

# **REAL TIME SIGN LANGUAGE TRANSLATOR**

**MUHAMAD HAIROL IKRAM BIN MOHD NAZRI**



اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

**UNIVERSITI TEKNIKAL MALAYSIA MELAKA**

# **REAL TIME SIGN LANGUAGE TRANSLATOR**

**MUHAMAD HAIROL IKRAM BIN MOHD NAZRI**

**This report is submitted in partial fulfilment of the requirements  
for the degree of Bachelor of Electronic Engineering with Honours**



**Faculty of Electronic and Computer Engineering  
Universiti Teknikal Malaysia Melaka**

**2022**

## DECLARATION

I declare that this report entitled Real Time Sign Language Translator is the result of my work except for quotes as cited in the references.



Signature : .....

---

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

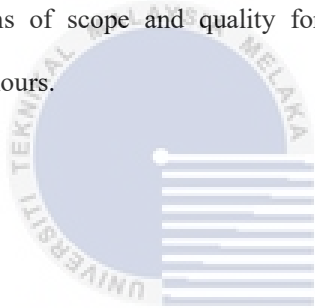
Author : MUHAMAD HAIROL IKRAM

BIN MOHD NAZRI

Date : 11 JANUARY 2022

## APPROVAL

I hereby declare that I have read this thesis, and, in my opinion, this thesis is sufficient in terms of scope and quality for the award of Bachelor of Electronic Engineering with Honours.



Signature

اونيورسيتي تیکنیکل ملیسیا ملاک

Supervisor Name

: DR SITI NORMI BINTI ZABRI @ SUHAIMI

Date

: 11 JANUARY 2022

## DEDICATION

This final is dedicated specially to my beloved parents and friends for their endless support and help whenever I need it, always praying the best for me and giving the useful advice

To my supervisor who always guide and gave supports to me during this final project.

Dr Siti Normi Binti Zabri @ Suhaimi

To my lecturer who gives a lecture on Artificial Intelligent that help me to understand this

اونيورسيتي تيكنيكل مليسيا ملاك project

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Dr. Wira Hidayat Bin Mohd Saad

To all my beloved friends who are always there to help me solve the problem, give me an idea, and mostly give me endless support for me to finish the project.

## ABSTRACT

Speech impairment is a disability that affects an individual's ability to communicate using speech and hearing. People who are affected by this use other media of communication such as sign language. Although sign language is ubiquitous in recent times, there remains a challenge for non-sign language speakers to communicate with sign language speakers or signers. With recent advances in deep learning and computer vision, there has been promising progress in the fields of motion and gesture recognition using deep learning and computer vision-based techniques. In recent years, deep learning has been used in image classification, object tracking, action recognition, and scene labeling. Traditionally, Image Processing techniques were used to solve any Computer Vision problems that occurred in an artificial intelligence system. However, in real-time identification, image processing cannot be used. This is where Deep Learning concepts are applied. The focus of this work is to create a real time sign language translator which offers sign language translation to text thus aiding communication between signers and non-signers. We built a simple Convolutional Neural Network for object detection. The model is trained, and multiple test cases are implemented in the TensorFlow environment to obtain accurate results.

## ABSTRAK

Kecacatan pertuturan ialah ketidakupayaan yang menjejaskan keupayaan individu untuk berkomunikasi menggunakan pertuturan dan pendengaran. Orang yang terjejas oleh ini menggunakan media komunikasi lain seperti bahasa isyarat. Walaupun bahasa isyarat ada di mana-mana sejak kebelakangan ini, masih terdapat cabaran bagi pengguna bukan bahasa isyarat untuk berkomunikasi dengan pengguna bahasa isyarat. Dengan kemajuan terkini dalam “Deep Learning” dan “Computer Vision”, terdapat kemajuan yang boleh diaplikasi dalam bidang pergerakan dan pengecaman gerak isyarat menggunakan “Deep Learning” dan teknik berasaskan “Computer Vision”. Dalam beberapa tahun kebelakangan ini, “Deep Learning” telah digunakan dalam klasifikasi imej, penjejakan objek, pengecaman tindakan dan pelabelan data. Secara tradisinya, teknik pemprosesan imej digunakan untuk menyelesaikan sebarang masalah “Computer Vision” yang berlaku dalam sistem “Artificial Intelligent”. Walau bagaimanapun, dalam pengenalan masa nyata, pemprosesan imej tidak boleh digunakan. Di sinilah konsep “Deep Learning” diterapkan. Fokus kerja ini adalah untuk mencipta penterjemah bahasa isyarat masa nyata yang menawarkan terjemahan bahasa isyarat kepada teks sekali gus membantu komunikasi antara pengguna dan bukan pengguna bahasa isyarat. Kami membina rangkaian “Neural Convolutional” yang mudah untuk pengesanan objek. Model ini dilatih dan berbilang situasi ujian dilaksanakan dalam persekitaran “TensorFlow” untuk mendapatkan hasil yang tepat.

## ACKNOWLEDGEMENTS

Bismillahirrahmanirrahim,

First of all, I would like to give all the praise to Allah S.W.T for giving me the strength and patience for the whole process of completing this project. Without blessed from Him, I cannot complete this project according to what has been planned. Also, not to forget for my family that always being there for me during the final project and gave me endless support and advice to complete this project.

Next, I would like to appreciate my supervisor, Dr. Siti Normi Binti Zabri @ Suhaimi for guiding me along the way to make this final year project complete. He gives me the idea and help me to complete this project.

Besides, I would like to thank all my friends for being with me from day one until I complete this final year project.

Last but not least, thank you to my lecturer, Dr. Wira Hidayat Bin Mohd Saad for teaching me the subject of Artificial Intelligent and it helps me to more understand my project.



## TABLE OF CONTENTS

<b>DECLARATION</b> .....	
<b>APPROVAL</b> .....	
<b>DEDICATION</b> .....	
<b>ABSTRACT</b> .....	<b>i</b>
<b>ABSTRAK</b> .....	<b>ii</b>
<b>ACKNOWLEDGEMENTS</b> .....	<b>iii</b>
<b>TABLE OF CONTENTS</b> .....	<b>iv</b>
<b>LIST OF FIGURES</b> .....	<b>vii</b>
<b>LIST OF TABLES</b> .....	<b>x</b>
<b>LIST OF APPENDICES</b> .....	<b>xi</b>
<b>1. CHAPTER 1 INTRODUCTION</b> .....	<b>1</b>
1.1 Introduction .....	1
1.2 Background of the project .....	1
1.3 Problem Statement .....	3
1.4 Motivation .....	5

1.5	Objectives .....	5
1.6	Scope of Project .....	6
1.7	Sign Language .....	6
1.8	Thesis Outline .....	9
<b>2.</b>	<b>CHAPTER 2 BACKGROUND STUDY .....</b>	<b>11</b>
2.1	Literature Review.....	11
<b>3.</b>	<b>CHAPTER 3 METHODOLOGY .....</b>	<b>22</b>
3.1	System overview.....	22
3.2	Methodology.....	23
3.3	Capture Image.....	24
3.4	Collect image .....	24
3.5	Image Labelling .....	25
3.6	Train the image/model.....	27
3.7	Evaluate the model.....	30
3.8	Image recognition .....	30
<b>4.</b>	<b>CHAPTER 4 RESULT AND DISCUSSION .....</b>	<b>31</b>
4.1	Processing time .....	31
4.2	Training and Evaluation.....	32
4.3	Ssd mobilenet v2 training result 2000 steps vs 10000 steps .....	34
4.3.1	2000 steps of training.....	34
4.3.2	10000 steps of training.....	36
4.4	Ssd mobilenet v2 evaluation result 2000 vs 10000.....	38

4.4.1	2000 steps .....	39
4.4.2	10000 steps .....	41
4.5	Detection and Translation Results in Real Time.....	44
4.5.1	Situation 1(one sign on one person) .....	44
4.5.2	Situation 2 (two sign on one person) .....	47
4.5.3	Situation 3 (Two sign on two people) .....	49
<b>5.</b>	<b>CHAPTER 5 CONCLUSION AND FUTURE WORK .....</b>	<b>51</b>
5.1	Conclusion .....	51
5.2	Future Work.....	52
<b>6.</b>	<b>REFERENCE .....</b>	<b>54</b>
	<b>APPENDIX A: DEFINE IMAGES TO COLLECT IMAGE.....</b>	<b>58</b>
	<b>APPENDIX B: CAPTURE IMAGE BY WEBCAM.....</b>	<b>59</b>
	<b>APPENDIX C: IMAGE LABELLING.....</b>	<b>60</b>
	<b>APPENDIX D: TRAIN THE MODEL.....</b>	<b>61</b>
	<b>APPENDIX E: EVALUATE THE MODEL .....</b>	<b>62</b>
	<b>APPENDIX F: DETECT FROM AN IMAGE .....</b>	<b>63</b>
	<b>APPENDIX G: REAL TIME DETECTION FROM WEBCAM.....</b>	<b>64</b>
	<b>APPENDIX H: ZIP AND EXPORT MODEL .....</b>	<b>65</b>

## LIST OF FIGURES

Figure 1.1: The essential elements related to sign language gesture formation .....	7
Figure 1.2: Sign Language Recognition Approaches.....	7
Figure 1.3: A flow chart of the processing steps used in the vision-based system for sign language translation.....	8
Figure 2.1: Proposed Sign Language System .....	13
Figure 2.2: Placement of sEMG electrodes.....	14
Figure 2.3: Kinect sensor components. ....	15
Figure 2.4: False recognition because of the similarity in the gestures.....	16
Figure 2.5: The result that shows an original image and image after Skin Colour Detection .....	17
Figure 2.6: System architecture for their smartphone depth image-based real-time sign language translation system.....	17
Figure 2.7: Proposed Data Glove and sensor position .....	18
Figure 2.8: Modification of some signs from ASL .....	19
Figure 2.9: Fail recognize phone call sign gesture in 50k model .....	20
Figure 2.10: Prototype of the designed glove uses low-cost hardware. ....	21
Figure 3.1: The overview of the system.....	23
Figure 3.2: Hand sign gesture (“Hello”, “I Love You”, “No”, “Thank You” and “Yes”) .....	24

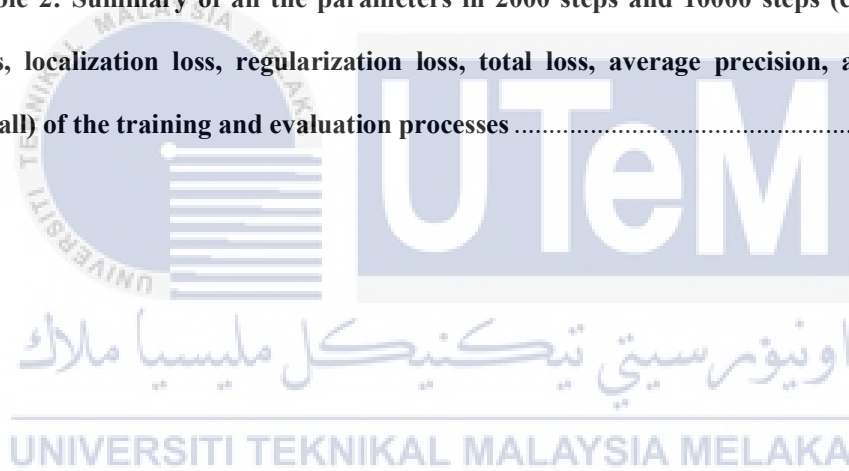
Figure 3.3: Command shows that webcam capture sign gesture “ILoveYou” was successfully collected.....	25
Figure 3.4: Hello gesture from Image Labelling package .....	26
Figure 3.5: I Love You gesture from Image Labelling package .....	26
Figure 3.6: Thank You gesture from Image Labelling package.....	26
Figure 3.7: No gesture from Image Labelling package .....	27
Figure 3.8: Yes, gesture from Image Labelling package .....	27
Figure 3.9: The speed and power consumption of the network is proportional to the number of MACs (Multiply-Accumulates).....	28
Figure 4.1: My CPU used 98% while training the data.....	32
Figure 4.2: The loss matric shown in the command prompt.....	34
Figure 4.3: The loss matric shown in TensorBoard.....	35
Figure 4.4: The loss matric shown in the command prompt.....	36
Figure 4.5: The loss matric shown in TensorBoard.....	37
Figure 4.6: The evaluation result shown on the command prompt.....	39
Figure 4.7: The evaluation result is shown on TensorBoard .....	40
Figure 4.8: The evaluation result is shown on Command Prompt.....	41
Figure 4.9: The evaluation result is shown on TensorBoard .....	42
Figure 4.10: Sign gesture for Hello in real time show 99%.....	44
Figure 4.11: Sign gesture for Yes in real time show 99%.....	45
Figure 4.12: Sign gesture for No in real time show 100%.....	45
Figure 4.13: Sign gesture for I Love You in real time show 94%.....	46
Figure 4.14: Sign gesture for Thank You in real time show 96% .....	46
Figure 4.15: Sign gesture for Hello and Yes with both hands in real time show 98%...	47
Figure 4.16: Sign gesture for I Love You and Yes with both hands in real time show 99% and 98% .....	47
Figure 4.17: Sign gesture in real time for Yes with right hand show 95% while left hand show 99%.....	48

<b>Figure 4.18: Sign gesture in real time for No with both hands show 99% .....</b>	<b>48</b>
<b>Figure 4.19: Sign gesture with two people in real time for Hello 99% and I Love You 100% .....</b>	<b>49</b>
<b>Figure 4.20: Sign gesture with two people in real time for I Love You 99% and Hello 100% .....</b>	<b>49</b>
<b>Figure 4.21: Sign gesture with two people in real time for Yes 98% and No 99% .....</b>	<b>50</b>
<b>Figure 4.22: Sign gesture with two people in real time for No 100% and Yes 100% ....</b>	<b>50</b>



## LIST OF TABLES

<b>Table 1: MobileNet Body Architecture</b> .....	<b>29</b>
<b>Table 2: Summary of all the parameters in 2000 steps and 10000 steps (classification loss, localization loss, regularization loss, total loss, average precision, and average recall) of the training and evaluation processes</b> .....	<b>43</b>



## LIST OF APPENDICES

<b>APPENDIX A: DEFINE IMAGES TO COLLECT IMAGE .....</b>	<b>61</b>
<b>APPENDIX B: CAPTURE IMAGE BY WEBCAM .....</b>	<b>62</b>
<b>APPENDIX C: IMAGE LABELLING .....</b>	<b>63</b>
<b>APPENDIX D: TRAIN THE MODEL .....</b>	<b>64</b>
<b>APPENDIX E: EVALUATE THE MODEL .....</b>	<b>65</b>
<b>APPENDIX F: DETECT FROM AN IMAGE.....</b>	<b>66</b>
<b>APPENDIX G: REAL TIME DETECTION FROM WEBCAM.....</b>	<b>67</b>
<b>APPENDIX H: ZIP AND EXPORT MODEL .....</b>	<b>68</b>



# CHAPTER 1



## INTRODUCTION



### 1.1 Introduction.

This chapter addresses the introductory theme of the project. It sets out the background of the project, the motivation of the work, the definition of the problem, the objectives of the project, its scope, and the thesis online.

### 1.2 Background of the project

One of the most fundamental aspects of human life is communication. Most individuals communicate verbally, however certain persons with restricted abilities must utilize sign language with hand and finger motions to communicate, even if not everyone understands what it means. A person who is deaf and hearing-impaired has an impairment to any portion of the ears as well as hearing loss. The biggest issue that Deaf individuals

encounter as a result of this handicap is that they have been unable to communicate effectively with others. They rely heavily on interpreters to help them communicate because not everyone, particularly youngsters, could use sign language [1]. Unfortunately, when these persons wish to connect with the public through sign language, the community is unable to provide constructive comments since they do not understand sign language. The greatest challenge for hearing-impaired persons is communicating with others [2] since they lack access to spoken and written language, especially if youngsters have not yet been introduced to sign language [3]. This difficulty is exacerbated when ordinary people do not comprehend sign language, and the majority of them do not study or understand the language [4].

Malaysia Sign Language (MSL) is the most widely used sign language in Malaysia [5]. Because sign language has not been standardized in the global standard [6,] the standardized sign language in one nation may differ from the standardized sign language in another. Malaysia Sign Language is related to American Sign Language (ASL) and French Sign Language (FSL) thus communication was not difficult. As a result, there is a need to design a system that can interpret sign language so that it may be utilized as a medium of communication or for learning sign language [7]. Camera technology may be used to identify objects, particularly human motion detection incomplete or specific body parts such as the face, facial expression, hand, or leg, on digital video shot by the camera or in real-time [8]. Despite the presence of digital processors in everyday life, humans have a great desire to make their instructions clearer to their computers. Human-computer interaction (HCI) does have limits. The majority of HCI forms are included in views of software and hardware interfaces like keyboards, mice, light pens, and so on. As a result, the transfer of meaning and notion is aimed at the lowest levels. The development of increasingly complex computers, equipped cameras, and software algorithms in machine learning, pattern recognition, and signal and image processing has enabled machines to watch their

surroundings. They drew inspiration from their perception creation using artificial intelligence (AI) [9].

### 1.3 Problem Statement

The main purpose of this research is to explore the challenges currently facing the deaf while interacting with unimpaired people and then provide a solution that can facilitate this communication. Hearing-impaired people need to communicate with others through sign language, a language that people without disabilities might ignore, leading to people with hearing impairments having trouble communicating in a social setting. A sign translator system can be developed using different approaches, such as using sensor gloves to collect accurate hand configuration data, or by using cameras to later process images and generate models to translate signs and so on. Furthermore, it is very popular since the development of new machine learning techniques and is the approach that will be followed by the proposed system. There are two traditional methods of communication between deaf people and hearing people who do not know sign language: interpreters and text writing. For everyday talks, interpreters are quite expensive, and their presence will result in a loss of privacy and freedom for deaf people. Texting is not an efficient technique to communicate since it is slower than speaking and the facial emotions made when doing sign language or speaking are lost. As a result, a low-cost, more efficient method of permitting communication between hearing and deaf persons is required. A sign language translator (SLT) system is a valuable tool for facilitating communication between deaf persons and hearing people who do not understand sign language by converting sign language into text. The technology, which may be worn by deaf persons who are unable to communicate, transforms the signs produced into text or voice on the mobile phone of those who can hear and speak. A deaf person's mobile phone's voice translation function converts spoken words into sign language visuals or movies. This study does not take into account the speech translator component. They can converse more easily and organically thanks to real-time translation. One component of this

purpose is to investigate the obstacles and alternative solutions for computer-supported Malaysia Sign Language (MSL) into text, with the primary users, in this case, being the disabled and the community.

The problem covered in this work is to develop a system that allows the translation of the language of Malaysia Sign Language and that gives autonomy to the person with disabilities using artificial vision. A real-time sign language translator system can be developed using different approaches, such as using OpenCV to collect image data for deep learning and by using the camera to recognize the gestures and generate the gestures to text. This study explores the challenges faced by deaf people while interacting with unimpaired. To understand and realize these challenges, sign languages, Malaysia Sign Language, and the history and development of MSL are also explored. The main goal of this thesis was to develop a system of an automated and generate system that converts Malaysia Sign Language (MSL) to text that can be used to facilitate communication among deaf and unimpaired. This work explores and analyses existing systems targeting this problem area.

Previous research has focused on vision-based and glove-based SLR systems, which gather signs using cameras and sensory glove devices, respectively. Vision-based approaches often need cameras to be installed in an area with a limited field of vision. Furthermore, the necessary infrastructure may not be accessible in all desirable places or maybe too expensive to implement. Concerns about user privacy restrict the utility of vision-based approaches. Glove-based SLR systems are not suitable for everyday usage due to the expensive cost of glove devices. So, in this study, we attempt to address the issue raised in the previous study by presenting a cost-effective solution that does not rely on gloves and does not overcomplicate the work system.

## 1.4 Motivation

Increased processing capacity, advances in machine learning techniques, and the amount of data available have improved the performance of image translator systems, allowing to development of projects in areas such as home automation, biology, genetics, augmented reality, security, among others. Many works have been carried out that try to address the problem of interpretation of sign languages, going to propose sign translator systems for different languages, including the Indian Sign Language, Japanese Sign Language, Mexican Sign Language, and so on. As for the development of a first-person sign translator system, most developed systems based on artificial vision involve the use of a camera supported by a third person, an approach that may be inappropriate in some cases, since the person with disabilities will not always meet people with who want to communicate that they have a translation system. Allowing hearing impaired people to have a system that can translate their signs can be considered more useful. Part of the system developed in this project serves as the basis for the development of sign language translation systems, especially the Malaysia Sign Language, which could benefit the population of hearing-impaired people in Malaysia.

## 1.5 Objectives

The overall objective of this work is to apply deep learning techniques for the translation of signs of the Malaysia sign language. The development of this project entails the definition of the following specific objectives:

- I. To design the dataset with first-person captured sign images
- II. To analyze sign language in real-time and label images for object detection.
- III. To identify the deep learning techniques for the recognition of sign language to be more accurate.

## 1.6 Scope of Project

The suggested idea is to create a Malay device system for Sign Language. The suggested initiative would also create responsiveness, allowing normal individuals to have a better communication channel with Deaf people. Aside from that, the project proposes to provide benefits to users for improved communication, such as allowing teachers or instructors to utilize them as a teaching tool.

This study focuses on mobile technology, where laptop users who use Windows may utilize the system to learn MSL to communicate more effectively with Deaf individuals because this system is only available in Windows. This research will also focus on Deaf individuals in Malaysia since this system was designed based on Malaysian duty to the Deaf population. Because not everyone knows how to use sign language, the intended user is among ordinary people. Deaf persons, on the other hand, can utilize the system as an extra tool. Comprehension of the language structure and having a comprehensive collection of signals for diverse scenarios require a solid understanding of MSL. As a result, more MSL research will be done to address these difficulties. The findings will be used to create a database of indications to be used in the system. This project's study focus consists of discovering the best design that can be used in constructing the system logic and interface. This initiative will help consumers since practically everyone nowadays, especially the younger age, understands how to utilize technology. Learning may be easily incorporated through the use of technology.

## 1.7 Sign Language

Sign language is a visual-spatial language that is based on positional and visual components such as finger and hand form, hand placement and orientation, and arm and body motions. These elements work together to express the meaning of a concept. Sign language's phonological structure typically consists of five parts (Figure 1.1). Each sign language motion is made up of five basic pieces. These five blocks reflect valuable features

of sign language and can be used by automated intelligence systems to translate sign language [10].

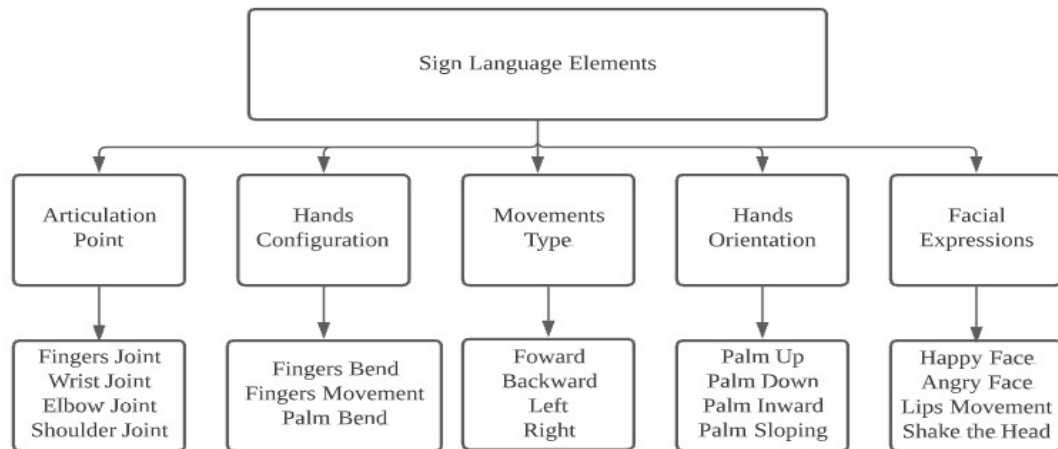


Figure 1.1: The essential elements related to sign language gesture formation

Scholarly approaches for overcoming disability-related challenges are many and methodical, and they differ according to the setting. One of the main interventions is sign language translation systems, which are used to convert sign language signs into the text to communicate with those who do not know these signals [11]. To record hand configurations and detect the related meanings of gestures, three techniques (Figure 1.2) are used [12]. These are vision-based, sensor-based, and a mix

of the two.

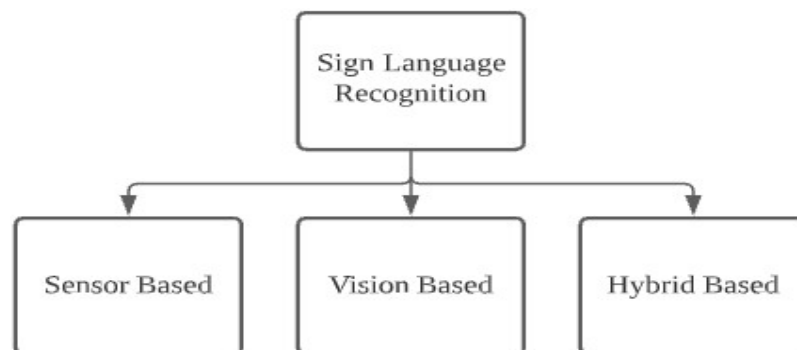


Figure 1.2: Sign Language Recognition Approaches

Cameras are the primary instruments used by vision-based systems to gather the essential input data (Figure 1.3). Cameras are very inexpensive, yet due to the blur generated by a web camera, most laptops employ a high-spec camera. However, despite the high-spec camera that most smartphones have [13], some issues must be addressed, including the limited field of view of the capturing device, high computational costs [14], and the need for multiple cameras to obtain results (due to depth and occlusion problems [15]). These issues are inherent to this system and render the entire system useless for the development of real-time recognition applications.

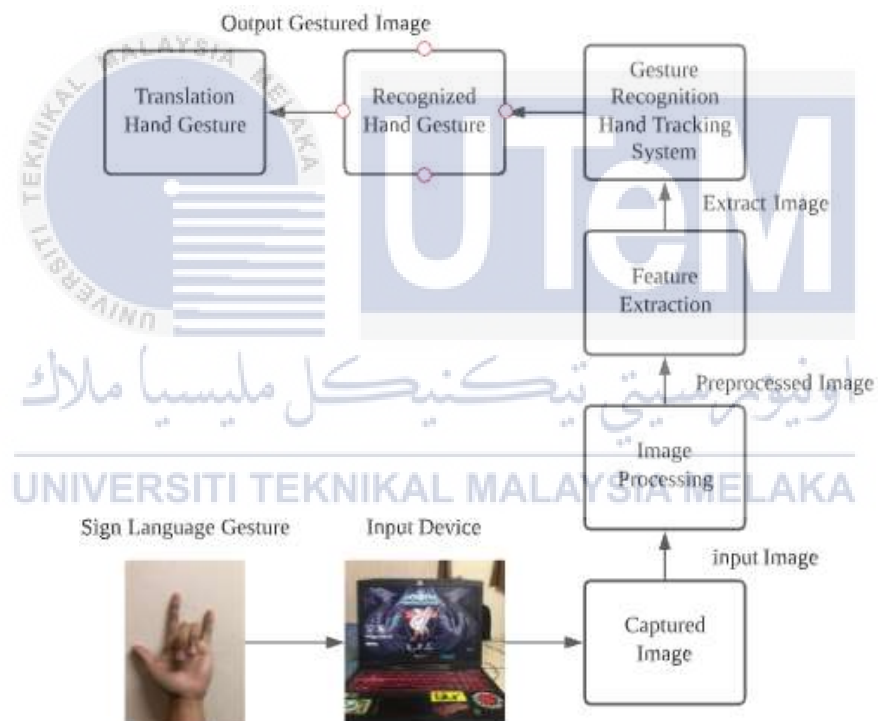


Figure 1.3: A flow chart of the processing steps used in the vision-based system for sign language translation.



We have collected 50 images for each of the 5 signs of MSL first and we train and test our system using samples collected for 2000 steps from 1 person and also test it with other people. 7 participants took part in the experiment. The experiment was conducted in two phases, firstly, we will take an image of a sign gesture, label the image, and train the image and test it with ourselves. Next to the second phase, the other six participants will test a system created based on our image of sign gestures taken, labeled, and trained. From this, we can observe the result test on the accuracy recognition of sign gestures that we developed. Furthermore, we do not use disabled people because it is hard to approach them because of this COVID-19 pandemic. Therefore, we use a colleague around us to make this system work.

### 1.8 Thesis Outline

This section describes the organization of the rest of the document.

Chapter 1 Introduction: In this chapter, presents the principle of sign language and machine learning which leads to the motivation of this work and the problem statement we are addressing, our motivation to choose this problem, and a preview of our solution.

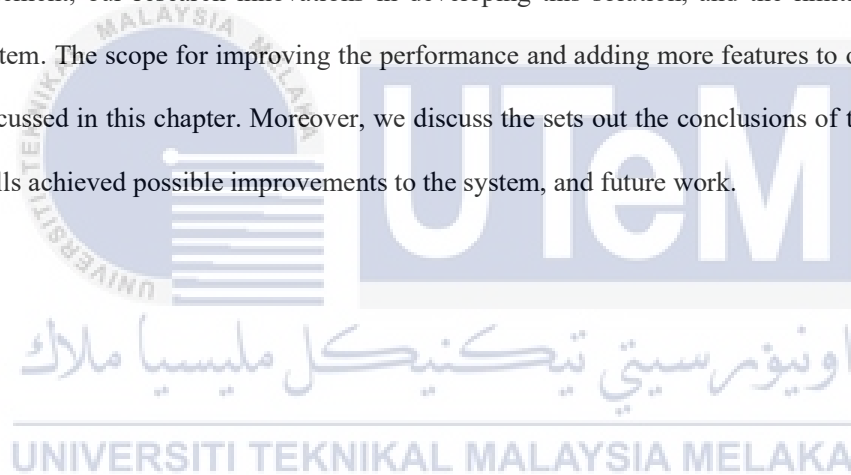
Chapter 2, This chapter provides a summary of our study on related work in literature. State of the Art presents a review of the previous research of the topics addressed in the work, such as linguistic models for sign languages, works that propose gesture recognition systems with different approaches, the state of convolutional neural networks, and analyses the current situation of hearing-impaired people in Malaysia from a demographic perspective and social, and their language of signs.

In Chapter 3, Methodology, Analysis, and Design, a sign language recognition system based on first-person vision is analyzed and proposed and its components are detailed. The reasons for certain design aspects are mentioned and the requirements of the system developed in this work are detailed. Creating the Dataset presents the steps taken to collect and debug a

dataset, the number of users involved, the selection of data for training, validation, and testing, and the tools used. Creating the Classifier describes the creation of convolutional neural networks. Next, we discuss the training process, architecture used, and tools used in development.

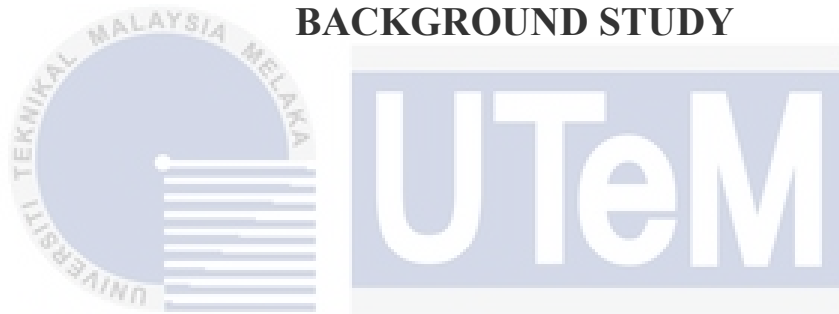
Chapter 4, Tests and Results, presents the test results of classifiers and their analysis. Other aspects of system design and classifier performance details are also evaluated. We discuss the performance of our system which is measured using various standard metrics like the accuracy of the recognition and translation.

Finally, in Chapter 5, In this chapter, we discuss how this solution addresses our problem statement, our research innovations in developing this solution, and the limitations of our system. The scope for improving the performance and adding more features to our system is discussed in this chapter. Moreover, we discuss the sets out the conclusions of the work, the skills achieved possible improvements to the system, and future work.



## CHAPTER 2

### BACKGROUND STUDY



#### 2.1 Literature Review

Sign language is an important study topic in computer vision, and several unique algorithms have been presented. For sign language recognition, the researchers utilized a variety of methods, including Artificial Neural Networks, the Hidden Markov Model, and Deep Learning using 3D CNN. CNN refers to an artificial neural network capable of extracting characteristics from images. LeNet-5, AlexNet, VGGNet, GoogLeNet, and ResNet are prominent CNN designs that have won the ImageNet Large Scale Vision Recognition Challenge since 2010.

This project will use one of the CNN architecture which is MobileNet. The MobileNet is an object detector released in 2017 as an efficient CNN architecture designed for mobile and embedded vision applications. This architecture uses proven depth-wise separable convolutions to build lightweight deep neural networks. The MobileNet model is

TensorFlow's first mobile computer vision model, and it is intended for usage in mobile applications. The improved results of these algorithms spawned a slew of new applications, including video tracking, motion estimation, scene reconstruction, object identification, sign language character and number recognition, and a slew of others.

Hand detection may be accomplished in a variety of methods, some of which need the use of specialized hardware such as gloves, depth-capable sensors such as the Microsoft Kinect sensor or Leap Motion cameras, and so on. Proprietary APIs for such technologies give essential information for feature extraction and sign categorization; yet they raise production costs, reducing commoditization. A vision-based technique capable of leveraging monocular cameras for hand identification, such as those found on mobile phones, tablets, and laptops, is required for a solution that seeks to reduce consumer prices by reusing existing mobile architectures. Several computer vision approaches might be employed to meet this need, some of which are sensitive to particular circumstances that may impair detection skills in complicated situations. So, this section will be discussing previous work related to sign language recognition and translation in real-time.

According to Kau, L.-J., & Zhuo, B.-X. (2016) [16], a proposed system, a wireless hand gesture recognition glove for real-time translation of sign language commonly used in Taiwan is developed. They designed this method to discern between distinct hand gestures; they have flex and inertial sensors built into the glove so that the three most significant criteria in Sign Language, such as finger position, palm orientation, and hand motion, can be detected without ambiguity. Periodically, the finger flexion, palm alignment, and motion trajectory will be sampled. Once the motion is identified as a legitimate signal, it is encoded and transmitted to the cell phone through Bluetooth Low Energy (BLE) for vocabulary look-up. Finally, the vocabulary will be converted to voice using Google Translate so that people may hear and understand what the sign language is saying. Deaf-mutes can readily converse with others using the suggested approach. In this system, that system has a limitation which is the user must always wear a glove everywhere if want to

communicate with others and this makes these gloves not preferred in real life. Moreover, this system that there are still some words that cannot be distinguished especially while the orientation of the palm. This system used Bluetooth so it cannot be used for higher data rates as offered by Wi-Fi and cellular technologies and cannot be used for long-distance. Furthermore, it is open to interception and attack due to wireless transmission or reception.



**Figure 2.1: Proposed Sign Language System**

Next, based on A Wearable System for Recognizing American Sign Language in Real-Time Using IMU and Surface EMG Sensors Wu, J., Sun, L., & Jafari, R. (2016) [17], a wearable system for recognizing American Sign Language (ASL) in real-time is proposed, fusing information from an inertial sensor and sEMG sensors. A feature selection strategy based on information gain is used to choose the best subset of characteristics from a large number of well-established features. Four prominent categorization methods are tested on four people for 80 regularly used ASL signals. The experimental findings reveal that using the specified feature subset and a support vector machine classifier, the average accuracies for intra-subject and intra-subject cross-session assessment are 96.16 percent and 85.24 percent, respectively. The need of including sEMG for ASL recognition is discussed, and the optimum sEMG channel is underlined. Unfortunately, this study does not thoroughly analyze a big number of indications and does not contain auto-segmentation, making it impossible to operate in real-time and difficult to test such a huge number of indicators with wearable inertial and sEMG systems. In a conclusion, whatever hardware that required gloves from this sign language project must have a problem in cost because due to the high

cost of glove devices and also the glove-based SLR systems are not ideal for use in daily life. It is very weird if users wear it in daily life and maybe in future work, this prototype can be more friendly.

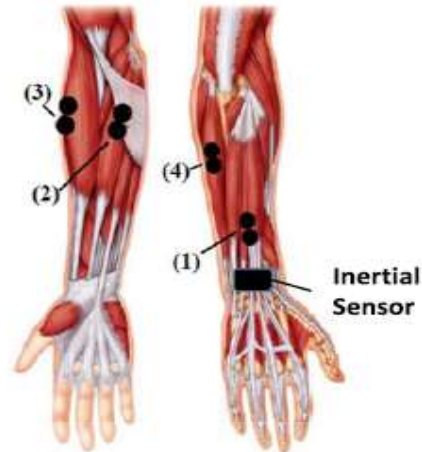
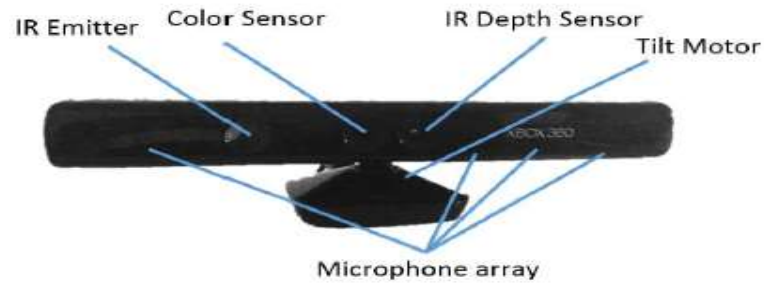


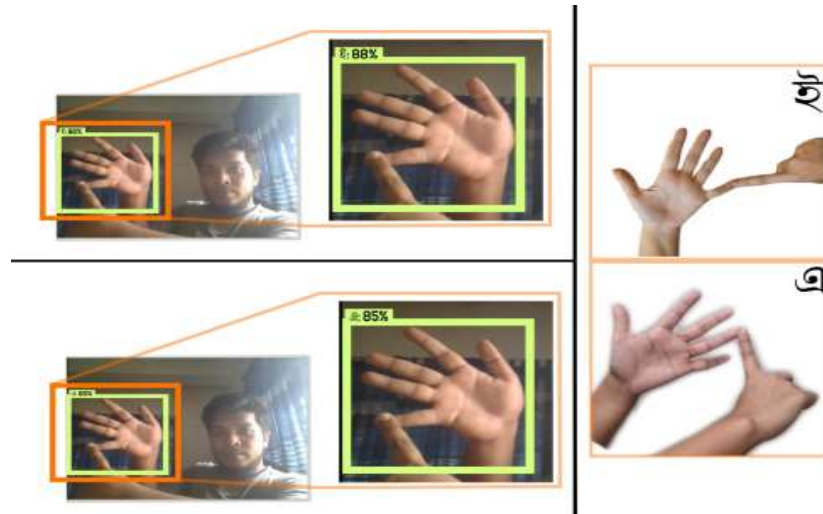
Figure 2.2: Placement of sEMG electrodes

While to solve the problem in the cost of glove-based system, Sosa-Jimenez, C. O., Rios-Figueroa, H. V., Rechy-Ramirez, E. J., Marin-Hernandez, A., & Gonzalez-Cosio, A. L. (2017) [18] were proposed a non-invasive sign detection application so that the interlocutor naturally interacts with the system interpreting a set of signs of the Mexico Sign Language using a bimodal cognitive vision system. Their application uses a Kinect sensor to obtain the gestures, geometric moments, and Hidden Markov models to process it, from experiments performed with 10 people, it was concluded that their application achieved an average sensitivity of 86% and an average specificity of 80%. However, they proposed the application using a Kinect sensor instead of the glove to save the cost and ideal for use in daily life, these applications make the user easily exposed to infrared cameras. The infrared can cause inflammation to the user and required huge costs on a component that was used.



**Figure 2.3: Kinect sensor components.**

Furthermore, based on Hoque, O. B., Jubair, M. I., Islam, M. S., Akash, A.-F., & Paulson, A. S. (2018) [19], they present a technique to detect Bangladesh Sign Language (BdSL) from images that perform in real-time. The approach detects the existence of signs in the picture region and classifies them using a Convolutional Neural Network-based object detection methodology. They used a Faster Region-based Convolutional Network technique for this goal and created a Bangladesh sign language image data set (BdSLImset) to train the system. Previous studies in detecting BdSL have relied on external equipment, and most other vision-based approaches do not function well in real-time. This system only uses a webcam instead of any other sensor device however, the system accuracy is not accurate while recognizing the letters, which have many similarities among their patterns because a system makes a false recognition in figure 2.4 showed. Furthermore, this system required a longer time in data training.



**Figure 2.4: False recognition because of the similarity in the gestures**

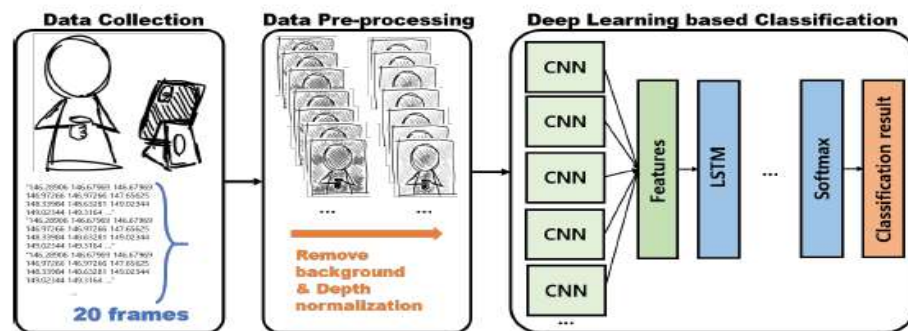
Khan, S., Ali, M. E., Das, S., & Rahman, M. M. (2019) [20] a system that can translate American Sign Language into the text from real-time video with a computer vision-based approach. This system is divided into two parts: Skin Color Detection (SCD) and Hand Gesture Recognition (HGR) (HGR). Skin pixels are extracted from images using heuristic criteria and an Artificial Neural Network (ANN). The ANN is trained using H, S, and V component values from the HSV colour space, as well as five texture characteristics. By visualizing the Q component value from YIQ colour space against the result from ANN for each pixel and applying K-means Clustering to it, an automatic threshold value for SCD is obtained. The XOR technique is utilized to extract a stationary hand picture from the SCD system's output photos. Another ANN is used for HGR. Two algorithms are used to extract features. They are the fingertip finding algorithm (a convex hull and K curvature combo) and pixel segmentation. The system's limitations are that it takes about 2.2 seconds to recognize each move and that it only divides skin colour from images but does not discern hand form.





**Figure 2.5: The result that shows an original image and image after Skin Colour Detection**

According to Park, H., Lee, J., & Ko, J. (2020) [21], This work presents our preliminary efforts in designing a mobile device-based sign language translation system using depth-only images. Their solution utilizes a convolutional neural network for feature extraction and image processing on smartphone-collected depth photos to emphasize the subject's hand and upper body actions. For word-level sign language translation, a succession of characteristics obtained from word-representing videos is run through a Long-Short Term Memory (LSTM) model. For 17 terms, we trained and tested our system with a total of 2,200 samples gathered from 26 users. With an effective picture pre-processing step, the classification accuracy of our proposed system utilizing self-collected data is 92 percent. However, the fact that these sensors give less accurate data, for example, under changing external light circumstances, has resulted in this system's shortcoming.



**Figure 2.6: System architecture for their smartphone depth image-based real-time sign language translation system.**

Furthermore, according to Ahmed, M., Zaidan, B., Zaidan, A., Salih, M. M., Al-qaysi, Z., & Alamoodi, A. (2021) [22]. The system proposed a new real-time sign recognition system based on a wearable sensory glove, which has 17 sensors with 65 channels. They present the Data Glove, which recognises diverse, potentially complicated hand motions in Malaysian Sign Language (MSL). Data Glove, with its 65 data channels, can meet the requirements provided by hand anatomy, kinematics, and gesture analysis. Four sensor groups were evaluated to find the best sensors for capturing hand gesture information. In addition, the sensor attached to the glove is validated using a 3D-printed humanoid arm. Five well-known MSL volunteers were chosen to make 75 gestures from the MSL numbers, letters, and sentences in an intensive set of trials to evaluate a system. The error rate was utilised to evaluate the system's performance. They conducted an analysis and debate that verifies their suggested system competes well with an advanced benchmark of prior works is up 100% based on 14 criteria in terms of the type of acquired signals, recognised gestures, and resolved difficulties. The results reveal that their system can recognise a wide range of gestures, with recognition accuracies of 99 percent, 96 percent, and 93.4 percent for numbers, alphabet letters, and MSL words, respectively. As previously stated, glove-based SLRs typically have hardware limitations. Because the hardware design of the data glove in this system is dependent on sensors and electrical connectors, the users are concerned about glove safety, and electrical connections can occasionally create a limitation of finger movement or some soldering point disconnection.

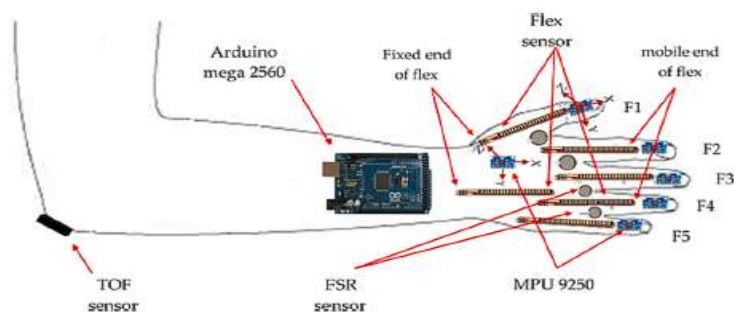


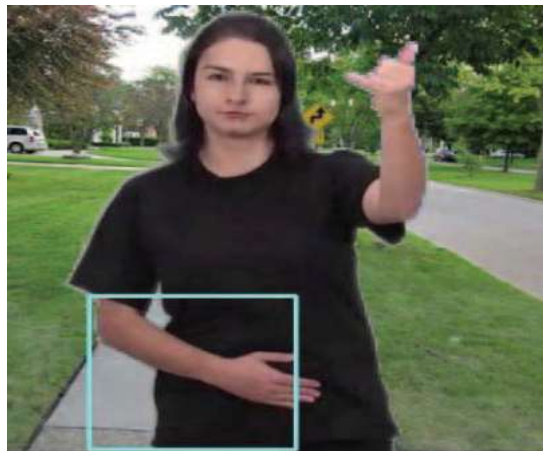
Figure 2.7: Proposed Data Glove and sensor position

Moreover, Islam, M. M., Siddiqua, S., & Afnan, J. (2017), represented a real-time HGR system based on American Sign Language (ASL) [23] recognition with greater accuracy. This method uses a mobile video camera to capture gesture pictures of ASL with a black backdrop for feature extraction. The system retrieves five characteristics throughout the processing phase: fingertip finder, eccentricity, pixel segmentation, and rotation. A novel approach for feature extraction is suggested that combines the K curvature and convex hull procedures. It is known as the "K convex hull" approach, and it is capable of detecting fingertips with great accuracy. In the system, an Artificial Neural Network (ANN) using a feedforward, backpropagation method is utilized to train a network using 30 feature vectors to detect 37 signs of the American alphabet and numerals correctly, which is useful for the HCI system. In a real-time scenario, this system's overall gesture recognition rate is 94.32 percent. However, in the rotation part, they have replaced some complex alphabet signs of ASL with other rotate ASL signs for a better recognition rate. So, this makes the ASL sign in this system are not a similar signal sign that was used in daily life. Furthermore, while in real-time performance analysis, this system required black background for collecting the image of the sign. So, if the system runs in an actual environment the accuracy of a sign will be affected.



Figure 2.8: Modification of some signs from ASL

Other than that, Zamora-Mora, J., & Chacon-Rivas, M. (2019) [24], trains the MobileNet V1 convolutional neural network against the Ego Hands dataset from Indiana University's UI Computer Vision Lab to determine if the dataset itself is sufficient to detect hands in LESCO videos, from five different signers that wear short-sleeve shirts under complex backgrounds. These requirements are critical in establishing the solution's utility. According to the cited bibliography, tests are conducted using single-color backdrops and long sleeve shirts to facilitate categorization tasks exclusively in controlled conditions. The first part of the two-step experiment yielded a mean average precision of (ninety-six dot one percent) 96.1 percent for the Ego Hands dataset and a (ninety-one percent) 91 percent average accuracy for hand detection across the five LESCO videos. Despite the excellent accuracy indicated in the testing, the hand identification module was unable to recognize certain hand forms such as clenched fists and open hands pointed perpendicular to the camera lens. Following that, the MobileNet V1 model successfully detected hands in LESCO videos until the two hundred thousand (200.000) training iteration, but the same algorithm under fifty thousand (50.000) iterations was unable to properly label hands in the five (5) test videos used, despite a small difference of 3.8 in mean average precision (mAP). The results demonstrate that the system was unable to recognise hands in a minor fraction of frames.



**Figure 2.9: Fail recognize phone call sign gesture in 50k model**

Lastly, according to Abraham, E., Nayak, A., & Iqbal, A. (2019) [25], This project aims to bridge this communication gap by proposing a novel approach to interpret the static and dynamic signs in the Indian Sign Language and convert them to speech. To obtain data about the activities, a sensor glove with flex sensors to detect finger bending and an IMU to read hand orientation is employed. This information is then wirelessly sent and categorized into voice outputs. Because of its capacity to learn long-term dependencies, LSTM networks were explored and applied for the categorization of gesture data. The proposed model was able to categorize 26 movements with a 98 percent accuracy, demonstrating the viability of employing LSTM-based neural networks for sign language translation. They solved the prior research's difficulty, which was the high cost of the hardware, in this study. In comparison to vision-based gesture detection systems that need cameras, the developed glove employs low-cost electronics. It also predicts signals that need two hands with a single right-handed glove and achieves a high level of accuracy. This is because most ISL signs include enough variety in the position and motion of the right hand to distinguish between distinct signs. This glove is also wireless, portable, and can be used with mobile devices to translate motions into text and voice. However, in my opinion, the glove-based sign language system is not practical for everyday usage.



**Figure 2.10: Prototype of the designed glove uses low-cost hardware.**

## CHAPTER 3



### 3.1 System overview

The overview of the proposed system is shown in figure 3.1 with a captured image, collect the image, image labeling, train the image, evaluate the image, detect the image, and real-time image. The sign gestured will be captured with a webcam and collect the image data. Next, the image will be labeled by using the label image package, and this model image was trained using CNN. The model was saved and loaded with OpenCV to recognize the gesture in real-time.

### 3.2 Methodology

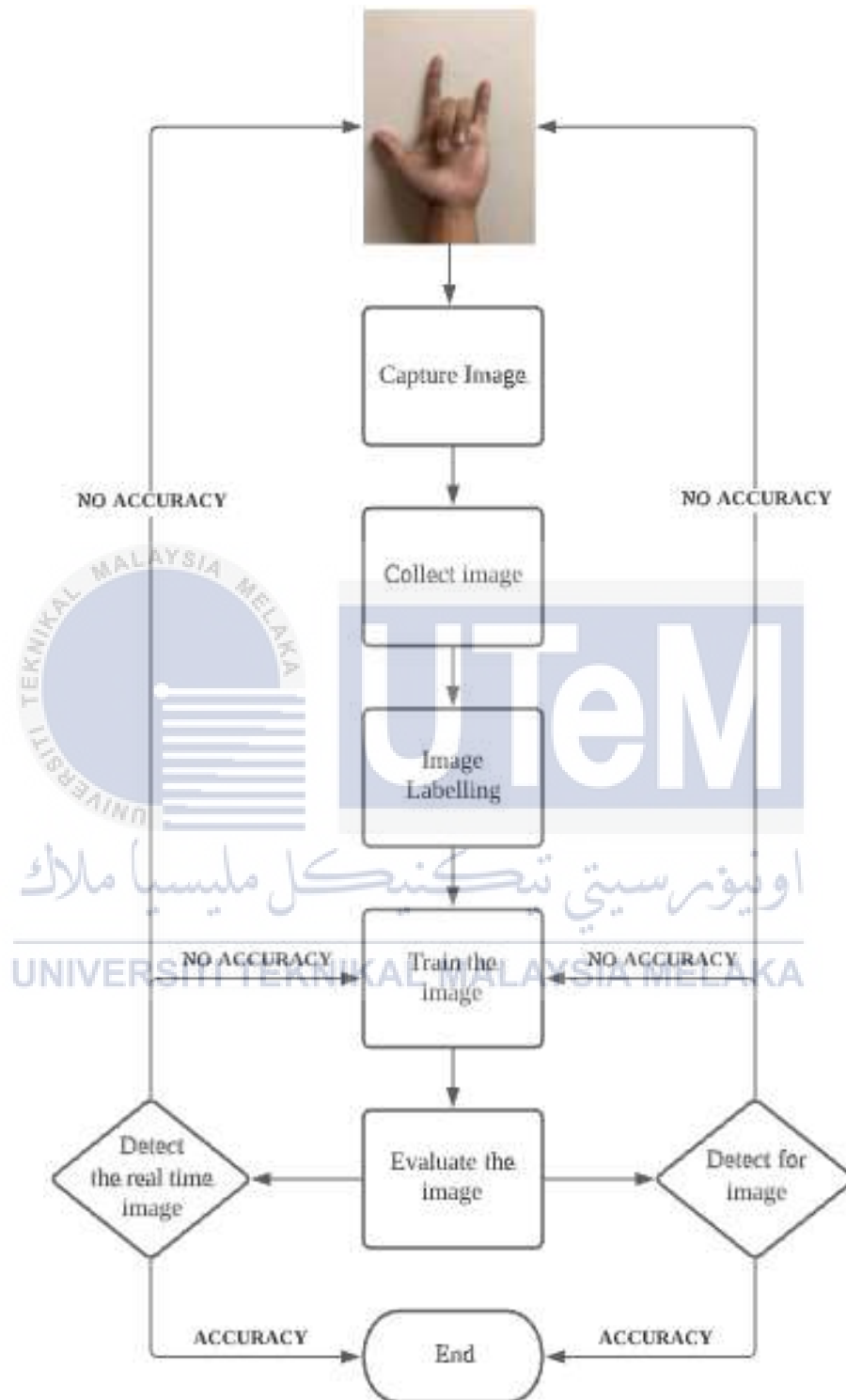


Figure 3.1: The overview of the system

### 3.3 Capture Image

In this stage, the gesture will be captured by using a webcam, so it is very simple hardware compared to previous work then use sensor or glove. So, it solves the problem in the previous project on the cost of the project.

### 3.4 Collect image

In this section, the five-sign gesture was collected which are “Hello”, “I Love You”, “No”, “Thank You” and “Yes”. The cv2.VideoCapture will be set to 0, this command is for connecting the jupyter notebook to the webcam. So, the sign gesture dataset was prepared with the image frames and the dataset consisted of 200 images of sign language as shown in figure 3.2. The hand gesture was collected from only 2 students. The student captured 40 images per category in various situations. The images are captured from various positions, angles, and with different backgrounds and lighting conditions.



Figure 3.2: Hand sign gesture (“Hello”, “I Love You”, “No”, “Thank You” and “Yes”)



## 4. Capture Images

```
In [9]: 1 IMAGES_PATH
Out[9]: 'Tensorflow\workspace\images\collectedimages'

In [54]: 1 for label in labels:
2         cap = cv2.VideoCapture(0)
3         print('Collecting images for {}'.format(label))
4         time.sleep(5)
5         for imgnum in range(number_imgs):
6             print('Collecting image {}'.format(imgnum))
7             ret, frame = cap.read()
8             imgname = os.path.join(IMAGES_PATH, label, label+'_'+str(uuid.uuid1()+1))
9             cv2.imwrite(imgname, frame)
10            cv2.imshow('frame', frame)
11            time.sleep(2)
12
13            if cv2.waitKey(1) & 0xFF == ord('q'):
14                break
15        cap.release()
16        cv2.destroyAllWindows()

Collecting images for iloveyou
Collecting image 0
Collecting image 1
Collecting image 2
Collecting image 3
Collecting image 4
```

Figure 3.3: Command shows that webcam capture sign gesture “ILoveYou” was successfully collected

### 3.5 Image Labelling

The Label Image package is a graphical image annotation tool, Annotations are saved as XML files in PASCAL VOC format, the format used by ImageNet. Besides, it also supports YOLO and CreateML formats. The annotations are bounding boxes, bounding boxes are the most commonly used type of annotation in computer vision. Bounding boxes are rectangular boxes used to define the location of the target object. They can be determined by the  $x$  and  $y$  axis coordinates in the upper-left corner and the  $x$  and  $y$  axis coordinate in the lower-right corner of the rectangle. Bounding boxes are generally used in object detection and localization tasks. The main purpose of the tool is to allow the users to highlight or capture a specific object in a picture. The images are highlighted to make them readable for the machines. Image labeling or image annotation are specifically used for artificial intelligence and machine learning. As the tool allows the users to use the highlighted images as the training data sets. The data sets are further while feeding with a deep learning algorithm. Therefore, with the help of the image labeling tools, so it makes we can develop a functional artificial intelligence model.

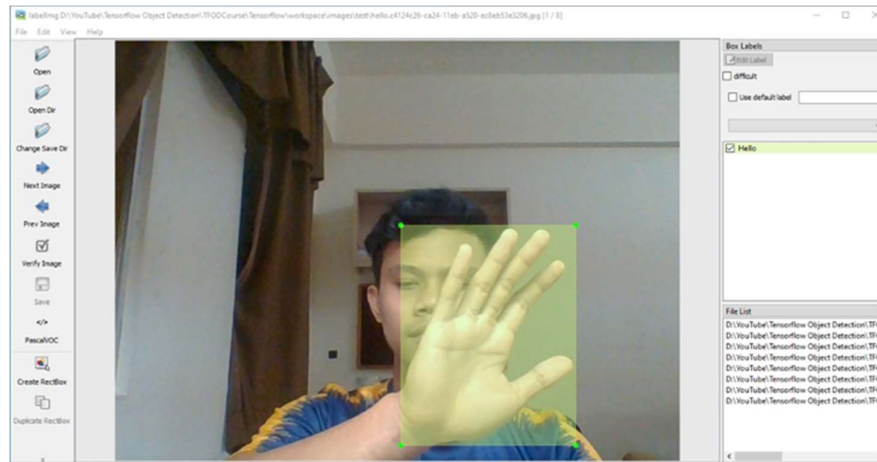


Figure 3.4: Hello gesture from Image Labelling package



Figure 3.5: I Love You gesture from Image Labelling package

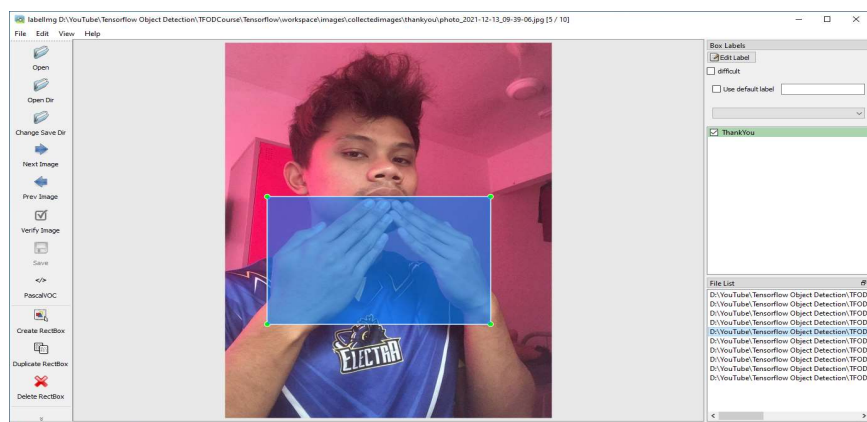


Figure 3.6: Thank You gesture from Image Labelling package

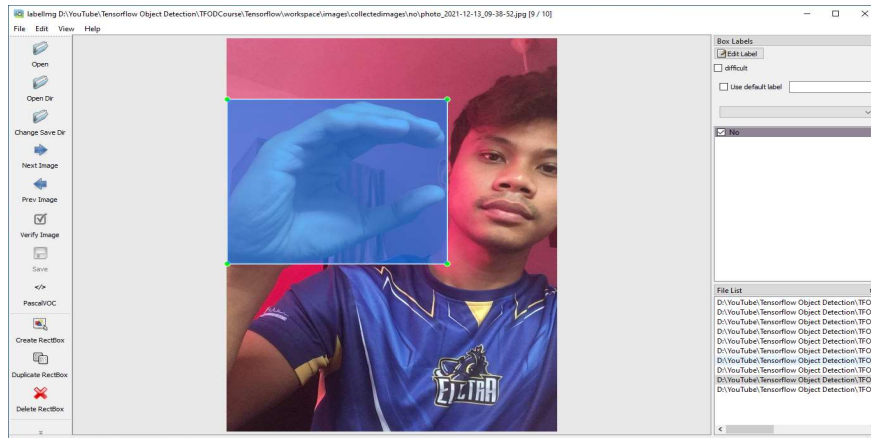


Figure 3.7: No gesture from Image Labelling package



Figure 3.8: Yes, gesture from Image Labelling package

### 3.6 Train the image/model.

This model was trained using MobileNet, a CNN class that was freely sourced by Google, and so provides us with an ideal starting point for training our classifiers that are crazy tiny and incredibly fast. Depthwise separable convolutions are used by MobileNet. When compared to the network with ordinary convolutions of the same depth in the nets, it greatly reduces the number of parameters. A depthwise separable convolution is created by combining two operations: depthwise and pointwise convolution.

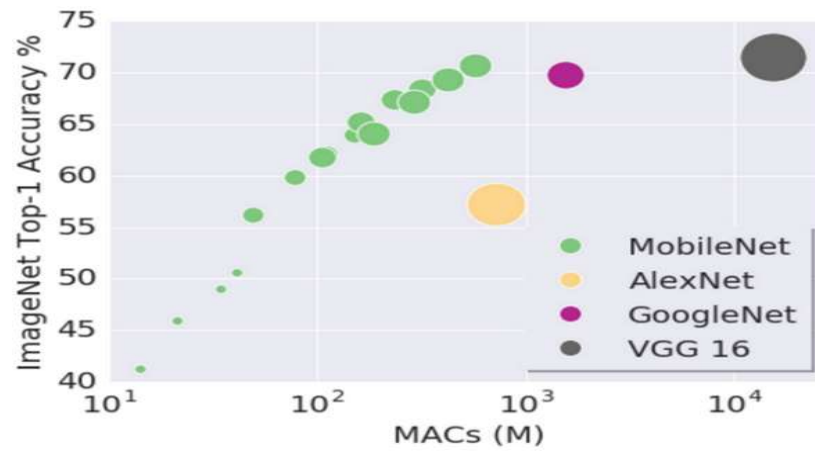


Figure 3.9: The speed and power consumption of the network is proportional to the number of MACs  
(Multiply-Accumulates)



Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool $7 \times 7$	$7 \times 7 \times 1024$
FC / s1	$1024 \times 1000$	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 1: MobileNet Body Architecture

### 3.7 Evaluate the model.

To evaluate object detection, the mean average precision (mAP) is used. The mAP compares the ground-truth bounding box to the detected box and returns a score. The higher the score, the more accurate the model is in its detections.

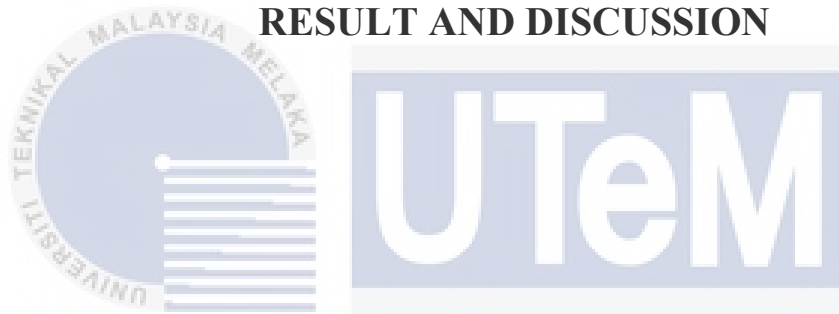
### 3.8 Image recognition

The trained model was put onto a laptop with TensorFlow as the backend, OpenCV to read picture frames, Visual Studio Code as the editor, and Python as the programming language. OpenCV reads real-time hand-shaped picture frames from the signer and rescales them to 64 x 64 x 3 pixels. The model identifies and predicts sign gestures with success. The suggested technology converts MSL digits to text.



## CHAPTER 4

### RESULT AND DISCUSSION



#### 4.1 Processing time

The processing time of a system has such a vital influence on image processing. Suitable hardware and software are needed to make the system run smoothly without taking a long time to load the programs. In real time, it is best to have a lower time of processing, so the frame rate per second will be faster. A new version of a personal computer's processor and large size of memory runs the software faster than the old versions. The webcam played another big role in processing time. Webcam with higher resolution provides better performance in detecting and classifying the hand sign gesture.

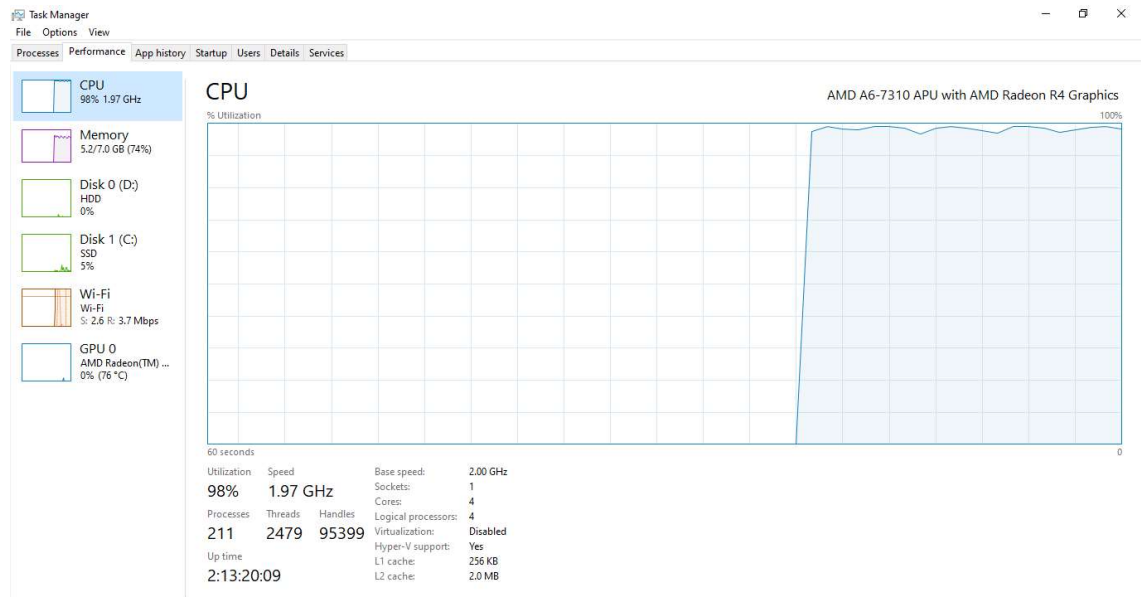


Figure 4.1: My CPU used 98% while training the data

## 4.2 Training and Evaluation.

TensorFlow has a variety of configuration papers that may be reformed based on a new training set. TensorBoard, a TensorFlow imaging platform, may be used to see the outcomes of the training and assessment stages. It can monitor a variety of data, including training duration, overall loss, number of steps, and much more. Furthermore, TensorBoard may be performed when the system is in training mode, allowing you to monitor the progress of the training and assessment activities to ensure that the training performance is excellent. The system checkpoint file is used as a starting point for fine-tuning. The trained model ratio of correct predictions for a label is shown by the mAP throughout the assessment. The IoU, on the other hand, is limited to object detection models. This metric represents the percentage of % overlap between the bounding box and the ground truth-bounding box. The mAP graph is an average of the proportion of accurate bounding boxes and labels of the model restarted with "correct" about bounding boxes that overlapped with their respective ground truth boxes by 50% or more.



The primary goal of training development is to keep overall loss as low as possible (less than 1). Tensorboard diagrams for the SSD-MobileNet model's AP, mAP, classification loss, localization loss, and total loss are presented below. When the overall loss decreases as the number of iterations/steps (epochs) increase, the training process is complete. The num steps variable specifies the number of training steps required before the training process is terminated. It is determined by the amount of the dataset as well as how long the investigator wants to train the model. The statistic used is a mean average precision (mAP). The area under the precision-recall curve is represented by a single value. The mAP is a measure of how good the model is in generating a bounding box in the test dataset that intersects with the ground truth bounding box at least 50% of the time. At 0.5IoU, the mAP value reached a greater level of confidence. The detector's performance improves as the mAP levels increase. Figures 4.3 and 4.5 display the validation of a gesture class that has been categorized and matched with an earlier trained class. The better the classification and detection accuracy, the smaller the classification loss that drops to zero. The localization loss curve is displayed in Figures 4.3 and 4.5. It specifies the predicate bounding box, which corresponds to the ground-truth bounding box.

## 4.3 Ssd mobilenet v2 training result 2000 steps vs 10000 steps

### 4.3.1 2000 steps of training

```

C:\ Command Prompt
Instructions for updating:
Use fn_output_signature instead
INFO:tensorflow:Step 100 per-step time 5.311s
I1213 13:57:16.699897 2160 model_lib_v2.py:707] Step 100 per-step time 5.311s
INFO:tensorflow:{'loss/classification_loss': 0.3707945,
'Loss/Localization_loss': 0.15911503,
'Loss/regularization_loss': 0.15386176,
'loss/total_loss': 0.68377125,
'learning_rate': 0.0319994}
I1213 13:57:16.614688 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.3707945,
'Loss/Localization_loss': 0.15911503,
'Loss/regularization_loss': 0.15386176,
'loss/total_loss': 0.68377125,
'learning_rate': 0.0319994}
INFO:tensorflow:Step 200 per-step time 4.264s
I1213 14:04:22.934233 2160 model_lib_v2.py:707] Step 200 per-step time 4.264s
INFO:tensorflow:{'loss/classification_loss': 0.17884709,
'Loss/Localization_loss': 0.08689452,
'Loss/regularization_loss': 0.15374656,
'loss/total_loss': 0.4194882,
'learning_rate': 0.0373328}
I1213 14:04:22.941220 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.17884709,
'Loss/Localization_loss': 0.08689452,
'Loss/regularization_loss': 0.15374656,
'loss/total_loss': 0.4194882,
'learning_rate': 0.0373328}
INFO:tensorflow:Step 300 per-step time 4.288s
I1213 14:11:31.680777 2160 model_lib_v2.py:707] Step 300 per-step time 4.288s
INFO:tensorflow:{'loss/classification_loss': 0.21560168,
'Loss/Localization_loss': 0.095508814,
'Loss/regularization_loss': 0.15358396,
'loss/total_loss': 0.46469444,
'learning_rate': 0.0426662}
I1213 14:11:31.684782 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.21560168,
'Loss/Localization_loss': 0.095508814,
'Loss/regularization_loss': 0.15358396,
'loss/total_loss': 0.46469444,
'learning_rate': 0.0426662}
INFO:tensorflow:Step 400 per-step time 4.290s
I1213 14:18:40.709211 2160 model_lib_v2.py:707] Step 400 per-step time 4.290s
INFO:tensorflow:{'loss/classification_loss': 0.12708287,
'Loss/Localization_loss': 0.060719408,
'Loss/regularization_loss': 0.1534743,
'loss/total_loss': 0.3412166,
'learning_rate': 0.047995908}
I1213 14:18:40.709314 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.12708287,
'Loss/Localization_loss': 0.060719408,
'Loss/regularization_loss': 0.1534743,
'loss/total_loss': 0.3412166,
'learning_rate': 0.047995908}

C:\ Command Prompt
'Loss/regularization_loss': 0.1449438,
'Loss/total_loss': 0.24716452,
'learning_rate': 0.07995972}
I1213 15:31:34.408031 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.064448915,
'Loss/Localization_loss': 0.037771795,
'Loss/regularization_loss': 0.1449438,
'Loss/total_loss': 0.24716452,
'learning_rate': 0.07995972}
INFO:tensorflow:Step 1800 per-step time 4.310s
I1213 15:58:45.320561 2160 model_lib_v2.py:707] Step 1800 per-step time 4.310s
INFO:tensorflow:{'loss/classification_loss': 0.04635403,
'Loss/Localization_loss': 0.019585446,
'Loss/regularization_loss': 0.14419322,
'loss/total_loss': 0.21013266,
'learning_rate': 0.0799474}
I1213 15:58:45.325567 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.04635403,
'Loss/Localization_loss': 0.019585416,
'Loss/regularization_loss': 0.14419322,
'loss/total_loss': 0.21013266,
'learning_rate': 0.0799474}
INFO:tensorflow:Step 1900 per-step time 4.281s
I1213 16:05:53.341564 2160 model_lib_v2.py:707] Step 1900 per-step time 4.281s
INFO:tensorflow:{'loss/classification_loss': 0.08632305,
'Loss/Localization_loss': 0.016702218,
'Loss/regularization_loss': 0.14340341,
'loss/total_loss': 0.24642769,
'learning_rate': 0.07993342}
I1213 16:05:53.346569 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.08632305,
'Loss/Localization_loss': 0.01670218,
'Loss/regularization_loss': 0.14340341,
'loss/total_loss': 0.24642769,
'learning_rate': 0.07993342}
INFO:tensorflow:Step 2000 per-step time 4.332s
I1213 16:13:06.543139 2160 model_lib_v2.py:707] Step 2000 per-step time 4.332s
INFO:tensorflow:{'loss/classification_loss': 0.04726407,
'Loss/Localization_loss': 0.017741915,
'Loss/regularization_loss': 0.14263014,
'loss/total_loss': 0.20763613,
'learning_rate': 0.079917811}
I1213 16:13:06.548144 2160 model_lib_v2.py:708] {'loss/classification_loss': 0.04726407,
'Loss/Localization_loss': 0.017741915,
'Loss/regularization_loss': 0.14263014,
'loss/total_loss': 0.20763613,
'learning_rate': 0.079917811}

```

Figure 4.2: The loss matrix shown in the command prompt

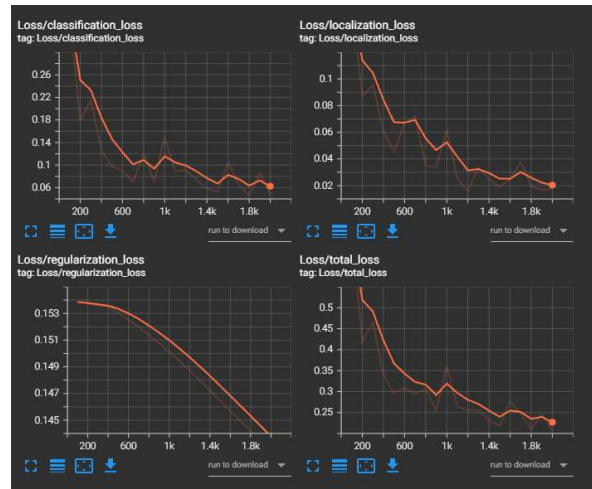


Figure 4.3: The loss matrix shown in TensorBoard



### 4.3.2 10000 steps of training

```

Command Prompt
Instructions for updating:
Use fn_output_signature instead
INFO:tensorflow:Step 2100 per-step time 5.581s
11215 21:26:18.841825 8240 model_lib_v2.py:707] Step 2100 per-step time 5.581s
INFO:tensorflow:('loss/classification_loss': 0.12669146,
'loss/localization_loss': 0.058909697,
'loss/regularization_loss': 0.14423314,
'loss/total_loss': 0.32901528,
'learning_rate': 0.07990656)
11215 21:26:18.852258 8240 model_lib_v2.py:707] ['loss/classification_loss': 0.12669146,
'loss/localization_loss': 0.058909697,
'loss/regularization_loss': 0.14423314,
'loss/total_loss': 0.32901528,
'learning_rate': 0.07990656]
INFO:tensorflow:Step 2200 per-step time 4.348s
11215 21:33:25.584906 8240 model_lib_v2.py:707] Step 2200 per-step time 4.348s
INFO:tensorflow:('loss/classification_loss': 0.15774351,
'loss/localization_loss': 0.056386936,
'loss/regularization_loss': 0.14380585,
'loss/total_loss': 0.3579363,
'learning_rate': 0.07988167)
11215 21:33:25.586169 8240 model_lib_v2.py:707] ['loss/classification_loss': 0.15774351,
'loss/localization_loss': 0.056386936,
'loss/regularization_loss': 0.14380585,
'loss/total_loss': 0.3579363,
'learning_rate': 0.07988167]
INFO:tensorflow:Step 2300 per-step time 4.444s
11215 21:40:49.925209 8240 model_lib_v2.py:707] Step 2300 per-step time 4.444s
INFO:tensorflow:('loss/classification_loss': 0.07633919,
'loss/localization_loss': 0.023733236,
'loss/regularization_loss': 0.14313787,
'loss/total_loss': 0.24321029,
'learning_rate': 0.07986114)
11215 21:40:49.936079 8240 model_lib_v2.py:707] ['loss/classification_loss': 0.07633919,
'loss/localization_loss': 0.023733236,
'loss/regularization_loss': 0.14313787,
'loss/total_loss': 0.24321029,
'learning_rate': 0.07986114]
INFO:tensorflow:Step 2400 per-step time 4.401s
11215 21:48:18.076208 8240 model_lib_v2.py:707] Step 2400 per-step time 4.401s
INFO:tensorflow:('loss/classification_loss': 0.08797198,
'loss/localization_loss': 0.03665625,
'loss/regularization_loss': 0.14238046,
'loss/total_loss': 0.26700896,
'learning_rate': 0.07983897)
11215 21:48:18.084362 8240 model_lib_v2.py:707] ['loss/classification_loss': 0.08797198,
'loss/regularization_loss': 0.09785225,
'learning_rate': 0.14808728,
'loss/total_loss': 0.073937014,
'learning_rate': 0.073937014]
11216 06:33:59.959950 8240 model_lib_v2.py:707] ['loss/classification_loss': 0.04121604,
'loss/localization_loss': 0.00901899,
'loss/regularization_loss': 0.09785225,
'loss/total_loss': 0.14808728,
'learning_rate': 0.073937014]
INFO:tensorflow:Step 9800 per-step time 4.238s
11216 06:41:03.825713 8240 model_lib_v2.py:707] Step 9800 per-step time 4.238s
INFO:tensorflow:('loss/classification_loss': 0.07228414,
'loss/localization_loss': 0.0075366325,
'loss/regularization_loss': 0.09734938,
'loss/total_loss': 0.17171016,
'learning_rate': 0.07390057)
11216 06:41:03.833842 8240 model_lib_v2.py:707] ['loss/classification_loss': 0.07228414,
'loss/localization_loss': 0.0075366325,
'loss/regularization_loss': 0.09734938,
'loss/total_loss': 0.17171016,
'learning_rate': 0.07390057]
INFO:tensorflow:Step 9900 per-step time 4.295s
11216 06:48:13.260248 8240 model_lib_v2.py:707] Step 9900 per-step time 4.295s
INFO:tensorflow:('loss/classification_loss': 0.028552698,
'loss/localization_loss': 0.007571084,
'loss/regularization_loss': 0.09685129,
'loss/total_loss': 0.13297507,
'learning_rate': 0.073662736)
11216 06:48:13.277341 8240 model_lib_v2.py:707] ['loss/classification_loss': 0.028552698,
'loss/localization_loss': 0.007571084,
'loss/regularization_loss': 0.09685129,
'loss/total_loss': 0.13297507,
'learning_rate': 0.073662736]
INFO:tensorflow:Step 10000 per-step time 4.270s
11216 06:55:20.389389 8240 model_lib_v2.py:707] Step 10000 per-step time 4.270s
INFO:tensorflow:('loss/classification_loss': 0.03246434,
'loss/localization_loss': 0.007928922,
'loss/regularization_loss': 0.09636565,
'loss/total_loss': 0.13675892,
'learning_rate': 0.07352352)

```

Figure 4.4: The loss matrix shown in the command prompt

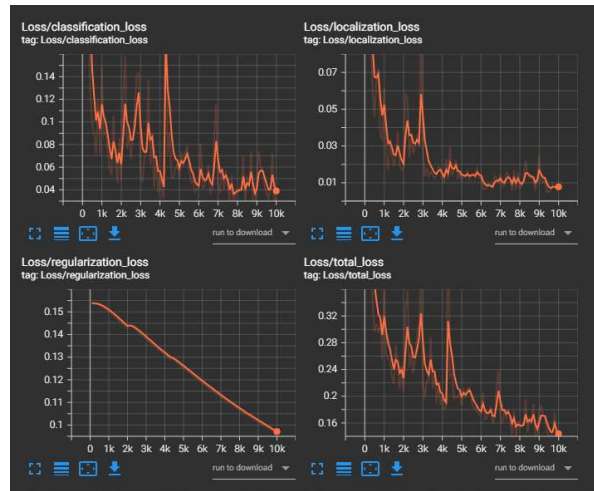


Figure 4.5: The loss matrix shown in TensorBoard



#### 4.4 Ssd mobilenet v2 evaluation result 2000 vs 10000

The mAP is the outcome of "precision-recall" operations on identifying bounding boxes. It is a good indicator of how successfully the network selects things of interest. The higher the mAP score, the more exact the detecting mechanism. The mAP primarily improves the "precision-recall" curve. A recall rate and an accuracy rate are then calculated for each guess. The average precision (AP) is the average of the forecasts estimated over numerous criteria. It is a determination to achieve detector properties in a single number. The figure below illustrates the AP for the sign gesture class based on pre-trained models (SSD-Mobilenet) in this study. A vertical axis represents the AP values, while a horizontal axis represents the steps (epochs).

Intersection over Union (IoU) is an accuracy estimate metric that indicates how accurate the object detector is on a linked database. Fixing the system prediction regarding the location of the specific object that is expected to be identified is key for object localization, as demonstrated in the figure below. Typically, this consists of drawing a bounding box around the item of interest. Then, based on the Intersection above the Union threshold, the localization function is calculated (IoU).

The TensorFlow Object Detection API employs "PASCAL VOC 2007 metrics," with a location estimated instance classified as a TP when the Intersection over Union (IoU) is more than 50%. At any one moment, only one item can be associated with a single bounding box. However, if many bounding boxes are predicted for an item, one is considered TP and the others as FP. Nonetheless, an item is predictable as an FN if it has no predicted bounding box that is associated with it.

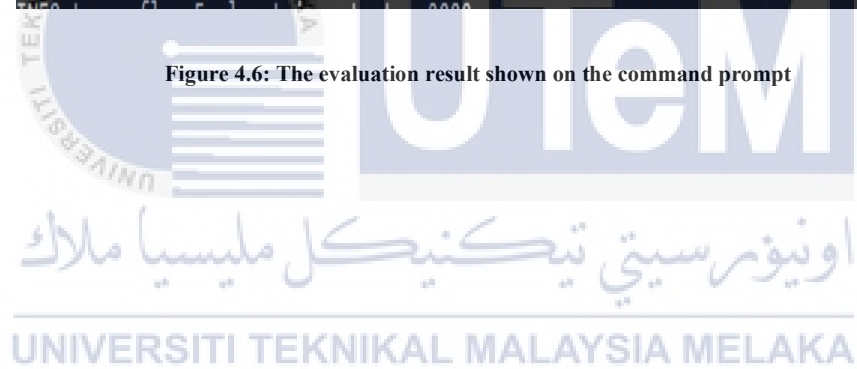
## 4.4.1 2000 steps

```

Average Precision (AP) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.755
Average Precision (AP) @[ IoU=0.50 | area= all | maxDets=100 ] = 0.926
Average Precision (AP) @[ IoU=0.75 | area= all | maxDets=100 ] = 0.912
Average Precision (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Precision (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Precision (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.755
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 1 ] = 0.760
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 10 ] = 0.820
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.820
Average Recall (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Recall (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Recall (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.820

```

Figure 4.6: The evaluation result shown on the command prompt



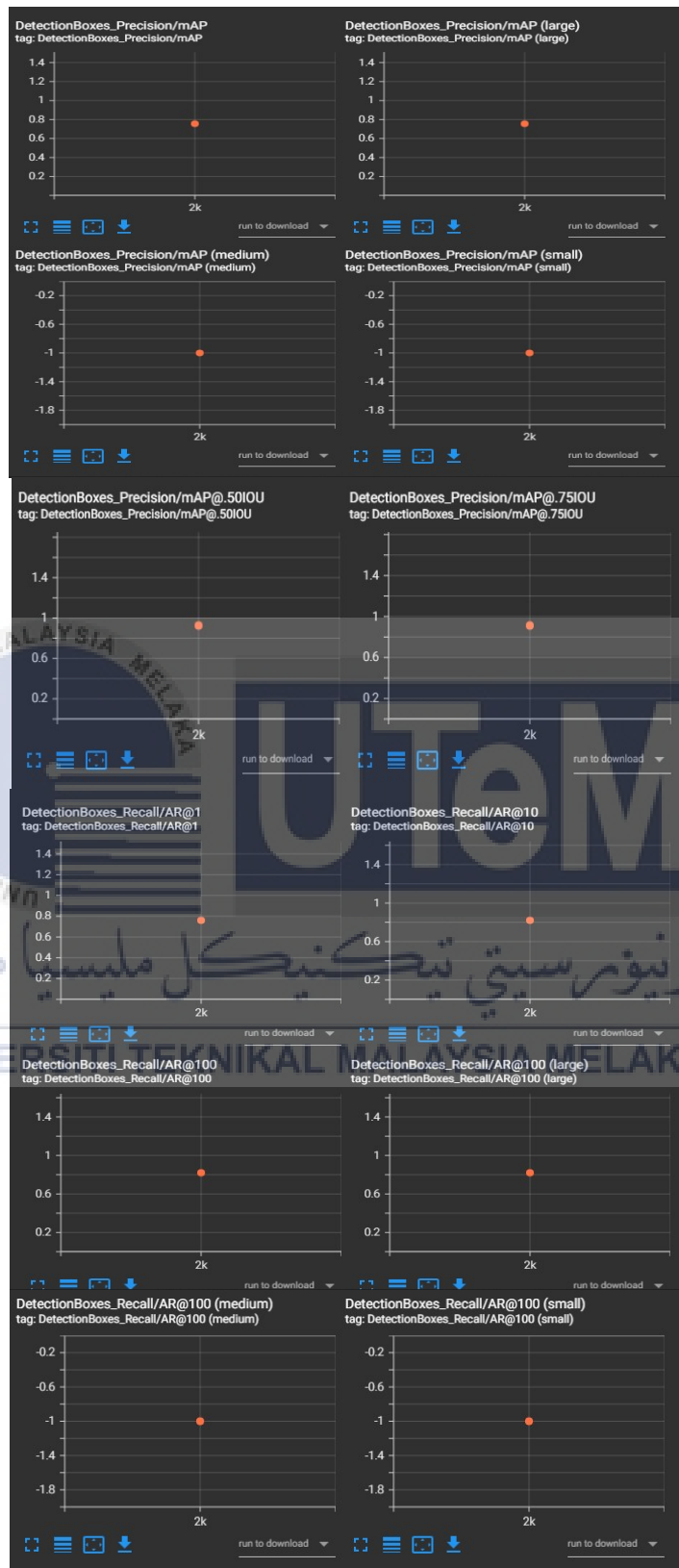


Figure 4.7: The evaluation result is shown on TensorBoard



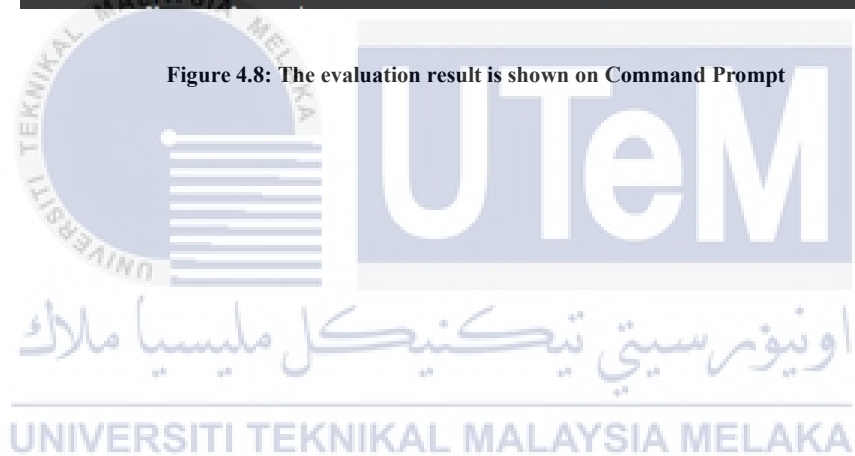
## 4.4.2 10000 steps

```

Average Precision (AP) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.907
Average Precision (AP) @[ IoU=0.50 | area= all | maxDets=100 ] = 1.000
Average Precision (AP) @[ IoU=0.75 | area= all | maxDets=100 ] = 1.000
Average Precision (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Precision (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Precision (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.907
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 1 ] = 0.915
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets= 10 ] = 0.915
Average Recall (AR) @[ IoU=0.50:0.95 | area= all | maxDets=100 ] = 0.915
Average Recall (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Recall (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Recall (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.915

```

Figure 4.8: The evaluation result is shown on Command Prompt



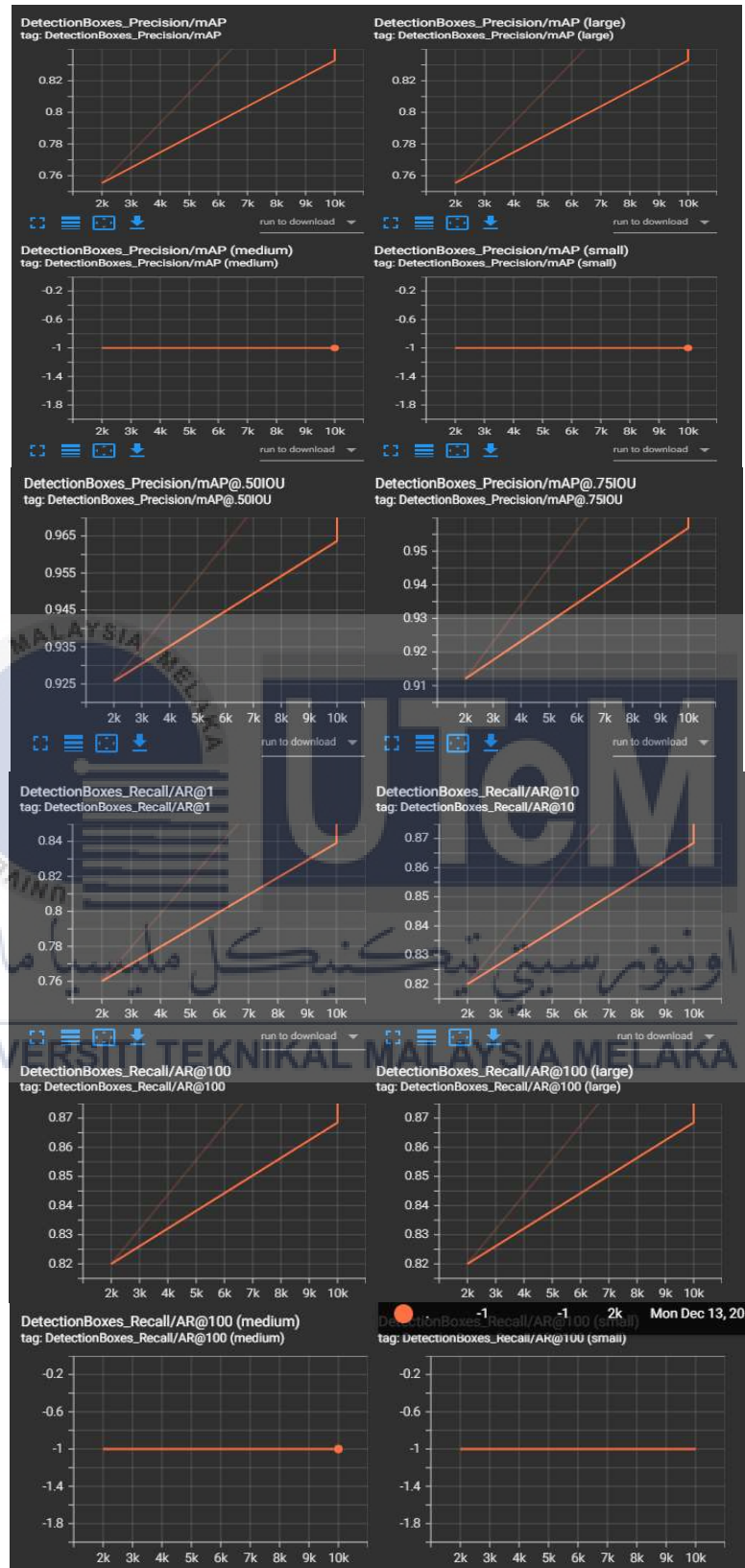


Figure 4.9: The evaluation result is shown on TensorBoard

Summary using TensorBoard data for 2000 steps vs 10000 steps

Properties	2000 steps	10000 steps
Loss/classification_loss	0.0475	0.0325
Loss/localization_loss	0.0177	0.0079
Loss/regularization_loss	0.1426	0.0964
Loss/total_loss	0.2076	0.1368
Average precision/mAP	0.755	0.907
Average precision/mAP@.50iou	0.926	1.000
Average precision/mAP@.75iou	0.912	1.000
Average precision/mAP(small)	-1.000	-1.000
Average precision/mAP(medium)	-1.000	-1.000
Average precision/mAP(Large)	0.755	0.907
Average recall/AR@1	0.760	0.915
Average recall/AR@10	0.820	0.915
Average recall/AR@100	0.820	0.915
Average recall/AR@100(small)	-1.000	-1.000
Average recall/AR@100(medium)	-1.000	-1.000
Average recall/AR@100(large)	0.820	0.915

**Table 2: Summary of all the parameters in 2000 steps and 10000 steps (classification loss, localization loss, regularization loss, total loss, average precision, and average recall) of the training and evaluation processes**

#### 4.5 Detection and Translation Results in Real Time

Comprehensive research on the datasets has been done to validate the detection accuracy for hand signals in real-time detection and translation. To obtain the maximum expected identification accuracy and proper translation, several classes of sign gesture photos, including hello, yes, no, thank you, and I love you, are evaluated. In this study, the detecting procedure is carried out with the use of an open CV and a laptop camera. To perform the detection procedure in both methods, the technique must be exported as a static inference model trained on the sign gesture dataset, together with the corresponding label map. The TensorFlow object detection API library returns the script, called export, using the most recent checkpoint number at the last phase of the training process. This result shows in 3 situations which are situation 1 (one sign on one person), situation 2 (two sign on one person), and lastly, situation 3 (Two sign on two people).

##### 4.5.1 Situation 1(one sign on one person)



Figure 4.10: Sign gesture for Hello in real time show 99%



Figure 4.11: Sign gesture for Yes in real time show 99%

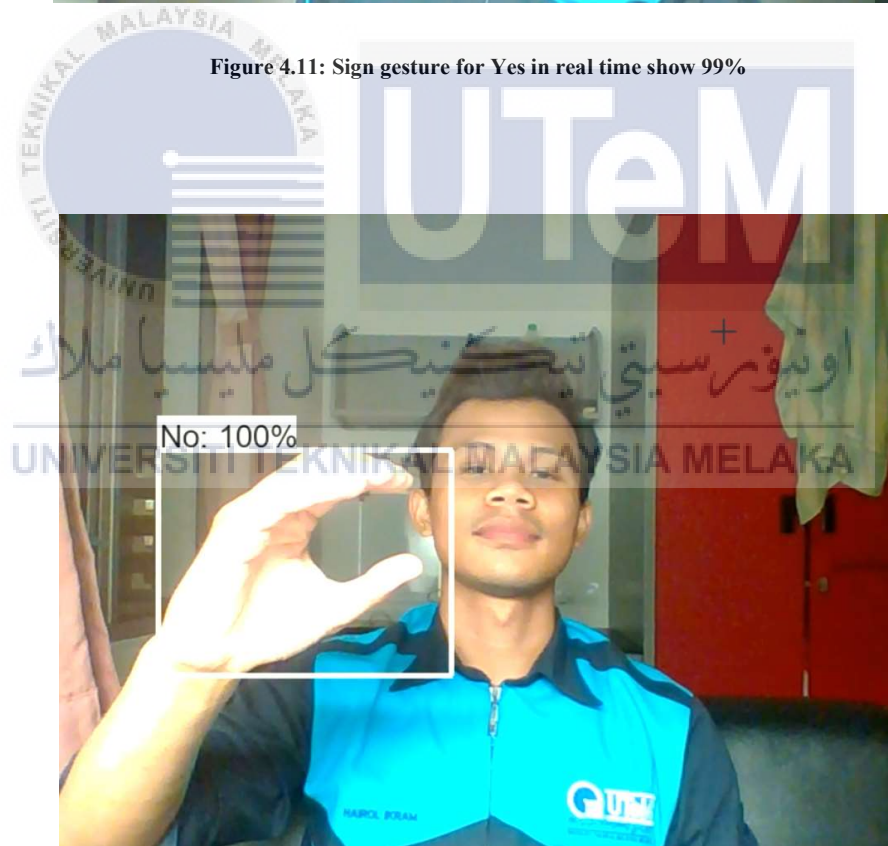


Figure 4.12: Sign gesture for No in real time show 100%

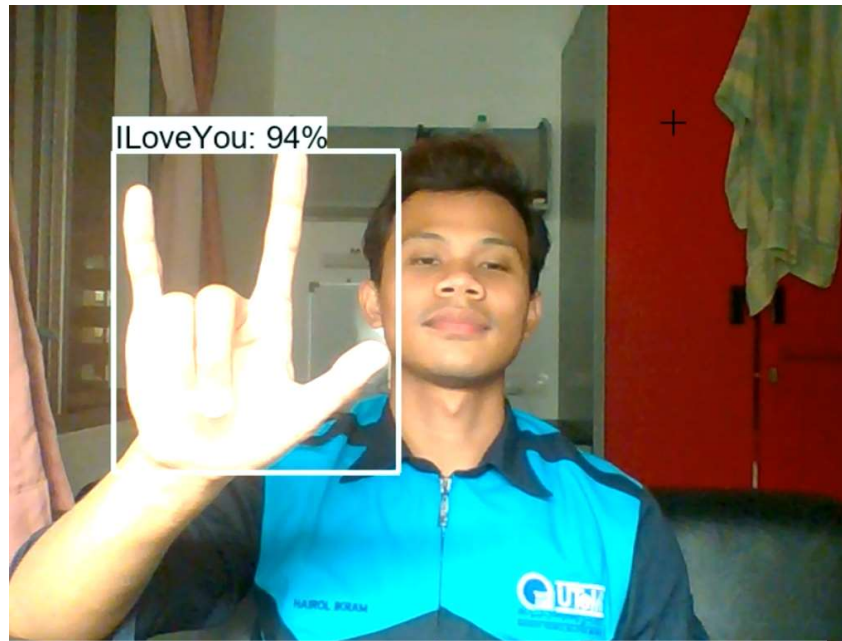


Figure 4.13: Sign gesture for I Love You in real time show 94%

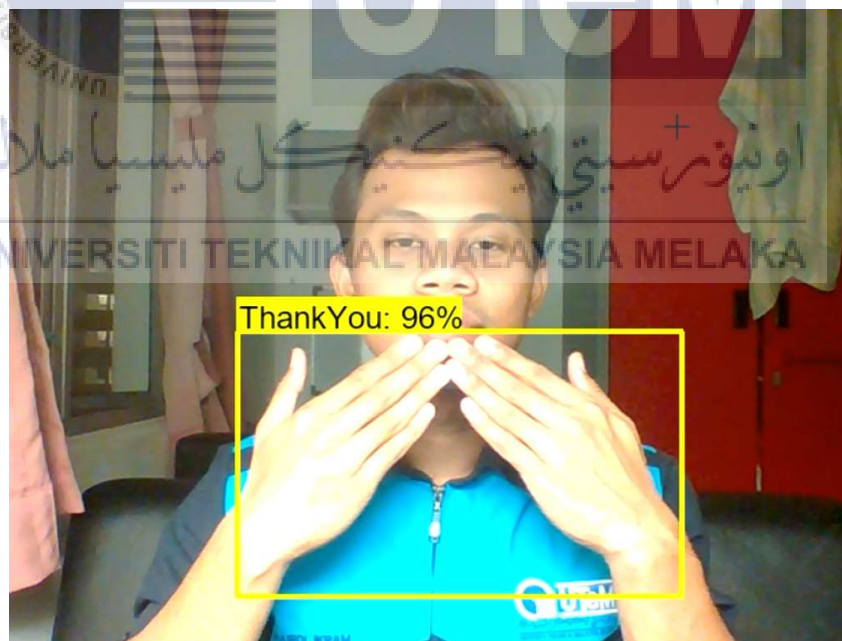


Figure 4.14: Sign gesture for Thank You in real time show 96%

#### 4.5.2 Situation 2 (two sign on one person)

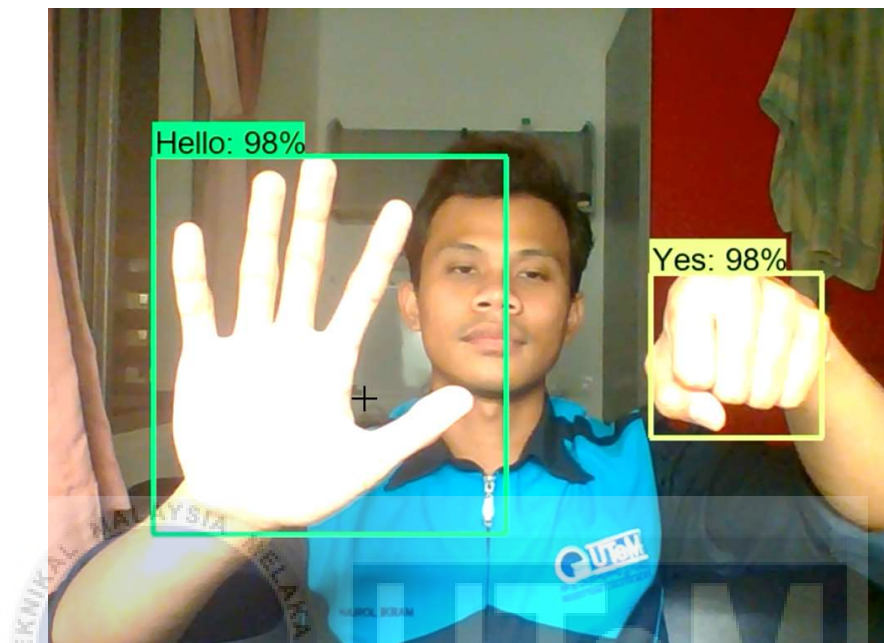


Figure 4.15: Sign gesture for Hello and Yes with both hands in real time show 98%

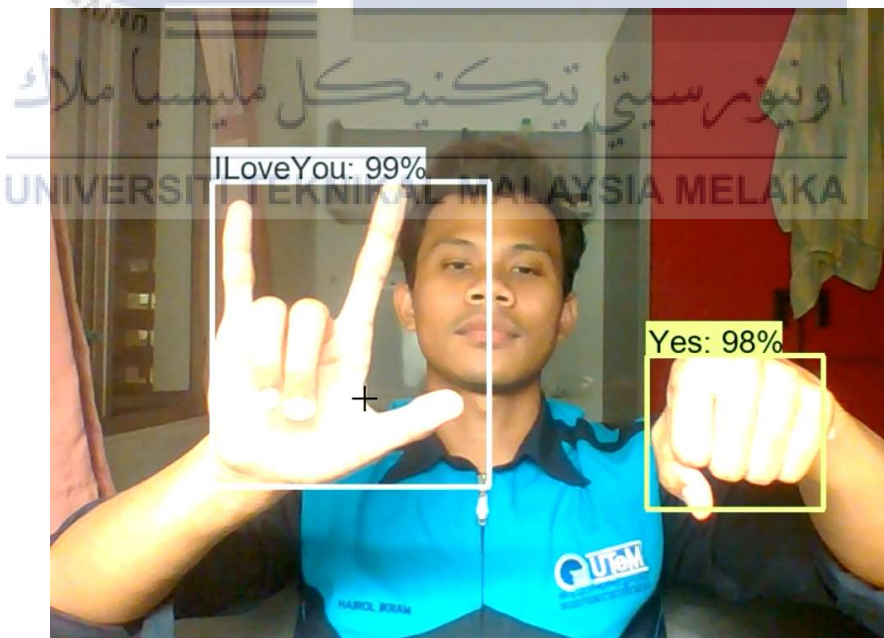


Figure 4.16: Sign gesture for I Love You and Yes with both hands in real time show 99% and 98%

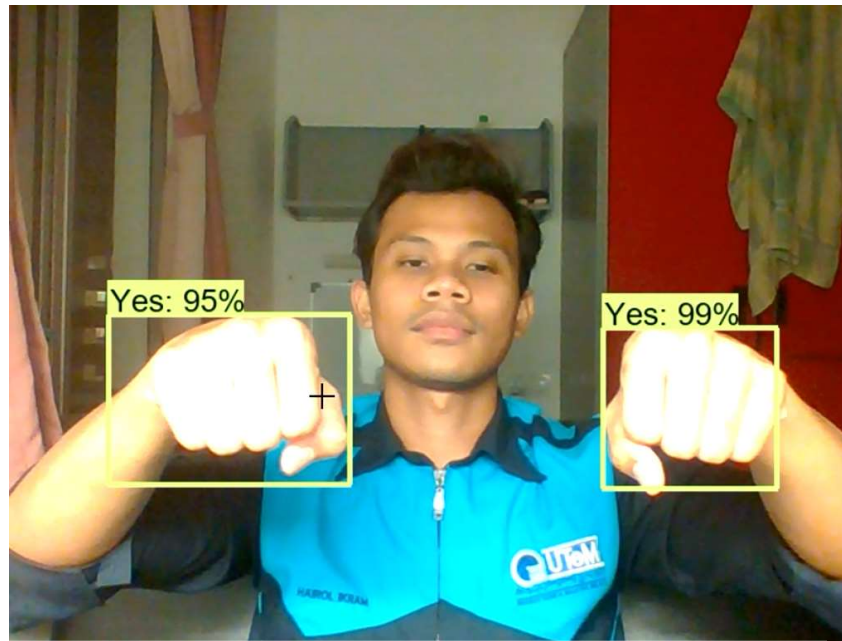


Figure 4.17: Sign gesture in real time for Yes with right hand show 95% while left hand show 99%

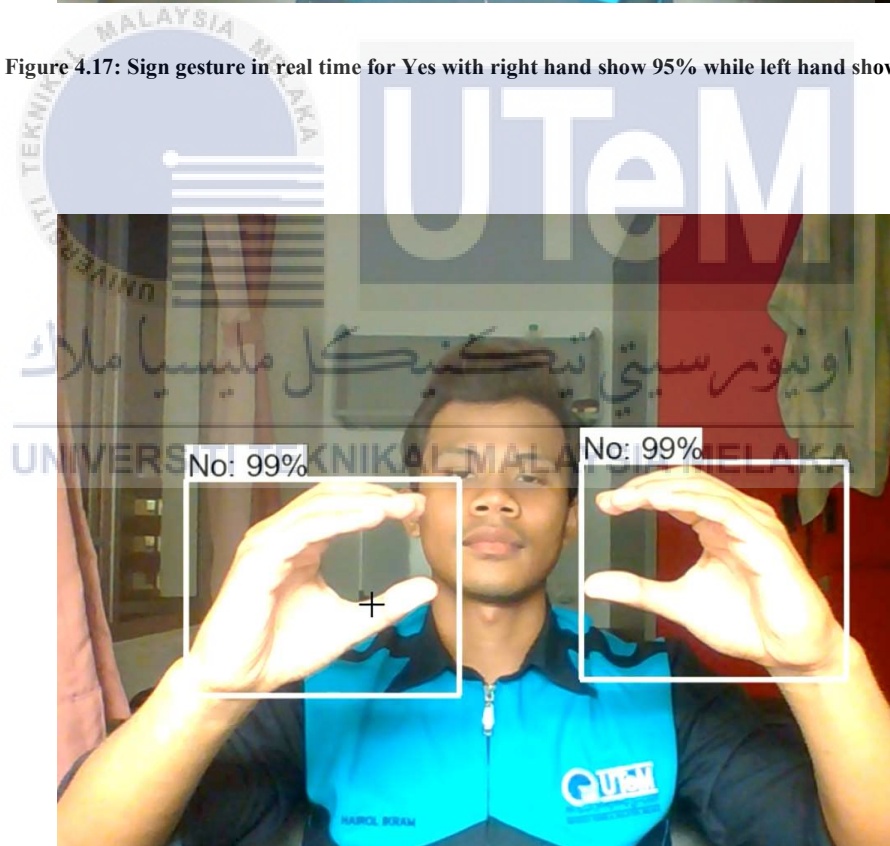


Figure 4.18: Sign gesture in real time for No with both hands show 99%



#### 4.5.3 Situation 3 (Two sign on two people)



Figure 4.19: Sign gesture with two people in real time for Hello 99% and I Love You 100%

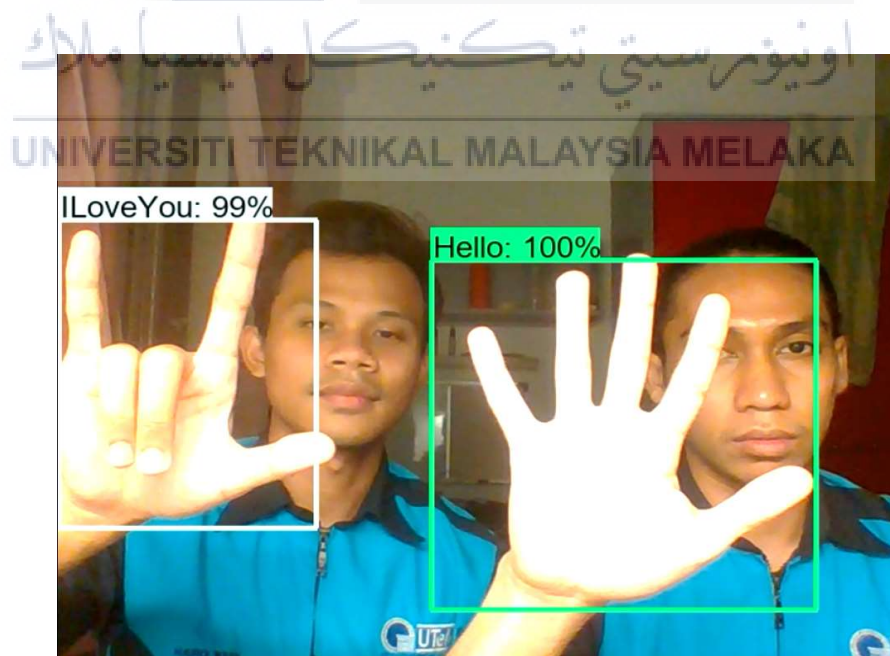


Figure 4.20: Sign gesture with two people in real time for I Love You 99% and Hello 100%



Figure 4.21: Sign gesture with two people in real time for Yes 98% and No 99%

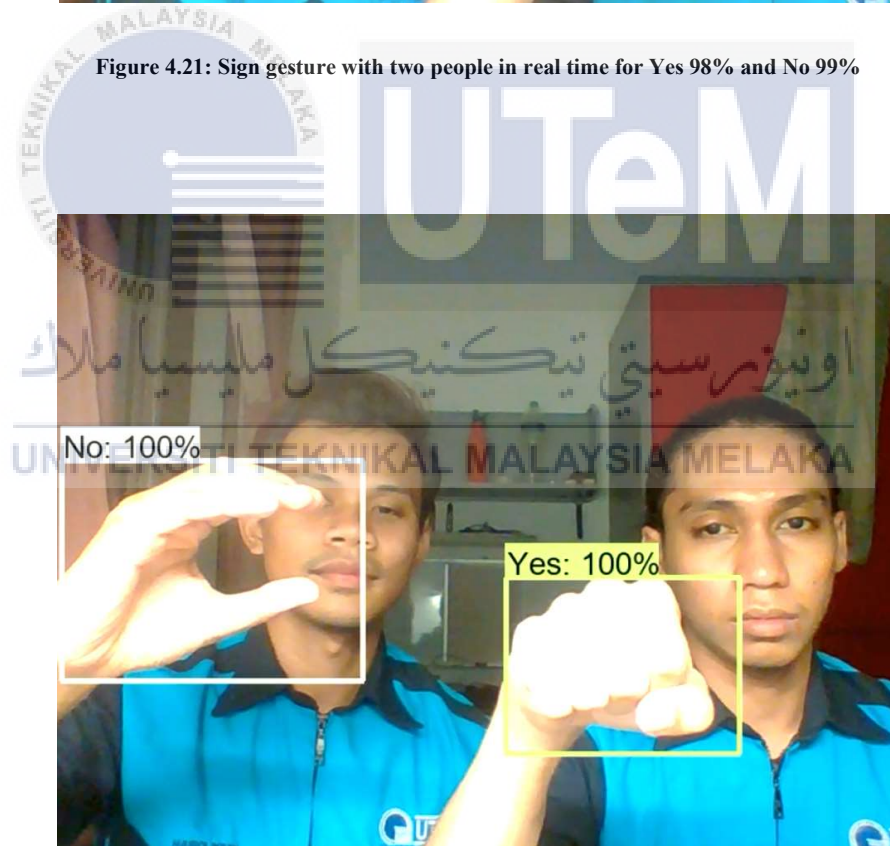


Figure 4.22: Sign gesture with two people in real time for No 100% and Yes 100%

## CHAPTER 5

### CONCLUSION AND FUTURE WORK



#### 5.1 Conclusion

Based on the research that has been done, successfully created an application to detect the object by using TensorFlow Object Detection API by utilizing Mobilenet SSD V2 as a pre-trained model. The system can detect 5 sign gesture class categories, namely Hello, Yes, No, Thank You, and I Love You. The system was able to display the measurement level of the accuracy of each class category. A system has been designed the dataset with the first person captured the sign images, and it makes the system can identify the deep learning techniques for the recognition of sign language be more accurate with an average of 90% above. The overall objective of this work has been to apply deep learning techniques for the recognition of signs of sign language, which has been developed and evaluated in this bachelor's degree project work. Different state-of-the-art solutions on gesture recognition were identified and analyzed, all situations of the signer to machine vision techniques.

A specific dataset was built with 208 images for 5 classes and divided into two which is about 168 images dataset that was used to train classifiers and some 40 images were collected for the test, with the participation of 1 signer and 2 signers and the support of algorithms that automated some of the work. The resulting model consisted of two CNN. Among other observations, it can be considered that the more users participate in the dataset, the better it will represent all the feature space with what can be found in a real environment, be it a skin of the hand, presence of dataset with different lighting different ways to execute the signs or variations in hand size.

In the data collection process for testing, it has been detected that the position of the signer is variable and affects the field of view of the system. This position restricts the position that the user must have when using the system, if the signer is far away from the camera to detect the sign gesture it makes, the executed sign would not be detected correctly, and it also comes up with a false translation. However, as long as the signer stays close to the camera will have no problems capturing the sign.

Finally, the result tells that this project was a success to analyze the sign language in real time and labeling images for object detection and then it directly goes to all objectives of the project were able to obtain.

## 5.2 Future Work

The proposed system focuses on five class categories of sign language which are Hello, Yes, No, Thank You and I Love You. However, there are still many other sign languages that are not included in this system. Therefore, in future work, it can further extend the number of classes by augmenting our training datasets with add on more examples of another sign language. This system is more focused on static sign gestures, so to make the improvement and more interesting, in the future, it can be able to add the dynamic

sign gesture into this project. However, the impact of COVID-19 on education systems, which is the implementation of online learning (OL) during Movement Control Order (MCO) to higher learning education institutions in Malaysia [26], so this system can be applied to help students on Sekolah Pendidikan Khas for hearing-impaired student on online learning. The proposed solution, despite not achieving sufficiently all the problems on past research, can be improved with the participation of more users in the dataset in other scenarios, which can significantly impact the performance of the classifier, increasing the size of the dataset, building the system with high performance computer and high resolution of the camera to capture the hand gesture and would be interesting. Also increasing the test dataset could lead to more profitable analysis.



## REFERENCE

- [1] Darus, N., Abdullah, N., & Mutalib, A. 2012, “*iMSL: Malay Sign Language for the Deaf and Hearing-impaired*”, Johor Bahru; Knowledge Management International Conference
- [2] Norziha, M. M. Z., Halimah, B. Z., & Azlina, A. (2010b). Developing Augmented Reality Book for Deaf in Science: The Determining Factors. In Proceedings of the International Symposium on Information Technology, Kuala Lumpur, Malaysia.
- [3] Siti-Zaharah, M. & Nor-Azan, M. Z. (2010). Courseware Accessibility for Hearing Impaired. In Proceedings of International Symposium in Information Technology, Kuala Lumpur, Malaysia
- [4] Paulraj, M. P., Sazali, Y., Hazry, D. & Wan-Mohd-Ridzuan, W. A. M. (2009). Gesture Recognition System for Kod Tangan Bahasa Melayu (KTBM) Using Neural Network. In Proceedings of the 5th International Colloquium on Signal Processing and Its Application, Kuala Lumpur, Malaysia
- [5] Norziha, M. M. Z., Halimah, B. Z., & Azlina, A. (2010a). A participatory Design in Developing Prototype an Augmented Reality Book for Deaf Students. In Proceedings of the Second International Conference on Computer Research and Development, Kuala Lumpur, Malaysia.

- [6] Siti-Zaharah, M. & Nor-Azan, M. Z. (2010). Courseware Accessibility for Hearing Impaired. In Proceedings of International Symposium in Information Technology, Kuala Lumpur, Malaysia.
- [7] Wijayanto, C.P. 2009. Membangun Aplikasi Pelatihan Bahasa Isyarat Berbasis Komputer pada Orang Tunarungu. STIMIK AMIKOM, Yogyakarta.
- [8] Huo, Feifei., Hendriks, E.A., Oomes, A.H.J. 2009. Detection Tracking and Recognition of Human Poses for a Real-Time Spatial Game. Delft University of Technology, Utrecht University. The Netherlands
- [9] Zare, A. A., & Zahiri, S. H. (2016). Recognition of a real-time signer-independent STATIC Farsi sign language based on Fourier coefficients amplitude. International Journal of Machine Learning and Cybernetics, 9(5), 727-741.
- [10] Ramli, S. GMT feature extraction for representation of BIM sign language. In Proceedings of the 2012 IEEE Control and System Graduate Research Colloquium (ICSGRC), Shah Alam, 16–17 July 2012; pp. 43–48.
- [11] Anderson, R.; Wiryana, F.; Ariesta, M.C.; Kusuma, G.P. Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on Input-Process-Output. Procedia Comput. Sci. 2017, 116, 441–448.
- [12] LaViola, J. A Survey of Hand Posture and Gesture Recognition Techniques and Technology; Brown University: Providence, RI, USA, 1999; Volume 29.
- [13] Anderson, R.; Wiryana, F.; Ariesta, M.C.; Kusuma, G.P. Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on Input-Process-Output. Procedia Comput. Sci. 2017, 116, 441–448.
- [14] Alvi, A.K.; Azhar, M.Y.B.; Usman, M.; Mumtaz, S.; Rafiq, S.; Rehman, R.U.; Ahmed, I. Pakistan sign language recognition using statistical template matching. Int. J. Inf. Technol. 2004, 1, 1–12.
- [15] Luqman, H.; Mahmoud, S.A. Transform-based Arabic sign language recognition. Procedia Comput. Sci. 2017, 117, 2–9.

- [16] Kau, L.-J., & Zhuo, B.-X. (2016). Live demo: A real-time portable sign language translation system. 2016 IEEE Biomedical Circuits and Systems Conference (BioCAS). <https://doi.org/10.1109/biocas.2016.7833748>
- [17] Sosa-Jimenez, C. O., Rios-Figueroa, H. V., Rechy-Ramirez, E. J., Marin-Hernandez, A., & Gonzalez-Cosio, A. L. (2017). Real-time Mexican Sign Language recognition. 2017 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC). <https://doi.org/10.1109/ropec.2017.8261606>
- [18] Hoque, O. B., Jubair, M. I., Islam, M. S., Akash, A.-F., & Paulson, A. S. (2018). Real-Time Bangladeshi Sign Language Detection using Faster R-CNN. 2018 International Conference on Innovation in Engineering and Technology (ICIET). <https://doi.org/10.1109/ciet.2018.8660780>
- [19] Khan, S., Ali, M. E., Das, S., & Rahman, M. M. (2019). Real-Time Hand Gesture Recognition by Skin Colour Detection for American Sign Language. 2019 4th International Conference on Electrical Information and Communication Technology (EICT). <https://doi.org/10.1109/eict48899.2019.9068809>
- [20] Park, H., Lee, J., & Ko, J. (2020). Achieving real-time sign language translation using a smartphone's true depth images. 2020 International Conference on Communication Systems & Networks (COMSNETS). doi:10.1109/comsnets48256.2020.9027420
- [21] Ahmed, M., Zaidan, B., Zaidan, A., Salih, M. M., Al-qaysi, Z., & Alamoodi, A. (2021). Based on wearable sensory device in 3d-printed humanoid: A new real-time sign language recognition system. *Measurement*, 168, 108431. doi:10.1016/j.measurement.2020.108431
- [22] Wu, J., Sun, L., & Jafari, R. (2016). A wearable system for Recognizing American sign language in real-time Using IMU and SURFACE EMG Sensors. *IEEE Journal of Biomedical and Health Informatics*, 20(5), 1281-1290. doi:10.1109/jbhi.2016.2598302



- [23] Islam, M. M., Siddiqua, S., & Afnan, J. (2017). Real time hand gesture recognition using different algorithms based on American Sign Language. 2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR). doi:10.1109/icivpr.2017.7890854
- [24] Zamora-Mora, J., & Chacon-Rivas, M. (2019). Real-time hand detection using convolutional neural networks for Costa Rican sign language recognition. 2019 International Conference on Inclusive Technologies and Education (CONTIE). doi:10.1109/contie49246.2019.00042
- [25] Abraham, E., Nayak, A., & Iqbal, A. (2019). Real-time translation of Indian sign language using LSTM. 2019 Global Conference for Advancement in Technology (GCAT). doi:10.1109/gcat47503.2019.8978343
- [26] Nordin, N. B., & Nordin, N. B. (2020). Impact of pandemic covid-19 to the online learning: Case of higher education institution in Malaysia. Universal Journal of Educational Research, 8(12A), 7607-7615. doi:10.13189/ujer.2020.082546

## APPENDIX A:

### DEFINE IMAGES TO COLLECT IMAGE

```
1 labels = ['hello', 'thankyou', 'yes', 'no', 'iloveyou']  
2 labels = ['yes']  
3 number_imgs = 10
```

اوتیور سیتی تکنیکل ملیسیا ملاک  
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## APPENDIX B:

### CAPTURE IMAGE BY WEBCAM

```
1 for label in labels:
2     cap = cv2.VideoCapture(0)
3     print('Collecting images for {}'.format(label))
4     time.sleep(5)
5     for imgnum in range(number_imgs):
6         print('Collecting image {}'.format(imgnum))
7         ret, frame = cap.read()
8         imgname = os.path.join(IMAGES_PATH, label, label+str(imgnum)+'.jpg'.format(str(uuid.uuid1())))
9         cv2.imwrite(imgname, frame)
10        cv2.imshow('frame', frame)
11        time.sleep(2)
12
13        if cv2.waitKey(1) & 0xFF == ord('q'):
14            break
15    cap.release()
16    cv2.destroyAllWindows()
```

## APPENDIX C:

### IMAGE LABELLING

```
1 !pip install --upgrade pyqt5 lxml
```

```
1 LABELIMG_PATH = os.path.join('Tensorflow', 'labelimg')
```


```
1 if not os.path.exists(LABELIMG_PATH):  
2     !mkdir {LABELIMG_PATH}  
3     !git clone https://github.com/tzutalin/labelImg {LABELIMG_PATH}
```

```
1 if os.name == 'posix':  
2     !make qt5py3  
3 if os.name == 'nt':  
4     !cd {LABELIMG_PATH} && pyrcc5 -o libs/resources.py resources.qrc
```

```
1 !cd {LABELIMG_PATH} && python labelImg.py
```

## APPENDIX D:

### TRAIN THE MODEL



```
1 TRAINING_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research', 'object_detection', 'model_main_tf2.py')
1 command = "python {} --model_dir={} --pipeline_config_path={} --num_train_steps=10000".format(TRAINING_SCRIPT,
2 paths['CHECKPOINT_PATH'], files['PIPELINE_CONFIG'])
1 print(command)
1 !{command}
```

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## APPENDIX E:

### EVALUATE THE MODEL




```
: 1 command = "python {} --model_dir={} --pipeline_config_path={} --checkpoint_dir={}".format(TRAINING_SCRIPT,  
2 paths['CHECKPOINT_PATH'],files['PIPELINE_CONFIG'], paths['CHECKPOINT_PATH'])  
  
: 1 print(command) /  
  
: 1 !{command}
```

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## APPENDIX F:

### DETECT FROM AN IMAGE



```

1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4 %matplotlib inline

1 category_index = label_map_util.create_category_index_from_labelmap(files['LABELMAP'])

1 IMAGE_PATH = os.path.join(paths['IMAGE_PATH'], 'test', 'photo_2021-12-13_12-06-00 (2).jpg')
2 img = cv2.imread(IMAGE_PATH)
2 image_np = np.array(img)
3
4 input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
5 detections = detect_fn(input_tensor)
6
7 num_detections = int(detections.pop('num_detections'))
8 detections = {key: value[0, :num_detections].numpy()
9               for key, value in detections.items()}
10 detections['num_detections'] = num_detections
11
12 # detection_classes should be ints.
13 detections['detection_classes'] = detections['detection_classes'].astype(np.int64)
14
15 label_id_offset = 1
16 image_np_with_detections = image_np.copy()
17
18 viz_utils.visualize_boxes_and_labels_on_image_array(
19     image_np_with_detections,
20     detections['detection_boxes'],
21     detections['detection_classes']+label_id_offset,
22     detections['detection_scores'],
23     category_index,
24     use_normalized_coordinates=True,
25     max_boxes_to_draw=5,
26     min_score_thresh=.8,
27     agnostic_mode=False)
28
29 plt.imshow(cv2.cvtColor(image_np_with_detections, cv2.COLOR_BGR2RGB))
30 plt.show()

```

## APPENDIX G:

### REAL TIME DETECTION FROM WEBCAM



```

1 |pip uninstall opencv-python-headless -y

1 cap = cv2.VideoCapture(0)
2 width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
3 height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
4
5 while cap.isOpened():
6     ret, frame = cap.read()
7     image_np = np.array(frame)
8
9     input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
10    detections = detect_fn(input_tensor)
11
12    num_detections = int(detections.pop('num_detections'))
13    detections = {key: value[0, :num_detections].numpy()
14                  for key, value in detections.items()}
15    detections['num_detections'] = num_detections
16
17    # detection_classes should be ints.
18    detections['detection_classes'] = detections['detection_classes'].astype(np.int64)
19
20    label_id_offset = 1
21    image_np_with_detections = image_np.copy()
22
23    viz_utils.visualize_boxes_and_labels_on_image_array(
24        image_np_with_detections,
25        detections['detection_boxes'],
26        detections['detection_classes']+label_id_offset,
27        detections['detection_scores'],
28        category_index,
29        use_normalized_coordinates=True,
30        max_boxes_to_draw=5,
31        min_score_thresh=.8,
32        agnostic_mode=False)
33
34    cv2.imshow('object detection', cv2.resize(image_np_with_detections, (800, 600)))
35
36    if cv2.waitKey(10) & 0xFF == ord('q'):
37        cap.release()
38        cv2.destroyAllWindows()
39        break

```



## APPENDIX H:

### ZIP AND EXPORT MODEL

