CLOUD-BASED OBSTACLE DETECTION SYSTEM FOR DRIVERS

By

Eio Hua Zen

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONS)

INFORMATION SYSTEMS ENGINEERING

Faculty of Information and Communication Technology
(Kampar Campus)

JAN 2020

REPORT STATUS DECLARATION FORM

Title: Cloud-Based Obstacle Detection System for Drivers

Academic Session: JAN 2020

I EIO HUA ZEN

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

_______________________ _______________________

(Author's signature) (Supervisor's signature)

Address:
566, Kampung Baru, Sungai Batu
34900 Pantai Remis,
Perak.

Lau Phooi Yee, PhD
____
Supervisor's name

Date: 20/4/2020

Date: 20 April 2020

CLOUD-BASED OBSTACLE DETECTION SYSTEM FOR DRIVERS

By

Eio Hua Zen

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONS)

INFORMATION SYSTEMS ENGINEERING

Faculty of Information and Communication Technology
(Kampar Campus)

JAN 2020

DECLARATION OF ORIGINALITY

I declare that this report entitled "**CLOUD-BASED OBSTACLE DETECTION SYSTEM FOR DRIVERS**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _________________________

Name : Eio Hua Zen

Date : 20/4/2020

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Dr. Lau Phooi Yee who has given me this bright opportunity to engage this project. A million thanks to you.

Finally, I must say thanks to my parents and my family and friends for their love, support, and continuous encouragement throughout the course.

ABSTRACT

Based on the past statistics and record, majority of the road accidents take place because driver is not concentrated enough in driving and causing lack of response time to instant traffic events. People expect to have an automated system that provides drivers the traffic sign information and detect the road condition. One of the most important functions is obstacle detection and recognition. This system involves the use of camera to capture the real-time road condition then identify the obstacle which are encountered by the vehicle, then provides correct information to the user. In this paper, the project proposed is cloud-based obstacle detection system for drivers. It is one of the most popular example of artificial intelligence system that used to detect obstacle. Artificial intelligence (AI) is something intelligent and it could perform things that only human can perform. It might even be more powerful than the human if it was well trained and developed. The system will be developed in mobile application. The application will provide information of the road condition to user once the obstacle is detected. The detected obstacle will be uploaded to database server whereby other user is able to access the information as well. To enable the mobile application to be more user-friendly, the information of detected object will be displayed in the form of icon on the map. User can simply click on the icon to know more details about the detected objects. For example, the date and time of detection, the name of user upload the data, the name of object detected, the actual location of the object and so on. The application will be developed with the help of Android Studio, Google Maps JavaScript API and TensorFlow API. The system can only to be operated when accessing to Internet. To study the performance of this cloud-based obstacle detection system, several evaluations were conducted.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ADB	Android Debug Bridge
AI	Artificial intelligence
API	Application Programming Interface
ASP	Active Server Page
B	Blue
COCO	Common Objects in Context
COM	Component Object Model
CPU	Central processing unit
CUDA	Compute Unified Device Architecture
DDoS	Distributed Denial-of-Service
Fast RCNN	Fast Regional-Based Convolutional Neural Network
Faster RCNN	Faster Regional-Based Convolutional Neural Network
FN	False Negative
FP	False Positive
G	Green
GPS	Global Positioning System
GPU	Graphics processing unit
IoU	Intersection over Union
GUI	Graphical User Interface
HTTP	HyperText Transfer Protocol
IoU	Intersection over Union
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
mAP	Mean Average Precision
ML	Machine Learning
OS	Operating System
PHP	Hypertext Preprocessor
R	Red
RCNN	Regional-Based Convolutional Neural Network
RPN	Regional Proposal Network
SDK	Software Development Kit
SDK	Software Development Kit

SSD	Single Shot Detector
TF	TensorFlow
TN	True Negative
TP	True Positive
URL	Uniform Resource Locator
Wi-Fi	Wireless Fidelity
YOLO	You Only Look Once
YOLOv2	You Only Look Once Version 2
YOLOv3	You Only Look Once Version 3

Chapter 1 Introduction

1.1 Problem Statement and Motivation

Every year, millions of people are killed and injured on roads (Jianmin Duan and Viktor, 2015). To solve this problem, it is advisable to develop system to assist human in driving. Therefore, auto driving is turning into a prominent point in numerous fields. It is critical to define, perceive traffic lights, street signs, humans on street, and other obstacles which help in the driving.

As we move towards progressively into image understanding, having more accurate object recognition gets significant. Human not only thinks about detecting object, yet additionally think about accurately defining the class and location of objects.

A very well-known application of object detection used in is the self-driving cars to detect objects and obstacle around such as cars, people, obstacles, traffic, pets and bicycle. With the application of object detection, autonomous car will be able to detect the obstacle in front and make the decision whether change its way or stop. Besides, autonomous car will be able to detect the sign board or traffic light in front and follow the instruction and traffic rule.

By using this application in autonomous car, the occurrence of accident will greatly reduce, solving many problems of citizen and country. This is because AI does not require rest, they able to fully concentrate all the time to prevent accident occur unlike human being.

In this project, the system will only able to detect and classify the object in front and upload the data to cloud system whereby other user can access to the information through the mobile application as well. Further work and research needed for the system to make the decision after facing the obstacle.

A mobile application will be developed to let the user to detect obstacles as well as upload or receive the information that what obstacle is detected on certain route.

The software required in this project is the Android Studio software as mobile application need to be developed. The choices of framework used would be the TensorFlow SSD-Mobile Net to run the object detection.

1.2 Objective

The objective of obstacle detection system is to detect and classify one or more obstacle captured by a smartphone in real-time. With the help of mobile application, the system provides the user real time information from road view. The system is able to show the real-time result from the obstacle detection on map in mobile application. All of the objectives have been achieved.

1.3 Project Scope and Direction

The project entails the development of machine learning model running on the smartphone which capable to identify and classify the obstacles accurately in real time form and allows the user to upload the data to cloud whereby other users can access.

First, we need to gather and label of the training and testing data. The dataset will be very large because the system needs to be trained with different objects which placed in different locations or backgrounds. To ease the work, COCO datasets is chosen to complete the detection system.

When the system begins to operate, the smartphone captures the scene in front of the user and process the captured frame. Deep learning algorithm in mobile application will be used to recognize objects in real-time, and it subsequently sends the results to the database server. This can be helped by connecting mobile application to the server with 000Webhost and PHP.

One of the restrictions is the speed of detection is not as fast as it could done in computer with strong GPU. The result of detection will be lagging few seconds behind. Besides, due to the change of weather conditions or viewing angles, some objects are difficult to be seen

1.4 Project Background

Artificial intelligence (AI) is a machine system that consist of sense of human minds. It is defined as an intelligent device which react according to its environment. When a machine can do some reasoning, self-correction and self-learning, it is known as an artificial intelligence system.

The birth of AI is during the year of 1956's at Dartmouth where Allen Newell, Herbert A. Simon and Cliff Shaw came out with the Logic Theorist (Brunette et al., 2009). They had initiated a series of research projects which related on the programming of computers to have human behaviors. The Logic Theorist was a software program that used to prove the theorems in symbolic logic. It is the first working program that able to simulate and solve some complex problem which require some aspect of human sense. This Logic Theorist and some other cognitive simulations developed by Allen Newell, Herbert A. Simon and Cliff Shaw had brought lots of opportunities for researcher to explore on the information-processing psychology developing field (Brunette et al., 2009). For current development, when dealing with some complex tasks studied in human factors psychology, Logic Theorist is still the central part of theory cognitive psychology, and the ideas from the Logic Theorist is still required for the use to solve those complex tasks.

During the late 1990s, AI start to take into concern in the real-world applicability. During the year of 1997, IBM's Deep Blue chess program had successfully defeated the world champion, Garry Kasparov and this is the time that prove that the AI system is getting more intelligent (May, 2017). Some real-world application such as image recognition and speech recognition are start being developed by the researchers. Researches are finding way to allow the algorithms learn the logical rules by themselves without structuring the logical rules which set by humans manually (May, 2017). The researchers then shift their focus into the Artificial Neural Networks (ANNs). The ANNs was first invented during the year 1940s to mimic how the human brains learn. After a few decades, this ANNs was been use widely as the concept of the backpropagation of gradient descent was improved quite a lot (May, 2017). The backpropagation method helps to reduce the number of permutations required. Hence, it become more efficient when comes to the training session. But there are still some limitations with current technology that had plagued their adoption even with the use of improved new algorithms. In 2006, some changes were made on the ANNs and now it is replaced by the term of deep learning neural networks (DNNs) which implemented by Geoffrey Hinton (May, 2017). Hinton added multiple layers of neural networks into ANNs in order to optimize the results obtained by each layer. Hence, there is a significant improve while the learning was now accumulated faster up the stack of layers. When the Graphical Processing Units (GPUs) was implemented, Andrew Ng of Stanford University make an improvement for the deep neural networks using the GPUs at year of 2012 (May, 2017). GPUs consist of massive parallel architecture that can handle multiple tasks simultaneously. Ng found that the training time used to train the deep learning neural networks is significantly reduced compared with the use of general-purpose CPUs (May, 2017).

Currently, there are 4 main factors which help to drive the AI today. First of it is the “Big Data” was introduced. AI require a huge amount of data in order to learn more effectively and precisely. Big Data provide lots of data information from different sources such as mobile computing, Internet of Things sensors and social.

The second factor is about the cheap computation power. Hardware is the one which remained as a constrain factor after the AI algorithms was improved at the past. Now, GPU gained the popularity in the AI community as it provides high computational power which can run the operations parallel and perform matrix multiplication in a more efficient manner. Not only the implementation of GPU, CPUs have also been improved as it now able to perform more efficient matrix computation and more effective parallelization with the new deep learning instruction set implement in the CPUs processor (May, 2017).

The third factor is a more sophisticated algorithms was implemented in recent. The state of the art in AI is depending on how the algorithms works. Hence, a more advanced algorithm would allow the AI to solve some of the specific problems such as speech recognition and image classification with a more precise accuracy.

Lastly, broader investment is also one of the main factors to drive the AI towards the future. In the past, the funding in this area is not enough combined with some challenging problems met in AI resulted in minimal progress. Now, AI investment is keeping increase as many people see the possibility and the benefits of AI which able to makes lots of working tasks become simple.

Deep learning might be the best way in training a better AI but when it comes to training process, it requires lots of data to make sure it would achieve a more precise accuracy. Just like a normal human brain, a lot of real-world experiences must be acquired in order to learn and deduce from it. For the artificial neural networks, it requires a huge amount of data to abstract more parameters in more details. For example, an image object recognition tasks require at least thousands of training data image in order to allow the machine extract and recognize the details of the object clearly.

In the previous decade, object detection is hard to be implement because of the insufficient of dataset and the lack of powerful CPU resources. After the implementation of GPU by NVIDIA in year 1999, developing a deep learning model is not a dream anymore. The time used for training and testing the dataset had been reduced significantly. A powerful GPU can at least decrease 70% of the training and testing time for a model. The amount of data used would also affect the quality of a classifier model. Insufficient data would lead to bad performance.

With current technology, we now could obtain data image easily from different type of resources and the data can also be transfer easily through high speed internet. But for specific item, it might hard to be found on the internet. Hence, we still need to capture our own image if we want to use it as our dataset. A classifier is just available to classify an image into one category of classes. For example, in an image which consist of a cat, a classifier would identify it as a cat. If an image contains more than one objects, the classifier will also available to categories it into a certain class only.

However, since the project has to be done in Android Smartphone. There is no GPU in Android phone, so the performance is not as good as the similar system operating in computer with a GPU. The good news is deep learning technology is still able to embed into mobile application.

1.5 Report Organization

This paper is organized in five (5) chapters. It contains introduction, literature review, system design, results and discussion, and conclusion and recommendations.

Chapter 1 is about the introduction, in which the problem statement and motivation, objective, project scope and direction, contribution and background information.

Chapter 2 is the literature review, which consists the reviews regarding on the previous similar systems and techniques such as object detection algorithms and cloud-based system.

Chapter 3 is the system design. The details of the design, methods and tools used in the project will be presented in this chapter.

Chapter 4 is the experimental results, analysis and discussions. All the testing results of the project are described and analyzed.

Chapter 5 is the conclusion and recommendations. This chapter includes the summary of the project as well as the suggestions and further development that can be mad

Chapter 2 Literature Review

2.1 Artificial intelligence (AI)

Artificial intelligence (AI) is the intelligence acquired by a machine. It would know how to do some human tasks such as reasoning, learning, planning and communicating (May, 2017). AI is generally a branch of computer science which focused on developing machines capable of intelligent behaviour. For machine and deep learning, both of them using the algorithms to learn from given data and make prediction on another new set of data (May, 2017). The algorithm required a huge amount of data in order to perform well in specific task. Machine and deep learning are the subset of the AI and deep learning is also the subset of the machine learning which focus even more narrowly on machine learning techniques that require "thought".

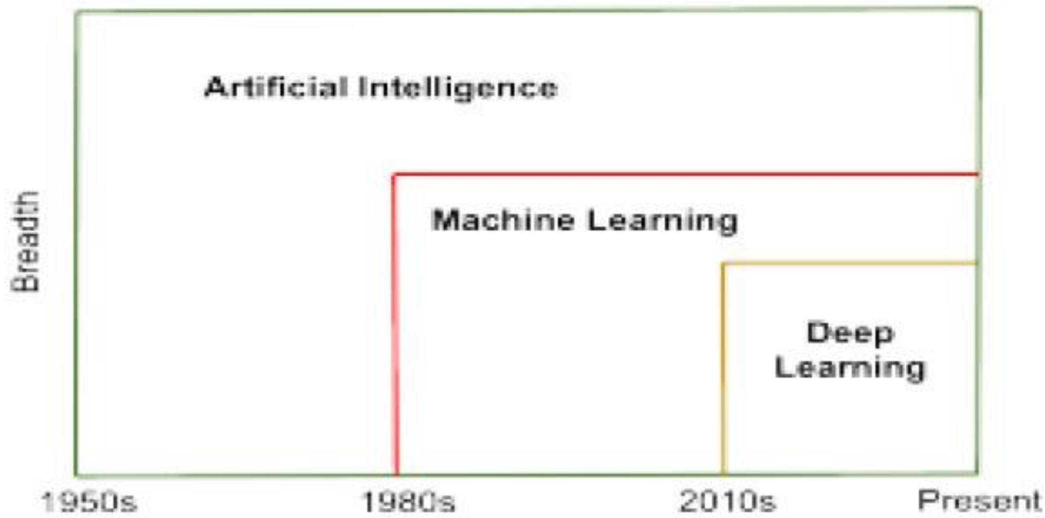


Figure 2-1: Relationship between Artificial Intelligence, Machine Learning and Deep Learning (May, 2017)

2.1.1 Machine Learning

According to Arthur Samuel in 1959, he mentioned that machine learning enables the "computers to learn without being programmed explicitly". Machine learning would learn from the data and make predictions on data through the explores study and construction of algorithms. There are two type of machine learning methods are widely used in recent real-world applications. One of the machine learning methods is the supervised learning and the another is the unsupervised learning method.

Supervised learning algorithms is given labelled samples and so the machine was trained to recognize the input and give a correct output based on the input given (Ongsulee, 2017). For examples, an image could be labelled either cat or dog and then when a new image was inserted, it would outcome a correct output based on the image given. The algorithm would compare the actual output with the correct outputs to find the errors and try to learn from it (Ongsulee, 2017). Supervised learning will use the pattern of a labelled input to predict the value on the unlabelled data. Supervised learning requires the supervision of a human and the labelled data was manually labelled by the programmer. There are 2 groups of supervised learning that it can be categorize which one is the regression and the another is the classification problem.

Unsupervised learning is a more interesting way of learning to study the representation of input patterns in a way that reflects the statistical structure of the overall patterns (Dayan, 2009). Unsupervised learning would react more likely to a human brain compared to the supervised learning. The training set used in unsupervised learning do not include any labels. The system was not told taught in recognizing the exact answer (Ongsulee, 2017). The algorithm must figure out by its own about what is being inputted. Unsupervised learning most of the time used in the transactional data and it is working quite well on it (Ongsulee, 2017).

Other than the supervised learning and the unsupervised learning, there are 2 more type of machine learning exist which is the semi-supervised learning and reinforcement learning. In semi-supervised learning, the word “semi” represents the need of the programmer in order to do correction whenever the machine did some mistake. It uses both unlabelled and labelled data for the training purposes. This method of learning is usually useful in the classification, regression and prediction process (Ongsulee, 2017). Whenever the cost associated is too high for labelling data, semi-supervised learning is recommended to be applied in order to reduce the cost by using more unlabelled data for training process (Ongsulee, 2017). For reinforcement learning, it basically learns through the trial and error process which would help to optimize the action yield accuracy (Ongsulee, 2017). Gaming, robotics and navigation application would often consider using this reinforcement learning method. Three primary components are required to take in consideration in this learning process which is the environment, the agent and the actions. The environment is referring to the things that the agent which is also known as the decision maker would interacts with. Other than that, the actions are basically what the agent can do. The main objective is to optimize the expected outcome for a given period by choosing the right actions by the agent (Ongsulee, 2017).

2.1.2 Deep Learning

Deep learning is a very powerful technique which is the enhancement of the machine learning. It is basically the integration of branch of machine learning. Deep learning has removed the need for feature engineering and replace it with a brittle, complex and engineering heavy pipelines with one end to one end trainable models with the help of different tensor operations. Deep learning has the capability to learn from the past prediction by its own and to continually improve their predictions based on new testing data (Ongsulee, 2017). Artificial neural networks have unique capabilities that enable Deep Learning models to solve tasks that Machine Learning models could never solve.

Through this continuous learning process, it would have the capability to detect different kinds of defect structure even though the defect structure is small and tiny. Thus, the deep neural networks were constructed and inputted into the AOI system for a better defect classification and evaluation (Ongsulee, 2017). A graphics processing unit (GPU) card was required to deal with huge amount of training set data for the classification of defect structures for different kinds of electronic components. A faster training process could be done with the help of GPUs.

2.2 Object Detection Method

There has been a lot of research done in object recognition using the old computer vision method. One of the traditional methods is by using the sliding window detector to detect the objects. Different size of window is used in order to find the location of object. After that, the feature is extracted using Histogram of Oriented Gradients, HOG feature extraction method and continued by using SVM classifier to classify it. This method is computationally expensive, and it is very slow as it need different scale of windows and to slide it step by step.

The accuracy for this method is significantly lower compare to the deep learning-based methods. The deep learning-based methods are divided into two categories which one is two stage detection method and the other is the unified detection method. Two stage detection methods consist of RCNN, Fast RCNN and Faster RCNN methods and for the unified detection method, it consists of YOLO and SSD methods. There are a few major concepts that are used in both techniques and it will be explained in the following.

2.2.1 RCNN (Regional-Based Convolutional Neural Network)

RCNN method is proposed by Ross Girshick et al. where this method used selective search to extract only 2000 regions of interest from an image. Selective search uses local cues like intensity colour, texture and measure of insideness to identify the region of interest. In this RCNN, selective search was done by generating initial sub-segmentation in order to have many candidate regions. Then, by using the greedy algorithm to recursively combine those similar regions into one. Next, the generated regions are used to produce the final candidate region of proposals. These candidate regions are fed into a CNN to continue with the feature extraction process. Lastly, by using the SVM method it would classify the extracted regions into certain classes. The flow diagram of RCNN is shown in figure below. There are several disadvantages using this RCNN method for object detection. First, the time used for training is huge as it need to classify those 2000 regions proposal in an image. Besides, it is similar for testing, it also requires long time to complete its testing process. Hence, it is not suitable to use in real-time application. Selective search is a fixed algorithm. It does not consist of any learning process. It might lead to generating bad candidate regions.

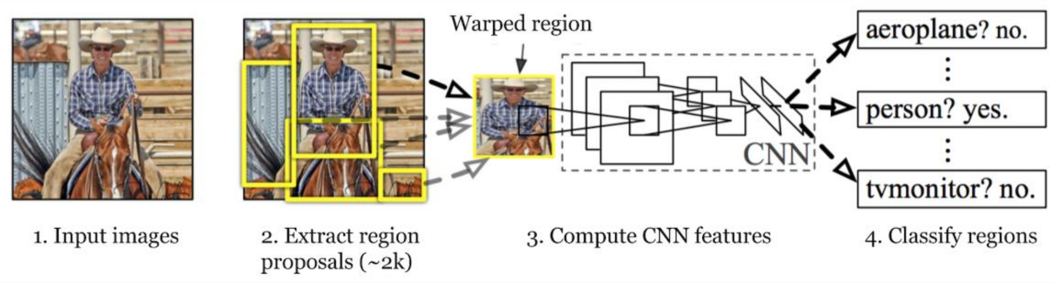


Figure 2-2: The architecture of R-CNN. (Girshick et al., 2014)

2.2.2 Fast RCNN (Fast Regional-Based Convolutional Neural Network)

Fast RCNN methods is almost similar with the approach of RCNN. The difference between Fast RCNN and RCNN is just only we change the sequence of inputting image to CNN. For the Fast RCNN approach, it feed the image into the CNN first to generate the convolution feature map. Then, it just identifies the region of proposals using selective search. Fast RCNN consist of an additional ROI Pooling layer to reshape the region of proposals into a fixed size. Lastly, it fed into a fully connected layer, most probably would be the SoftMax layer to perform the classification process. Refinement process for the bounding boxes will also be performed. The overflow of diagram for Fast RCNN are shown in Figure 3 below. Compare to RCNN, Fast RCNN has the benefit of the convolution is done only once per image rather than those 2000 region of proposal which will take a long time to run.

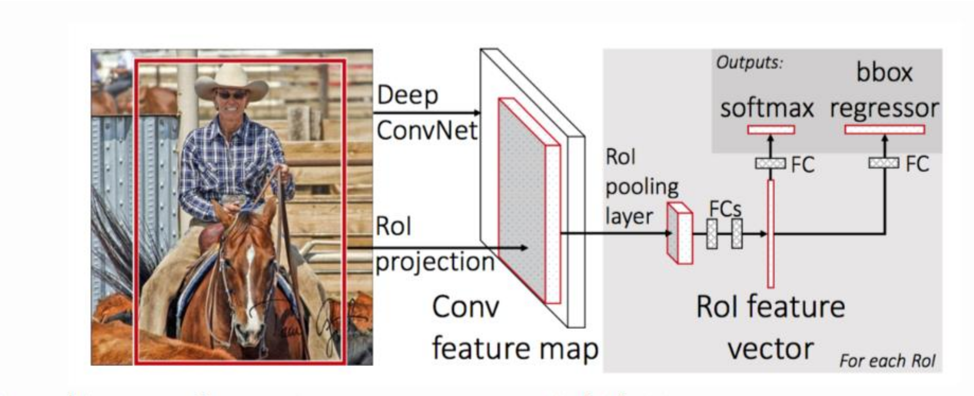


Figure 2-3: The architecture of Fast R-CNN. (Girshick, 2015)

2.2.3 Faster RCNN (Faster Regional-Based Convolutional Neural Network)

Faster RCNN is an improved method of previous Fast RCNN and RCNN. Faster RCNN does not use selective search method to find those candidate regions. Selective search method is a slow and time-consuming process. Faster RCNN allows the network to learn how to find the region proposals. The figure below shows the process in Faster RCNN method.

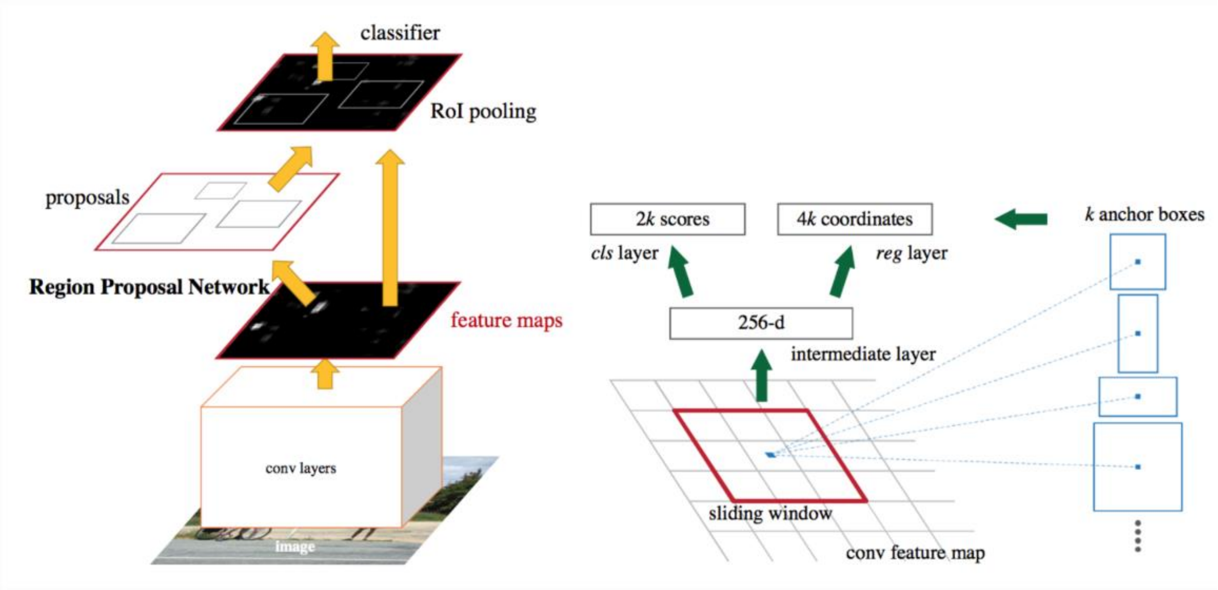


Figure 2-4: An illustration of Faster R-CNN model. (Ren et al., 2016)

Like Fast RCNN, image was first inputted into a CNN first to generate the feature map. Later, the generated convolution feature map will be into a region proposal network. Region proposal network would find the region of interest but there is no feature extracted. In the ROI Pooling layer, the region of interest found would match with the feature map. It will be resizing into a fixed shape. The resized region of interest would be input into a fully connected layer for classification. The bounding box is then refined in order to eliminate duplicated boxes.

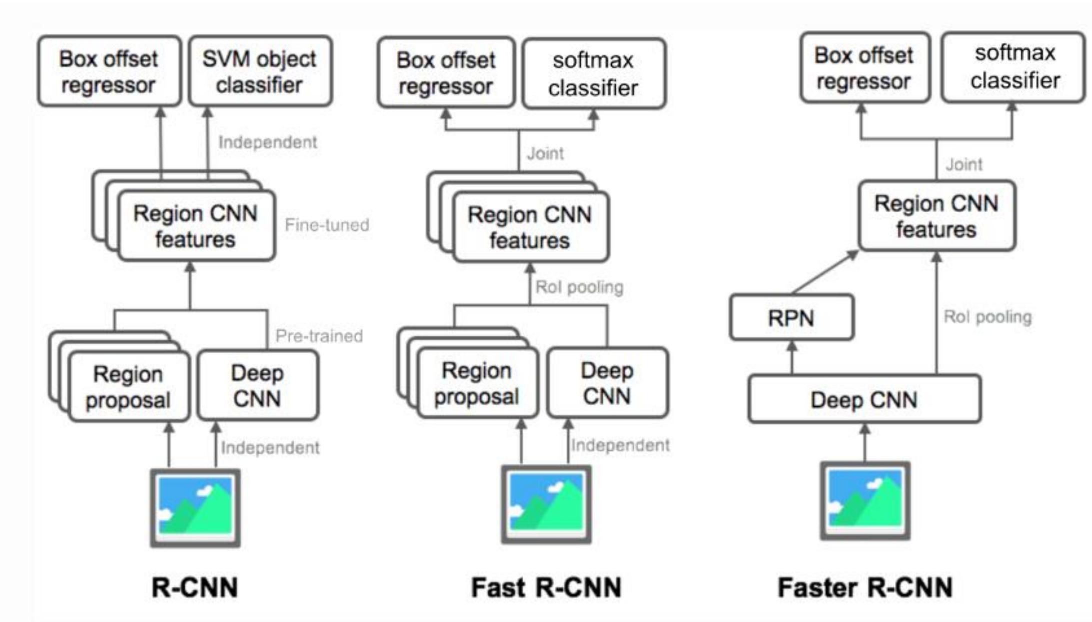


Figure 2-5: Comparison between R-CNN, Fast R-CNN and Faster R-CNN (Wang, 2019)

2.2.4 You Only Look Once (YOLO)

YOLO is an object detector which uses features learned by a deep convolutional neural network to detect an object. Unlike two-stage detectors like RCNN, YOLO does not use region proposal method. YOLO divides image into grid unit instead to detect object.

It is a method which look at the entire image. Image was split into S x S grid. Each grid would contain number of bounding boxes. For each bounding box, it has its own class probability. Single convolutional neural network would predict the bounding boxes and the class probabilities for all these boxes. Therefore, the detection speed is faster than traditional method. If the bounding boxes confidence score and class probability is above the value of threshold set, then the bounding box will be used to locate the object.

However, YOLO only predict one type of class in each grid. Hence, it would be struggling in detecting a small object or very close object. Detection accuracy is low and localization errors are huge because of the processing of the grid unit.

To solve the issues, YOLOv2 has been introduced. YOLOv2 improves the detection accuracy compared to YOLO by applying anchor box and using batch normalization for the convolution layer as well as fine-grained features. However, the detection accuracy is still low for small objects. Hence, YOLOv2 is not qualified enough applied in self-driving car, where requires high accuracy for small objects such as traffic sign.

YOLOv3 has been finally proposed in 2018 to solve the problems of YOLOv2. To improve detection accuracy, YOLOv3 is constructed of convolution layers and deep network, as shown in Figure 2.6. Residual skip connection is applied to solve the vanishing gradient problem of deep networks. Concatenation and up-sampling method are used to maintain fine-grained features which is important for detection in small object. In order to detect varied size of objects, theory of feature pyramid network is applied in YOLOv3 by detecting object in three different scales.

The flow of detection is shown in Figure 2.6. When an image is input into the YOLOv3 network, information is output from three detection layers. The predicted results of the three detection layers are combined. The results is then processed by using NMS (non-maximum suppression). Lastly, YOLOv3 has determined the final detection result.

The speed of detection of YOLOv3 is still as fast as YOLOv2. This is because YOLOv3 is a fully convolutional network which is consisted of small-sized convolution filers of 3 × 3 and 1 × 1. Hence, YOLOv3 is qualified for self-driving applications in term of its speed and accuracy.

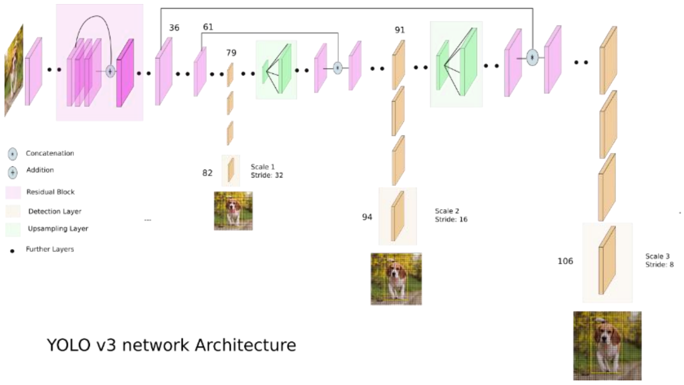


Figure 2-6: YOLOv3 Architecture (Choi, Chun, Kim and Lee, 2019)

As shown in Figure 2-7, YOLOv3 process the image grid or pixels through feature map and output the bounding box coordinates, objectness score and class scores. The objectness score is the probability of certain object present in the bounding box. The class scores are probability of certain category of the object presented in the bounding box.

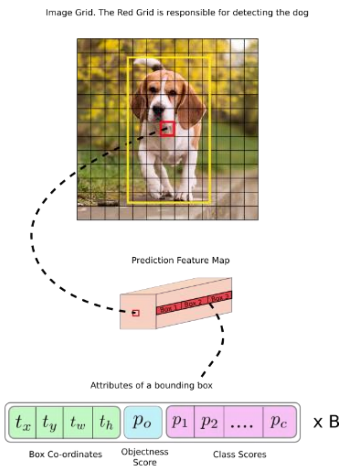


Figure 2-7: Output and Input of YOLOv3 (Choi, Chun, Kim and Lee, 2019)

Four coordinates for each bounding box t_x, t_y, t_w, t_h are predicted by the YOLOv3 network. As shown in Figure 2.8, if the bounding box has width and height p_w, p_h and the cell is offset from the top left corner of the image by (c_x, c_y), then the predictions correspond to:

$$b_x = \sigma(t_x) + c_x$$

$$b_y = \sigma(t_y) + c_y$$

$$b_w = p_w e^{t_w}$$

$$b_h = p_h e^{t_h}$$

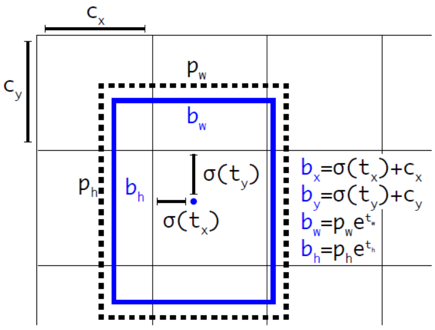


Figure 2-8: Bounding box with location prediction and dimension (Redmon and Farhadi, 2018)

In YOLO v2, a deep architecture Darknet-19 is used. It is a 19-layer network for object detection with 11 more layers. Therefore, it has total 30-layer architecture. However, YOLO v2 face difficulty when detecting small object. This is because lack of fine-grained features since the input is down sampled by the layers. Besides, the architecture of YOLOv2 has no skip connections, no residual blocks as well as no up-sampling.

As the improvement, YOLO v3 uses a Darknet-53 ad its architecture as shown in Figure 2-9.

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

Figure 2-9: Darknet-53 (Redmon and Farhadi, 2018)

The importing feature of YOLOv3 compared to YOLOv2 is that bounding boxes are predicted in three different scales. In each scale, the features of objects are extracted, which is similar to technique in feature pyramid network. This technique allows fine-grained information from previous feature map and unsampled features respectively. Thanks to detections at different layers, the problem of detecting small objects in YOLOv2 is solved.

In YOLOv3 architecture, the 52 x 52 layer is responsible to detect small object, the 26 x 26 layer detects the medium-sized object, and the 13 x 13 layer detects large object. As an example, if the input is an image of size 416 x 416, YOLOv3 will predicts ((52 x 52) + (26 x 26) + 13 x 13)) x 3 = 10647 bounding boxes. Nevertheless, there is only one object in the image, only one bounding box is needed. NMS technique is used to solve the problem.

NMS (Non-Maximum Suppression) is used in many computer vision algorithms. It is a class of algorithms to select one bounding box out of many overlapping entities. The selection criteria can be chosen to arrive at particular results. Most commonly, the criteria are some form of probability number along with some form of overlap measure. For example, in Figure 2-10, there are 3 bounding boxes of the red grid cell detect the same object, after processing NMS, it will be only one bounding box with most accurate coordinate.

Figure 2-10: Non-Maximum Suppression (Sambasivarao, 2020)

2.2.5 Single Shot Detector (SSD)

SSD achieve good balance between the speed and accuracy performance. SSD used auxiliary convolutional layers to extract features at multiple scales. These extracted feature maps will be input into a convolutional kernel to predict the bounding box and classification probability. The score of the class probability and 4 offsets (coordinates of bounding box) is computed (Biswas et al., 2019). Those scores which exceed the threshold point would be use as the final box to locate the object.

As its name proposes, the SSD network decides all bounding box probabilities in single shot. It is a fast detector model. However, the accuracy of detection may be lower (Biswas et al., 2019).

As explained earlier, R-CNN use region proposal network to create regions of interest and use convolutional layers to characterize the regions. SSD does the two task in the same time, which is known as "single shot". Therefore, SSD has the faster speed than RCNN family. This is the reason SSD is suitable to be utilised in embedded device or smartphone.

Generally, SSD will be utilized together with the MobileNet model. It is normally utilized in mobile applications or in other resource limited devices such as Raspberry Pi.

There are two type of deep neural networks which are base network and detection network. MobileNet are base networks. Base network offers high level features for detection. There is a fully connected layer at the end of this networks, it can be removed and replaced with detection networks, like SSD to achieve higher accuracy of detection.

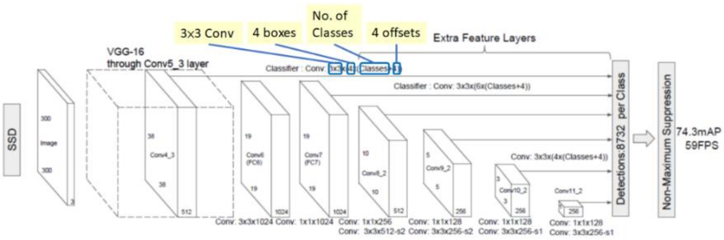


Figure 2-11: Network Architecture of SSD (Tsang, 2020)

2.2.6 Cloud AutoML Vision Object Detection

In order to permit engineers with restricted machine learning knowledge and ability to prepare good models, Cloud AutoML is introduced. It depends on state-of-the-art transfer learning and neural architecture search technology of Google (Cloud AutoML Vision Object Detection documentation | Cloud AutoML Vision Object Detection | Google Cloud, 2019). It can train custom machine learning models that are explicit to our needs, with the least exertion and machine learning expertise.

Cloud AutoML is completely coordinated with Google Cloud services, predictable method to get to the whole Google Cloud administration line, including storing our training data in Cloud Storage (Cloud AutoML Vision Object Detection documentation | Cloud AutoML Vision Object Detection | Google Cloud, 2019). To create a prediction on our trained model, we can utilize the current Vision API by using adding a parameter for custom model or Cloud ML Engine's online prediction service.

User can likewise utilize the self-labelled data to train a custom model. But fee will be charged for all the services, unlike the object detection mentioned before.

To embed the models into Android phone. Firebase ML Kit is introduced. In this modern era, consumers expect mobile applications to not merely be instinctive, yet additionally have the option to give incredible highlights. Therefore, machine learning has become important to mobile development.

Engineers are progressively depending on machine learning to upgrade their application's client experience and just with calibrated machine learning models can convey those ground-breaking highlights to charm their clients. But what if we don't have machine learning expertise? What if we have an ML model, but don't want to deal with hosting it or serving it to a mobile device?

To solve the problem, Firebase introduces ML Kit, a machine learning SDK. It could bring powerful machine learning features to our app. No matter we are freshmen in machine learning, or we are an experienced machine learning developer, ML Kit make machine learning become as simple as it could be. ML Kit accompanies a lot of readily APIs, concentrated on basic cases, perceiving content, identifying faces, perceiving tourist spots, examining barcodes, and marking images. We can simply give information to the library and it will give all of us sorts of data. ML Kit APIs can run on gadget or in the cloud, counting on the functionality, and some offer two options.

APIs which is on-device can process data without accessing to network. Meanwhile, APIs which is cloud-based use the strength of Google Cloud Platforms associated with machine learning technology to provide a higher level of accuracy. We can always upload our own TensorFlow Lite model to the Firebase console, ML Kit will deal with facilitating it and serving it to client. ML Kit acts as an API layer to our custom model, making it simple to run and utilise in our mobile application. Machine Learning is on our fingertips.

In summary, ML Kit is a SDK in mobile which is able to bring Google's machine learning to Android and iOS applications in a simple bundle. Regardless of whether we are new to this field or expertise in machine learning, the functionality can be easily created by typing in only a couple of lines of code. There are several APIs that can recognize faces, text and so on. However, if we need to some feature which are not supported by the APIs, such as recognizing different types of fruits from an image, then we need to train our own model. This is how Cloud AutoML Vision Object Detection can help user.

2.3 Cloud-based system

Cloud-based means services and applications made accessible to clients through the Internet from a distributed computing supplier's server. In other word, cloud computing alludes to storing and accessing data over the Internet rather than hard drive of computer. There are a lot of cloud-based system platform available in market such as Firebase or Microsoft Azure. In this project, Firebase will be discussed.

2.3.1 Firebase

Firebase is a Backend-as-a-Service (BaaS) grew up into an app-development platform on Google Cloud Platform.

Firebase a Realtime Database. In future, most databases expect us to make HTTP calls to get and adjust our data. At the point when we associate our application to Firebase, we are interfacing through a WebSocket. WebSockets are much quicker than HTTP. Our information matches up amazingly through that solitary WebSocket as quick as our customer's system can convey it. Firebase sends us new information when it is refreshed. At the point when our customer spares a change to the information, every single associated customer gets the refreshed information right away.

It is likewise a file storage gives a basic method to spare records straightforwardly from the customer to Google Cloud Storage. In order to protect Cloud bucket, Firebase Storage has its security system, while providing detailed privileges to authenticated clients.

In short, Firebase is a completely managed framework for creating Android, iOS, and web applications that offer automated information synchronization, encryption, messaging, file storage, analytics, and more. Starting with Firebase is a best and simple approach to assemble or model portable backend services.

However, Firebase is not chosen as the database server for this project because it has less online tutorial.

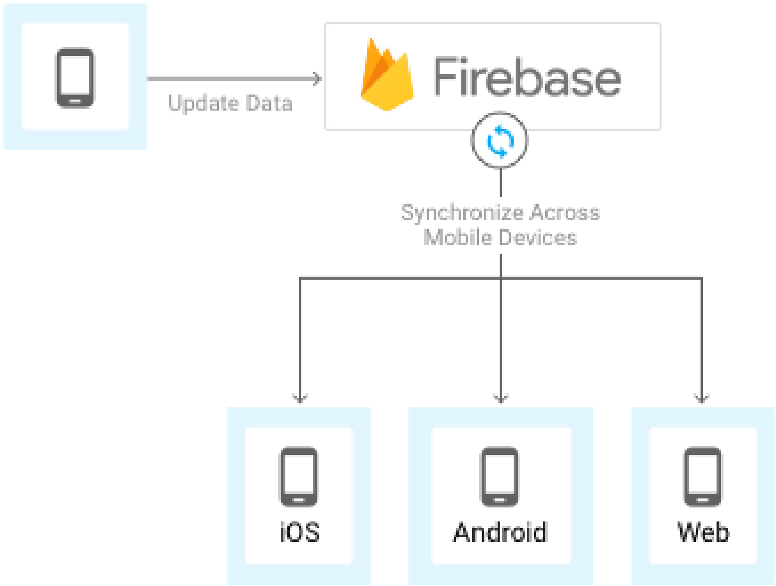


Figure 2-12: The Relationship between Cloud and Hard Drive (Mobile app backend services | Solutions | Google Cloud, 2019)

2.3.2 Hypertext Pre-processor (PHP)

PHP is a popular general-purpose scripting language that is especially suited to web development. There are several reasons why PHP is famous and widely used. firstly, it is open source and can easily learn from internet as there are many online tutorials of learning PHP. Besides, PHP has simple installation and cross platform availability. No matter which OS we are using, we can easily start with PHP development.

PHP is language created for web development and unlike C# or Java. It is all about the web. There is hosting available for PHP. Majority of hosting provider is supporting PHP while other programming languages support is not often found on every hosting.

Besides PHP, there is another option which is ASP.NET. It is open-source server-side web-application framework designed for web development. It gives user much easier to develop web pages rather than Java or PHP. Because asp.net is not language as Java and PHP, it's a framework, which makes programmer life as a developer easier. However, there are several reason PHP is chosen to complete the project.

PHP has fast load time and results in faster site loading speeds. PHP codes runs much faster than ASP because it runs in its own memory space while ASP uses an overhead server and a COM based architecture.

In working with PHP, most tools associated with the program are open source software, such as WordPress, so user need not pay for them. As for ASP, user might need to buy additional tools to work with its programs.

PHP has less expensive hosting. ASP programs need to run on Windows servers with IIS installed. Hosting companies need to purchase both of these components in order for ASP to work, this often results in a more expensive cost for monthly hosting services. On the other hand, a PHP would only require running on a Linux server, which is available through a hosting provider at no additional cost.

PHP is flexible for database connectivity. It can connect to several databases the most commonly used is the MySQL. MySQL can be used for free. If ASP is used, MS-SQL, a Microsoft product must be purchased.

Table 2-1: Comparison between ASP.NET and PHP

The Basis of Comparison Between ASP.NET vs PHP	**ASP.NET**	**PHP**
Type	Web application framework created by Microsoft	Server-side scripting language created by Rasmus Lerdorf
Support	Large to medium size enterprise applications	Small to medium sized web solutions
Cost	License cost attached	Freely available all over the web
Solutions	More focused on Security and functionalities	More focused on client facing, user interfaces
Community	Dedicated community with the fewer developers	Large size community since its open source
Security	Highly secure	Less built-in security feature than .NET
Speed	Decent speed, fast enough for desktop application	Not suitable and slower for desktop application
Customization	Less prone to customization	Allow customization causes bugs, thus poor coding than .NET

There is also another option which is Firebase. However, PHP is suitable in web while Firebase backed by google cloud base backend mostly used for mobile apps. If we want to launch web version with mobile apps, PHP is a great tool which can handle both perfectly. In short, PHP is chosen to work with the database server to complete this project.

2.3.3 000Webhost

Whenever user visit a website on the Internet or use an app on mobile device, they are effectively sending a request and getting some response. They expect the website or app to give them a response anytime.

However, this means that in order to be accessible anytime, a website which is basically just a collection of files needs to run on a computer that is constantly on and has an uninterrupted Internet connection. This is a tough task for normal computer, so websites use specialized, powerful computers called servers instead.

Servers are not cheap. Moreover, since they are working non-stop, servers consume a decent amount of electricity, not to mention the fact that they require regular maintenance by qualified specialists.

Figure 2-13: How Server Works

If a server costs money, and its operation costs money, and its maintenance costs money. Anyone offering to host website for free is either a charity or has other sources of income to cover its expenses. There are two popular ways to do that which is placing ads on the hosted websites and offering paid upgrades and extra services, while severely restricting the range of features available to the free users.

000Webhost can act as a convenient testing ground for personal and other small-scale web projects, yet its limitations and safety issues make it almost unthinkable for hosting any serious website. It does provide multiple benefits. User can easily create a fully functional website.

000Webhost has had its share of hacks and security issues a couple of years back. However, 000webhost has overcome these issues and has incorporated mandate security features even for their free hosting. 000Webhost uses an advanced firewall and incorporates DDoS protection. It provides regular updates and has continuous monitoring. It supports instant backup to ensure user data has a backup always. While it does not have too many out of the box security features, still sufficiently good security support.

Generally, most users may not expect a free hosting service to have too many features. Contrary to this assumption, 000webhost has a good feature list. It supports PHP and MySQL with its cPanel. They get a free website builder. 000webhost supports multiple other PHP features.

000Webhost provides the user with a one-click auto-installer that can be used to install other applications such as WordPress, Drupal, and around 50 other scripts.

With the free hosting, user get 10 GB bandwidth, 1 GB Disk space, free domain name hosting, free website builder, 2 websites, WordPress auto installer and instant account activation with no added cost.

Besides, Managing Database is easy, and user can add/delete database very quickly. The options are self-explanatory, and user may not require expert help to manage databases.

However, there are also some limitation. For starters, there is the monthly bandwidth cap, which means there is a maximum amount of times visitors can view your website. This quickly puts a ceiling on your audience growth and the type of content you can use. Videos and high-quality images are not able to be the contents.

Next, there is the lower availability, which means the hosting provider does not guarantee that its servers will work 24/7/365, but more like 23/7/365. In other words, up to 5% of the time your website will be offline.

The website will not be fast, it will be quite slow by any standard. This makes sense because there is no incentive for the hosting provider to allocate too much of a server's resources such as memory and computing power to free accounts.

Finally, free hosting does not allow using any custom address for website, for example website.com. Only a sub-address of a specific domain name fixed by hosting provider, for example website.000webhostapp.com.

In short, 000Webhost is a great helper to create a free server for beginner. Getting a free host is always an exciting deal, especially when we are running low on budget. Having said this, since this would be accessed by multiple users, it is important that the hosting service is easy to use.

Chapter 3 System Design

3.1 Design Specifications

The OS platform of the smart phone is Android mobile operating system 6.0 “Marshmallow”. The programming language used is Android, Java, PHP and JavaScript. The database of the system in MySQL. JSON will be used as web service.

There are also third-party libraries included in my system to complete the project, which is Google Maps JavaScript API and TensorFlow API. The internet browser using is Google Chrome.

There are two main modules in the project which is Detection Module and Map Module. Google Maps JavaScript API is used in Map module whereas TensorFlow API is used in Detection Module.

3.1.1 Obstacle detection Mobile Application

Object detection task is one of the most popular example of artificial intelligence system that used to identify and classify objects. Inside the object detection task, it consists of deep convolutional neural networks as a classifier. This classifier is work together with other object detection technique to detect the region of interest of a image. There are many different types of open source frameworks such as TensorFlow, pytorch, Caffe and Keras are available online. Many researches had been done using TensorFlow by those huge company such as Nvidia, Uber and Snapchat in detecting object or face. TensorFlow is consider as low-level language which is more flexible in design.

In this project, we use the TensorFlow Object Detection API which is an open source framework for object detection related task to identify and classify different types of objects when we are driving.

TensorFlow is an open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in machine learning and developers easily build and deploy Machine Learning-powered applications.

Fortunately, TensorFlow is able to be built in android phone. The simplest way to use TensorFlow on Android is to use Android Studio. There is another way to build TensorFlow in Android device which is building with Bazel and deploying with ADB (Android Debug Bridge) on the command line. Since I am more familiar with Android Studio, I choose the first method. Firstly, we need to install android studio by following the instructions on their website.

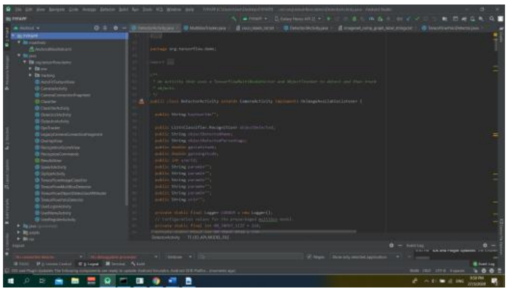


Figure 3-1: Android Studio

Then, clone the TensorFlow repository from Github:
https://github.com/tensorflow/tensorflow/tree/master/tensorflow/examples/android

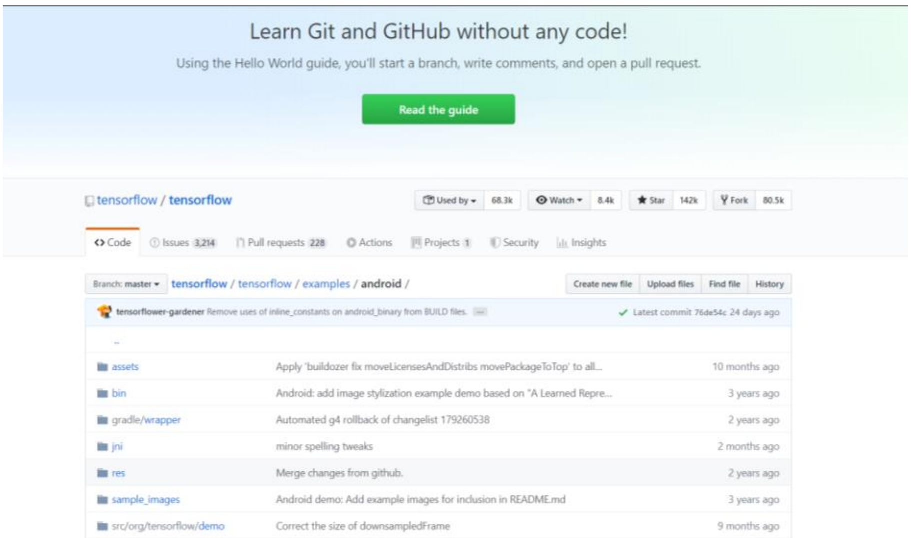


Figure 3-2: TensorFlow Github

There are four TensorFlow sample models in the Github which is TF Classify, TF Detect, TF Stylize and TF Speech. In this project, TF Detect is used as reference to complete the project. TF Detect aims to demonstrate an SSD-Mobilenet model trained using the TensorFlow Object Detection API to localize and track objects (from 80 categories) in the camera preview in real-time.

3.1.2 Datasets for Obstacle Detection

One important element of deep learning and machine learning at large is dataset. A good dataset will contribute to a model with good precision and recall. There are some household names commonly used and referenced by researchers such as COCO, PASCAL and ImageNet. In this project, COCO datasets are used as datasets of object detection module in the project.

The full name of COCO is Common Objects in Context. Based on full name, we could know that images in COCO dataset are taken from everyday scenes thus attaching "context" to the objects captured in the scenes. COCO was an initiative to collect natural images, the images that reflect everyday scene and provides contextual information. In everyday scene, multiple objects can be found in the same image and each should be labelled as a different object and segmented properly. COCO dataset provides the labelling and segmentation of the objects in the images. It is convenience for those machine learning developer. They can easily take benefit of the labelled and segmented images as their datasets.

There are many advantages of using COCO datasets because it contains 2.5 million labelled instances in 382,000 images (Lin, Hays, Maire and Perona, 2015). It has the most images among all the sources of datasets such as KITTI, PASCAL, ImageNet and so on. With the help of COCO, machine learning developer save the time of preparing and annotating datasets. Besides, object detection model can also perform better because of large number of datasets in COCO. As the more datasets we have, the higher the accuracy and precision of detection system.

According to the COCO research paper, there are total of 91 object categories in COCO. However, only 80 object categories of labelled and segmented images which is integrated and trained in the project. Table below shows the 80 objects in the COCO datasets. In other word, the object detection module is able to detect these 80 objects.

Table 3-1: Lists of 80 Objects in COCO Datasets (Lin, Hays, Maire and Perona, 2015)

airplane	bench	cake	dog	horse	pizza	sports ball	truck
apple	bicycle	car	donut	hot dog	potted plant	stop sign	umbrella
backpack	bird	carrot	kite	elephant	sandwich	suitcase	Wine glass
banana	boat	cat	tv	refrigerator	sheep	surfboard	zebra
baseball bat	bottle	chair	fork	motorcycle	skateboard	tennis racket	dining table
baseball glove	bowl	couch	frisbee	broccoli	skis	tie	toilet
bear	orange	cow	giraffe	parking meter	snowboard	traffic light	fire hydrant
bed	bus	cup	handbag	person	spoon	train	laptop
mouse	remote	oven	cell phone	microwave	keyboard	toaster	sink
knife	book	clock	vase	scissors	teddy bear	hair drier	toothbrush

3.1.3 Web and Database Development

One of the objectives of the project is to share the real-time information online with other users. In order to achieve the objective, a website is developed for user to upload and share the data. The language used to develop the website is PHP with the help of 000Webhost. It can act as a convenient testing ground for small-scale web projects.

Firstly, an account needs to be created at https://www.000webhost.com/. After logging in, a new website can be easily created by inserting website name and password.

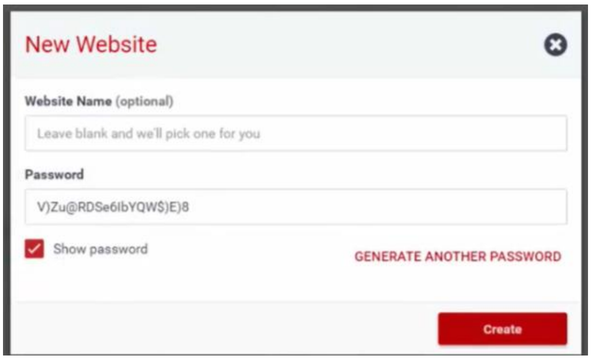


Figure 3-3: Inserting Website Name and Password

After that, three method of developing website is available to choose. In this project, “Upload Own Website” will be more suitable since the option allow to upload own prebuild website and files.

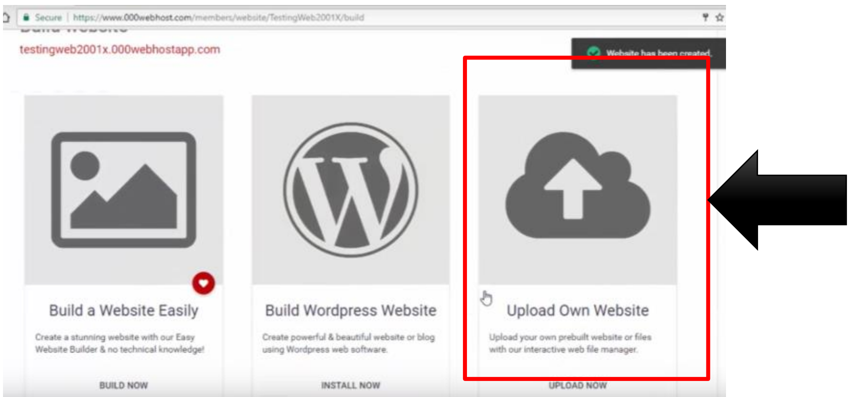


Figure 3-4: Choose to Upload Own Website

Then, there is a folder of “public_html”. The website can be easily created by uploading the prebuild website file into the “public_html” like Figure 3-5.

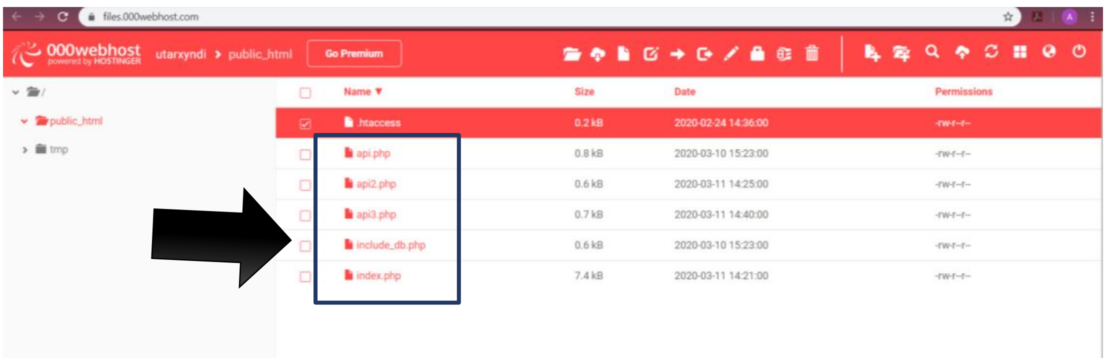


Figure 3-5: Upload Own File

Go back to the main menu, the status of the website will be running. The URL of website is showed in blue words. By coping the link and paste it in internet browser, the website can be easily accessed by any device in any time.

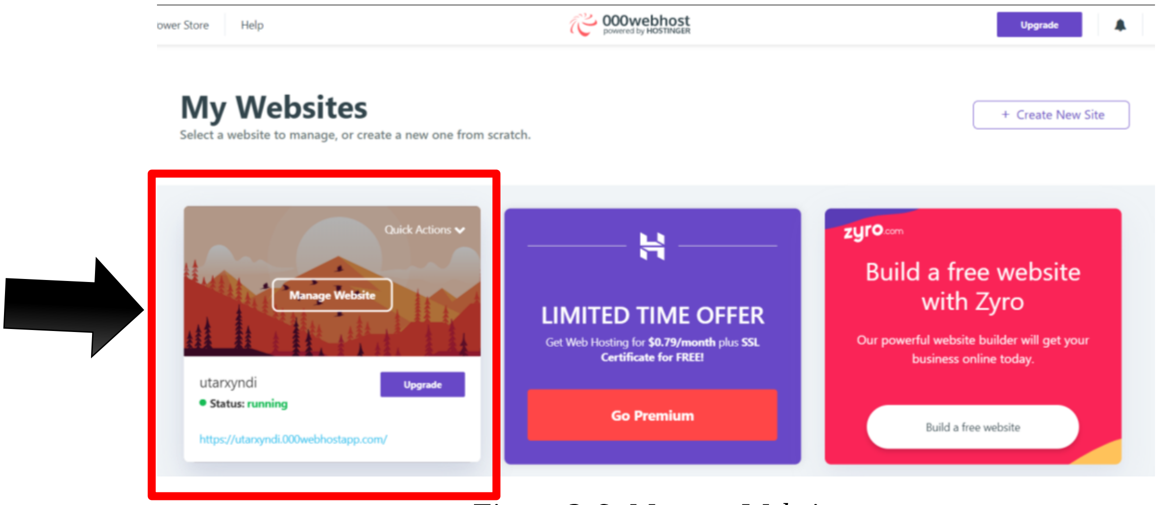


Figure 3-6: Manage Website

The website layout of the project is shown in Figure 3-7. This website will be displayed on the mobile apps for driver to check the obstacle in front.

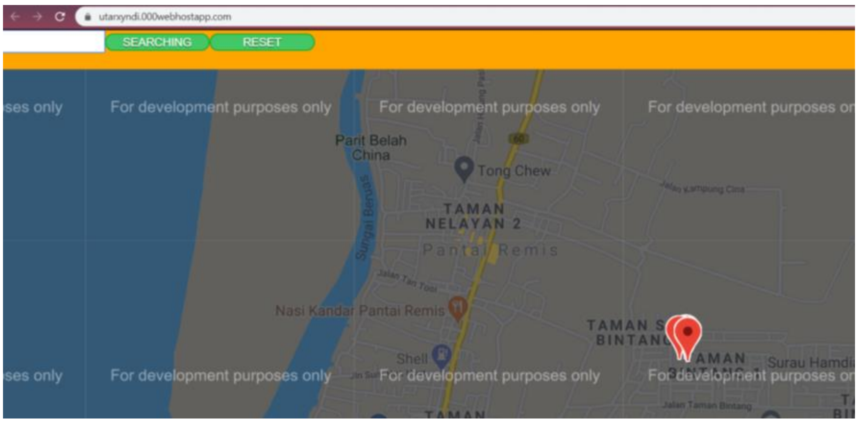


Figure 3-7: Website Layout

The website can be managed in dashboard. The details and usage of the website is shown in the dashboard. Since it is free version, there is certain limit of disk space quota, monthly bandwidth quota and so on. However, the quota provided by the free version is enough for the project to operate.

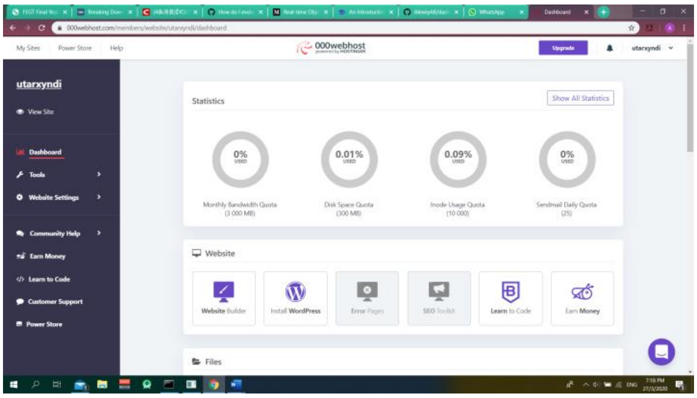


Figure 3-8: Dashboard of Website

Database of the website can be easily managed in dashboard. Besides, another benefit of using 000webhost is 000webhost offer free hosting with almost unrestricted PHP and MySQL support. There are two databases in the project, which is result of object detection system and the user registration system.

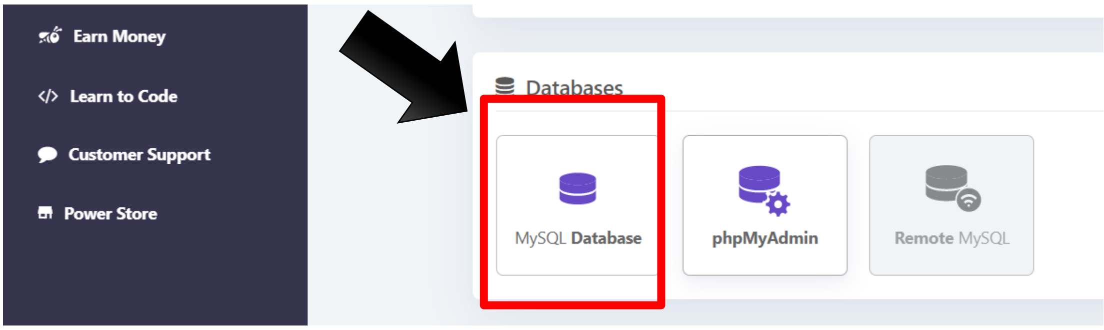


Figure 3-9: Choose MySQL Database

In the database table of detection, there are columns of the name of the object detected, the percentage of matching of the object, the location coordinate, the time of uploading, the user who upload the data and so on.

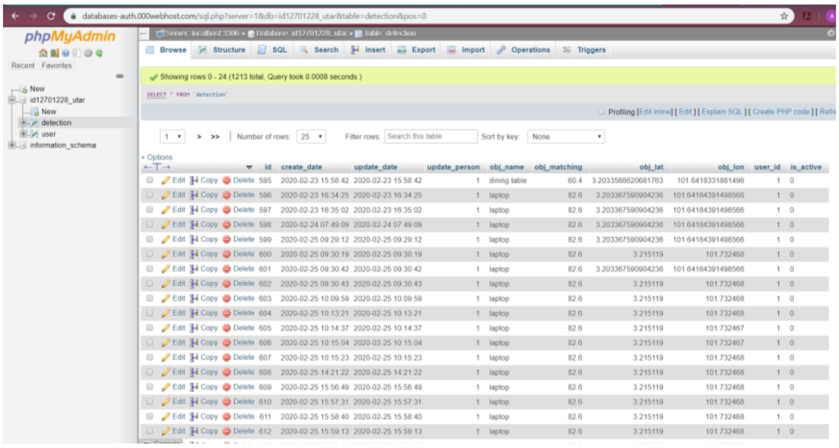


Figure 3-10: Detection Database

In the table of user registration, there are columns of user id for recording purpose, username and password.

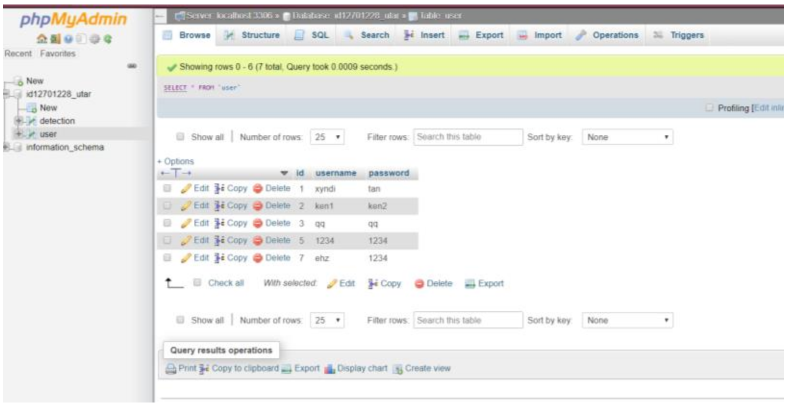


Figure 3-11: User Database

3.2 System Flow

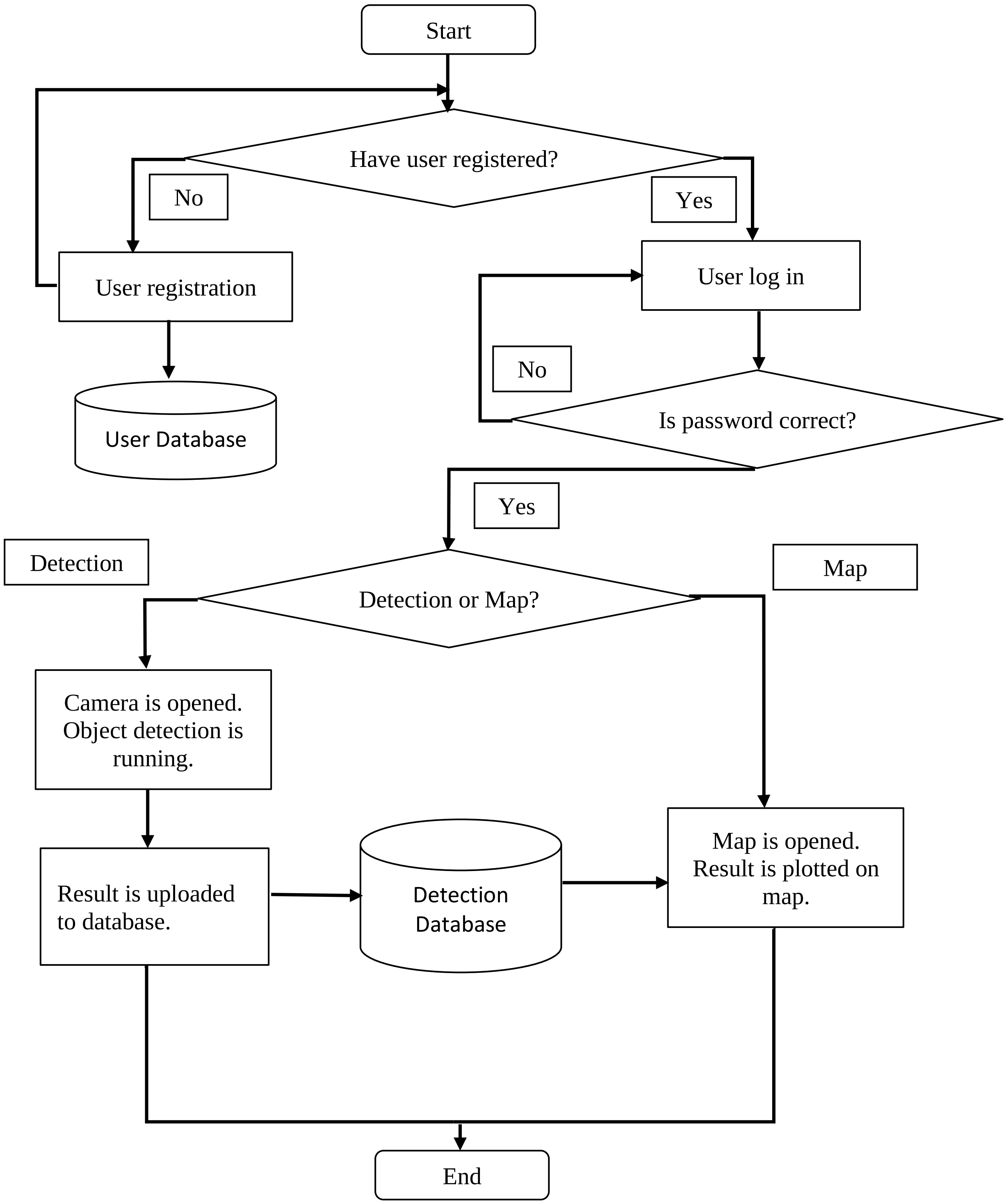


Figure 3-12: System Flow Chart

Since the system is designed for driver, an android smartphone needs to be set up and operated inside the car.

User need to register before using the apps by key in username and password. After user registered, user can log in the apps by entering the registered username and respective password.

After user log in, user can choose either Detection module or Map module. In Detection module, mobile application will start detecting and classifying the object detected in real-time. The 80 types of object can be detected. The list of objects is listed in Table 3-1. A bounding box will describe the target location. The name of predicted object will be shown above the bounding box. Once the object is being detected and classified, the result will upload to the database together with the details such as, username, location coordinate, name of the object, percentage of matching (confidence score), date and time of detection.

In Map module, there will be map with some marker on it. The maker indicates the object is detected in this location. When user clicked the marker, the detail of detection will be displayed such as the object name, confidence score, date and time of detection as well as the name of user upload the data. If there is no user has done detection before, there will be no maker on the map.

Besides, there is Search column at the top left corner of the layout. This allows user to search the location of certain objects on map. There is also Reset function to enable user to clear all the detection data in database. In other word, the Reset button is used to clear all marker on map.

Use Case Diagram

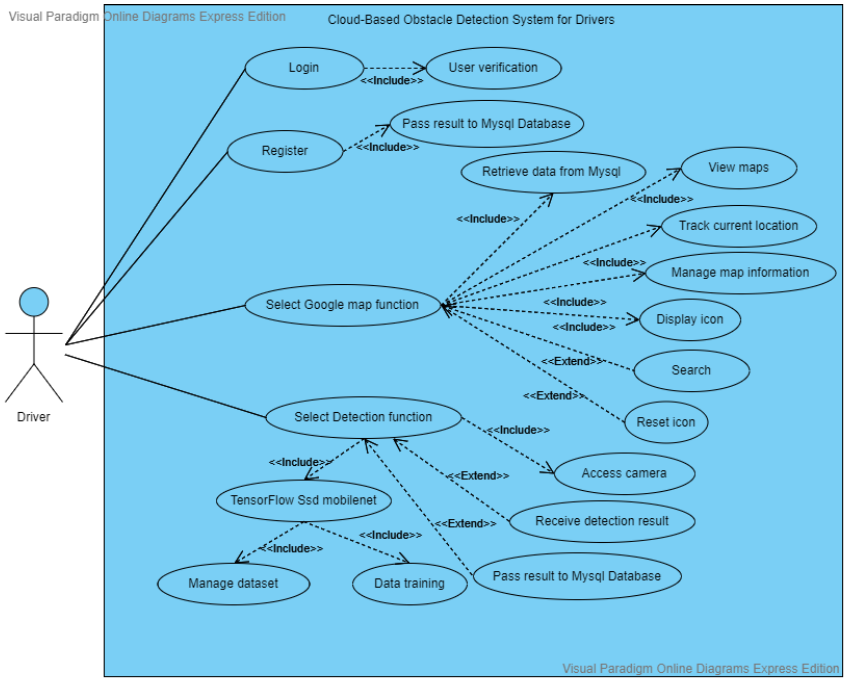


Figure 3-13: Use Case Diagram of Cloud-based Obstacle Detection System

Sequence Diagram

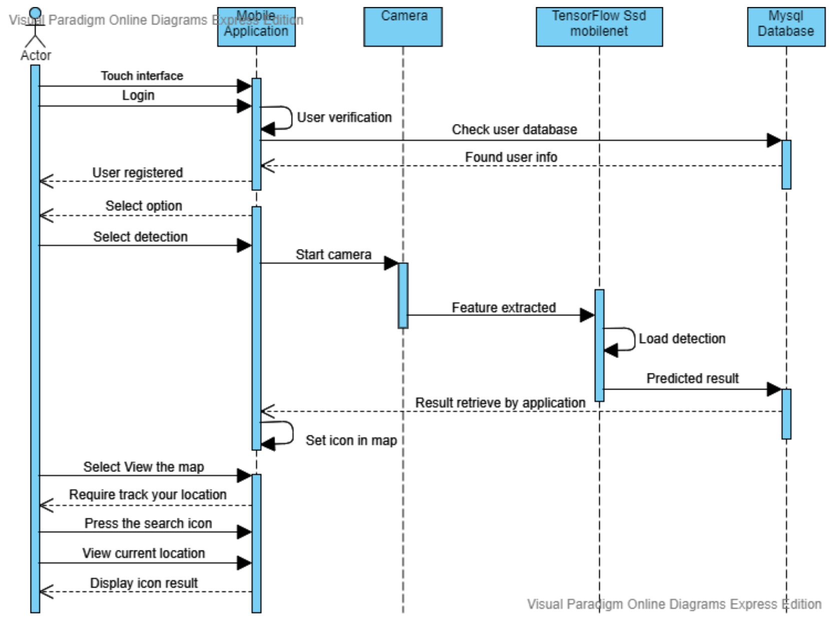


Figure 3-14: Sequence Diagram of Cloud-based Obstacle Detection System

Block diagram

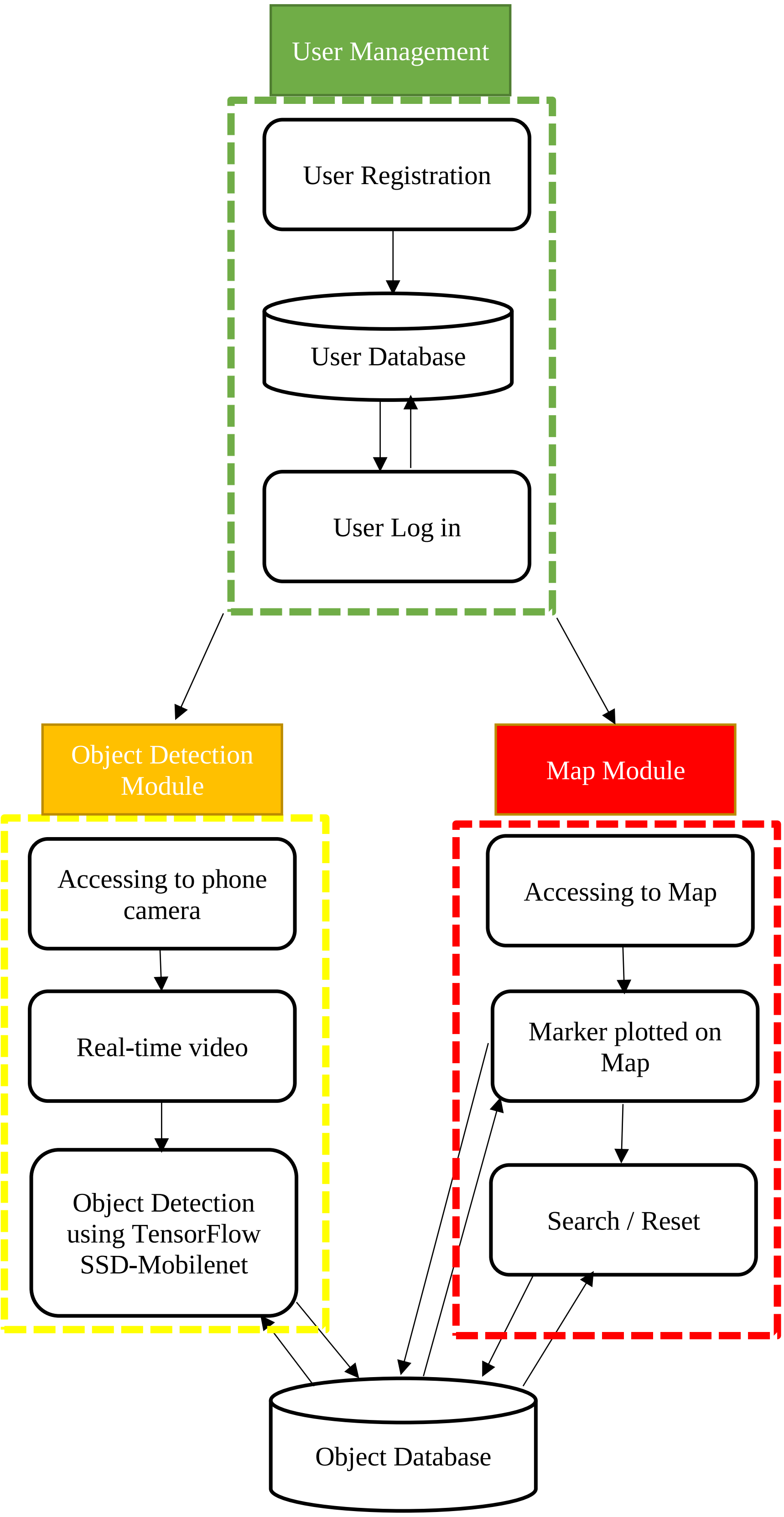


Figure 3-15: Block Diagram of Cloud-based Obstacle Detection System

The system is separated into 3 modules, to ease the arrangement of task which is User Management, Detection Module and Map Module.

In the User Management Module, the user must register for the first-time log in. After logging in into the mobile application, user can either go to the Detection Module or Map module.

In Detection function, the camera of Android phone will be activated. The mobile application will start detecting obstacles in front in real-time. The result will be shown with the help of bounding box and label. At the same time, the result and GPS location coordinate as well as user identity will be automatically uploaded to database server.

In Map Module, user can track their location and access to detected object on specific location with the form of map marker. The details of detected object can be viewed by clicking on the map marker. There are also Search function and Reset function in this module. Search function is to allow user to search for specific object on the map to find out their location. Reset function is to allow user to clear all the marker on map. In other word, it is to clear the data in database.

Chapter 4 Experimental Result

4.1 User Manual

4.1.1 User Management Module

The basic requirement for user is having an Android phone with minimum available internal storage of 200MB. The minimum OS requirement for android phone is Android 5.0 Lollipop (API 21).

After installing the APK file in Android smartphone, the apps can be launched by clicking the icon of “UTAR Xyndi”.

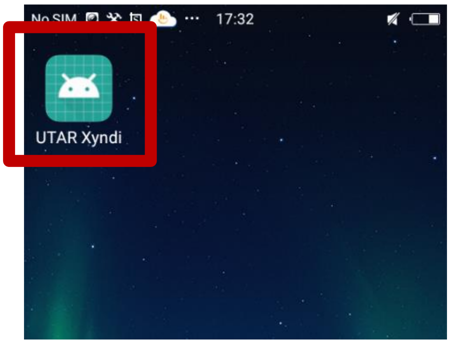


Figure 4-1: Icon of Mobile Application

There will be a Login Page where user needs to key in their username and password. However, this is only valid to registered user.

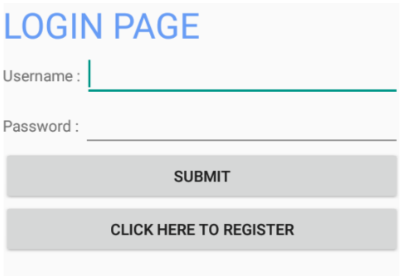


Figure 4-2: Login Page

When there is blank in username or password, error message will be shown at bottom. User is not able to proceed to next page.

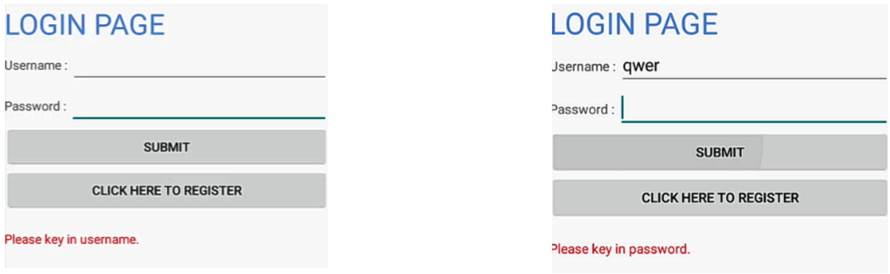


Figure 4-3 : Warning When Leaving Blank in Username and Password

When the user has not registered before logging in or there is wrong username or password, the warning of “Incorrect combination of username and password” will be displayed and user is not able to proceed.

In order to proceed, user must register before logging in. To register, user just have to click the column of “CLICK HERE TO REGISTER”.

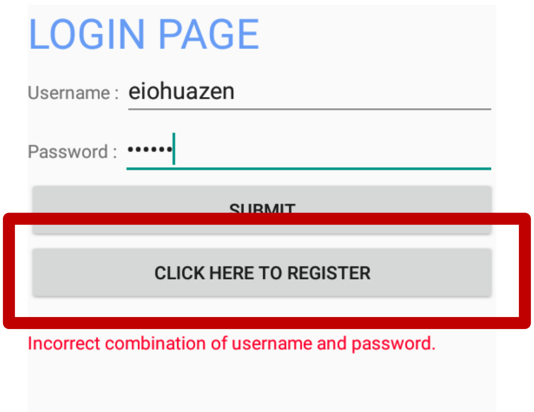


Figure 4-4: Incorrect Username and Password

After that, user will enter the Register Page then they just have to enter their username and password and click “SUBMIT” to complete registration.

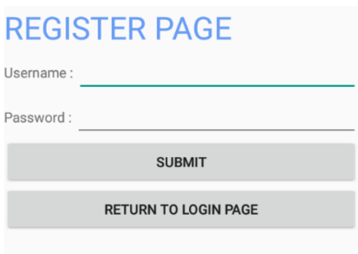


Figure 4-5: Register Page

After registration, there will be a message of “Registration success, please proceed to login page” displaying at the bottom.

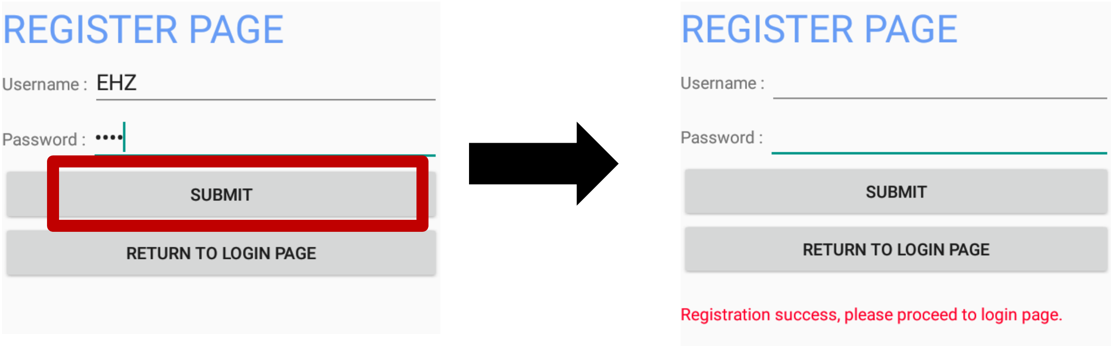


Figure 4-6: Registration Success

When user returns to Login Page, user will be successfully log in with their registered username and password and clicked “SUBMIT”. There will be “Login success.” at the bottom.

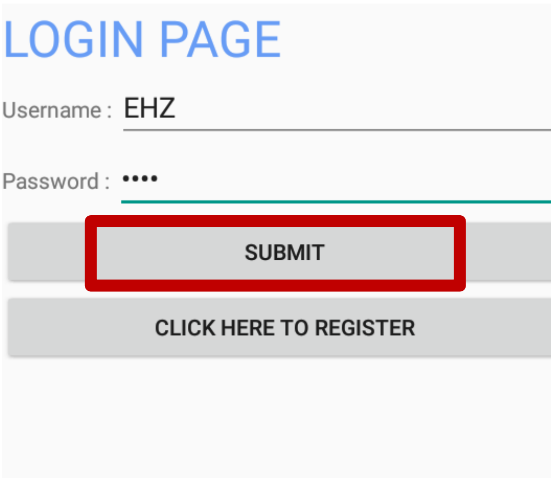


Figure 4-7: Enter Registered Username and Password

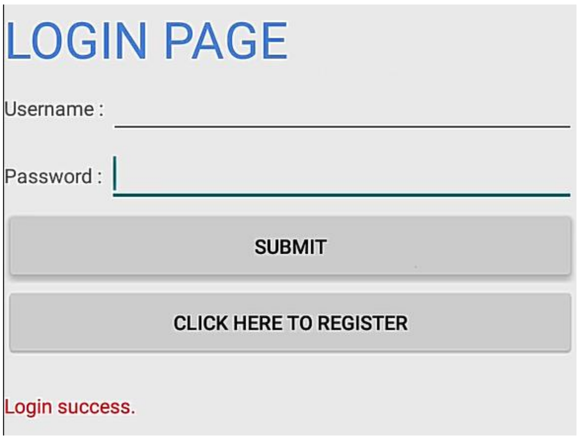


Figure 4-8: Login Success

After logging in, there will be two modules in main menu, which is "DETECTOR VIEW" and "MAP VIEW".

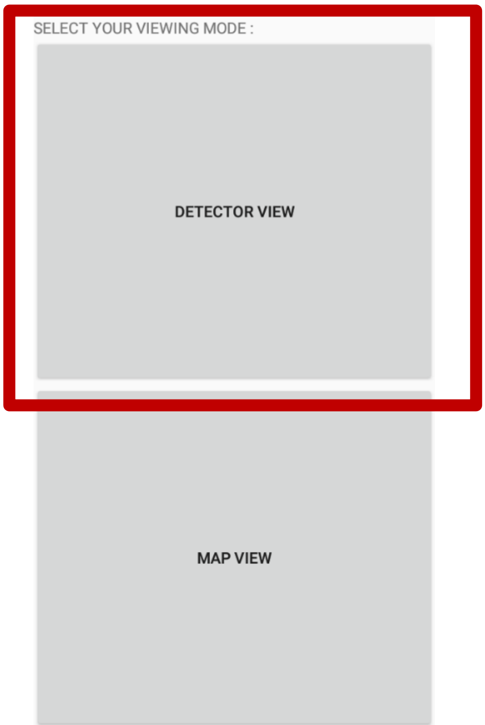


Figure 4-9: Select Detector View in Main Menu

If user starts with Detector View, the real-time obstacle detection system will be operating with the help of phone camera. When the system detects something, which is matching with its datasets, a bounding box will be pop up to enclose the object. The name of the object and the percentage of matching will be displayed at the bottom of bounding box.

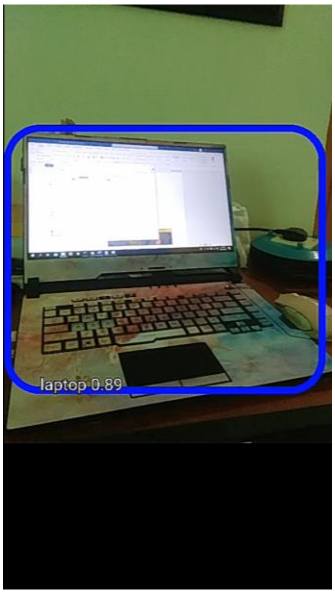


Figure 4-10: Detection View

Meanwhile, the result of detection system will be sent to the database in the web server which allowing data to be shared with other users. To validate the result, user needs to return to main menu and proceed to “MAP VIEW”

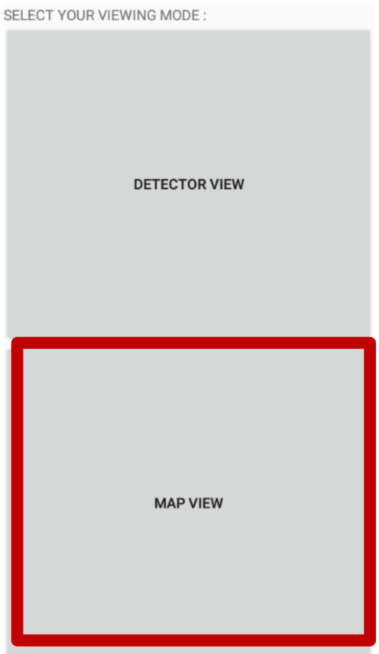


Figure 4-11: Select Map View in Main Menu

The result of detection such as the name of detected object, the percentage of matching, the date and time of detection, user who uploaded the data and GPS location coordinate will be sent to the database server. In the "MAP VIEW", the marker will be plotted on the coordinate where the user detected the object. When the user clicked on the marker, the details of detection will be pop up.

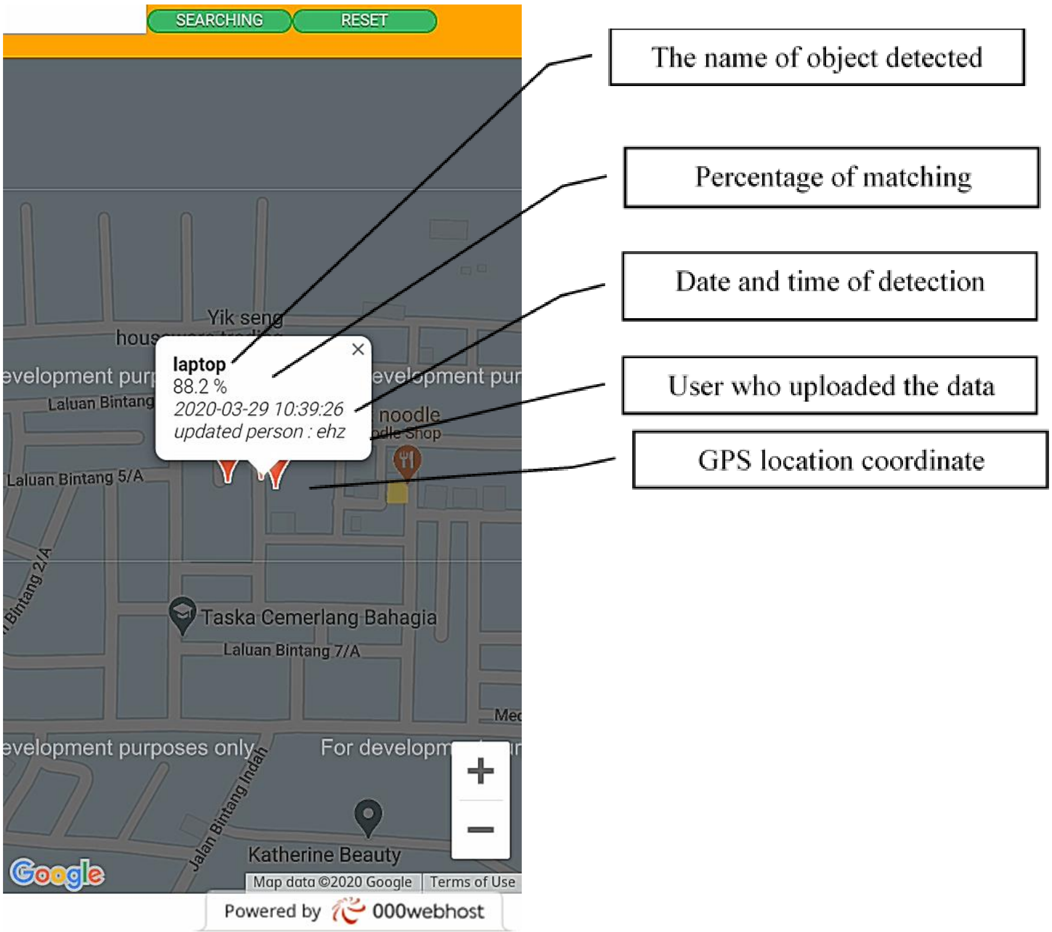


Figure 4-12: Map View

There is also "Searching" and "Reset" Button on the top of the page. "Searching" button allows user to key in the name of objects in the white column and search the location of the object on map. "Reset" button is for user to clear all the data and marker on the map.

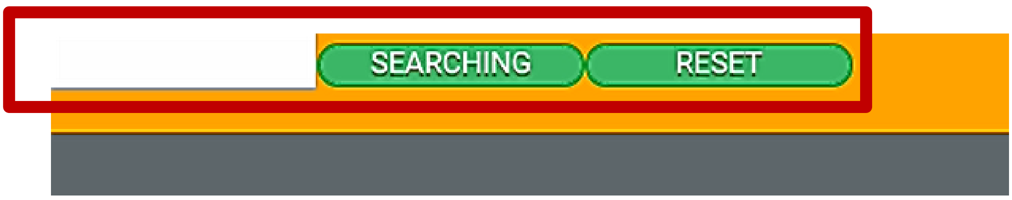
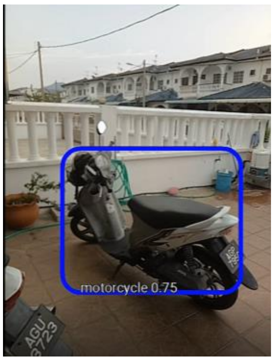


Figure 4-13: Searching and Reset Function

4.1.2 Object Detection Module

To evaluate the performance of the system, different objects are tested as shown in Table 4-1. The object tested in this part are car, motorcycle, person, traffic light, bicycle, potted plant and vase, dining table and truck. The observation, percentage of matching and average time taken to detect will be noted down.

Table 4-1: Detection Result of Different Object

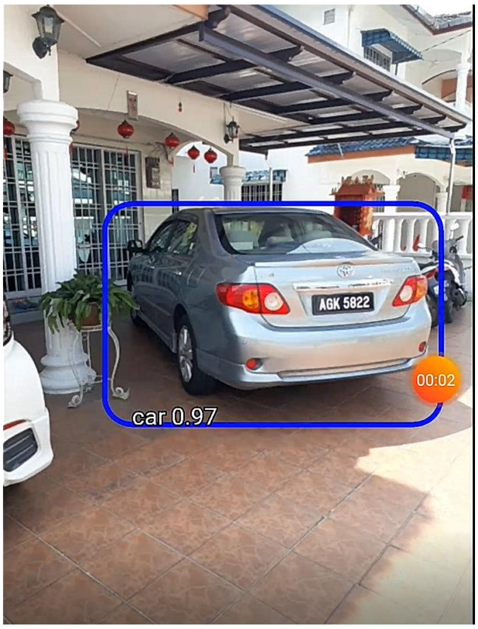
Types of object	Observation from system
(1) Car 	*Object detected* = car
	Percentage of matching = 98%
	Average time taken to detect = 0.5 seconds
	Observation: The result of detection for car is correct.
(2) Motorcycle 	*Object detected* = motorcycle
	Percentage of matching = 75%
	Average time taken to detect = 0.5 seconds
	Observation: The result of detection for motorcycle is correct.
(3) Person 	*Object detected* = 5 persons
	Percentage of matching = 98%,65%, 97%, 94%, 95%
	Average time taken to detect = 0.5 seconds
	Observation: The result of detection for person or pedestrian is correct. However, some people who are being hidden is not able to detect.

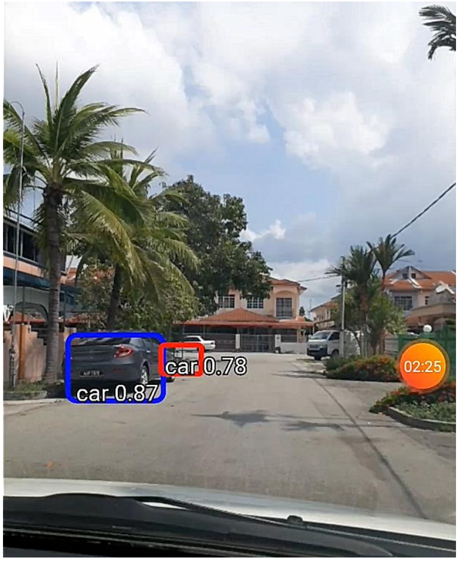
(4) Traffic Light 	*Object detected* = traffic light *Percentage of matching* = 62% *Average time taken to detect* = 1.0 seconds *Observation*: The result of detection for traffic light is correct. However, there are 2 traffic lights in the image. The system may need more time to detect another traffic light.
(5) Bicycle 	*Object detected* = bicycle *Percentage of matching* = 78% *Average time taken to detect* = 0.5 seconds *Observation*: The result of detection for traffic light is correct. However, there are 3 bicycles in the image. However, some parts of another 2 bicycles is hidden, the system may not to detect it as bicycle successfully.
(6) Potted plant and vase 	*Object detected* = potted plant and vase *Percentage of matching* = 68% and 83% *Average time taken to detect* = 0.5 seconds *Observation:* The result of detection for potted plant and vase is correct.
(6) Dining Table	*Object detected* = dining table

	*Percentage of matching* = 61%
	Average time taken to detect = 0.5 seconds
	Observation: The result of detection for dining table is correct.
(7) Truck	*Object detected* = truck
	*Percentage of matching* = 60%
	Average time taken to detect = 0.5 seconds
	Observation: The result of detection for truck is correct.

Besides, the system is also tested with objects in different distance in real-time. The observation, percentage of matching and average time taken to detect will be noted down.

Table 4-2: Detection Result of Different Distance of Object from User

<table>
<tr><td>Distance of object</td><td>Observation from system</td></tr>
<tr><td rowspan="4">(1) Near
</td><td>Object detected = car</td></tr>
<tr><td>Percentage of matching = 74%</td></tr>
<tr><td>Average time taken to detect = 0.5 seconds</td></tr>
<tr><td>Observation:
When object is near, the result of detection is correct.</td></tr>
<tr><td rowspan="4">(2) Moderate
</td><td>Object detected = car</td></tr>
<tr><td>Percentage of matching = 97%</td></tr>
<tr><td>Average time taken to detect = 0.5 seconds</td></tr>
<tr><td>Observation:
When distance of object is moderate from the user, the result of detection is correct.</td></tr>
</table>

(3) Far  	*Object detected* = 2 cars *Percentage of matching* = 87% and 78% *Average time taken to detect* = 1.0 seconds *Observation*: When distance of object is far from the user, the result of detection is correct. However, the bounding box is not bounded the actual object. This is because of the system is not able to catch up the updated location of the objects.
(4) Very Far  	*Object detected* = 4 cars *Percentage of matching* = 64%, 68%, 67%, 68% *Average time taken to detect* = 2.0 seconds *Observation*: When object is too far away from the system, some of the objects can still be detected but some of the objects may not be detected because it is appeared too small. There are 5 cars in the image, but there are only 4 cars in the result. The result of detection is partially correct.

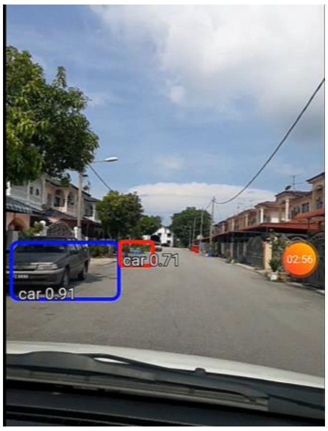
Furthermore, the system is used to detect multiple of objects to check the performance.

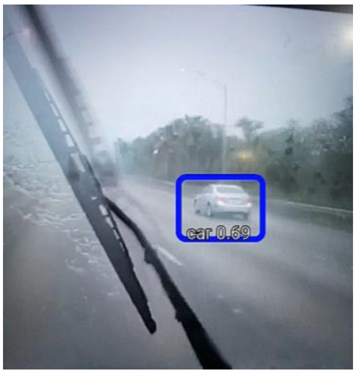
Table 4-3: Detection Result of Different Number of Object

Numbers of object	Observation from system
(1) Single object  	*Object detected* = motorcycle *Percentage of matching* = 75% *Average time taken to detect* = 0.5 seconds *Observation*: The system is able to detect single object.
(2) Multiple objects  	*Object detected* = motorcycle and potted plant *Percentage of matching* = 80%, 76%, 96% *Average time taken to detect* = 0.5 seconds *Observation*: The system is able to detect multiple type objects in the same time.
(3) Overlap  	*Object detected* = 2 motorcycles *Percentage of matching* = 87% and 62% *Average time taken to detect* = 1.0 seconds *Observation*: The system is able to detect multiple overlapped objects in the same time. However, it takes time to detect. Sometimes it may fail because some part of the object is hidden, and system cannot detect it.

The system is used to detect objects under different weather such as sunny, cloudy, rainy and during night.

Table 4-4 : Detection Result under Different Weather

<table>
<tr><th>Weather</th><th>Observation from system</th></tr>
<tr><td rowspan="4">(1) Sunny

</td><td>Object detected = 2 cars</td></tr>
<tr><td>Percentage of matching = 91% and 71%</td></tr>
<tr><td>Average time taken to detect = 0.5 seconds</td></tr>
<tr><td>Observation:
During sunny day, the system can perform well and fast.</td></tr>
<tr><td rowspan="4">(2) Cloudy

</td><td>Object detected = car</td></tr>
<tr><td>Percentage of matching = 79%</td></tr>
<tr><td>Average time taken to detect = 0.5 seconds</td></tr>
<tr><td>Observation:
During cloudy day, the system can perform fast, but some shaded and dark region cannot be detected correctly. For example, the truck is not being detected in the image.</td></tr>
<tr><td rowspan="4">(3) Rainy

</td><td>Object detected = car</td></tr>
<tr><td>Percentage of matching = 69%</td></tr>
<tr><td>Average time taken to detect = 1.0 seconds</td></tr>
<tr><td>Observation:
During rainy day, the system can perform well but may take some time to perform detection.</td></tr>
</table>

(1) Night 	*Object detected* = car and person
	Percentage of matching = 80% and 62%
	Average time taken to detect = 1.0 seconds
	Observation: During nighttime, the system can perform well if the light condition is in visible range. However, some shaded and dark region cannot be detected correctly. For example, there are a few persons on the street, but system can only detect one person.

4.1.3 Map Module

Since one of the objectives of this project is sharing data on cloud. Therefore, to achieve this objective, besides the “MAP VIEW” in the mobile application, other user can view the Map with marker by accessing URL “http://utarxyndi.000webhostapp.com/” in internet browser. The figure showed that results will be stored even if the user is moving place or multiple users are using it.

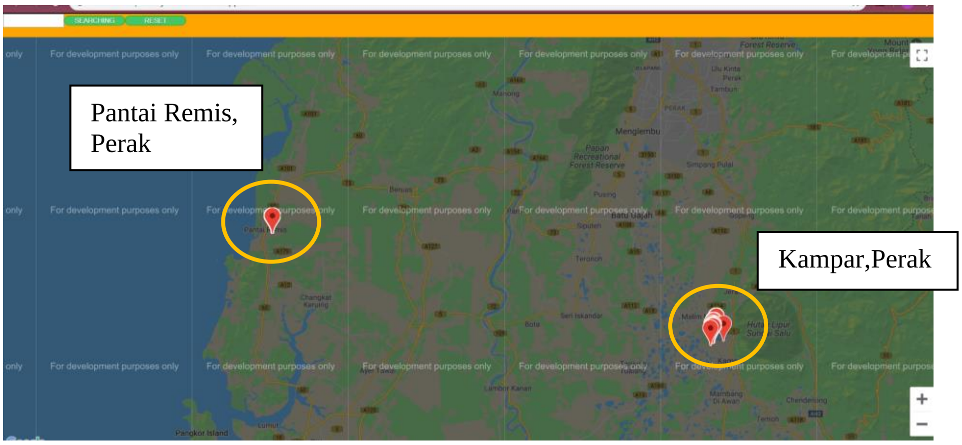


Figure 4-14: Map Module in Web

The red marker (yellow circle) is the place where the detection module detected something. When zooming in the map, the location of the detection will be clearer to be observed as shown in Figure 4-15.

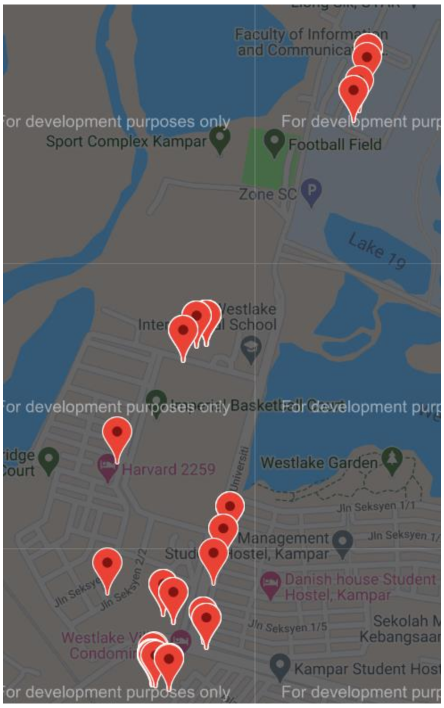


Figure 4-15: Multiple Marker

By clicking on the red marker, the detail information of detection result will be shown in a column as shown in Figure 4-16. The information includes the name of object detected, percentage of matching, date and time of detection and user who uploaded the data as well as GPS location coordinate of the objects.

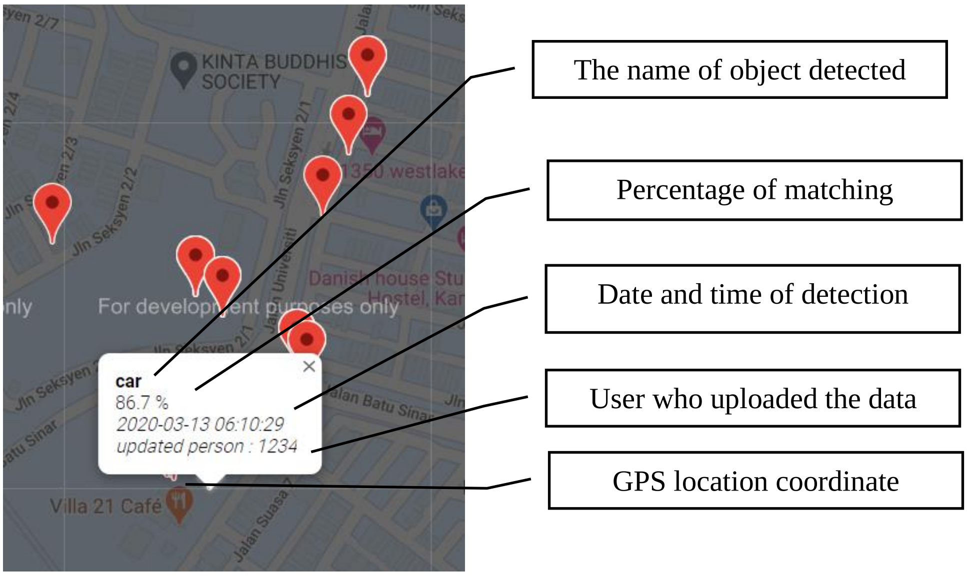


Figure 4-16: Maker with Column

No matter user is using the mobile application or accessing URL via internet browser, user can still search for the objects he wishes to search on map.

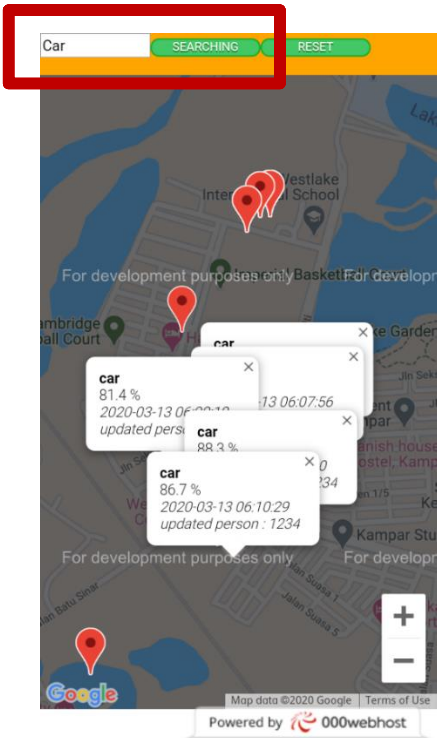


Figure 4-17: Searching function

Figure 4-17 showed that user using the "Searching" function to search for "Car", then the map will jump to where the car is locating and only the "Car" marker is displayed on map.

To validate the GPS result of the system, the apps is launched in the car and start detecting along the way.

Figure 4-18: Detection Result after Driving

Based on the observation, there is inaccurate marker plotted on wrong location sometimes which is already marked with yellow box. However, majority of the marker is plotted on correct coordinate.

It is recommended to enable high accuracy location mode in Android phone when launching the mobile application. Besides, use GPS, Wi-Fi or mobile networks will help user to get the most accurate location.

4.2 Analysis Between Different Object Detection System

4.2.1 YOLOv3

Numerous experiments were conducted to test the performance of the object detection system, which has tested under different lighting conditions, different weather, different distance, different speed and different objects.

The evaluation of performance is depending on the four numbers on result which is True Positive, True Negative, False Positive and False Negative. Confusion matrix is a table with two columns and two rows and show the relationship of the dour numbers. After having these 4 number from the result of detection, a detailed analysis can be made to calculate accuracy of the system.

Table 4-5: Matrix of Confusion

		Actual Class	
		Object detected correctly	Object detected wrongly
Predicted Class	Predicted Object Detected	**True Positive (TP)**	**False Positive (FP)**
	No Object Detected	**False Negative (FN)**	**True Negative (TN)**

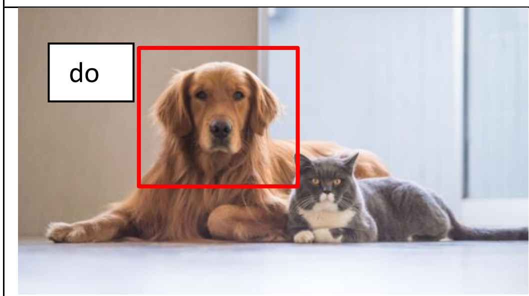
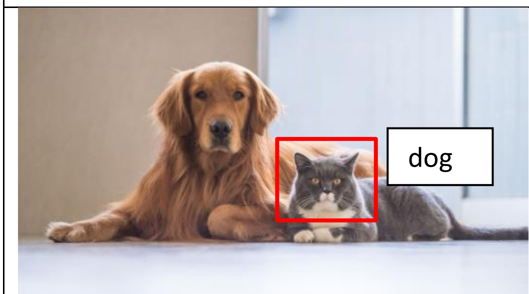
Table 4-5 shows the result description of object detection, where

- True positive (TP) is the condition when object A is detected as the object A, and it is true.
- True negative (TN) is the condition when there is no object A detected, and it is true.

- False positive (FP) is the condition when object B is detected as object A, which is wrong.
- False negative (FN) is the condition when is object A is not detected as object A, which is wrong.

In order to provide a clearer explanation, the illustration of the element of confusion matric with example image and statement is provided in Table 4-6.

Table 4-6: Illustration of the element of confusion matric

Example of image after detection	Statement
	<u>True positive</u> When dog is detected as "dog", and the result is correct.
	<u>True negative</u> When dog is not detected in the image, and the result is correct.
	<u>False positive</u> When cat is detected as "dog", the result is wrong.
	<u>False negative</u> When the dog is not detected as "dog", the result is wrong.

The performance of the classification model of object detection system can be evaluated by several metrics such as accuracy, precision, recall. To calculate the value of the metrics, the value of TP, TN, FP and FN plays the important roles.

Accuracy is defined as “the fraction of quantity of correct classification over the entire number of samples.”

$$\boldsymbol{Accuracy = \frac{TP + TN}{TP + TN + FP + FN}} \tag{4.1}$$

Precision is the ratio of correct detected result (TP) to the detected result (TP+FP)
.

$$\boldsymbol{Precision = \frac{TP}{TP + FP}} \tag{4.2}$$

Recall is the ratio of correct detected result (TP) to the actual result (TP+FN).

$$\boldsymbol{Recall = \frac{TP}{TP + FN}} \tag{4.3}$$

Besides those evaluation metrics, there is also some basic terms of foundational concepts which is confidence score and IoU.

Confidence score is the probability that an anchor box contains an object. It is usually predicted by a classifier. In other word, confidence score is the probability the predicted object matching the actual object. Intersection over Union (IoU) is defined as the area of the intersection divided by the area of the union of a predicted bounding box and a ground-truth box (actual object). Both IoU and confidence score are used as the important element that determine whether a detection is a or false positive a true positive.

A detection is considered a true positive (TP) only if it satisfies three conditions:

- Confidence score is greater than threshold.
- The predicted class matches the class of a ground truth.
- The predicted bounding box has an IoU greater than a threshold (e.g., 0.5) with the ground-truth.

False positive happens when confidence score is greater than threshold but there is violation of either of the latter two conditions which is the predicted class matches the class of a ground truth and the predicted bounding box has an IoU greater than a threshold (e.g., 0.5) with the ground-truth.

The detection counts as a false negative (FN) when the Confidence score of a detection that is supposed to detect a ground-truth is lower than the threshold

The detection counts as a true negative (TN) when the Confidence score of a detection that is not supposed to detect anything is lower than the threshold.

To have a better understanding of the evaluation metrices of performance of classification model. Figure below shows the illustration of calculation of precision, recall and IoU.

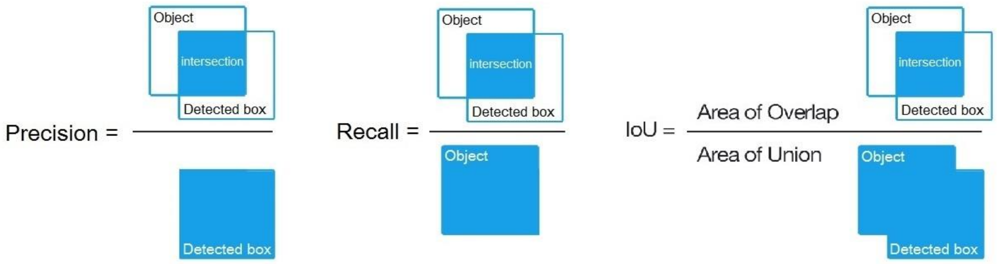


Figure 4-19: Illustration of calculation of precision, recall and IoU

To evaluate the performance of system in this paper, the accuracy, precision and recall of the result are calculated based on the value of TP, TN, FP and FN using the formulas in (4.1), (4.2) and (4.3). Confidence score is the value already measured by the system as the output. Since actual area detected is difficult to measure from the image, the Intersection over Union (IoU) is not calculated in result.

In order to make sure the object detection system is evaluated in all aspect; the system is running in different conditions. A real-time video is taken for each different condition and the total number of frames of real time video is counted. The number of frames for TP, TN, FP and FN are taken to calculate accuracy, precision and recall. Besides, the average time taken to detect object is measured.

The evaluation of performance is done on two sets of classification model which is YOLOv3 with custom trained model and YOLOv3 with COCO pretrained model. The difference between both models is the number of datasets for training. Figure below explained the structure of the result in table-form.

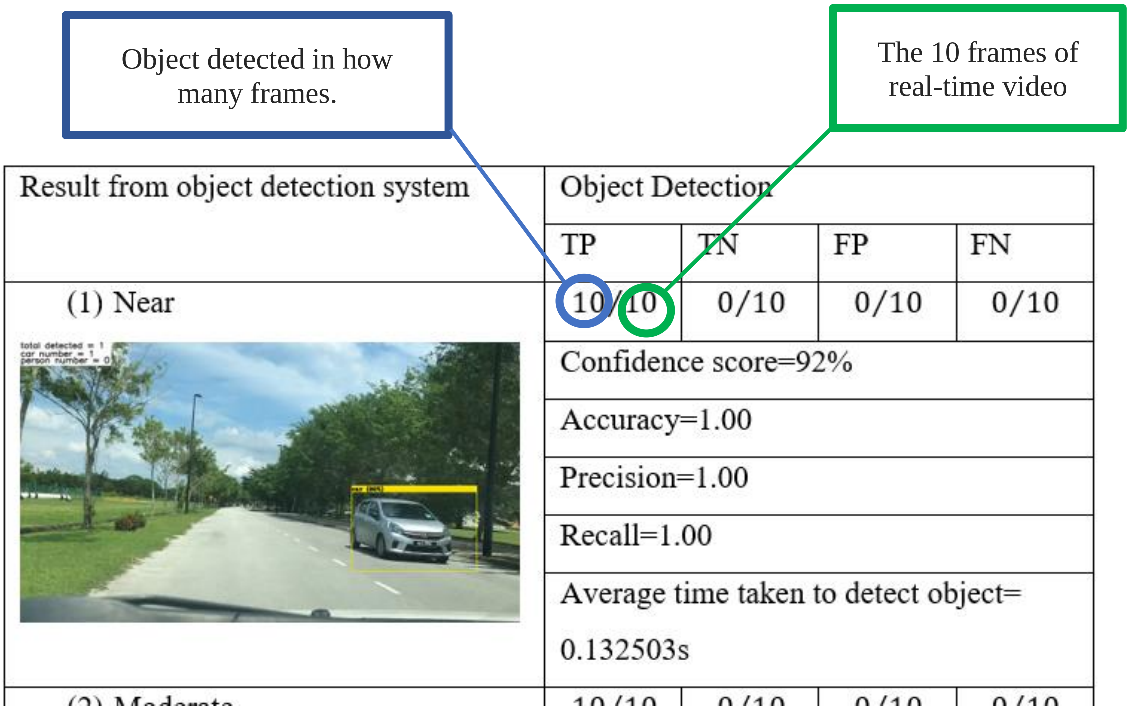


Figure 4-20: The Structure of Result Presented

Different Object

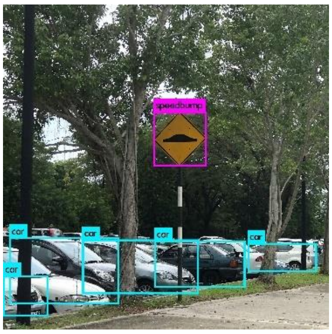
First and foremost, the trained models must be tested with the trained objects which is aimed to be detected. The trained model is supposed to detect speedbump signboard, one-way signboard, speed-limit signboard, cyclist signboard, no-right-turn signboard, parking signboard, no-entry signboard, stop signboard, pedestrian signboard, exit signboard, bus-stop signboard, no-left-turn signboard, enter signboard, car, person and bike.

In COCO pretrained model, there are 80 objects which is already trained and ready to be detected. COCO dataset contains 2.5 million labelled instances in 382,000 images and its training takes long time as well. The 80 objects are listed in Table 3-1 in Chapter 3.

In order to proceed to evaluation of performance of the object detection system, the classification model must be made sure is functional. In this part, only the self-trained model is tested because COCO pretrained model is the verified model by expertise.

Table 4-7 showed the result of using self- trained model in YOLOv3. The result of the model is correct and verified. Therefore, the model is ready to be used in the object detection system.

Table 4-7: Result using Self-trained Model with Different Object

Result from object detection system	Object Detection			
Light condition	TP	TN	FP	FN
(1) Speedbump signboard 	45/45	0/45	0/45	0/45
	Confidence score= 100%			
	Accuracy=1.0000			
	Precision=1.0000			
	Recall=1.0000			
	Average time taken to detect object=0.115803s			
(2) One-way signboard 	43/45	0/45	0/45	2/45
	Confidence score=100%			
	Accuracy=0.8600			
	Precision=1.0000			
	Recall=0.8600			
	Average time taken to detect object=0.115440s			

<table>
<tr><td rowspan="6">(3) Speed limit signboard

</td><td>35/45</td><td>0/45</td><td>1/45</td><td>9/45</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=0.7778</td></tr>
<tr><td colspan="4">Precision=0.9722</td></tr>
<tr><td colspan="4">Recall=0.7955</td></tr>
<tr><td colspan="4">Average time taken to detect object=0.116193s</td></tr>
<tr><td rowspan="6">(4) Cyclist signboard

</td><td>38/45</td><td>0/45</td><td>3/45</td><td>4/45</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=0.8444</td></tr>
<tr><td colspan="4">Precision=0.9268</td></tr>
<tr><td colspan="4">Recall=0.9048</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.114744s</td></tr>
<tr><td rowspan="6">(5) No-right-turn signboard

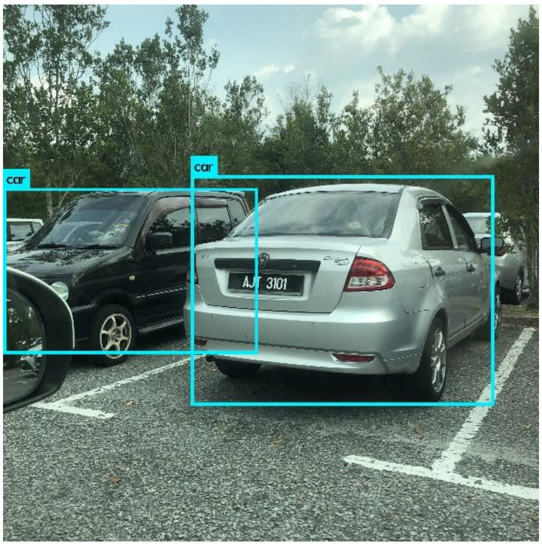
</td><td>40/45</td><td>0/45</td><td>2/45</td><td>3/45</td></tr>
<tr><td colspan="4">Confidence score= 97%</td></tr>
<tr><td colspan="4">Accuracy=0.8889</td></tr>
<tr><td colspan="4">Precision=0.9524</td></tr>
<tr><td colspan="4">Recall=0.9302</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.115648s</td></tr>
</table>

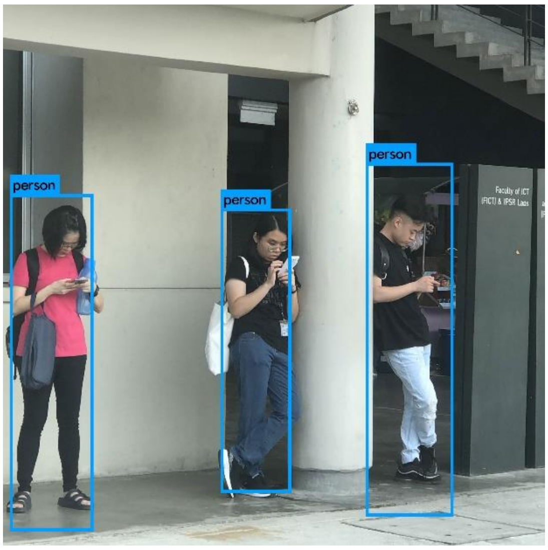
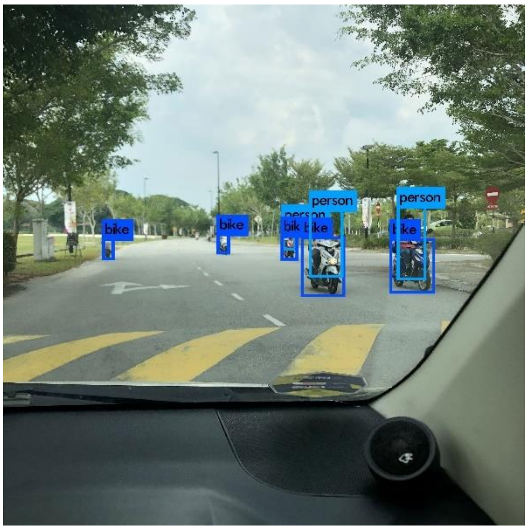
<table>
<tr><td>(6) Parking signboard</td><td>43/45</td><td>0/45</td><td>0/45</td><td>2/45</td></tr>
<tr><td rowspan="5">
</td><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=0.9556</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.9556</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.115719s</td></tr>
<tr><td>(7) No-entry signboard</td><td>45/45</td><td>0/45</td><td>0/45</td><td>0/45</td></tr>
<tr><td rowspan="5">
</td><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.115744s</td></tr>
<tr><td>(8) Stop signboard</td><td>43/45</td><td>0/45</td><td>0/45</td><td>2/45</td></tr>
<tr><td rowspan="5">
</td><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=0.9556</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.9556</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.114880s</td></tr>
</table>

(9) Pedestrian signboard	40/45	0/45	0/45	5/45
	Confidence score= 100%			
	Accuracy=0.8889			
	Precision=1.0000			
	Recall=0.8889			
	Average time taken to detect object= 0.115365s			
(10) Exit signboard	40/45	0/45	0/45	5/45
	Confidence score=100%			
	Accuracy=0.8889			
	Precision=1.0000			
	Recall=0.8889			
	Average time taken to detect object= 0.113743s			
(11) Bus stop signboard	33/45	0/45	0/45	12/45
	Confidence score=100%			
	Accuracy=0.7333			
	Precision=1.0000			
	Recall=0.7333			
	Average time taken to detect object= 0.115657s			

<table>
<tr><td rowspan="6">(12) No-left-turn signboard

</td><td>37/45</td><td>0/45</td><td>0/45</td><td>8/45</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=0.8222</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.8222</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.113922s</td></tr>
<tr><td rowspan="6">(13) Enter signboard

</td><td>42/45</td><td>0/45</td><td>0/45</td><td>3/45</td></tr>
<tr><td colspan="4">Confidence score=96%</td></tr>
<tr><td colspan="4">Accuracy=0.9333</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.9333</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.115335s</td></tr>
<tr><td rowspan="6">(14) Car

</td><td>45/45</td><td>0/45</td><td>0/45</td><td>0/45</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.114828s</td></tr>
</table>

<table>
<tr><td rowspan="6">(15) Person

</td><td>40/45</td><td>0/45</td><td>0/45</td><td>5/45</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=0.8889</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.8889</td></tr>
<tr><td colspan="4">Average time taken to detect object=0.115615s</td></tr>
<tr><td rowspan="6">(16) Bike

</td><td>35/45</td><td>0/45</td><td>3/45</td><td>7/45</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=0.7778</td></tr>
<tr><td colspan="4">Precision=0.9211</td></tr>
<tr><td colspan="4">Recall=0.8333</td></tr>
<tr><td colspan="4">Average time taken to detect object=0.115553s</td></tr>
</table>

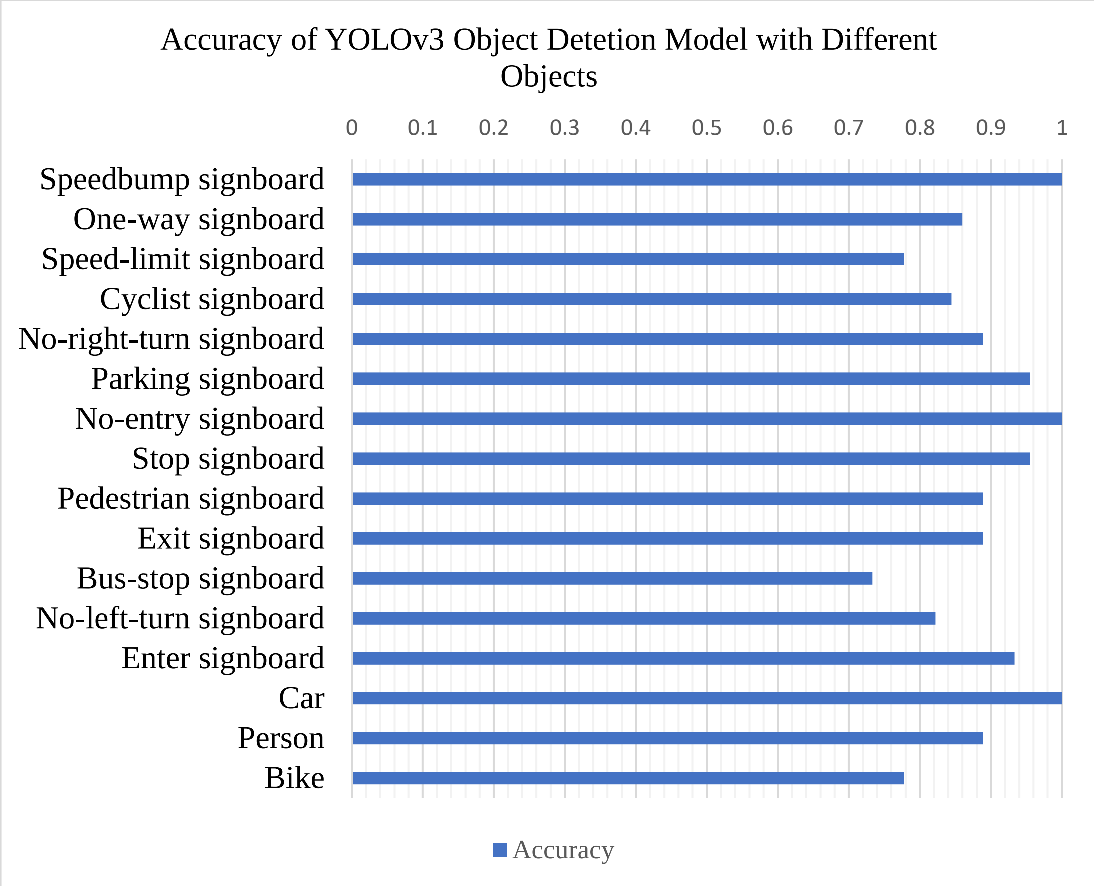


Figure 4-21: Accuracy of Self-Trained Model in Detecting Different Objects

The result is taken in real-time. All objects have been captured in real-time video for few seconds. Targeted objects are bounded by bounding box and the respective label names. To analyse the result, the 45 frames, which object is detected, is calculated frame by frame, whether it is TP, TN, FP or FN.

Based on Figure 4-21, the overall accuracy is high enough to be the classification model of object detection system. The accuracy of an image classifier depends on the amount of training data used. The more the amount of training data, the higher of the accuracy of image classifier, till a saturation point which is known as "overfitting".

The overfitting happens when the system learned a complex pattern in data and leads to memorization of the data. It has only memorized the training data and is unable to detect new case. This happens because of the training time is too long.

Different Light Condition

In this part, self-trained model and COCO pretrained model are tested under different light condition such as low light condition, normal light condition and high light condition. The evaluation and results are shown in Table 4-8 and Table 4-9 respectively.

Table 4-8: Result using Self-trained Model under Different Light Condition

<table>
<tr><td rowspan="2">Result from object detection system
Light condition</td><td colspan="4">Object Detection(motorbike)</td></tr>
<tr><td>TP</td><td>TN</td><td>FP</td><td>FN</td></tr>
<tr><td rowspan="6">(1) Low Light
</td><td>0/177</td><td>0/177</td><td>0/177</td><td>177/177</td></tr>
<tr><td colspan="4">Confidence score=0%</td></tr>
<tr><td colspan="4">Accuracy=0.000</td></tr>
<tr><td colspan="4">Precision=0.000</td></tr>
<tr><td colspan="4">Recall=0.000</td></tr>
<tr><td colspan="4">Average time taken to detect object=/</td></tr>
<tr><td rowspan="6">(2) Normal Light
</td><td>293/308</td><td>0/308</td><td>2/308</td><td>12/308</td></tr>
<tr><td colspan="4">Confidence score= 69%</td></tr>
<tr><td colspan="4">Accuracy=0.9513</td></tr>
<tr><td colspan="4">Precision=0.9932</td></tr>
<tr><td colspan="4">Recall=0.9607</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.160078s</td></tr>
<tr><td rowspan="6">(3) High Light
</td><td>0/180</td><td>0/180</td><td>0/180</td><td>180/180</td></tr>
<tr><td colspan="4">Confidence score=0%</td></tr>
<tr><td colspan="4">Accuracy=0.000</td></tr>
<tr><td colspan="4">Precision=0.000</td></tr>
<tr><td colspan="4">Recall=0.000</td></tr>
<tr><td colspan="4">Average time taken to detect object=/</td></tr>
</table>

Table 4-9: Result using COCO Pretrained Model under Different Light Condition

<table>
<tr><td rowspan="2">Result from object detection system
Light condition</td><td colspan="4">Object Detection (motorbike)</td></tr>
<tr><td>TP</td><td>TN</td><td>FP</td><td>FN</td></tr>
<tr><td rowspan="7">(1) Low Light

</td><td>170/200</td><td>0/200</td><td>20/200</td><td>10/200</td></tr>
<tr><td colspan="4">Confidence score=82%</td></tr>
<tr><td colspan="4">Accuracy=0.8500</td></tr>
<tr><td colspan="4">Precision=0.8947</td></tr>
<tr><td colspan="4">Recall=0.9444</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.167306s</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td rowspan="6">(2) Normal Light

</td><td>195/200</td><td>0/200</td><td>5/200</td><td>0/200</td></tr>
<tr><td colspan="4">Confidence score=97%</td></tr>
<tr><td colspan="4">Accuracy=0.9750</td></tr>
<tr><td colspan="4">Precision=0.9750</td></tr>
<tr><td colspan="4">Recall=1.000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.163356s</td></tr>
<tr><td rowspan="6">(3) High Light

</td><td>175/200</td><td>0/200</td><td>20/200</td><td>5/200</td></tr>
<tr><td colspan="4">Confidence score=87%</td></tr>
<tr><td colspan="4">Accuracy=0.8750</td></tr>
<tr><td colspan="4">Precision=0.8974</td></tr>
<tr><td colspan="4">Recall=0.9722</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.161997s</td></tr>
</table>

Figure 4-22 showed the difference of accuracy of using self-trained model and COCO pretrained model under different light condition.

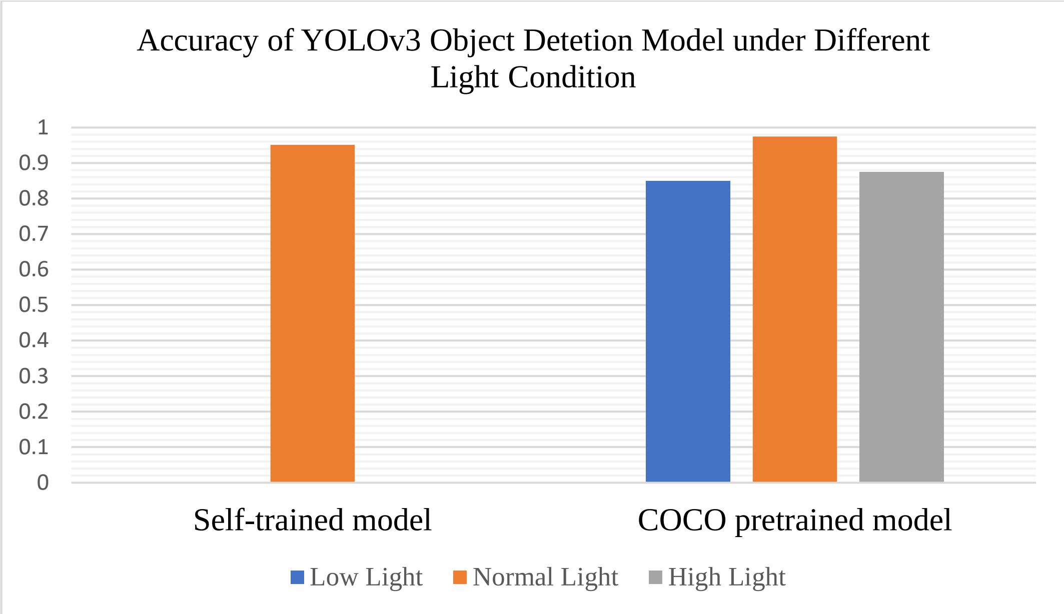


Figure 4-22: Comparison of Accuracy between Self-trained Model and COCO Pretrained Model under Different Light Condition

Based on Table 4-8 and Figure 4-22, object detection system using self-trained models can only detect objects under normal light condition only. Meanwhile, object detection system using COCO pretrained model can detect objects in all condition as shown in Table 4-9. In order to evaluate the performance precisely, the result only focused on the detection of motorbikes at the front, even though some background objects such as car are detected as well.

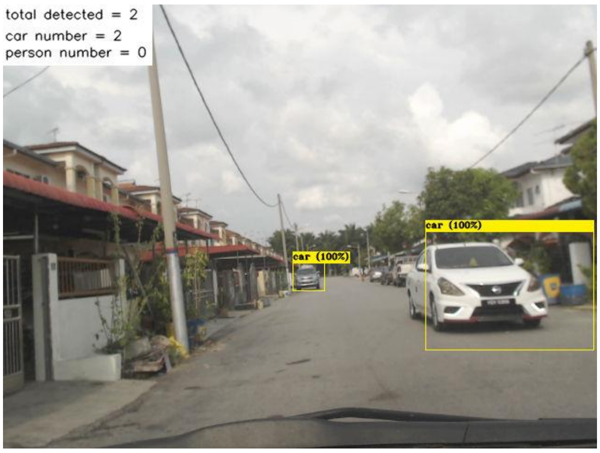
The main reason of the differences is lack of datasets for training in self-trained models. COCO datasets contained the image of objects under most of the situation. For example, there are images of objects lying under shadow and images of objects in bright environment in COCO datasets. However, the datasets in self-trained datasets are the images under normal condition. Therefore, using self-trained models, the objects in low light condition and objects under high light exposure failed to be detected and classified.

Different Weather

In this part, self-trained model and COCO pretrained model are tested under different weather such as sunny day, rainy day and cloudy day. The evaluation and results are shown in Table 4-10 and Table 4-11 respectively.

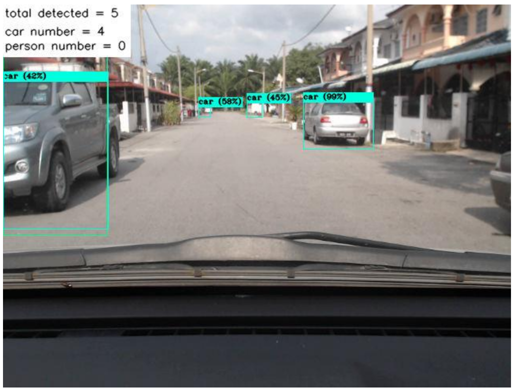
The challenges faced is the difficulty to capture the road condition of rainy and cloudy in real-time. Therefore, the videos of road in rainy and cloudy day are downloaded from online source. Videos are played in the phone and the video will be facing to the object detection system. Object detection system will detect the objects in the video as if it is happening in real-time.

Table 4-10: Result using Self-trained Model under Different Weather

Result from object detection system under different weather	Object Detection (Car)			
	TP	TN	FP	FN
(1) Sunny 	40/40	0/40	0/40	0/40
	Confidence score= 100%			
	Accuracy=1.0000			
	Precision=1.0000			
	Recall=1.0000			
	Average time taken to detect object= 0.169455s			
(2) Rainy 	35/50	0/50	5/50	15/50
	Confidence score= 100%			
	Accuracy=0.7000			
	Precision=0.8750			
	Recall=0.7778			
	Average time taken to detect object= 0.163670s			

(3) Cloudy 	41/50	0/50	0/50	9/50
	Confidence score=100%			
	Accuracy= 0.8200			
	Precision=1.0000			
	Recall=0.8200			
	Average time taken to detect object= 0.163550s			

Table 4-11: Result using COCO Pretrained Model under Different Weather

Result from object detection system	Object Detection (Car)			
	TP	TN	FP	FN
(1) Sunny 	45/45	0/45	0/45	0/45
	Confidence score=99%			
	Accuracy=1.0000			
	Precision=1.0000			
	Recall=1.0000			
	Average time taken to detect object= 0.171867s			
(2) Rainy 	43/45	0/45	0/45	2/45
	Confidence score=97%			
	Accuracy=0.9556			
	Precision=1.0000			
	Recall=0.9556			
	Average time taken to detect object= 0.161375s			

(3) Cloudy 	42/45	0/45	0/45	3/45
	Confidence score=95%			
	Accuracy=0.9333			
	Precision=1.0000			
	Recall=0.9333			
	Average time taken to detect object= 0.151757s			

Figure 4-23 showed the difference of accuracy of using self-trained model and COCO pretrained model under different weather.

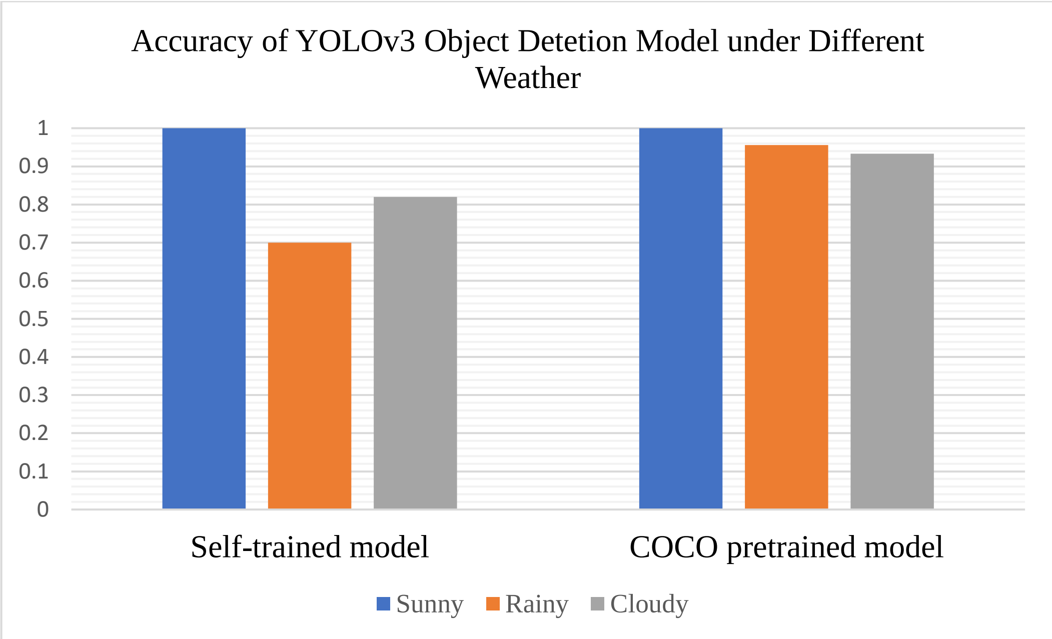


Figure 4-23: Comparison of Accuracy between Self-trained Model and COCO Pretrained Model under Different Weather

Based on Table 4-10 and Table 4-11, object detection system can perform under different weather by using self-trained model or COCO pretrained model. During sunny day, both models can perform well in detecting objects with the accuracy of 1 in the experiment. However, COCO pretrained model has higher accuracy during cloudy and rainy day with the accuracy of 0.9333 and 0.9556. Meanwhile, self-trained model has only accuracy of 0.8200 in cloudy day and 0.7000 in rainy day.

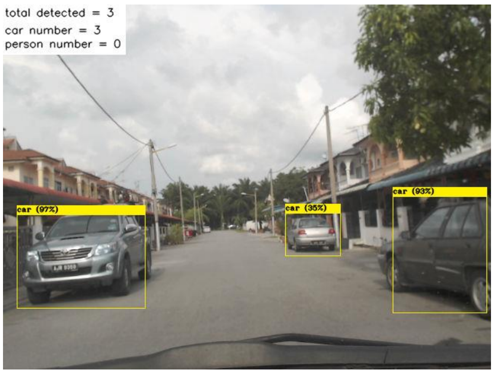
Similarly, the main reason of the differences is lack of datasets for training in self-trained models. This is because COCO datasets contained the image of objects under most of the situation while self-trained datasets have only limited image under normal condition.

In short, this experiment has shown that the object detection system with both models does not have problem under different weather.

Different Distance

Self-trained model and COCO pretrained model are tested to detect objects from different distance which is from about 100 meters, 60 meters, 30 meters and 10 meters. The evaluation and results are shown in Table 4-12 and Table 4-13 respectively.

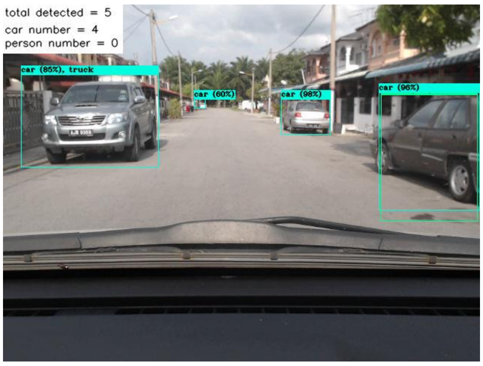
Table 4-12: Result using Self-trained Model under Different Distance

<table>
<tr><td rowspan="2">Result from object detection system</td><td colspan="4">Object Detection (car)</td></tr>
<tr><td>TP</td><td>TN</td><td>FP</td><td>FN</td></tr>
<tr><td rowspan="6">(1) Near (10 meters)

</td><td>20/20</td><td>0/20</td><td>0/20</td><td>0/20</td></tr>
<tr><td colspan="4">Confidence score=97%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.161508s</td></tr>
<tr><td rowspan="6">(2) Moderate (30 meters)

</td><td>30/30</td><td>0/30</td><td>0/30</td><td>0/30</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.162061s</td></tr>
</table>

<table>
<tr><td rowspan="6">(3) Far (60 meters)
</td><td>39/40</td><td>0/40</td><td>0/40</td><td>1/40</td></tr>
<tr><td colspan="4">Confidence score=99%</td></tr>
<tr><td colspan="4">Accuracy=0.9750</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.9750</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.162109s</td></tr>
<tr><td rowspan="6">(4) Very Far (100 meters)
</td><td>33/50</td><td>0/50</td><td>0/50</td><td>17/50</td></tr>
<tr><td colspan="4">Confidence score=40%</td></tr>
<tr><td colspan="4">Accuracy=0.6600</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.6600</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.163561s</td></tr>
</table>

Table 4-13: Result using COCO Pretrained Model under Different Distance

<table>
<tr><td rowspan="2">Result from object detection system</td><td colspan="4">Object Detection</td></tr>
<tr><td>TP</td><td>TN</td><td>FP</td><td>FN</td></tr>
<tr><td rowspan="6">(1) Near (10 meters)
</td><td>20/20</td><td>0/20</td><td>0/20</td><td>0/20</td></tr>
<tr><td colspan="4">Confidence score=85%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.167852s</td></tr>
</table>

<table>
<tr><td rowspan="6">(2) Moderate (30 meters)
</td><td>30/30</td><td>0/30</td><td>0/30</td><td>0/30</td></tr>
<tr><td colspan="4">Confidence score=91%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.161136s</td></tr>
<tr><td rowspan="6">(3) Far (60 meters)
</td><td>40/40</td><td>0/40</td><td>0/40</td><td>0/40</td></tr>
<tr><td colspan="4">Confidence score=92%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.163208s</td></tr>
<tr><td rowspan="6">(4) Very Far (100 meters)
</td><td>50/50</td><td>0/50</td><td>0/50</td><td>0/50</td></tr>
<tr><td colspan="4">Confidence score=91%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object=
0.165481s</td></tr>
</table>

Figure 4-24 showed the difference of accuracy of using self-trained model and COCO pretrained model in detecting objects from different range of distance.

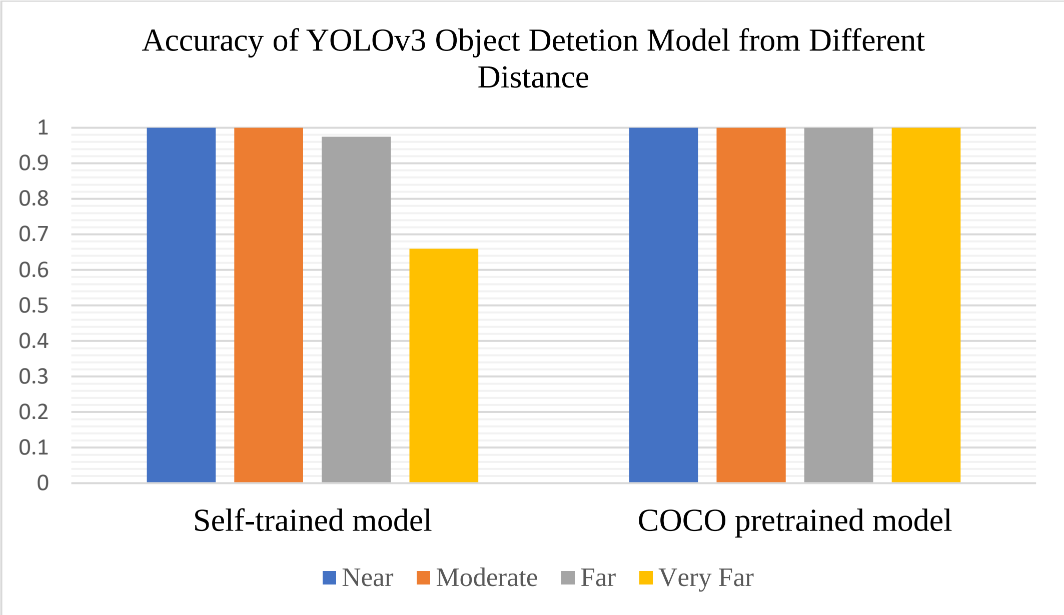


Figure 4-24 : Comparison of Accuracy between Self-trained Model and COCO Pretrained Model from Different Distance

Based on Table 4-12 and Table 4-13, object detection system can perform well in detecting objects from different range of distance by using self-trained model or COCO pretrained model. COCO pretrained model maintains high accuracy of 1 in detecting objects in all range of distance as long as the objects can be seen by human's eyes. Meanwhile, self-trained model has only accuracy of 0.9750 and 0.6600 in distance of 60 meters and 100 meters but achieve accuracy of 1 when the object is near.

In short, this experiment has shown that the object detection system with both models does not have problem in detecting objects in different range.

Different Speed

In this part, self-trained model and COCO pretrained model are tested in detecting object which is under different speed of motion. The evaluation and results are shown in Table 4-14 and Table 4-15 respectively.

The challenges faced is the difficulty to catch the opportunity to capture the real-time car movement in fast speed. Therefore, the car movement video is speeded up and played in phone. Videos are played in the phone and the video will be facing to the object detection system. The object detection system will detect the objects in the video played by the phone as if it is happening in real-time.

The car-movement video is not directly inserted into the object detection system because the speed of video will be optimised so that the detection system will be able to catch the detection. Therefore, directly inserting video into the system is not a good way to evaluate the performance of detection in real-time. In order to capture the real-time movement, the video is played in phone and faced to the camera.

Table 4-14: Result using Self-trained Model under Different Speed

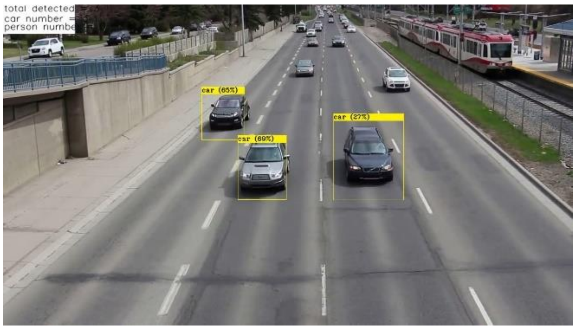
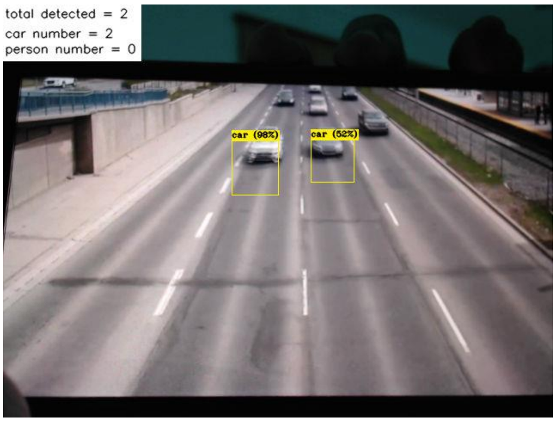
<table>
<tr><td rowspan="2">Result from object detection system</td><td colspan="4">Object Detection</td></tr>
<tr><td>TP</td><td>TN</td><td>FP</td><td>FN</td></tr>
<tr><td rowspan="6">(1) Moderate movement
</td><td>35/40</td><td>0/40</td><td>0/40</td><td>15/40</td></tr>
<tr><td colspan="4">Confidence score=69%</td></tr>
<tr><td colspan="4">Accuracy=0.8750</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.8750</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.154391s</td></tr>
<tr><td rowspan="6">(2) Fast movement

</td><td>20/45</td><td>0/45</td><td>0/45</td><td>25/45</td></tr>
<tr><td colspan="4">Confidence score=98%</td></tr>
<tr><td colspan="4">Accuracy=0.4444</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.4444</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.179541s</td></tr>
</table>

Table 4-15: Result using COCO Pretrained Model under Different Speed

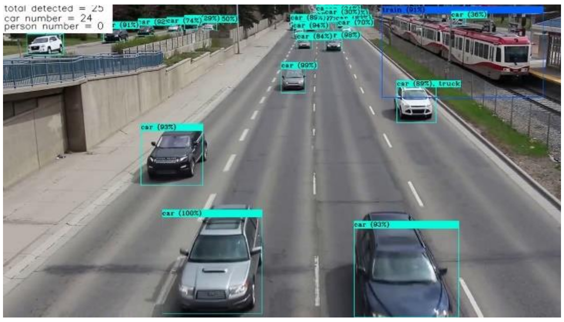
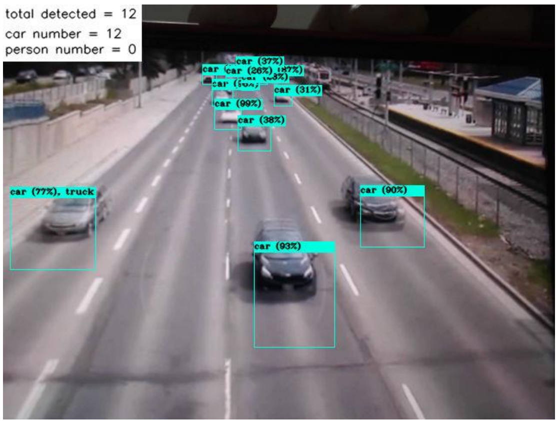
<table>
<tr><td rowspan="2">Result from object detection system</td><td colspan="4">Object Detection</td></tr>
<tr><td>TP</td><td>TN</td><td>FP</td><td>FN</td></tr>
<tr><td rowspan="6">(1) Moderate movement
</td><td>45/45</td><td>0/45</td><td>0/45</td><td>0/45</td></tr>
<tr><td colspan="4">Confidence score=100%</td></tr>
<tr><td colspan="4">Accuracy=1.0000</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=1.0000</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.093288s</td></tr>
<tr><td rowspan="6">(1) Fast movement

</td><td>40/45</td><td>0/45</td><td>0/45</td><td>5/45</td></tr>
<tr><td colspan="4">Confidence score=99%</td></tr>
<tr><td colspan="4">Accuracy=0.8889</td></tr>
<tr><td colspan="4">Precision=1.0000</td></tr>
<tr><td colspan="4">Recall=0.8889</td></tr>
<tr><td colspan="4">Average time taken to detect object= 0.094388s</td></tr>
</table>

Figure 4-25 showed the difference of accuracy of using self-trained model and COCO pretrained model which is under different speed of motion.

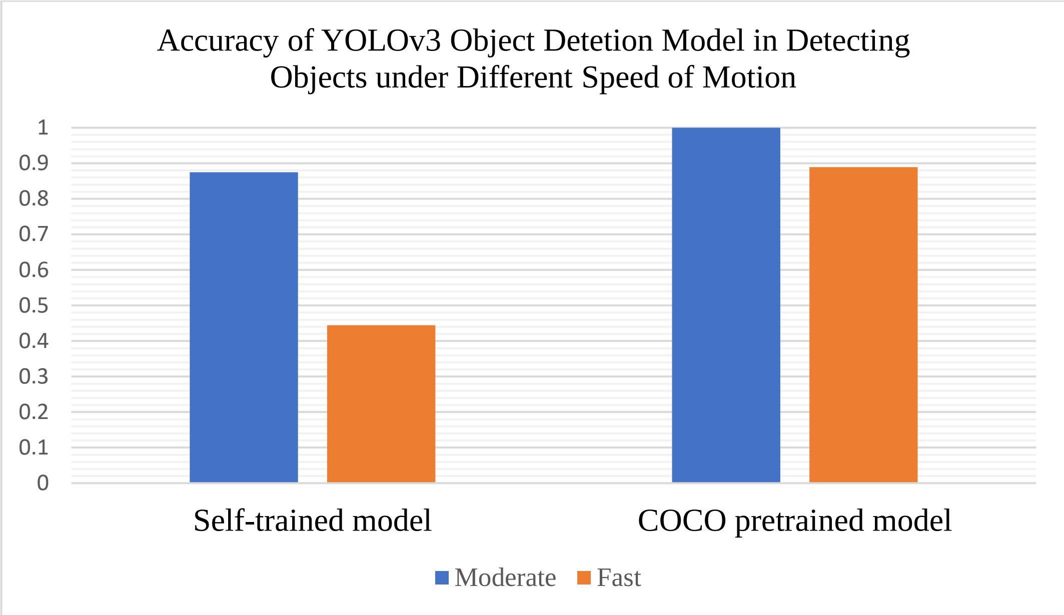


Figure 4-25: Comparison of Accuracy between Self-trained Model and COCO Pretrained Model under Different Speed of Motion

Based on Table 4-11 and Table 4-15, object detection system can perform well in detecting objects which is under different speed of motion by using self-trained model or COCO pretrained model. COCO pretrained model maintains high accuracy of 1 in detecting objects if the objects are moving in moderate speed. When the objects are moving fast, the bounding box may not able to bound the objects is actual position. This is because the system is not fast enough to catch up the actual location of the objects. Nevertheless, the detection result is correct.

Meanwhile, self-trained model has only accuracy of 0.8750 and 0.4444 when objects moving in moderate speed and fast speed respectively. The condition of wrong localization of bounding box happens as well in self- trained model.

In short, this experiment has shown that the object detection system with both models does not have much problem in detecting objects when objects is moving at various speed.

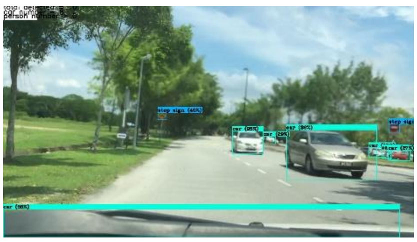
Multiple Object

In this part, self-trained model and COCO pretrained model are tested in detecting multiple object in real-time. The evaluation and results are shown in Table 4-16 and Table 4-17 respectively.

Table 4-16: Result using Self-trained Model to Detect Multiple Objects

Numbers of object	Observation from system
(1) One object 	The car is detected. The result is correct.
(2) Multiple objects 	The speedbump signboard, no-entry signboard and 2 driving cars are detected. The cars parking at the back is note counted. The result is correct.

Table 4-17: Result using COCO Pretrained Model to Detect Multiple Objects

Numbers of object	Observation from system
(1) One object 	The car is detected. The result is correct.
(2) Multiple objects 	The signboard and 8 cars are detected including the cars parked at the back. However, the signboard is wrong detected as stop signboard.

Based on Table 4-16 and Table 4-17, object detection system can perform well in detecting multiple objects by using self-trained model or COCO pretrained model.

COCO pretrained model is able to detect more objects with higher accuracy than self-trained model. As shown in Table 4-16, COCO pretrain model can help to detect the cars at the back whereas self-trained model is not able to do so. The limitation of COCO pretrained model in this experiment is specific signboard cannot be recognised correctly. All signboards are recognised as “stop sign” in COCO pretrained model.

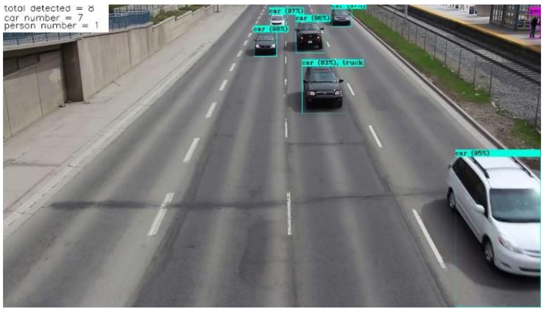
With self-trained model, most of the specific signboards can be recognised correctly because it is specifically trained. The accuracy is also high enough to be the classification model for the system.

In short, this experiment has shown that the object detection system with both models does not have much problem in detecting multiple objects.

Counter

In this part, the evaluation and results of counter are shown in Table 4-18.

Table 4-18: Analysis of Counter Result

Result of Object Detection System	Result of counter	
	Total number of objects detected	Actual number
	8	8
	Car Number	Actual number
	7	7
	Person Number	Actual number
	1	1
	Total number of objects detected	Actual number
	4	4
	Car Number	Actual number
	3	3
	Person Number	Actual number
	0	0
	Total number of objects detected	Actual number
	105	122
	Car Number	Actual number
	0	3
	Person Number	Actual number
	100	117

The result of counter depends on the result of object detection system. The counter will be counting correctly as long as the result of detection system is correct.

There is a limitation in counter too. The default maximum bounding box of YOLOv3 object detection method is 200. If user want to detect more objects, the value in source code must be modified to a higher value.

However, there is difficulty in detecting too many objects in one single frame. The accuracy of detection will decrease when the objects in the frame increase as shown in the result in Table 4-18.

To increase the accuracy when detecting large number of objects, the objects must be made sure that the objects are clear and large to be seen. Besides, increasing the number of datasets training is one of the ways too. The datasets trained should contain the image of large number of labelled objects such as group photo.

4.2.2 Firebase ML Kit

Machine learning is the part of AI. Nowadays, machine learning has become an integral part of mobile development. Big companies like Uber, Facebook, Microsoft etc. rely heavily on machine learning for their businesses. It helps them to know their users better and provide them with a better experience on their apps. In this project, the method of Firebase ML Kit is also tested to build the object detection system in Android phone.

First, we used ML Kit in Firebase console to label and train our dataset. we need to put together a training dataset of labelled images. The images must be in certain format, for example JPEG. Each image must be 30MB or smaller. It is better to Include at least 100 or more examples of each label. For higher accuracy, multiple angles, resolutions, and backgrounds for each label should be included. Besides, the models cannot generally predict labels that humans cannot assign. So, if a human cannot assign labels by looking at the image for 1 or 2 seconds, the model likely cannot be trained to do it either.

To simplify the task, it is advisable to organize our training images into directories, each named after a label and containing images that are examples of that label. Then, compress the directory structure into a zip archive and upload.

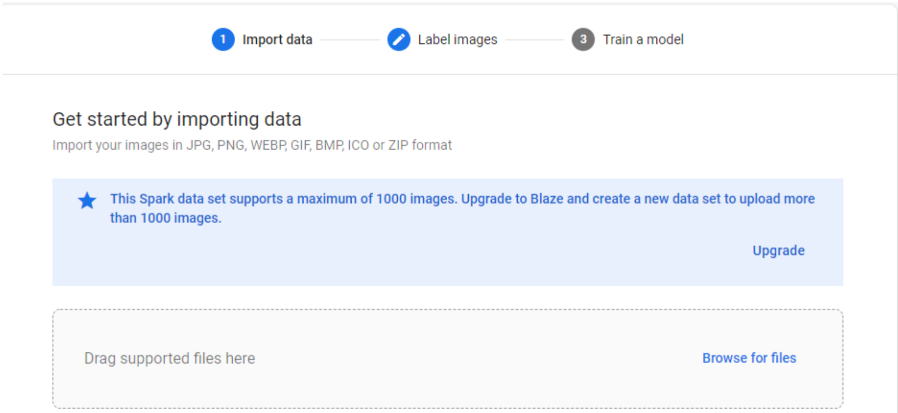


Figure 4-26: Import Dataset

Then, check again the labelled image whether there is any miss out or mis-labelled.

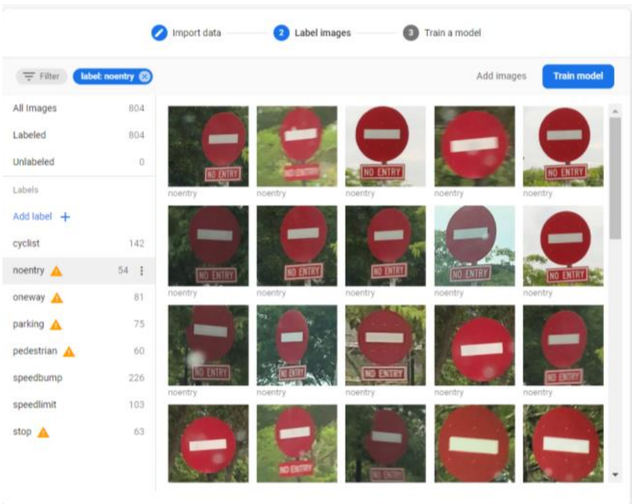


Figure 4-27: Label Dataset

After that, we can start training the datasets. We can configure the following settings, which govern the performance of the generated model. We can choose to train faster but accuracy is low. In other hand, longer training time will result in higher accuracy.

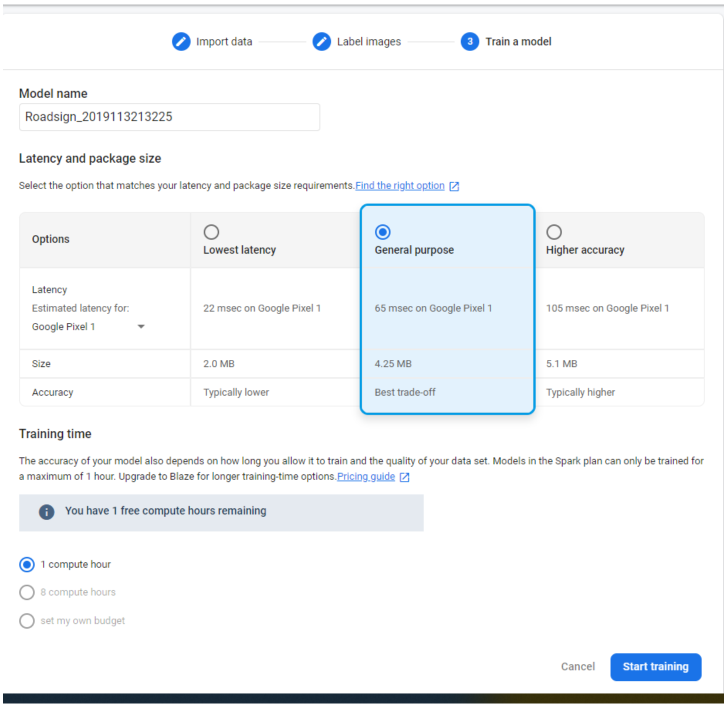


Figure 4-28: Choose the number of hours to train

However, we need to pay for the training time. The first 3 compute hour is free for new user. Following compute hour need to be paid.

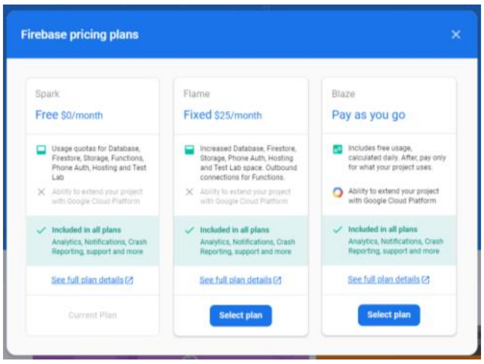


Figure 4-29: Choose the plan and pay the training fee

A notification email will be sent to us once the training is completed, then we can choose to download model to bundle it with our app or just publish it to Firebase and we can load it to our app anytime. For simplicity, publishing it will be less complex.

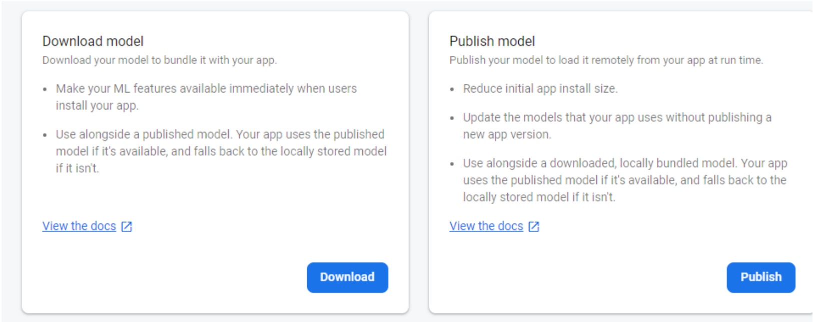


Figure 4-30: Choose how to deploy the trained model

When training completes, we can see performance metrics for the model. The objective is to determine the score threshold that works best for our model. The score threshold is the minimum confidence the model must have for it to assign a label to an image. By moving the score threshold slider, we can see how different thresholds affect the model's performance. Model performance is measured using two metrics: precision and recall.

In this project, since the datasets has less than 100 images for one label, the score threshold set to 0.5 to get average result. More datasets trained are allowed us to have higher score threshold in project which result in higher accuracy.

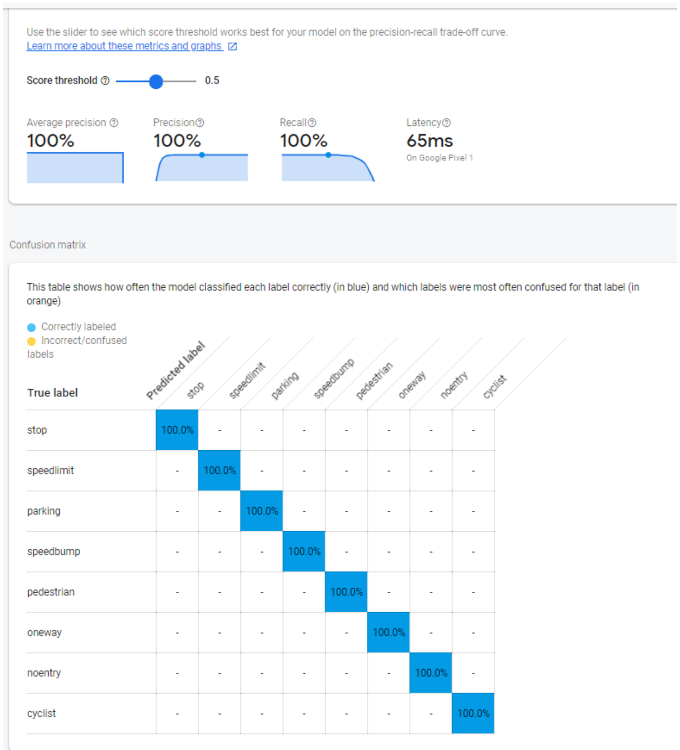


Figure 4-31: Result of training

The figures below are the result in Android Emulator using the datasets trained in ML Kit. 12 results of detection are recorded in figures below. 8 out of 12 results are correct while the rest of results are wrong.

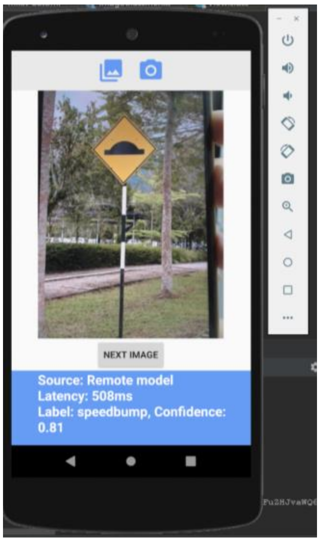


Figure 4-32: Speedbump signboard with accuracy of 81%

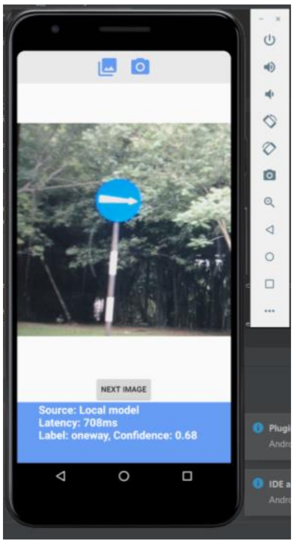


Figure 4-33: One-way signboard with accuracy of 68%

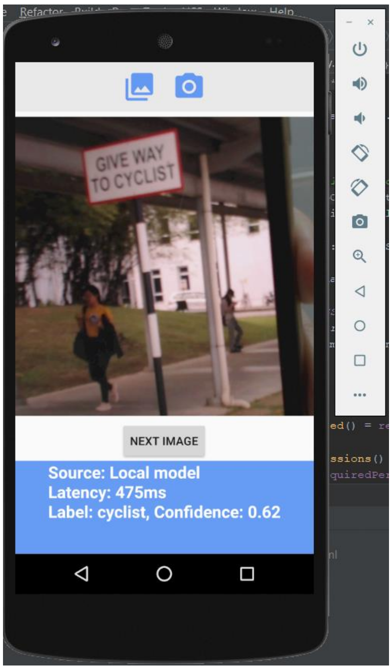


Figure 4-34: Cyclist signboard with accuracy of 62%

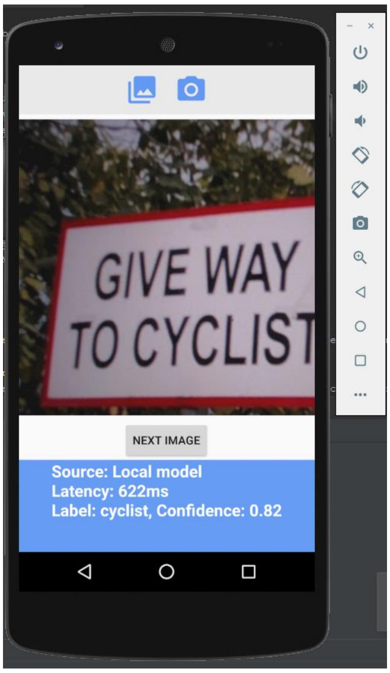


Figure 4-35: Cyclist signboard with accuracy of 82%

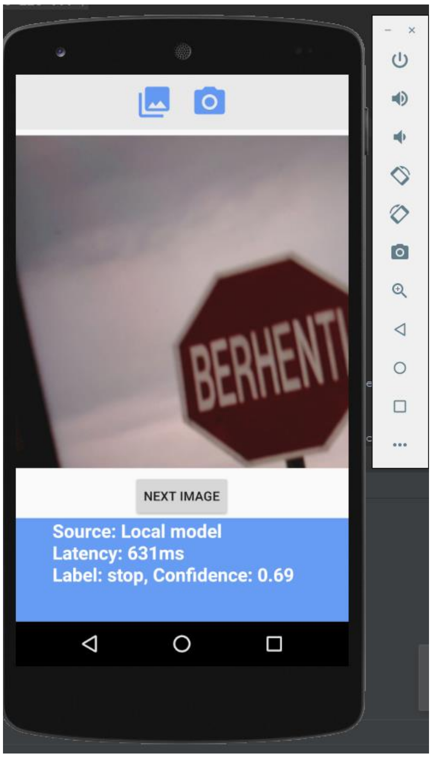


Figure 4-36: Stop signboard with accuracy of 69%

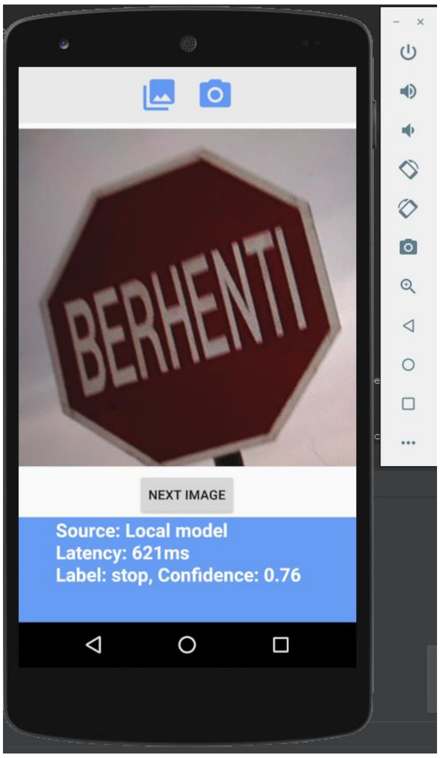


Figure 4-37: Stop signboard with accuracy of 76%

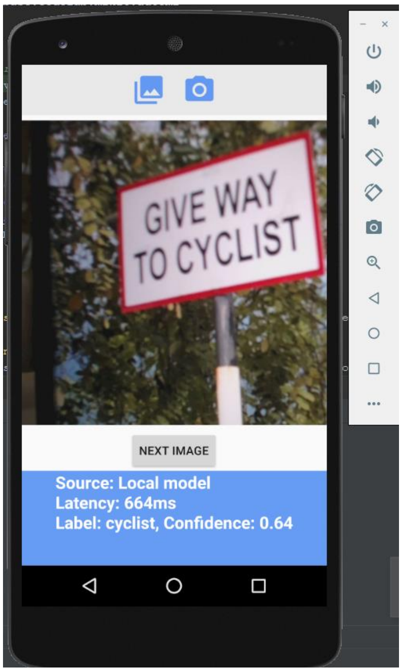


Figure 4-38: Cyclist signboard with accuracy of 64%

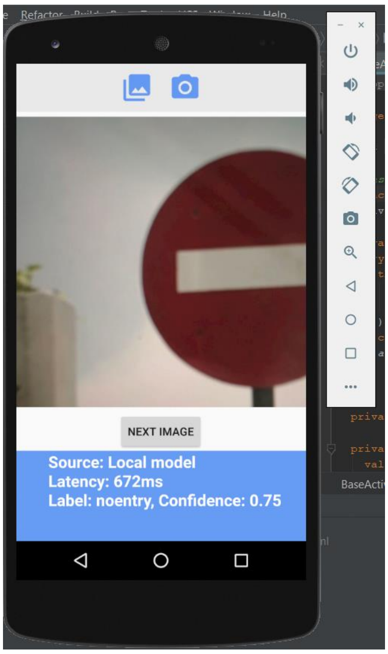


Figure 4-39: No-entry signboard with accuracy of 75%

There are 4 out of 12 results are wrong as shown below. There are 3 results failed to detect the presence of signboards. There is one result recognise wrongly the signboards (Figure 4-43). The reason of failure might be lack of datasets for training the system.

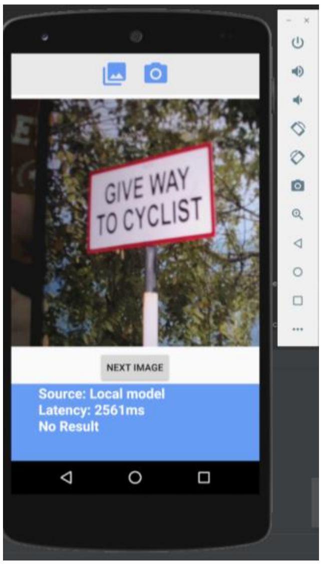


Figure 4-40: Failed to detect cyclist signboard

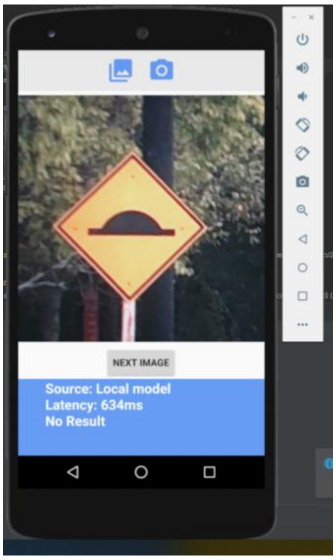


Figure 4-41: Failed to detect speedbump signboard

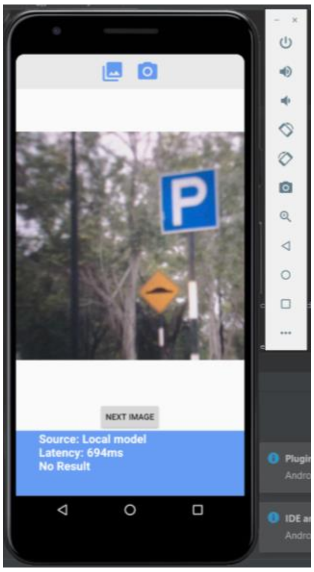


Figure 4-42: Failed to detect speedbump signboard and parking signboard

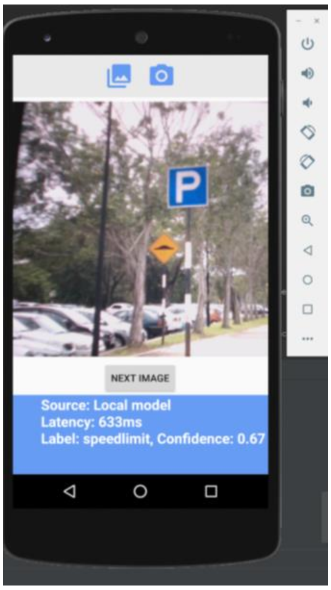


Figure 4-43: Wrong detection of signboard

4.2.3 Comparison of Object Detection Method

Table 4-19: Comparison of Object Detection Method

Object detection method	***YOLOv3***	***SSD-Mobile net in Android***	***Firebase ML Kit***
Detection input	Real-time video	Real-time video	Photo
Detection speed	Fast	Moderate	Slow
Fee	Free of charge	Free of charge	Charged Fee
Accuracy	Accurate	Less accurate	Least accurate
Implementation in Android phone	Cannot be implemented in phone	Can be implemented in phone	Can be implemented in phone
Is it chosen for project?	Not chosen because it cannot be implemented on phone	Chosen because it is accurate and can be implemented on phone.	Not chosen because it is not accurate and not in real-time.

From the analysis, the performance of YOLOv3 running in laptop is the best. It has highest accuracy and fastest speed. YOLOv3 is most suitable for real-time processing among the three method. However, YOLOv3 cannot be implemented in Android phone so far. Android phone does not have the strong GPU computability to support the processing of YOLOv3. Therefore, YOLOv3 is not chosen in this project.

Firebase introduces ML Kit, a machine learning SDK. It could bring powerful machine learning features to mobile application. No matter user are freshmen in machine learning, or an experienced machine learning developer, ML Kit make machine learning become as simple as it could be. However, user need to pay fee to enjoy the service. There is also free trial of ML Kit. But the accuracy of the system is very low. To increase the accuracy, fee must be paid to increase the training time. Another reason of not choosing Firebase ML Kit is it is not performing in real time. Hence, processing time is very slow.

Lastly, SSD-Mobile net in TensorFlow framework is chosen. The main reason is because it can perform in Android phone in the form of real-time. This is because it supports hardware acceleration with the Android Neural Networks API. However, the speed of detection is not as fast as the speed in laptop because of the processor in Android phone is limited. The accuracy of detection is also within the acceptance range.

Chapter 5 Conclusion

5.1 Conclusion

The objective of this project is to implement a cloud-based obstacle detection system for driver with the help of Artificial Intelligent. The cloud-based obstacle detection system for driver is able to detect and classify one or more obstacle captured by a smartphone. With the help of mobile application, the system provides the user real time information from road view. The information allows the driver to take suitable and correct decisions in order to drive safely on road. Besides, the system is able to show the real-time result from the obstacle detection on map in mobile application. All the objectives have been achieved.

From the result of main project, the cloud-based object detection system has been developed into Android smart phone. The objective of this part is to evaluate the performance of object detection system without the help of GPU and strong CPU as computer have. TensorFlow SSD-MobileNet algorithm was used to complete the object detection module. The object detection module is able to perform well to detect objects from different distance, under different weather and detect multiple objects in the same time. From overall observation, the percentage of accuracy of the object detection module is about 80%. Besides, from the result of map module, the GPS location of object detected can be retrieved and plotted on map correctly in most of time.

To develop object detection system in smartphone, YOLOv3 algorithm can no longer be used. Instead, and Firebase ML Kit are used to develop object detection system in Android smartphone. In short, the accuracy and speed of detection is very poor compared to YOLOv3 with the help of computer.

In part two, other object detection system such as YOLOv3 and Firebase ML Kit have been analysed. Performance of YOLOv3 is evaluated with different objects under different light condition, different weather, different distance, different speed and multiple objects. The performance of the system varies among these different conditions. The system is also tested with two models with different number of datasets. This is to investigate the relationship between the number of datasets training and the performance of object detection system. Based on the analysis, the more the training datasets, the better the performance of object detection system.

Besides, the object detection system built with Firebase ML Kit is also tested with simple input images of signboards. However, the accuracy is comparably lower than YOLOv3 and TensorFlow SSD-MobileNet. Besides, user need to pay fee to enjoy the service. Another reason of not choosing Firebase ML Kit is it is not performing in real time. Hence, processing time is very slow.

As a conclusion for the analysis, SSD-Mobile net in TensorFlow framework is chosen. The main reason is because it can perform in Android phone in the form of real-time. This is because it supports hardware acceleration with the Android Neural Networks API. However, the speed of detection is not as fast as the speed in laptop because of the processor in Android phone is limited. The accuracy of detection is also within the acceptance range.

5.2 Challenges

Three object detection method have been tried in this project. The most time-consuming part is the part of training data. It may take days to train a model in order to perform the object detection system. Besides, researching for these 3 methods are time-consuming too.

Initially, I used YOLO algorithm to implement the detection and recognition of road signs. The result of detection is accurate and fast. The advantage is it can perform well in real time. However, when I come to upload result to database, I found difficulty to upload the result to cloud and retrieve the result to mobile application.

Hence, I switch to use ML Kit in firebase to perform the signboard detection. Since Firebase is a mobile and web application development platform, this method will help me greatly in the further work on mobile application part.

However, the steps in ML Kit is comparably easier than implementing YOLOv3, so there is pricing in this feature. If there are more dataset need to be trained, more training hours is needed, higher the fee.

Eventually, I changed to use the SSD-MobileNet method in TensorFlow framework. It is successfully working in Android phone in real-time. Besides, the result can be uploaded to cloud and retrieved back to the mobile application. Its speed of detection is slower than YOLOv3, but faster than Firebase ML Kit.

Another challenge is I found difficulty when looking for similar application online to get some idea. However, there is no similar project online.

Besides, datasets must be more to increase the accuracy of result. The images collected must be in different angle and location. Therefore, problem will be faced when collecting such a larger amount of data.

5.3 Recommendation of improvement

The project can be improved by using an Android phone with better specification and processor. This is because the project is running with the CPU of the phone. With the help of better phone processor, the speed of detection can be increased.

Furthermore, it is recommended to train a custom model for the detection system. In this project, COCO datasets are used in order to save time because collecting and annotating datasets is time-consuming. However, there is total of 80 objects in COCO datasets which is too many for a obstacle detection system for driver. Therefore, it is recommended to train a custom model of datasets of the obstacle which is normally faced by driver. For example, road sign boards and traffic light signal.

Besides, the alertness to the driver as audio output can be embedded into the project. However, if the driver is not following the alert, the incident may occur too.

Hence, it is recommended the automatic braking system should be embedded and get activated when alerted so the speed of the vehicle gets regulated based on the signboard.

Reference

Brunette, E. S., Flemmer, R. C. and Flemmer, C. L. (2009). A review of artificial intelligence. ICARA 2009 - Proceedings of the 4th International Conference on Autonomous Robots and Agents, pp. 385–392.

Choi, J., Chun, D., Kim, H. and Lee, H., 2019. Gaussian YOLOv3: An Accurate and Fast Object Detector Using Localization Uncertainty for Autonomous Driving.

Girshick, R. Fast R-CNN. In Proc. IEEE Intl. Conf. on computer vision, pp. 1440-1448. 2015.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proc. IEEE Conf. on computer vision and pattern recognition (CVPR), pp. 580-587. 2014.

Google Cloud. 2019. Cloud Automl Vision Object Detection Documentation | Cloud Automl Vision Object Detection | Google Cloud. [online] Available at: <https://cloud.google.com/vision/automl/object-detection/docs/> [Accessed 4 November 2019].

Google Cloud. 2019. Mobile App Backend Services | Solutions | Google Cloud. [online] Available at: <https://cloud.google.com/solutions/mobile/mobile-app-backend-services> [Accessed 3 November 2019].

Hechri, A. and Mtibaa, A., 2012. Automatic detection and recognition of road sign for driver assistance system. 2012 16th IEEE Mediterranean Electrotechnical Conference.

International Journal of Innovative Technology and Exploring Engineering, 2019. Object Detection Method Based on YOLOv3 using Deep Learning Networks. 9(1), pp.1414-1417.

Jianmin Duan and Viktor, M., 2015. Real time road edges detection and road signs recognition. 2015 International Conference on Control, Automation and Information Sciences (ICCAIS).

Lin, T., Hays, J., Maire, M. and Perona, P., 2015. *Microsoft COCO: Common Objects In Context*. [online] Arxiv.org. Available at: <https://arxiv.org/pdf/1405.0312.pdf> [Accessed 31 March 2020].

May, (2017). Deep Learning and the Artificial Intelligence Revolution. Mongo DB. [online] Available at: https://www.mongodb.com/collateral/deep-learning-and-the-artificial-intelligence-revolution [Accessed 5 Aug. 2019].

Ongsulee, P. (2017). Artificial Intelligence, Machine Learning and Deep Learning. Fifteenth International Conference on ICT and Knowledge Engineering, pp. 1–6.

Phon-Amnuaisuk, S., Murata, K.T., Pavarangkoon, P., Yamamoto, K. and Mizuhara, T. (2018). Exploring the Applications of Faster R-CNN and Single-Shot Multi-Box Detection in Smart Nursery Domain. Available at: https://arxiv.org/pdf/1808.08675.pdf [Accessed 5 Aug. 2019].

Ren, S., He, K., Girshick, R. and Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), pp.1137-1149.

Tsang, S., 2020. *Review: SSD — Single Shot Detector (Object Detection)*. [online] Medium. Available at: <https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11> [Accessed 31 March 2020].

Wang, C., 2018. Research and Application of Traffic Sign Detection and Recognition Based on Deep Learning. 2018 International Conference on Robots & Intelligent System (ICRIS).

Weng, L. (2019). Object Detection for Dummies Part 3: R-CNN Family. [online] Lil'Log. Available at: https://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-dummies-part-3.html [Accessed 8 Aug. 2019].

Final Year Project Poster

CLOUD-BASED OBSTACLE DETECTION FOR DRIVERS

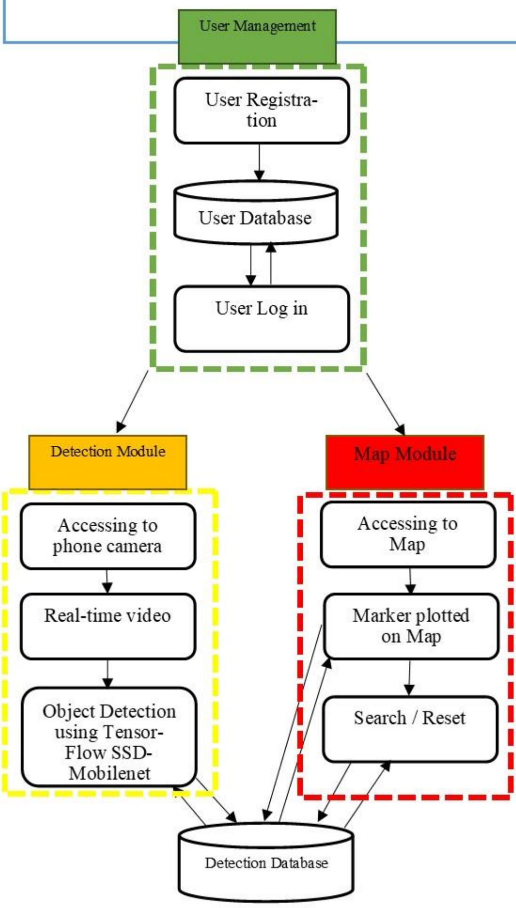
EIO HUA ZEN

Problem Statement

Every year, millions of people are killed and injured on roads. To solve this problem, it is advisable to develop system to assist human in driving. Therefore, auto driving is turning into a prominent point in numerous fields. It is critical to define, perceive traffic lights, street signs, humans on street, and other

Objective

- to detect and classify one or more obstacle captured by a smartphone in real-time.
- to show the real-time result from the obstacle detection on map in mobile application.



System Design

User Management Module
User must register for the first-time log in.

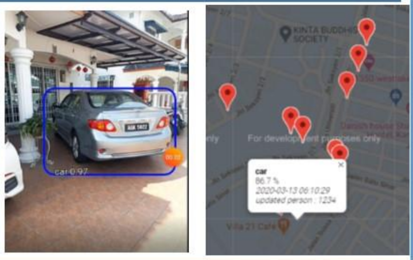
Detection module
Apps start detecting obstacles in front in real-time with camera.

Map Module
User can access to detected object on specific location with the form of map marker.

Search function is to allow user to search for specific object on the map to find out their location.

Reset function is to allow user to clear all the marker on map.

Result



Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Eio Hua Zen
ID Number(s)	1505979
Programme / Course	Bachelor of Information Systems (Hons) Information Systems Engineering
Title of Final Year Project	Cloud-Based Obstacle Detection System for Drivers

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: __18____ **%** **Similarity by source** Internet Sources: _____11______% Publications: ___5___ % Student Papers: ___12_____ %	
Number of individual sources listed of more than 3% similarity: _0______	

Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

____
Signature of Supervisor

Name: Lau Phooi Yee, PhD

Date: 20 April 2020

____
Signature of Co-Supervisor

Name: __________________________

Date: __________________________

1505979 FYP2

ORIGINALITY REPORT

18%	11%	5%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	hostingpill.com Internet Source	1%
2	Submitted to University of Warwick Student Paper	1%
3	blog.zenggyu.com Internet Source	1%
4	gaffis.com Internet Source	1%
5	firebase.google.com Internet Source	1%
6	tech.amikelive.com Internet Source	1%
7	www.educba.com Internet Source	1%
8	www.opencodez.com Internet Source	1%
9	Jiwoong Choi, Dayoung Chun, Hyun Kim, Hyuk-Jae Lee. "Gaussian YOLOv3: An Accurate and	<1%

	Fast Object Detector Using Localization Uncertainty for Autonomous Driving", 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019 Publication	
10	Submitted to Indian Institute of Technology Roorkee Student Paper	<1%
11	Submitted to University of Northumbria at Newcastle Student Paper	<1%
12	Submitted to International University - VNUHCM Student Paper	<1%
13	codelabs.developers.google.com Internet Source	<1%
14	cloud.google.com Internet Source	<1%
15	Hashir Ali, Mahrukh Khursheed, Syeda Kulsoom Fatima, Syed Muhammad Shuja, Shaheena Noor. "Object Recognition for Dental Instruments Using SSD-MobileNet", 2019 International Conference on Information Science and Communication Technology (ICISCT), 2019 Publication	<1%
16	Submitted to Curtin University of Technology Student Paper	<1%

17	www.tensorflow.org Internet Source	<1 %
18	Submitted to University of Sydney Student Paper	<1 %
19	Leo Gugerty. "Newell and Simon's Logic Theorist: Historical Background and Impact on Cognitive Modeling", Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 2016 Publication	<1 %
20	Submitted to Universiti Sains Malaysia Student Paper	<1 %
21	"Computer Vision and Image Processing", Springer Science and Business Media LLC, 2020 Publication	<1 %
22	Submitted to CSU, San Jose State University Student Paper	<1 %
23	morioh.com Internet Source	<1 %
24	Submitted to City University of Hong Kong Student Paper	<1 %
25	blog.paperspace.com Internet Source	<1 %

Submitted to University of Hong Kong

26	Student Paper	<1%
27	Submitted to Universiteit van Amsterdam Student Paper	<1%
28	towardsdatascience.com Internet Source	<1%
29	Submitted to Indian Institute of Technology, Madras Student Paper	<1%
30	Submitted to University of Macau Student Paper	<1%
31	Rabindra Bista, Awanish Ranjan. "A new approach to extract meaningful clinical information from medical notes", 2017 11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), 2017 Publication	<1%
32	Submitted to KDU College Sdn Bhd Student Paper	<1%
33	Ekaba Bisong. "Building Machine Learning and Deep Learning Models on Google Cloud Platform", Springer Science and Business Media LLC, 2019 Publication	<1%

34	intraceuticalsacasa.ro Internet Source	<1%
35	Submitted to Cork Institute of Technology Student Paper	<1%
36	Submitted to Universiti Teknologi MARA Student Paper	<1%
37	Submitted to University of Glasgow Student Paper	<1%
38	enbeeone3.com Internet Source	<1%
39	scholars.unh.edu Internet Source	<1%
40	Submitted to Universiti Putra Malaysia Student Paper	<1%
41	Submitted to Cranfield University Student Paper	<1%
42	image.ing.bth.se Internet Source	<1%
43	link.springer.com Internet Source	<1%
44	machinelearningmastery.com Internet Source	<1%
45	Submitted to Griffith College Dublin Student Paper	

		<1%
46	Submitted to University of East London Student Paper	<1%
47	Submitted to Thammasat University Student Paper	<1%
48	eprints.utar.edu.my Internet Source	<1%
49	Tanmoy Sen, Haiying Shen. "Machine Learning based Timeliness-Guaranteed and Energy-Efficient Task Assignment in Edge Computing Systems", 2019 IEEE 3rd International Conference on Fog and Edge Computing (ICFEC), 2019 Publication	<1%
50	Soo Siang Teoh, Thomas Bräunl. "Symmetry-based monocular vehicle detection system", Machine Vision and Applications, 2011 Publication	<1%
51	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1%
52	"Recent Advances in Information and Communication Technology 2020", Springer Science and Business Media LLC, 2020 Publication	<1%

53	en.wikipedia.org Internet Source	<1%
54	Submitted to AUT University Student Paper	<1%
55	Submitted to University of Bristol Student Paper	<1%
56	Submitted to Universidad Carlos III de Madrid Student Paper	<1%
57	studentsrepo.um.edu.my Internet Source	<1%
58	www.renewableenergy.go.ke Internet Source	<1%
59	Submitted to Bahcesehir University Student Paper	<1%
60	tekinfoway.in Internet Source	<1%
61	Submitted to Netaji Subhas Institute of Technology Student Paper	<1%
62	library.binus.ac.id Internet Source	<1%
63	eprints.utm.my Internet Source	<1%

64	maxwellsci.com Internet Source	<1%
65	d-nb.info Internet Source	<1%
66	Submitted to Universiti Malaysia Perlis Student Paper	<1%
67	electronics360.globalspec.com Internet Source	<1%
68	Submitted to DeVry, Inc. Student Paper	<1%
69	Submitted to Multimedia University Student Paper	<1%
70	Submitted to Turun yliopisto Student Paper	<1%
71	Xu Zhang, Haipeng Wang, Dazhou Zhou, Jianxiang Li, Haibo Liu. "Abnormal Detection of Substation environment based on Improved YOLOv3", 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), 2019 Publication	<1%
72	Submitted to University of Queensland Student Paper	<1%
73	Gu Hao, Yang Yingkun, Qu Yi. "General Target	<1%

Detection Method Based on Improved SSD", 2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), 2019

Publication

74	Submitted to The University of Manchester Student Paper	<1%
75	www.mdpi.com Internet Source	<1%
76	Submitted to The Hong Kong Polytechnic University Student Paper	<1%
77	Submitted to University of Ulster Student Paper	<1%
78	Gong Cheng, , Junwei Han, Peicheng Zhou, and Lei Guo. "Scalable multi-class geospatial object detection in high-spatial-resolution remote sensing images", 2014 IEEE Geoscience and Remote Sensing Symposium, 2014. Publication	<1%
79	Zhang, Guofeng, Weida Zhou, Weihua Ren, Shuang Liu, Mingyue Ding, F. Wahl, and Yaoting Zhu. "", MIPPR 2007 Pattern Recognition and Computer Vision, 2007. Publication	<1%
80	clarendonhills.us	

Internet Source

<1%

---

81 Submitted to Queen Mary and Westfield College <1%
Student Paper

---

82 P.R. Sriram, Sandeep Kumar Ramani, Ram V Shrivatsav, Muthu M.Mankiandan, Nithin Ayyappaa. "Autonomous Drone for Defence Machinery Maintenance and Surveillance", 2019 Third World Conference on Smart Trends in Systems Security and Sustainablity (WorldS4), 2019 <1%
Publication

---

83 Submitted to Universiti Teknologi Malaysia <1%
Student Paper

---

84 Submitted to Islamic University of Gaza <1%
Student Paper

---

85 Shyla Raj, D.S. Vinod. "Automatic defect identification and grading system for 'Jonagold' apples", 2016 Second International Conference on Cognitive Computing and Information Processing (CCIP), 2016 <1%
Publication

---

86 Submitted to University of Bath <1%
Student Paper

---

87 Submitted to University of California, Los <1%

Angeles
Student Paper

88	"Biometric Recognition", Springer Science and Business Media LLC, 2017 Publication	<1%
89	Submitted to Asian Institute of Technology Student Paper	<1%
90	Lecture Notes in Computer Science, 2013. Publication	<1%
91	Submitted to University of Hertfordshire Student Paper	<1%
92	Submitted to Laureate Education Inc. Student Paper	<1%
93	Submitted to University of Bedfordshire Student Paper	<1%
94	Submitted to University of Surrey Student Paper	<1%
95	Submitted to Higher Education Commission Pakistan Student Paper	<1%
96	arxiv.org Internet Source	<1%
97	Elizabeth Hull, Ken Jackson, Jeremy Dick. "Requirements Engineering", Springer Science and Business Media LLC, 2011	<1%

Publication

98	Hisham El-Amir, Mahmoud Hamdy. "Deep Learning Pipeline", Springer Science and Business Media LLC, 2020 Publication	<1%
99	"Machine Learning and Deep Learning", International Journal of Innovative Technology and Exploring Engineering, 2019 Publication	<1%
100	worldwidescience.org Internet Source	<1%
101	S. Onis, H. Sanson, C. Garcia. "Iterative unsupervised object detection system", 2008 15th International Conference on Systems, Signals and Image Processing, 2008 Publication	<1%
102	"Innovations in Smart Cities Applications Edition 3", Springer Science and Business Media LLC, 2020 Publication	<1%
103	Hüseyin Kutlu, Engin Avci, Fatih Özyurt. "White blood cells detection and classification based on regional convolutional neural networks", Medical Hypotheses, 2020 Publication	<1%
104	Submitted to Wageningen University Student Paper	<1%

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	1505979
Student Name	Eio Hua Zen
Supervisor Name	Dr. Lau Phooi Yee

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Front Cover
√	Signed Report Status Declaration Form
√	Title Page
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report. ______________________ (Signature of Student) Date:20/4/2020	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. ______________________ (Signature of Supervisor) Date: 20 April 2020