmentation and instance segmentation. Semantic segmentation anticipates pixel-wise classifiers to give each pixel a particular category mark, thereby offering an even more detailed picture. Instance segmentation is proposed to distinguish distinct entities and assign a separate pixel level mask to each. Besides that, Vahab et al. (2008) state that object detection is the primary skill most computers and robot vision systems need. It mainly addresses the locating and position of particular objects in an image.

According to Shah & Kapdi (2017), deep neural network (DNN) defined as a multi-layered artificial neural network (ANN) between input and output layers. DNN are those with depths exceeding three or more than three layers. In addition, Galvez et al. (2019) state that Convolutional neural network (CNN) is a subset of deep neural networks. Besides, CNN

is most used to evaluate graphic imagery. It also has deeper layers, biases, performances and weights by a non-linear simulation.

Furthermore, convolutional neural networks neurons are organized volumetrically such as height, distance, and depth. According to Pehlivan et al. (2020), MobileNet-SSD architecture is the combination of MobileNet architecture and Single Shot Detector (SSD) architecture where MobileNet architecture is used for prediction SSD architecture is to identify the result of classification. MobileNet-SSD achieves best-precision trade-off for real-time implementations inside the fastest detectors. It means that almost 20 object category can be identified.

Based on the literature reviews of the dataset reviewed, the MS-COCO dataset is the gold standard benchmark for measuring computer vision model performance. Besides that, MS-COCO dataset is labelled with data to train supervised computer vision models that can recognize familiar objects in the dataset. MS-COCO dataset is intended to identify and segment objects in a natural context. Next, MS-COCO dataset is considerably higher in the number of instances per category than the PASCAL VOC and SUN datasets. By comparing with PASCAL VOC dataset, MS-COCO dataset performs more categories and more instances per category. Additionally, an average of MS-COCO dataset contains 3.5 categories per image and 7.7 instances per image while comparing with PASCAL VOC dataset and ImageNet dataset which both averages of datasets have less than 2 categories per image and 3 instances per image (Lin et al., 2014).

Based on the research, Raspberry Pi is a lightweight, low-cost device with low weight and size like a credit card. It has a 32-bit ARM processor that provides 700 to 1000 MHz processing speed and 4 GPU core footage. Additionally, Raspberry Pi supports innovative technology like OpenGL ES2 and accelerated audio or video encoding through hardware. These features render the Raspberry Pi an outstanding computer vision tool. As extra information, Raspberry Pi's processor operates default to 700MHz (Askar et al., 2015). The Raspberry Pi Camera Module v2 is a high-quality 8-megapixel Sony IMX219 specially built Raspberry Pi image sensor with a fixed focus lens. Besides that, It can hold 3280 x 2464

pixel static images and also supports 640x480p60/90,1080p30 and 720p60 footage. Other than that, the Raspberry Pi camera module can be used for high-definition video and even images (Sunitha et al., 2019).

Based on several literature reviews of image processing software reviewed, OpenCV software is the most suitable software for use in computer vision applications. This is can be proved OpenCV is an open-source library developed to process images and contain data structures designed for image processing. The development of image processing for human-robot interaction is critical in the entire image and video processing field, for modifying the programming environment, or in other applications. Hence, OpenCV is recommended to use in the project.

CHAPTER 3
METHODOLOGY

Chapter 3 discusses the methodology of this project. This methodology's discussions emphasised the components, method, knowledge, and procedures for detecting the object detection system's general object. The methodology was constructed by the project planning, schedule planning, designing the experiments and the methods used to complete this project.

3.1 Project Planning

Plan design describes the project's model's structure and implementation with the most detailed approach and purpose. Decision-making and action are important to accomplish the overall goals of this program. The main goal is to arrange the time, expenses and capital for the activities needed and effectively handle risks by forecast implementation. One of the most successful approaches is to draw up a Gantt map to display and schedule the project's phases to ensure all the project phases are finished on time. The flow chart is also an easy way to guarantee every step and process in sequence, schedules and error avoidance in the project planning job.

3.1.1 Project flow chart

The flow chart is a technique used from the first phase to complete the project to visualize and demonstrate process sequences. Besides that, each step is explicitly arranged and defined according to the sequences. As seen in Figure 3.1, the flow chart illustrates the procedures and processes sequences of this project. Some discussions and comparisons have been made from the literature review level in order to determine the most suitable components and approaches to be chosen and applied in this project.

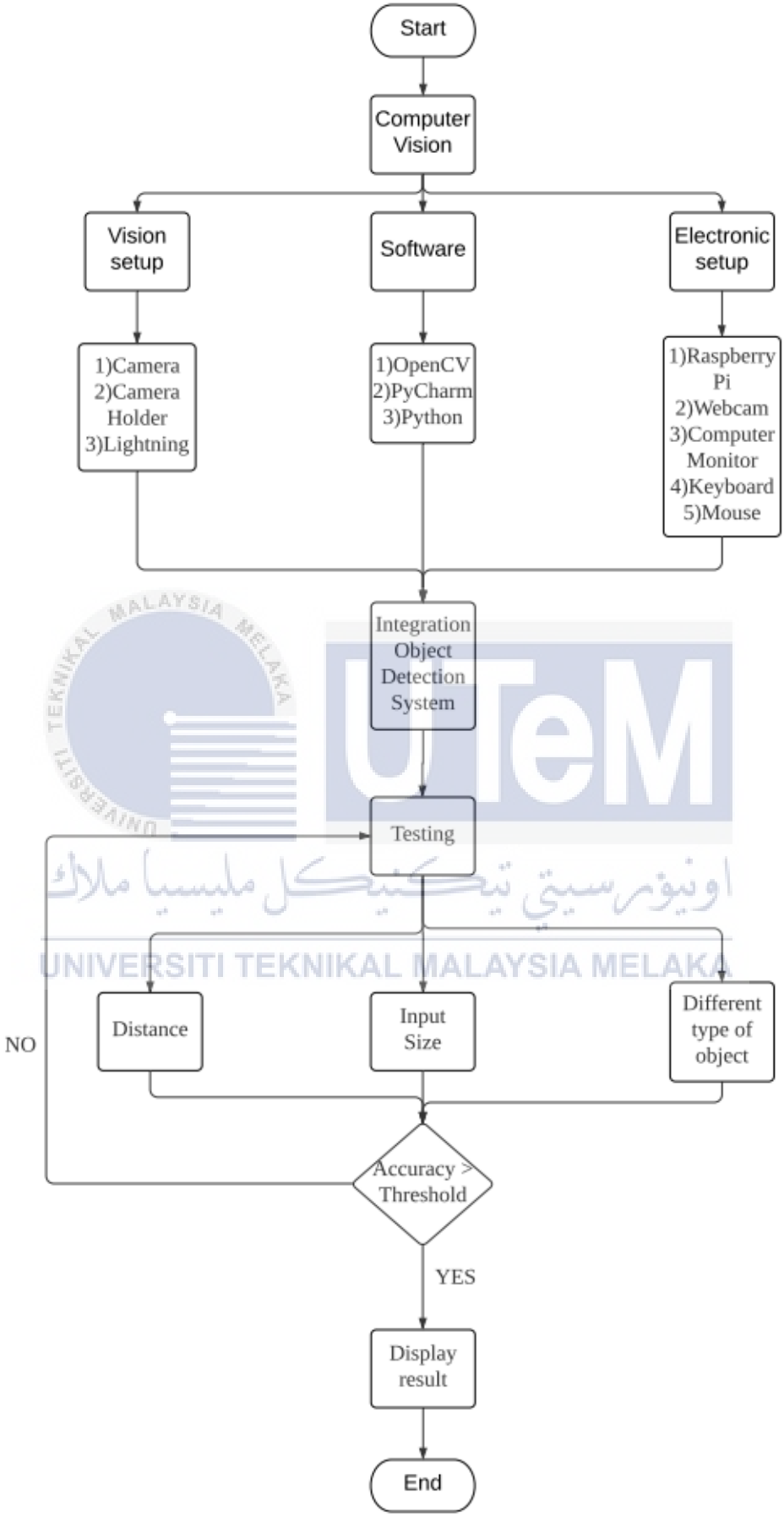


Figure 3.1: Flow Chart For the Project.

3.2 Project Design And Development

In the early stage, the structure, technique, software and suitable components have been selected. Components that have been selected are needed to capture the general object's image and collect the data from the real-time detection process. The selection of software is to ensure the image processing process can be implemented successfully. After completing the selection, the object detection system will need to be built according to the most efficient image processing algorithm for detecting and locating the general object. Purpose of this project is to replace the traditional desktop system with Raspberry Pi to create a low-cost detection system.

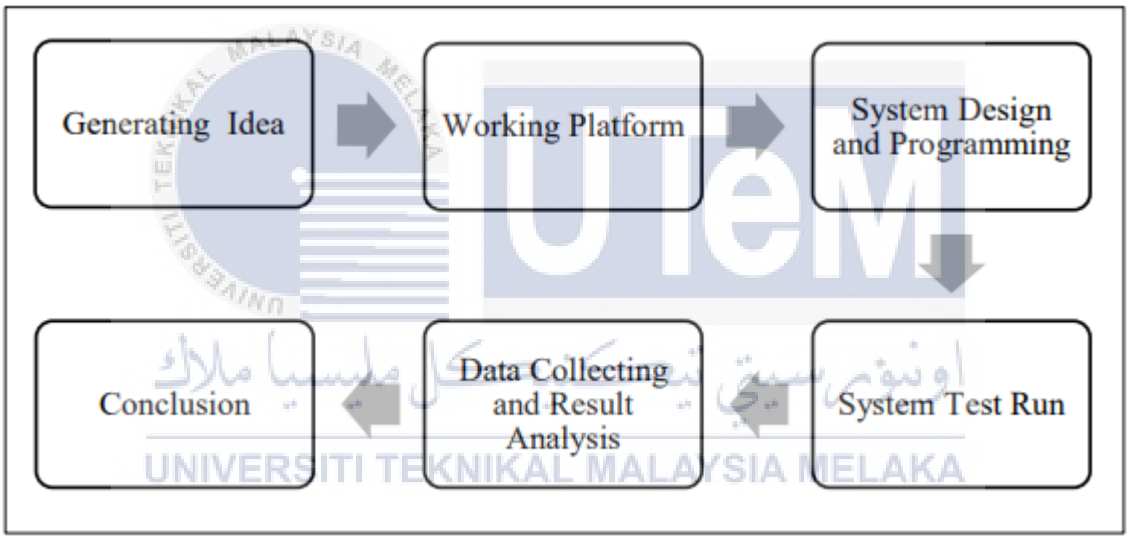
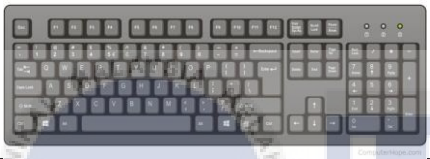


Figure 3.2: Project Design and Development Flow Chart

3.2.1 List of Components

Table 3.1: List of components

No.	Components	Function
1	Raspberry Pi 	• As a single-board computer working on the Linux operating system, it can process all of the data and then connect to the server.
2	Webcam 	• It is used to capture an object's image when the signal is given to it by computer. • The cost is cheap compare to the professional digital camera. • Consist of USB. Its interfacing becomes easy with computers.
3	Computer Screen Monitor 	• Display the output of the computer to the user. • Consist of HDMI and VGA. Its interfacing becomes easy with computers.
4	Power Adapter 	• Generate current to the computer. • Consists of USB type C.

5	HDMI Cable 	• To transmit high-definition audio and video signals. • Consists of Micro-HDMI to Standard HDMI Cable.
6	Memory Card 	• Data storage.
7	Keyboard 	• Connect to the computer. • Input characters into the computer.
8	Computer Mouse 	• Give commands to the computer by controlling the movement of the pointer.
9	Tripod Webcam 	• Used to holding and stabilize the webcam to reduce the camera shake and increase the quality of photography. • Allows adjusting the shooting angles of webcam.
10	LED Downlight 	• For illumination purpose when capturing the image of electronic devices.

3.2.2 Design of Object Detection System Hardware

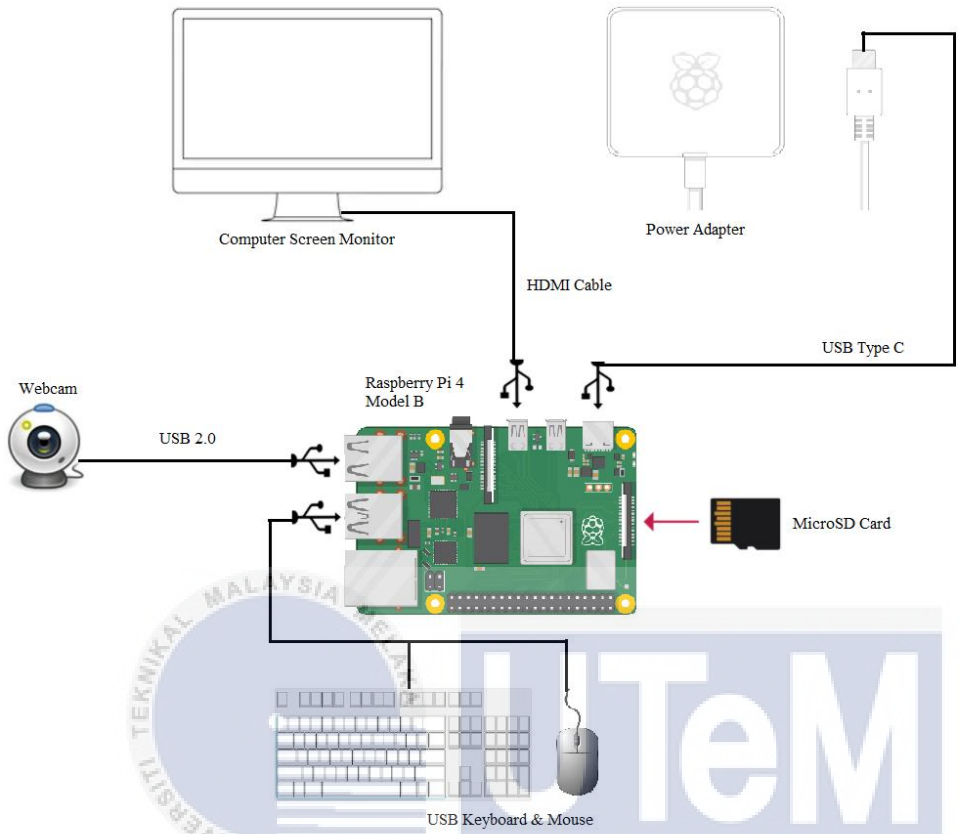


Figure 3.3: Object Detection System.

Figure 3.3 shows the setup of the whole object detection system. First, a power adapter RPi with 15W is chosen as the power supply for the Raspberry Pi. The input and output current for the power adapter is 0.5A and 3.0A, respectively. Other than that, the range of input voltage for the power adapter is divided into two categories one is for rating, which is 100VAC to 240VAC, and another one is for operating, which is 96VAC to 264VAC. For long cables and high current usage, the output voltage of this adapter is cleverly balanced to a higher standard at 5.1VDC. The voltage will remain at 5.1V at low current usage, and it is sufficient for most 5.0V applications since 5.5V is the higher limit of the voltage. Meanwhile, the voltage will drop through the cable in the case of a high current

draw, and the extra 0.1V will compensate for the loss. In addition, the captive cable with a USB Type-C plug is suitable to power the Raspberry Pi 4 Model B.

Secondly, Raspberry Pi is a single-board computer, and it acts as an important component for the object detection system. Compared to the previous generation Raspberry Pi 3 Model B+, this series of Raspberry Pi provides ground-breaking improvements in processing speed, multimedia performance, memory, and networking while maintaining backward compatibility and comparable power consumption. Besides that, the Cortex A72 SoC of Raspberry Pi 4 Model B, which has the faster 1.5GHz 64-bit quad-core CPU combined with the up to 4GB LPDDR4 RAM offers processor performance equivalent to entry-level x86 PC systems for the end consumer. In addition, this model of Raspberry Pi has two USB 3.0 ports and two USB 2.0 ports. Obviously, the transfer speed of USB 3.0 is 10 times faster than USB 2.0 and is ideal for linking fast peripherals such as SSDs and flash drives. The true gigabit ethernet purposely developed super-speedy wired networking maintaining PoE, rendering it suitable as an internet media server and network-related ventures. Next, the Raspberry Pi 4 Model can support dual monitor display and is able to support the resolutions up to 4K multimedia through a pair of micro-HDMI ports. Besides that, for this series of Raspberry Pi, it maintains backward compatibility on the 40-pin GPIO same as the previous model. Other than that, the USB type C on the Raspberry Pi 4 Model B as the power input and it is capable of obtaining more power (15W or above), which provides more power for its peripherals like USB 3.0, for instance, hard disk drive (HDD) and solid-state drive (SSD). Besides that, the microSD card that used on the Raspberry Pi 4 Model B is 16GB as data storage. It acts as an initial capacity for the operating system (OS) and the data. Not only that, via several kinds of USB attached peripherals, storage may be expanded. the storage may be expanded by several forms of peripherals attached to USB.

Besides that, the Logitech C310 High Definition (HD) webcam is selected as the vision set up for the object detection system. This webcam is easy to use compared with the Raspberry Pi camera module. The key features of this model webcam include the size is small and able to adjust the angle of the webcam, widescreen HD 720P video calls, the maximum resolution is 720p/30 fps, USB type 2.0 plug with 1.5m cable, fixed focus lens, built-in-microphone and supports operating system such as macOS, Google Chrome OS and

Microsoft Window 7 and above. The functionality of the webcam installed on the Raspberry Pi 4 Model B is to detect and capture the image or video in real-time tasks. Using the webcam tripod to fix and locate the webcam's position to ensure the webcam is under stabilization condition. Apart from that, light brightness is considered as the compulsory factor that will affect the performance of the image. Using LED light as the vision set up to improve the webcam vision to ensure that can easily capture the image with high quality.

Furthermore, the computer screen monitor, computer keyboard and computer mouse are the compulsory devices for the object detection system. The computer screen monitor is connecting with the Raspberry Pi 4 Model B by using an HDMI cable. It is used to display the detecting process while a webcam is tracing the object. It is also used to show the information of the single-board computer in pictorial form. Other than that, the computer keyboard is an input device, whereas the computer mouse is a hand-held device. The function of a computer keyboard is to insert text and coding into the single-board computer, while the function of a computer mouse is to control the position of the cursor on the computer screen in order to give a command to start the program.

3.3 Implementation Of Software And Programming Code

For developing the object detection system, software to process the image and produce the output for further classification is essential for object detection. The software used in this project is OpenCV, while Python Language is the coding language for implementing pseudocode. In Python Language Programming and the OpenCV Library, PyCharm is an integrated development environment.

3.3.1 OpenCV software

OpenCV is the object detection software that has been selected to apply in this project. OpenCV is an acronym for the Open Source Computer Vision Center. It is intended to include the required tools for solving computer vision issues. OpenCV is the platform that is ideally adapted for the project relative to other software because it is free to use. It is a series of various interconnected features, primarily intended for real-time use. The features are continuously invented and update-to-date since the program is open-source. The important thing is that the software is free of charge; it allows industry and non-commercial users to use it.

Figure 3.6 shows the four major elements include in OpenCV. The first is the CV portion involves simple image processing and machine viewing algorithms at higher stages; ML is the machine learning library consisting of several statistical classification and clustering tools. The HighGUI provides I / O routines and features to store and load video and images, and CXCore comprises the basic data and content structures. (Neelima & Saravanan, 2014).

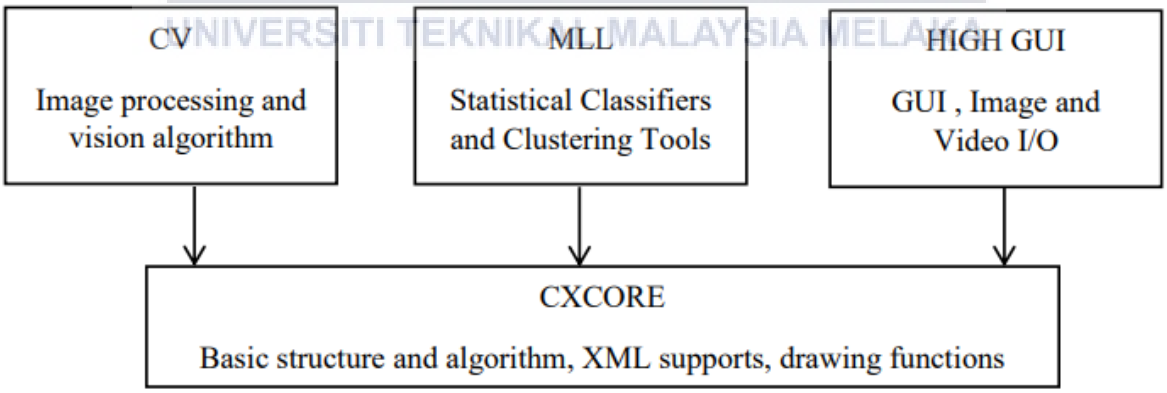


Figure 3.4: Four major elements in OpenCV (Neelima & Saravanan, 2014).

The OpenCV feature allows the user to communicate with the operating system, file system, and devices, such as cameras, in a library called HighGUI ("high-level graphical user interface"). HighGUI offers users the probability of opening windows, viewing images,

reading, writing files related to graphical images and videos, and managing basic mouse, cursor, and keyboard events. The OpenCV HighGUI library can be divided into three components: the hardware portion, the device component, and the GUI component. The hardware component mostly deals with camera operation. Interaction with a camera is a repetitive and frustrating task in most operating systems. HighGUI enables simple queries to be made and the current camera picture restored.

OpenCV image processing algorithms are constructed by writing programming script files. The quality of images should be tested after the algorithms have been used. This results in visual and statistical analyses to test the accuracy of the images processed. The OpenCV image processing algorithms for detecting the output of constructed algorithms with statistics analyzed have been developed. For the construction of image processing algorithms, OpenCV's computer vision library is used. The figure below shows the proposed processing methods for image algorithms that use OpenCV.

3.3.2 PyCharm software

PyCharm is used to analyse the data of Python coding. PyCharm was selected in this project due to it is a beginner-friendly software. JetBrains create PyCharm in the Czech Republic. PyCharm is an integrated virtual development environment (IDE), which is especially suitable for the Python language of computer programming. In addition, an integrated development environment provides tools for code review, debugger and debugging tools as well as solutions for version control and supports both Django and Anaconda's Web creation. Besides that, PyCharm is able to work with multiplatform such as macOS, Windows and Linux. IDE helps developers with the aid of numerous available APIs in developing Python plugins. It enables the user to deal directly with a variety of databases without merging them into other resources. While built explicitly for Python, it can also be produced with this IDE to include HTML, CSS, and JavaScript data.

3.3.3 Python

Python is a programming language that commonly selected by fresh programmers due to it is a beginner-friendly software. Python was created by Guido van Rossum back in the late 1980s and released to the public in 1991. Besides that, Python can think like human and implementing the code. It also limits the need to write syntactically appropriate programs to add extra keywords. In addition, Python is easier to understand and use than other languages such as C++ or Java languages. This is because it can be done with uncomplicated commands and less text than other languages. Building a Python interpreter is common practice, and it is useful for quick tests and experiments. Python code also has a comprehensive standard library. To select a few random instances, Python ship with a range of XML parsers, csv & zip file readers & authors, libraries that use almost an internet protocol and data form, etc. Python coding also has great support for creating web apps. The language interpreted is Python, so it is easily verifying how the operators or functions operate by using the command-line interpreter. Python interpreter has an integrated support module that can enhance the comprehension of various language aspects.

3.3.3.1 Process of object detection

Firstly, run the network with architecture model. Next, the webcam is activated and producing a frame. As extra information, images must be preprocessed and transformed into Binary Large Object (BLOB) before it sends into the network. This preprocessed involves redimensioning the size of images, scaling the images and normalization the images. Not only that, but it also to ensure that the images accommodate to the network input by adjusting the size and scale of the images. Besides, normalisation is accomplished by subtracting the RGB value of images with the average RGB value of the training set's images. This is achieved in order to counteract variations in light in the input images.

When the binary large object is primed, a forward pass proceeds where the blob is propagated across the network and the predictions are computed. The processing time for

the project will be recorded as the inference time. After that, the detector will undergo two filtering stages before revealing the final predictions to eliminate the bad detections. The first stage is to remove the score of predictions that is lower than the threshold value of confidence. The ways to overcome this problem are by setting the value to 0.5 as a confidence threshold value. If the prediction value is lower than 50%, it will be considered false detection; in other words, the detection will be not counted. Next, non-maximum suppression (NMS) is the second stage in filtering used to ensure that an object is only detected once and eliminates smaller overlapping bounding boxes. After the filtering is completed, the bounding boxes are drawn on the detected object together with the tag of predicted categories and the confidence values. Lastly, the result will display on the monitor.

3.4 Experimental Setup

The project aims to determine the appropriateness of operating real-time object detection on a Raspberry Pi. One experiment will be carried out on the different type of objects such as computer mouse, cell phone, remote, laptop and keyboard. In this section, this experiment is to measure the percentage of detection accuracy for each object by using the different range of distance between camera and object, which is 0.6m, 0.7m, 0.8m, 0.9m and 1m and the different size of the input image which is 280 x 280 pixels, 320 x 320 pixels and 360 x 360 pixels. Two types of algorithms will be used in these experiments: the SSD algorithm and the MobileNet-SSD algorithm. This experiment is performed to determine the performance of both algorithms while detecting the objects. It captures 80 frames of video with an object at four different distances from the camera and calculating the detector's average confidence. The purpose of this experiment is to measure the percentage of detection accuracy between the SSD algorithm and MobileNet-SSD algorithms in order to prove which one is the most precise and suitable to run in a real-time objection on Raspberry Pi. Table 3.2 shows each model's detection accuracy by using different parameters consisting of different distance and input size.

Table 3.2: Detection accuracy for each model by using different parameters.

Distance (meters)	Input Size (pixels)	SSD (%)	MobileNet-SSD (%)
0.6	280 x 280	XX	XX
0.6	320 x 320	XX	XX
0.6	360 x 360	XX	XX
0.7	280 x 280	XX	XX
0.7	320 x 320	XX	XX
0.7	360 x 360	XX	XX
0.8	280 x 280	XX	XX
0.8	320 x 320	XX	XX
0.8	360 x 360	XX	XX
0.9	280 x 280	XX	XX
0.9	320 x 320	XX	XX
0.9	360 x 360	XX	XX
1	280 x 280	XX	XX
1	320 x 320	XX	XX
1	360 x 360	XX	XX

Average Precision (AP) is a common indicator used for identifying the accuracy of both algorithms in object detection. The mean Average Precision (mAP) is calculating by using the equation below:

$$IoU = \frac{\text{Area of intersection}}{\text{Area of Union}} \quad \text{(Equation 3.1)}$$

where IoU = Intersection over Union

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

$$Recall = \frac{\text{TP}}{\text{TP+FN}} \quad \text{(Equation 3.2)}$$

where TP = True Positive

FP = False Positive

FN = False Negative

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad \text{(Equation 3.3)}$$

where N is the number of queries in the set

AP_i is the average precision (AP) for a given query, i

3.4.1 Programming Setup

Step 1: Installation of Python and PyCharm software.

Firstly, install the Python software version 3.7.6 by accessing the URL link: https://www.python.org/downloads/. This version works well in OpenCV, and the installer is suitable for the Windows OS. While the installer is ready to run, add Python 3.7 to the path and start installing. Once the installation is done, close the dialogue box. Next, install the PyCharm software by accessing the URL link: https://www.jetbrains.com/pycharm/download/#section=windows. The version is "Community Version" due to it is open source and free for download. Besides that, PyCharm is an integrated development environment (IDE) used to edit the code. It is a code editor that will allow the user to write the code, and it also has a lot of functionalities that help the user in the coding process. While the installer is ready to run, make sure to associate .py files to PyCharm software and add the launcher directory to the path. Once the installation is done, restart the computer. After that, open the PyCharm software and start to create a new project. Then, the PyCharm environment will appear at the desktop computer and ensure to add the opencv-python package by installing at the project interpret. After the package is finish installed, create a new Python file and start to write coding.

Step 2: Importing the MS-COCO dataset, the weight file, MobileNet-SSD architecture, and SSD architecture in the module.

Figure 3.5 shows the essential files for object detection. All of these files are compulsory for creating an object detector module. Based on the figure below, the coco.names file represents the dataset for object detection to detect and classify the common objects such as a person, accessories, animal, vehicle, outdoor object, sports, kitchenware, food, furniture, appliance, electronic and indoor objects. Next, the frozen_inference_graph.pb represents the weight file applied in object detection. Other than that, ssd_inception_v2_coco.config file represents the configuration of SSD else ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt file represents the configuration of MobileNet-SSD. In addition, all of the code will write in the ObjectDetectionModule.py file.

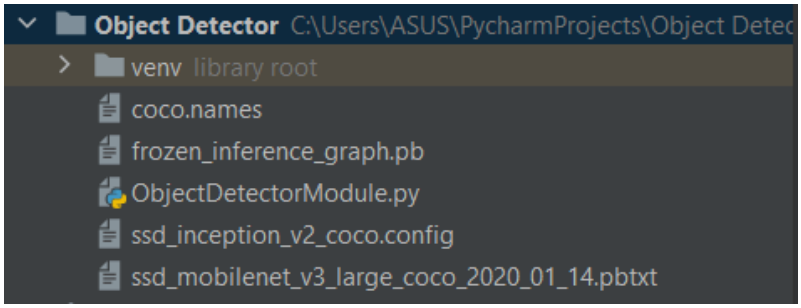


Figure 3.5: Importing the dataset, configuration file, and weight file.

Step 3: Creating object detector module coding.

Firstly, import the OpenCV package by writing a code import cv2. Based on this coding, the cv2 is standing for computer vision. Next, to identify the object's class names, it needs to import the MS-COCO dataset in the class file to support reading the detected object. Figure 3.6 shows the coding for import the MS-COCO dataset in the module.

```
classNames = []
classFile = 'coco.names'
with open(classFile,'rt') as f:
    classNames = f.read().rstrip('\n').split('\n')
```

Figure 3.6: Coding for import MS-COCO dataset.

Secondly, import the configuration file and weight file into the module. The first thing is two configuration files need to read in the config path. The two configuration files are ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt represents MobileNet-SSD configuration and ssd_inception_v2_coco.config represents SSD configuration. These two configurations are totally different, and both of them are used for comparing the performance and the detection accuracy while detecting the object. Next, frozen_inference_graph.pb is the weight file that needs to be applied in the module. Figure 3.7 illustrates the configuration file and weight file that importing into the module.

```
configPath = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
weightsPath = 'frozen_inference_graph.pb'
```

Figure 3.7: Coding for the import configuration file and weight file.

Figure 3.8 shows the parameter setting of the detected objects. These functions will allow the user from ID to get the names of the object detected. The first is to write in the weight path and configuration path into the cv2.dnn_DetectionModel. Next, setting the input size which is (280x280) pixels, (320x320) pixels, and (360x360). Besides that, input scale, input mean, and input swap are set as default settings.

```
net = cv2.dnn_DetectionModel(weightsPath,configPath)
net.setInputSize(320,320)
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)
```

Figure 3.8: Coding for parameter setting of the detected objects.

The next step is to send the image to the model, and it will get the confidence value. The first thing is to define the class id, confidence value, and bounding box. Next, write in the image that wants to detect and define the confidence threshold. The confidence threshold is set as equal to or more than 0.5 that will declare as acceptable otherwise, the confidence threshold is less than 0.5 will declare as ignore. Besides that, the bounding box information that needs is to create a rectangle on the object detected and write the name based on the class id. Furthermore, apply the non-maximum suppression method in the model to suppress or remove the non-maximum detect section that has the lower threshold and keep the maximum detect section with a high threshold. Not only that, the non-maximum suppression method is mainly used for removing the duplicates bounding box while detecting an object. While the detecting system is run, it will display the information based on the class id, the confidence value and the bounding box on the detected object.

```
def getObjects(img,thres,nms,draw=True,objects=[]):
    classIds, confs, bbox = net.detect(img,confThreshold=thres,nmsThreshold=nms)
```

Figure 3.9: Apply non-maximum suppression in the function.

Next, create the main script to be able to access Raspberry Pi. Activate the external webcam by writing the code cap is equal to cv2.VideoCapture(1). The class id one of video capture means that it will use the external webcam. Certain parameters need to be defined to know the size of the image, which are cap.set(3, 640) and cap.set(4, 480). Based on cap.set(3,640), the prop id is three means width. Otherwise, cap.set(4, 480) the prop id is four means height. Starting the while loop, it will first read the image and send the image to detect method in the net. Then, it will give the confidence values, the bounding box, and the class id. Lastly, the information displays a rectangle around the object and puts the text in terms of name and confidence value.

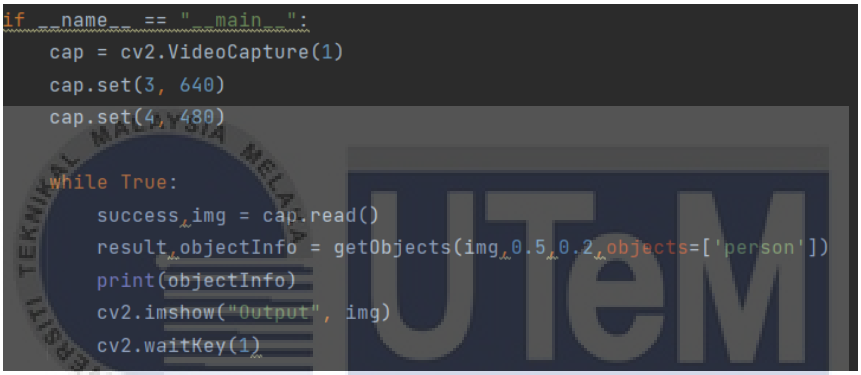


Figure 3.10: Coding for webcam setup and output setup.

3.4.1.1 Computer Hardware Setup

Figure 3.11: Computer hardware setup.

Figure 3.11 shows the computer hardware setup for object detection. Firstly, insert the webcam into the USB port for the computer and observe the computer's response to identify the webcam can either directly use or not. In case the webcam is cannot directly use, the user needs to install the software called ManyCam to activate the external webcam. This software is user-friendly, and it can be used to support a maximum of six cameras. After the webcam setup is done, open the coding for the object detector module using PyCharm software and run it simultaneously. Lastly, the object detection process will run in real-time, and the detection result will be obtained and show on the computer desktop.

3.4.1.2 Raspberry Pi Hardware Setup

Figure 3.12: Raspberry Pi hardware setup.

Figure 3.12 illustrates the Raspberry Pi hardware setup for object detection. Firstly, preparing a set of computer monitor, computer mouse and keyboard. Next, connect the computer monitor to the Raspberry Pi by inserting it into the micro HDMI port of the Raspberry Pi. The function of the computer monitor is to display the object detection process and the detecting result. The next step is to connect the computer mouse and keyboard to the Raspberry Pi. The function of the computer mouse is to control the position of the cursor on the computer screen to give a command to start run the object detection process. In contrast, the function of a computer keyboard is to insert text and coding into the code editor software in Raspberry Pi. After the hardware has been completely set up, switch on the power adapter, and the Raspberry Pi will automatically start. The Raspbian desktop will display on the computer monitor. At first, the user needs to check the version of OpenCV using a few codes in the terminal to make sure the version is 4.3 and onwards. The problem is the method using a detection module that is not available in the 4.1 and 4.2 versions of OpenCV. After finished

checking the version of OpenCV, then it can proceed to the coding part. The next step is creating a new folder name as an object detector module on the desktop. The folder contents should include the configuration files, weight file, MS-COCO dataset file, and the object detector module file. Next, open the mu editor software in the programming option and using mu editor software to load the object detector module. Then, the object detection process will run in real-time in the Raspberry Pi. Lastly, the detection result will be obtained and shown on the computer desktop.

3.5 Bill Of Material (B.O.M)

Bill of material is a hierarchical listing of components used to assemble a system that lists the system's cost, inventory management, and tracking the needs of components. The bill of material can be used as a shopping list to purchase the final product components needed to create this project. BOM helps the get ready for procurement arrangements and decreases the possibility of errors. By referring to this project's objective, the component used in this project is being compared with various types of the same categories' products to get the lowest price guarantee quality component. The Raspberry Pi 4 Model B and RPi 15W power adapter is purchase from the Cytron marketplace due to the reasonable price. Besides that, many webcam types are available in the market, but the decision is selected the Logitech webcam because it is easy to use and reasonable. For the LED Light, there are different types of LED available in the market. The decision to use this ZEN LED round shape downlight as the component of this project is that the LED is the cheapest and most appropriate for image acquisition step among the other types of LED available in the visited shop. Table 3.3 shows the bill of material for the project.

Table 3.3: Bill of Material

No.	Components	Qty	Unit Price (RM)	Total Price (RM)
1	Raspberry Pi 4 Model B	1	239.00	239.00
2	RPi 15W (5V/3A) PSU USB C Power Adapater	1	35.00	35.00
3	PS3 EYE Camera	1	50.00	50.00
4	HDMI Cable	1	4.60	4.60
5	Micro-SD card 16GB	1	36.00	36.00
6	ZEN LED Light	2	20.00	40.00
			Total	404.60

CHAPTER 4
RESULT AND DISCUSSION

This chapter discusses the results that have been obtained through the object detection process. These processes are repeated on the five different types of objects: computer mouse, cell phone, remote, laptop, and keyboard to determine detection accuracy. In addition, experimental tests and analyses were conducted in order to validate the results of the study. Finally, the data obtained from the experimental testing are analyzed and discussed using several methods such as tabulation and graph method.

4.1 Result Of Object Detection Process

In this section, there are provided five different tables by obtaining the result. For additional information, five materials have been selected to undergo the experiment of object detection, such as computer mouse, cell phone, remote, laptop, and keyboard. The main purpose of this experiment is to compare and analyze the performance of the SSD algorithm and MobileNet-SSD algorithm by using different parameters such as the distance between camera and object and using different input sizes. Additionally, the final result for each object is illustrated in the table below.

4.1.1 Interpretation on the Result of Computer Mouse and Cell Phone

In Table 4.1 and Table 4.2, the result for the computer mouse and cell phone are shown. The data obtained in the form of distance, input size, SSD algorithm, and Mobile-SSD algorithm. According to the table, five different distances have been listed for the experiment to test in different input sizes such as 280x280 pixels, 320x320 pixels, and 360x360 pixels. Besides that, both tables use the same parameters to identify the detection rate for detecting the objects using different configurations such as the SSD algorithm and Mobile-SSD algorithm. Based on the result given below, five readings are recorded for both configurations during the object detection test and calculate the average value to finalize the outcome.

From Table 4.1, it can observe that while the set up for the distance between camera and object is 0.6 meters, both configurations are performed well in 360x360 pixels of input size. In this scenario, the SSD algorithm can achieve a 72% average detection rate, whereas the MobileNet SSD algorithm can achieve an 85% average detection rate. Therefore, the difference in the average detection rate for both configurations is around 13%. In addition, by comparing the detection rate with different distances and input size set up, both configurations are performed the highest detection rate in the experiment with the distance between camera and object is 0.6 meters and input size is 360x360 pixels. On the contrary, the lowest detection rate for the SSD algorithm is in the distance of 0.9 meters, and input size as 280x280 pixels, and the average detection rate is 60%. On the other hand, while the lowest detection rate for the MobileNet-SSD algorithm is in the distance of 1 meter, and the input size is 280x280 pixels, and the average detection rate is 70%. Figure 4.1 shows the sample result of object detection for the computer mouse.

Next, it can be seen in Table 4.2 that both configurations are accomplished well for detection rate in the distance between camera and object as 0.6 meters and input size as 360x360 pixels. The SSD algorithm can achieve a 72% average detection rate based on the result given, whereas the MobileNet SSD algorithm can achieve an 83% average detection rate. Hence, the difference in the average detection rate for both configurations is almost 11%. Furthermore, the highest detection rate for both configurations based on the result

obtained is 72% and 83%, representing the SSD algorithm and MobileNet-SSD algorithm, respectively. Otherwise, from the result given, the lowest detection rate for both configurations is in the distance of 1 meter with an input size of 280x280 pixels. The average detection rate for the SSD algorithm is 61%, whereas the average detection rate for the MobileNet-SSD algorithm is 71%. Figure 4.2 shows the sample result of object detection for the cell phone.

Based on the final result of the computer mouse and cell phone, it can be concluded that the MobileNet-SSD algorithm is more accurate and precise than the SSD algorithm. Both configurations are undergoing the same experiment to determine the reading of detection rate by using different distance and input size with the help to calculate the average detection rate that can be proved in the object detection process. Obviously, the detection accuracy rate for MobileNet-SSD algorithms is slightly more than the SSD algorithm are shown in Table 4.1 and Table 4.2. Meanwhile, the MobileNet-SSD algorithm is more suitable for object detection system to run in real-time using Raspberry Pi.

Table 4.1: Result of Computer Mouse.

Distance(m)	Input Size (pixels)	SSD (%)						MobileNet-SSD (%)					
		x_1	x_2	x_3	x_4	x_5	$\bar{x}$	x_1	x_2	x_3	x_4	x_5	$\bar{x}$
0.6	280 x 280	70	68	70	68	69	69	79	78	79	79	80	79
0.6	320 x 320	67	67	65	65	66	66	76	74	76	76	78	76
0.6	360 x 360	72	73	73	72	70	72	84	86	85	84	86	85
0.7	280 x 280	61	61	62	60	61	61	73	72	72	74	74	73
0.7	320 x 320	63	64	63	65	65	64	77	75	75	76	77	76
0.7	360 x 360	68	66	67	68	66	67	79	78	79	77	77	78
0.8	280 x 280	62	61	62	62	63	62	71	72	71	70	71	71
0.8	320 x 320	66	64	65	66	64	65	75	73	73	75	74	74
0.8	360 x 360	63	62	64	64	62	63	76	76	75	77	76	76
0.9	280 x 280	59	61	60	59	61	60	72	71	71	70	71	71
0.9	320 x 320	63	61	62	61	63	62	73	73	74	73	72	73
0.9	360 x 360	65	65	64	65	66	65	76	76	77	76	75	76
1	280 x 280	63	64	63	63	62	63	70	70	71	69	70	70
1	320 x 320	66	65	66	64	64	65	73	74	74	73	71	73
1	360 x 360	68	66	67	66	68	67	75	75	74	75	76	75

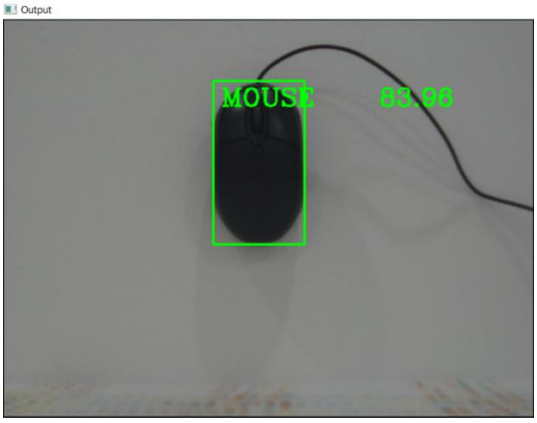


Figure 4.1: Result of object detection for a computer mouse.

Table 4.2: Result of Cell Phone.

Distance(m)	Input Size (pixels)	SSD (%)						MobileNet-SSD (%)					
		x_1	x_2	x_3	x_4	x_5	$\bar{x}$	x_1	x_2	x_3	x_4	x_5	$\bar{x}$
0.6	280 x 280	68	66	66	68	67	67	80	80	81	79	80	80
0.6	320 x 320	62	63	61	62	62	62	76	78	78	76	77	77
0.6	360 x 360	73	73	71	71	72	72	82	82	84	83	84	83
0.7	280 x 280	65	63	63	64	65	64	74	74	74	73	75	74
0.7	320 x 320	65	66	64	65	65	65	75	74	75	75	76	75
0.7	360 x 360	64	62	62	63	64	63	76	78	78	76	77	77
0.8	280 x 280	65	65	64	66	65	65	75	75	74	75	76	75
0.8	320 x 320	68	69	68	67	68	68	80	78	80	79	78	79
0.8	360 x 360	67	67	65	65	66	66	75	77	77	75	76	76
0.9	280 x 280	63	63	62	63	64	63	74	73	73	72	73	73
0.9	320 x 320	65	65	65	64	66	65	75	74	76	74	76	75
0.9	360 x 360	64	64	63	64	65	64	73	75	75	74	73	74
1	280 x 280	62	60	60	61	62	61	70	72	72	70	71	71
1	320 x 320	65	65	64	66	65	65	72	73	73	73	74	73
1	360 x 360	64	63	62	62	64	63	72	72	73	71	72	72

Figure 4.2: Result of object detection for cell phone.

4.1.2 Interpretation on the Result of Remote and Laptop

First and foremost, the remote and laptop results are shown in Table 4.3 and Table 4.4, respectively. Besides that, the data obtained in the form of distance, input size, SSD algorithm, and Mobile-SSD algorithm. A few parameters are used for the experiment of object detection, such as setting up the distance between the camera and the object with 0.6 meters to 1 meter and setting up the input sizes into 280x280 pixels, 320x320 pixels, and 360x360 pixels. From Table 4.3, it can proved that both configurations are done well using 0.6 meters as the distance and 320x320 pixels as input size. This is because while the distance is set up to 0.6 meters and the input size used is 320x320 pixels, the SSD algorithm can achieve almost 72% of detection rate, whereas the MobileNet-SSD algorithm can achieve up to 83% of detection rate. Therefore, the detection rate previously mentioned for both configurations is the highest value compared with other parameters. Besides that, the difference in the average detection rate between these two configurations is nearly 11%. When the input size is 280x280 pixels, and the range distance is between 0.9 meters to 1 meter, the result of both models is significantly lower. Furthermore, it can be proved that the average detection rate for the SSD algorithm is only 61%, and the average detection rate for the MobileNet-SSD algorithm is 70% which both results are the lowest detection rate in the experiment framework. Figure 4.3 shows the sample result of object detection for the remote.

Based on Table 4.4, it can be seen that the highest detection rate for the SSD algorithm achieved 69% of the average detection rate with 0.7 meters as the distance and 320x320 pixels as the input size. In contrast, the highest detection rate for the MobileNet-SSD algorithm can be achieved until 87% of the average detection rate with 0.6 meters as the distance and 360x360 pixels as the input size. Besides that, it also can be proved that the MobileNet-SSD algorithm performs well with the distance from 0.6 meters to 0.8 meters and the input size is 320x320 pixels and 360x360 pixels. In this condition, the average detection rate for the MobileNet-SSD algorithm can reach from 80% to 87%, which means that the MobileNet-SSD can achieve very high accuracy and better performance in the object detection system compare to others. However, when the distance is 1 meter, and the input size is 280x280 pixels, both algorithms perform slightly worst. By proving this scenario, the average detection rate for the SSD algorithm can only achieve 61%, whereas the average detection rate for the MobileNet-SSD algorithm can only achieve only 71%, which

represents the lowest detection rate from the result. Figure 4.4 illustrates the sample result of object detection for the laptop. At last, it can be concluded that the MobileNet-SSD algorithm is more accurate than the SSD algorithm based on the result obtained.

Table 4.3: Result of Remote.

Distance(m)	Input Size (pixels)	SSD (%)						MobileNet-SSD (%)					
		x_1	x_2	x_3	x_4	x_5	$\bar{x}$	x_1	x_2	x_3	x_4	x_5	$\bar{x}$
0.6	280 x 280	68	68	69	68	67	68	77	78	77	79	79	78
0.6	320 x 320	73	72	71	72	72	72	83	84	83	82	83	83
0.6	360 x 360	66	65	66	67	66	66	77	77	77	78	76	77
0.7	280 x 280	63	64	62	63	63	63	75	74	76	75	75	75
0.7	320 x 320	68	70	70	68	69	69	80	80	80	81	79	80
0.7	360 x 360	67	68	67	67	66	67	79	78	79	77	77	78
0.8	280 x 280	63	61	61	62	63	62	72	73	72	74	74	73
0.8	320 x 320	67	66	68	66	68	67	77	78	78	79	78	78
0.8	360 x 360	66	65	64	64	66	65	74	75	75	75	76	75
0.9	280 x 280	61	61	60	62	61	61	71	73	73	72	71	72
0.9	320 x 320	64	66	66	64	65	65	75	76	77	76	76	76
0.9	360 x 360	63	63	62	63	64	63	73	72	73	74	73	73
1	280 x 280	63	63	62	64	63	63	70	69	70	70	71	70
1	320 x 320	68	69	68	68	67	68	75	75	74	76	75	75
1	360 x 360	63	63	65	65	64	64	71	72	71	71	70	71

Figure 4.3: Result of object detection for a remote.

Table 4.4: Result of Laptop.

Distance(m)	Input Size (pixels)	SSD (%)						MobileNet-SSD (%)					
		x_1	x_2	x_3	x_4	x_5	$\bar{x}$	x_1	x_2	x_3	x_4	x_5	$\bar{x}$
0.6	280 x 280	63	63	62	63	64	63	79	80	79	79	78	79
0.6	320 x 320	65	67	67	65	66	66	86	86	86	85	87	86
0.6	360 x 360	65	65	64	66	65	65	86	88	86	87	88	87
0.7	280 x 280	62	62	61	62	63	62	77	77	78	76	77	77
0.7	320 x 320	63	65	63	65	64	64	83	85	83	84	85	84
0.7	360 x 360	67	67	68	66	67	67	85	85	84	86	85	85
0.8	280 x 280	64	63	63	65	65	64	73	75	73	74	75	74
0.8	320 x 320	65	65	65	66	64	65	81	79	80	81	79	80
0.8	360 x 360	68	70	70	68	69	69	84	84	85	83	84	84
0.9	280 x 280	66	64	64	65	66	65	72	74	73	72	74	73
0.9	320 x 320	66	66	65	67	66	66	77	75	75	77	76	76
0.9	360 x 360	67	69	69	68	67	68	80	78	78	80	79	79
1	280 x 280	61	61	61	60	62	61	70	72	71	70	72	71
1	320 x 320	63	63	62	63	64	63	73	73	74	72	73	73
1	360 x 360	66	66	67	65	66	66	75	73	73	75	74	74

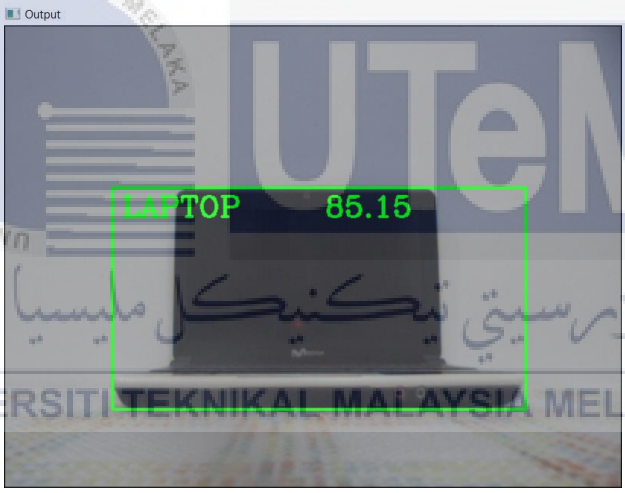


Figure 4.4: Result of object detection for laptop.

4.1.3 Interpretation on the Result of Keyboard

First of all, the result of the keyboard is shown in Table 4.5. Apart from that, the data acquired in the form of distance, input size, SSD algorithm, and Mobile-SSD algorithm were also included. Some criterion is set up for the experiment, such as using different distances from 0.6 meters to 1 meter and input size consists of 280x280 pixels,320x320 pixels and 360x360 pixels. Both configurations are run the object detection testing five times, and the

average detection rate is calculated. Based on the result given, the SSD algorithm performs well in the distance of 0.7 meters and 1 meter with the input size of 280x280 pixels and 360x360 pixels. Therefore, it can be proved that these two variables5 get the same detection rate value of 68%. In contrast, the MobileNet-SSD algorithm performs well in the distance of 0.6 meters and 0.7 meters, with an input size of 280x280 pixels. The average detection rate obtained is nearly the same for the distance of 0.6 meters; the average detection rate is 80%, while for the distance of 0.7 meters, the average detection rate is 81%. Therefore, the MobileNet-SSD algorithm can achieve within 80% and above detection rate in these two variables. Besides that, when the distance is 0.8 meters and the input size is 360x360 pixels, the average detection rate for the SSD algorithm is 61%. While the distance between camera and object is setting up to 1 meter and the input size is 360x360 pixels, the average detection rate for the MobileNet-SSD algorithm is 70%. Therefore, both models perform slightly worst in the variables mentioned previously. Figure 4.5 shows the sample result of object detection for the keyboard. In conclusion, it can be concluded that the MobileNet-SSD algorithm is more accurate than the SSD algorithm based on the result obtained. In addition, the MobileNet-SSD algorithm is more suitable for object detection systems to run in real-time with Raspberry Pi due to its high detection rate.

Table 4.5: Result of Keyboard.

Distance(m)	Input Size (pixels)	SSD (%)						MobileNet-SSD (%)					
		x_1	x_2	x_3	x_4	x_5	$\bar{x}$	x_1	x_2	x_3	x_4	x_5	$\bar{x}$
0.6	280 x 280	66	68	68	67	66	67	80	80	79	81	80	80
0.6	320 x 320	66	64	65	64	66	65	77	77	76	77	78	77
0.6	360 x 360	63	62	62	61	62	62	70	71	71	71	72	71
0.7	280 x 280	68	68	69	68	67	68	81	80	82	82	80	81
0.7	320 x 320	66	66	65	67	66	66	78	77	78	79	78	78
0.7	360 x 360	64	63	63	62	63	63	76	76	75	77	76	76
0.8	280 x 280	66	65	67	65	67	66	78	78	77	78	79	78
0.8	320 x 320	65	64	63	64	64	64	77	76	76	77	74	76
0.8	360 x 360	61	61	60	61	62	61	78	79	78	78	77	78
0.9	280 x 280	63	62	63	63	64	63	77	76	75	76	76	76
0.9	320 x 320	64	64	63	65	64	64	73	72	71	73	71	72
0.9	360 x 360	66	66	67	65	66	66	74	75	74	74	73	74
1	280 x 280	66	66	68	67	68	67	75	76	75	74	75	75
1	320 x 320	63	62	63	63	64	63	70	70	69	70	71	70
1	360 x 360	69	67	67	68	69	68	77	77	78	76	77	77

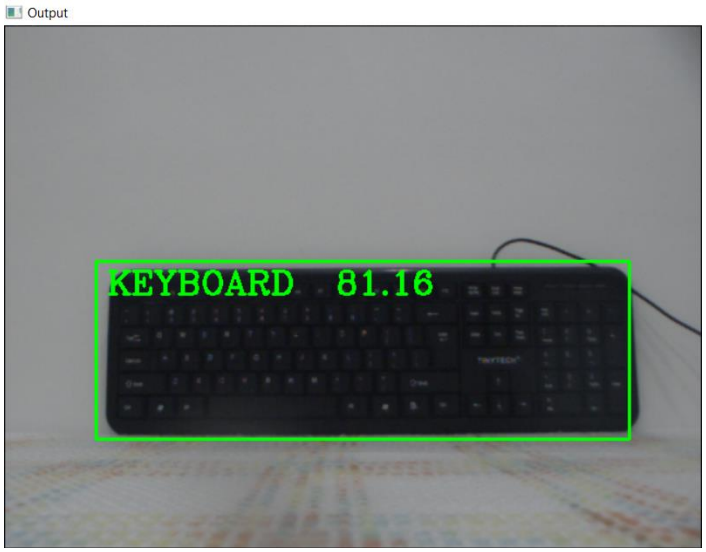


Figure 4.5: Result of object detection for the keyboard

4.2 Analysis and Discussion

First and foremost, there are two line graphs published in Figure 4.6 and Figure 4.7. In Figure 4.6, the line graph shows the relationship between the average detection rate with input size. While in Figure 4.7, the line graph shows interconnection with the average detection rate versus distance. The main purpose of these two graphs is to show the overall conclusion for the object detection process while the experiment is completed. Apart from that, the major objective of both graphs is showing the comparison between SSD algorithm and MobileNet-SSD algorithm in the term of detection rate, accuracy and performance. In fact, input size and distance represent the manipulated variable for the object detection process. Indirectly, the detection rate is manipulated by the different input sizes and distance setups. There are three input sizes mentioned in Figure 4.6 which is 78400 pixels^2 as 280x280 pixels, 102400 pixels^2 as 320x320 pixels and 129600 pixels^2 as 360x360 pixels. Besides that, five different distances are revealed in Figure 4.7, which is 0.6 meters, 0.7 meters, 0.8 meters, 0.9 meters, and 1meter. Based on the graph given in Figure 4.6 and Figure 4.7, it can be proved that the average value of detection rate for using the MobileNet-SSD algorithm as the configuration is higher than the SSD algorithm. In addition, it also can be demonstrated that the bigger the size of the object, the higher the detection rate will be obtained. This statement shows that the laptop and keyboard size is bigger than other materials such as cell

phone, computer mouse, and remote, so both get a slightly high detection rate in the object detection process. Furthermore, it can be determined that the shorter the distance setup, the higher the detection rate for the object will be achieved. Figure 4.7 shows the evidence to support the statement above, which can be seen in the average detection rate for the MobileNet-SSD algorithm while detecting the laptop. The average detection rate for laptop is gradually decreased due to the distance change. Additionally, it can be concluded that the size of the object and the distance setup between the object and camera will directly react to the detection rate for the object.

In conclusion, the objective for this project has been achieved. Obviously, the MobileNet-SSD algorithms can perform a high detection rate compare with the SSD algorithms. Based on the table and graph that shown, it can be proved that the MobileNet-SSD algorithm is suitable as the configuration for the object detection system due to its high detection accuracy while detecting the object in real-time. Additionally, a few limitations can be listed, such as the webcam's specification and the speed with which an object is detected. The PS3 EYE camera is chosen for this project's vision set up due to its reasonable price. The main issues are that the resolution pixel of the webcam and focal length of the webcam's lens is limited so that while the distance is a change to more than 1 meter, the webcam is unable to detect the object and fails to pass through the threshold, which is 50% and above. Therefore, the suggestion can change the webcam to a high-end camera such as the FUJI camera and LEICA camera to perform the object detection process.

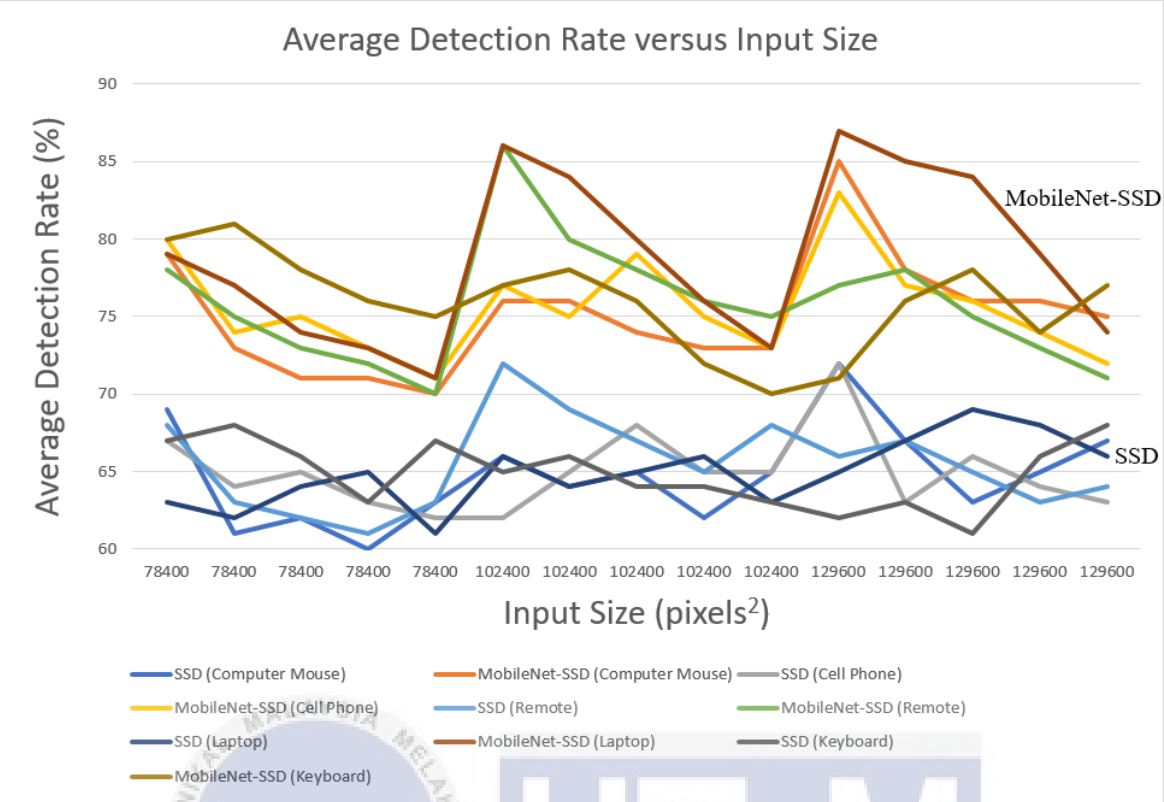


Figure 4.6: Graph of average detection rate versus input size.

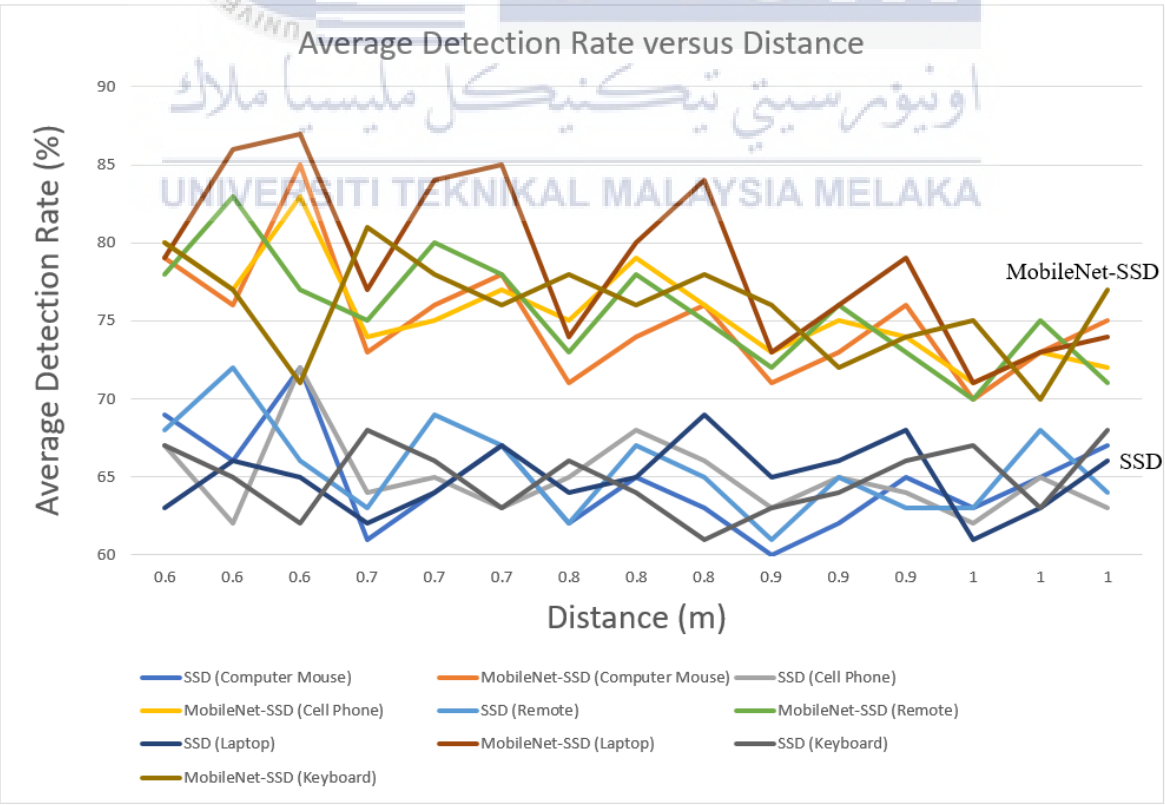


Figure 4.7: Graph of average detection rate versus distance.

4.2.1 Colour Analysis

Figure 4.8 shows the cell phone as the example used in the experimental framework to determine the performance of the object detection system while detecting multicolour objects in real-time on Raspberry Pi. Based on the result given, it can prove that the object detection system is able to detect the same types of material with different colours. Furthermore, even though the size and shape of the objects are different, the detection system still can detect and give the detection rate on the object. Meanwhile, both algorithms are performed in different colours, shapes, and sizes of objects to get the detection rate in real-time on the object detection system. Obviously, it can demonstrate that the object's colour will not affect the object detection system during the experiment. Besides that, the class name and detection rate will also be shown on the detected object. Therefore, it can be concluded that the five objects of 91 categories selected from the MS-COCO dataset can proceed with the experiment by applied the object detection system on Raspberry Pi in order to determine the detection rate for objects, even though the size, shape, and colour of an object are different.

Figure 4.8: Examples of the multicolour object.

CHAPTER 5
CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This project aims to implement the MobileNet-SSD algorithm into a low-cost vision detection system using OpenCV to create high accuracy and precision cost-effectiveness detection system. Besides that, it also focuses on investigating the suitability of running a real-time object detection system on a Raspberry Pi. There are two types of algorithms: Single-Shot Detectors (SSD) and MobileNet-SSD, which are implemented and tested in accuracy at different input sizes and distance setup. It concludes that the size of an input image and the distance between camera and object act as manipulating each model architecture's detection accuracy. Based on the experiment's observation, it can be proved that the speed of both configurations while detecting the object is very slow and that only suitable for those applications that do not require high speed using the Raspberry Pi as hardware. There is a trade-off with accuracy if higher speeds are achieved since there is not enough computational power to have both. Meanwhile, choosing a proper input size is essential for obtaining the right balance of accuracy needed for a particular application.

Furthermore, based on the result obtained, it can determine that the object's size will impact the detection rate because the data gained shows clearly that the detection rate for the bigger size of an object is much higher than the smaller size of the object. Therefore, selecting the same size of an object is considered a compulsory process to get the right balance of accuracy. In addition, the webcam is not only represented as the vision setup, but

it also plays an important role in the object detection system because the webcam's performance will affect the result of detection rate for the detecting object. Hence, the specification of the webcam acts as an important part in order to make the object detection system successful. At the same time, this study is beneficial to those interested in creating an object detection system with similar hardware to find the balance.

5.2 Future Recommendations

On behalf of successfully implementing this project, some limitations still act as the constraints of this project. For example, the lower resolution pixels of the webcam and the focal length of the webcam lens will affect the quality of video capture and the detection rate while running the object detection system in real-time on Raspberry Pi. For the recommendation of vision setup, the high end of the camera should be selected and used for the object detection system in order to increase the quality of images and ensure the detection accuracy rate for the object will be more relevant. In addition, for improving the hardware setup, add the external LED light on a Raspberry Pi. The preparation needs a breadboard, an LED light, a 330-ohm resistor, and two male-female jumper wires. The purpose of assembling an external LED light on Raspberry Pi is to instruct the user to identify which object has been detected based on the programming's colour set up. For example, the red colour is represented as the keyboard, and the green colour is represented as cell phone, so the red light will light up while the keyboard is undergoing the object detection process. Furthermore, the object detection system also can implement in the production line for the inspection process. For instance, while the product is put on the conveyor, the webcam can still detect and give the signal to the operator. Due to the time limitation, a few things should have been undertaken to reinforce this study's results, such as testing different object categories, distances, and input sizes. Another thing is to train the architecture model using different datasets with higher resolution and observe whether it will increase detection accuracy rate with smaller objects. Besides that, one thing left out in this project is investigating the influence lightning has on models' ability to detect objects, which should be discussed in future work.

5.3 Sustainability

First and foremost, electronic devices are now integrated with human beings and have become indispensable in human life. Moreover, in the new era of globalization, most people will have electronic products such as smartphones, television, computers, etc. In other words, the development of science and technology has become the most concerned topic in the world. Therefore, electronic devices can be a tremendous long-term sustainability strategy if the publics can move beyond short-term economics. Furthermore, based on the trend, most electronic devices are made free of hazardous chemicals and recyclable. Apart from that, green manufacturing plays a vital role in the electronic industry to maintain the product's sustainability. The primary process of green manufacturing is to produce things that can be used for an extended lifetime of the product and then disassembled and reused, thus promoting a more sustainable manner of building and using technology. Besides that, green manufacturing is concerned with finding alternatives to the waste and toxic materials commonly utilised to construct modern technologies. At last, the components and materials used in this system were user-friendly components that kind to an environment operated without harmful emissions and low electric consumption.

5.4 Complexity

There are minimal complications encountered throughout this project's completion. Firstly, the complexity of detecting electronic devices of different sizes and shape such as computer mouse, keyboard, cell phone, remote and laptop. Next, the setup for camera, lighting and the object must be located at the correct position to ensure the image for the object when detecting is clear and appropriate to produce a successful output in real-time object detection on Raspberry Pi. Besides that, based on eight types of algorithms, select the suitable algorithm for the object detection system. The procedure was to identify and categorise the object's class names and calculate the detection rate based on the study from the literature review. Before executing this project, it is necessary to have a thorough understanding of the theoretical concepts and functions behind each process flow in order to ensure that the object detection system operates smoothly. Some various codes and functions

need to be applied when implementing the object detection algorithm. Before completing the object detection module, some compulsory files need to be prepared, such as the weight file, configuration file, and database as the module's support. From previous research, there was no specific instruction on how to construct the coding for creating an object detection system; therefore, a self-learning practice to create the code and run it without error is challenging work. Lastly, the primary code that controlled the system's whole functionality was complicated since the system required to get accurate data during the object detection process in real-time. Additionally, the coding for the object detection system must be written in appropriate values and commands to ensure the output of the result will be accurate and correct.

5.5 Lifelong Learning

During the assessment of this project, the object detection system can serve as a sense of lifelong learning which can classify and categorise the general object through different extraction functions. When implemented in the industry, this technique will improve production efficiency by cutting down labour costs, optimising usage time, and minimising trash disposed of more ecologically. Furthermore, this initiative will prepare companies to decrease labour costs by lengthening cycle times and monitoring inefficient applications using production engineering knowledge. The object detection system can connect with a machine for functioning to monitor and check the product's condition, whether it is pass or not, with the threshold set by the company. Using this detection system not only can save a lot of labour costs, but the human factor issue will also reduce. Indirectly, the visibility and productivity of the company will be increasing. Apart from that, the rapid development and advancement of technology make the companies increase the productivity and demand as much as possible by changing the conventional methods into an automated system.

5.6 Entrepreneurship

The execution of this project may benefit from many critical learnable skills that can help it achieve more success and influence as an entrepreneur. The ability to prioritise and stay focused on a goal is a critical component of entrepreneurship. It is critical to possess the personal discipline necessary to establish the project's goals and objectives and work diligently to achieve them. Conviction is also a necessary component of being an ardent advocate. Believing in something powerful enough to transform passion into action, this individual has developed into an advocate. This strength and influence may be utilised to convince others to make the right choice. The broad knowledge gained via this project enables similar achievement in other business areas, thus assisting in overcoming future obstacles.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., … Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016*, 265–283.

Ali, H., Khursheed, M., Fatima, S. K., Shuja, S. M., & Noor, D. S. (2019). Object recognition for dental instruments using SSD-mobilenet. *2019 International Conference on Information Science and Communication Technology, ICISCT 2019*, 1–6. https://doi.org/10.1109/CISCT.2019.8777441

Anaz, A. S. (2020). *Comparison Between OpenCV and MATLAB Performance In Real Time Comparison between Open CV and MATLAB Performance in Real Time Applications Ammar Sameer Anaz Diyaa Mehadi Faris Abstract :* □□□□□□□ □□□□□ □□□□□□□ □□ *(MATLAB)* □□□□ *(Open CV)* □□□ □□□□ □□□ □□□. *March*.

Askar, W., Elmowafy, O., Youssif, A., & Elnashar, G. (2015). Object Tracking Using Vision on Raspberry Pi. *International Conference on Aerospace Sciences and Aviation Technology*, *16*(AEROSPACE SCIENCES), 1–9. https://doi.org/10.21608/asat.2015.22940

Chiu, Y. C., Tsai, C. Y., Ruan, M. Da, Shen, G. Y., & Lee, T. T. (2020). Mobilenet-SSDv2: An Improved Object Detection Model for Embedded Systems. *2020 International Conference on System Science and Engineering, ICSSE 2020*, 0–4. https://doi.org/10.1109/ICSSE50014.2020.9219319

Deepthi, R. S., & Sankaraiah, S. (2011). Implementation of mobile platform using Qt and OpenCV for image processing applications. *2011 IEEE Conference on Open Systems, ICOS 2011*, 284–289. https://doi.org/10.1109/ICOS.2011.6079235

Deng, J., Dong, W., Socher, R., Li, L., Li, K., & Fei-fei, L. (2009). *ImageNet : a Large-Scale Hierarchical Image Database ImageNet : A Large-Scale Hierarchical Image Database*. *May 2014*. https://doi.org/10.1109/CVPR.2009.5206848

Epiphany, J. L. (2014). On Teaching Digital Image Processing with MATLAB. *American Journal of Signal Processing , January 2014*. https://doi.org/10.5923/j.ajsp.20140401.02

Everingham, M., Eslami, S. M. A., Gool, L. Van, Williams, C. K. I., & Winn, J. (2015). *The PASCAL Visual Object Classes Challenge : A Retrospective*. 98–136. https://doi.org/10.1007/s11263-014-0733-5

Everingham, M., Gool, L. Van, Williams, C. K. I., & Winn, J. (2010). *The PASCAL Visual Object Classes (VOC) Challenge*. 303–338. https://doi.org/10.1007/s11263-009-0275-4

Feng, X., Jiang, Y., Yang, X., Du, M., & Li, X. (2019). Computer vision algorithms and hardware implementations: A survey. *Integration*, *69*(April), 309–320. https://doi.org/10.1016/j.vlsi.2019.07.005

Galvez, R. L., Bandala, A. A., Dadios, E. P., Vicerra, R. R. P., & Maningo, J. M. Z. (2019). Object Detection Using Convolutional Neural Networks. *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, *2018-Octob*(May 2019), 2023–2027. https://doi.org/10.1109/TENCON.2018.8650517

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., & Krasin, I. (2017). *The Open Images Dataset V4 Unified image classification , object detection , and visual relationship detection at scale*.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8693 LNCS*(PART 5), 740–755. https://doi.org/10.1007/978-3-319-10602-1_48

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2020a). Deep Learning for Generic Object Detection : A Survey. *International Journal of Computer Vision*, *128*(2), 261–318. https://doi.org/10.1007/s11263-019-01247-4

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2020b). Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, *128*(2), 261–318. https://doi.org/10.1007/s11263-019-01247-4

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2015). SSD: Single Shot MultiBox Detector. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9905 LNCS*, 21–37.

Matuska, S., Hudec, R., & Benco, M. (2012). The comparison of CPU time consumption for image processing algorithm in Matlab and OpenCV. *Proceedings of 9th International Conference, ELEKTRO 2012*, 75–78. https://doi.org/10.1109/ELEKTRO.2012.6225575

Neelima, K. B., & Saravanan, T. (2014). *Image Detection and Count Using Open Computer Vision (Opencv)*. *4*(9), 107–109.

Pehlivan, B., Kahraman, C., Kurtel, D., Nakip, M., & Guzelis, C. (2020). *Real-Time Implementation of Mini Autonomous Car Based on MobileNet - Single Shot Detector*. 1–6. https://doi.org/10.1109/asyu50717.2020.9259830

Pi, R. (2020). Raspberry Pi 4 Computer Model B. *Raspberrypi.Org*, *May*, 1–6. www.raspberrypi.org

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, *115*(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y

S.Sathiyamoorthy. (2014). Industrial Application of Machine Vision. *International Journal of Research in Engineering and Technology*, *03*(19), 678–682. https://doi.org/10.15623/ijret.2014.0319120

Shah, M., & Kapdi, R. (2017). Object detection using deep neural networks. *Proceedings of the 2017 International Conference on Intelligent Computing and Control Systems, ICICCS 2017*, *2018-Janua*, 787–790. https://doi.org/10.1109/ICCONS.2017.8250570

Sharma, G. (2017). Performance Analysis of Image Processing Algorithms using Matlab for Biomedical Applications. *International Journal of Engineering and Manufacturing*, *7*(3), 8–19. https://doi.org/10.5815/ijem.2017.03.02

Sunitha, M. R., Khan, F., Gowtham Ghatge, R., & Hemaya, S. (2019). Object Detection and Human Identification using Raspberry Pi. *1st IEEE International Conference on Advances in Information Technology, ICAIT 2019 - Proceedings*, 135–139. https://doi.org/10.1109/ICAIT47043.2019.8987398

Vahab, A., Naik, M. S., Raikar, P. G., & R, P. S. (2008). Applications of Object Detection System. *International Research Journal of Engineering and Technology*, 4186. www.irjet.net

Wu, X., Sahoo, D., & Hoi, S. C. H. (2020). Recent advances in deep learning for object detection. *Neurocomputing*, *396*, 39–64. https://doi.org/10.1016/j.neucom.2020.01.085

Yongxiang, W. (2009). Research on bank intelligent video image processing and monitoring control system based on OpenCV. *2009 3rd International Conference on Anti-Counterfeiting, Security, and Identification in Communication, ASID 2009*. https://doi.org/10.1109/ICASID.2009.5276928

APPENDICES

A Specifications Of Object Detection System Hardware

Model	Raspberry Pi 4 Model B
SoC Type (Processor)	Broadcom BCM2711 (with metal cover)
Core Type	Cortex-A72 64-bit (ARMv8)
No.of Cores	Quad-Core
GPU	VideoCore VI
Multimedia	H.265 decode (4Kp60) H.264 decode (1080p60) H.264 encode (1080p30) OpenGL ES 1.1,2.0,3.0 Graphics
Memory/OS storage	microSD
RAM	LPDDR4:1GB,2GB,4GB and 8GB options
Ethernet	True Gigabit Ethernet
USB Port	2 x micro HDMI support Dual Display
WIFI	802.11 b/g/n/ac (2.4GHz+5GHz & Shielded)
Bluetooth	5.0 + BLE (Shielded)
Antenna	PCB Antenna (Similar to Rpi Zero W)
GPIO	40 pins (Fully backwards-compatible with previous boards)
Operating System (OS)	Latest Raspbian (>24 June 2019)
Dimension	85mm x 56mm
Power Input	5V via USB Type C (up to 3A)

Model	Logitech C310 HD Webcam
Resolution and Frame Rate	480p at up to 30 fps 720p at up to 30 fps
Sensor	CMOS
Sensor Resolution	1.3 MP 5 MP (Software Enhanced)
Field of View	60°
Zoom	None
Focus Type	Fixed
Host Connection	USB Type-A USB 2.0
Power Source	Bus Powered
Low-Light Correction	No
Operating Systems (OS)	Window 7 or Later
Microphone	Yes
Lens Material	Plastic
Privacy Shutter	No
Attachment Method	Clip
Cable Length	1.5m

Model	ASUS VS229H-P
Panel Size	Wide Screen 21.5"(54.6cm) 16:9
True Resolution	1920x1080
Full HD 1080P	Yes
Pixel Pitch	0.248mm
Brightness (Max)	250 cd/㎡
ASUS Smart Contrast Ratio (ASCR)	50000000:1
Viewing Angle (CR≧10)	170°(H)/160°(V)
Response Time	5ms
Display Colors	16.7M
SPLENDID Video Preset Modes	6 Modes
Skin-Tone Selection	3 Modes
Colour Temperature Selection	4 Modes
Signal Input	HDMI, D-Sub, DVI-D
Earphone jack	3.5mm Mini-Jack
Analog Signal Frequency	30~80 kHz(H)/ 55~75 Hz(V)
Digital Signal Frequency	30~80 kHz(H)/ 55~75 Hz(V)

Model	RPi 15W (5V/3A) PSU USB C UK Plug-Black
Input Voltage	Range: • 100-240VAC (rated) • 96-264VAC (operating)
Input Current	0.5A (MAX)
Output Voltage	DC 5.1V
Output Current	3.0A
Output Connector	USB Type-C
Cable	1.5m 18 AWG Captive Cable
Output Power	15.3W (MAX)
Frequency	50/60Hz ±3Hz
Efficiency	72% to 81%

Model	Male-to-male Micro HDMI to Standard HDMI cable
Cable Length	1.5m
Colour	Black
Feature	Support both image and audio

Model	Kingston Micro SD
Capacity	32GB
Data Transfer Rate	Up to 80MB/s
Speed Class	Class 10
Dimension	11mm x 15mm x 1mm
Operating Temperature	-25°C to 85°C
Storage temperature	-40°C to 85°C

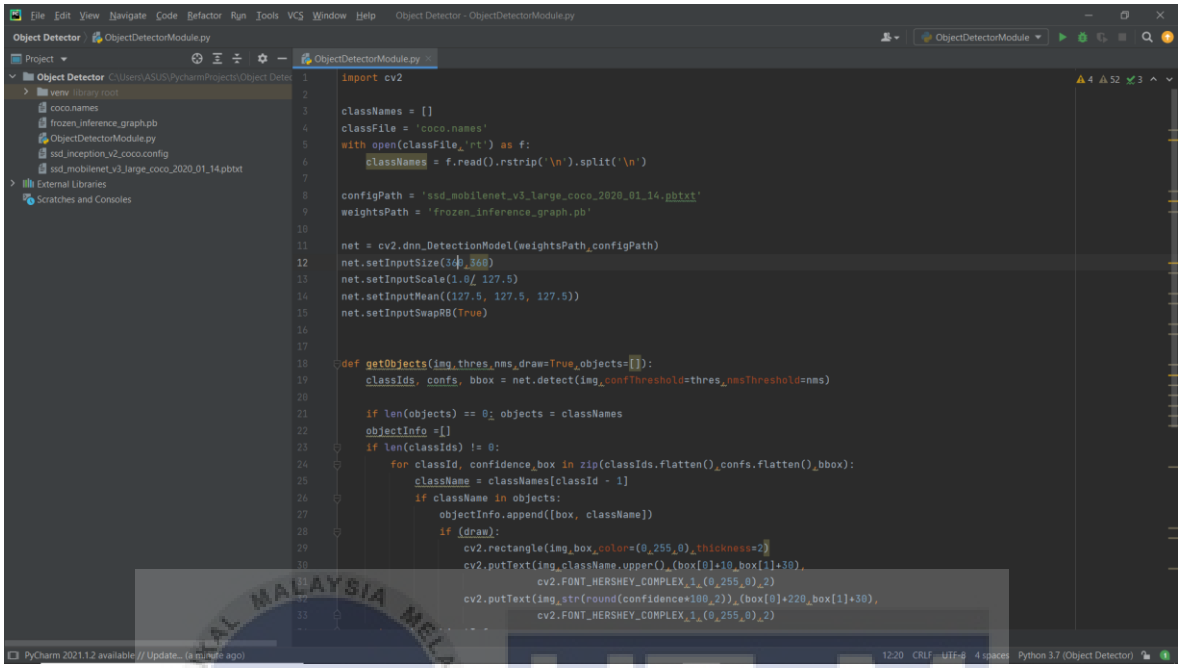
Model	Keyboard KM-720
Hotkeys	12 FN Multimedia Hotkeys
Rating	5V / 30mA
Dimension	450 × 150 × 25 mm
Weight	585g
Cable Length	150cm
Hardware Connectivity	USB

Model	Computer Mouse OP-620D
Report Rate	125Hz
Sensor Technology	Optical
Resolution	1000 DPI
Rating	5V / 100mA
Dimension	119 × 63 × 36 mm
Weight	90g
Cable Length	150cm
Hardware Connectivity	USB
System Requirements	Windows XP / Vista / 7 / 8 / 8.1 / 10

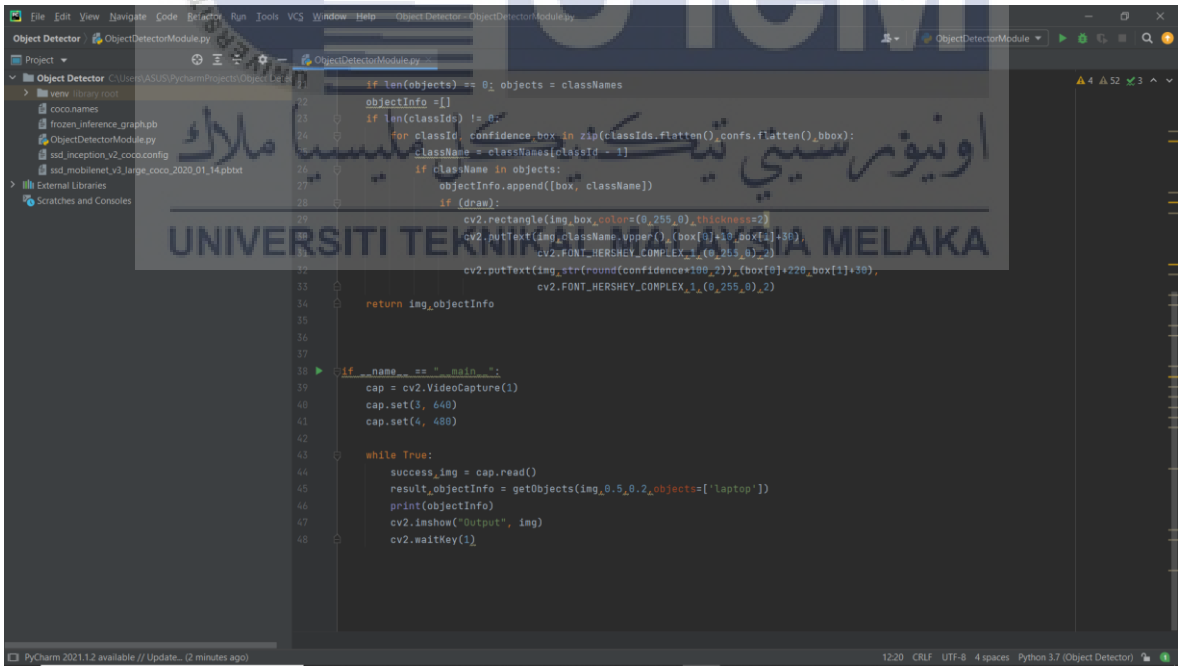
Model	Tripod
Folded Height	15cm
Height (MAX)	20cm
Screw Size	1/4"
Material	Aluminium Alloy
Weight	49.2g

Model	Zen LED Downlight
Lumens	1.711 lm
Power (W)	25
Voltage Range	220 – 240V, 50/60Hz
Power factor	0.9
Lifetime (hrs.)	50,000
Temperature (MAX)	+40°C
Temperature (MIN)	-20°C

B Coding Of Object Detection



```
import cv2

classNames = []
classFile = 'coco.names'
with open(classFile,'rt') as f:
    classNames = f.read().rstrip('\n').split('\n')

configPath = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
weightsPath = 'frozen_inference_graph.pb'

net = cv2.dnn_DetectionModel(weightsPath,configPath)
net.setInputSize(320,320)
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)


def getObjects(img,thres,nms,draw=True,objects=[]):
    classIds, confs, bbox = net.detect(img,confThreshold=thres,nmsThreshold=nms)

    if len(objects) == 0: objects = classNames
    objectInfo =[]
    if len(classIds) != 0:
        for classId, confidence,box in zip(classIds.flatten(),confs.flatten(),bbox):
            className = classNames[classId - 1]
            if className in objects:
                objectInfo.append([box, className])
                if (draw):
                    cv2.rectangle(img,box,color=(0,255,0),thickness=2)
                    cv2.putText(img,className.upper(),(box[0]+10,box[1]+30),
                                cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
                    cv2.putText(img,str(round(confidence*100,2)),(box[0]+220,box[1]+30),
                                cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
```



```
    if len(objects) == 0: objects = classNames
    objectInfo =[]
    if len(classIds) != 0:
        for classId, confidence,box in zip(classIds.flatten(),confs.flatten(),bbox):
            className = classNames[classId - 1]
            if className in objects:
                objectInfo.append([box, className])
                if (draw):
                    cv2.rectangle(img,box,color=(0,255,0),thickness=2)
                    cv2.putText(img,className.upper(),(box[0]+10,box[1]+30),
                                cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
                    cv2.putText(img,str(round(confidence*100,2)),(box[0]+220,box[1]+30),
                                cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
    return img,objectInfo


if __name__ == "__main__":
    cap = cv2.VideoCapture(1)
    cap.set(3, 640)
    cap.set(4, 480)

    while True:
        success,img = cap.read()
        result,objectInfo = getObjects(img,0.5,0.2,objects=['laptop'])
        print(objectInfo)
        cv2.imshow("Output", img)
        cv2.waitKey(1)
```

C Gantt Chart For Fyp 1

	Month	Oct-20				Nov-20				Dec-20				Jan-21				Feb-21			
No	Week / Activity	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Submission of FYP Title																				
2	Meeting with FYP supervisor																				
3	FYP Title Briefing																				
4	Talk 1: Chapter 1: Introduction																				
5	Talk 2: Chapter 2: Literature Review																				
6	Talk 3: Technical English																				
7	Talk 4: Chapter 3: Methodology																				
8	Talk 5: Reference and Format																				
9	Report Progress: Chapter 1																				
10	Report Progress: Chapter 2																				
11	Report Progress: Chapter 3																				
12	Logbook Submission																				
13	Presentation																				
14	FYP 1 Submission																				
15	Preparation of Raw Material and Equipment																				