

**“Real-Time Automatic Number Plate Recognition System
Using Deep Learning Approach”**

VIGNESH A/L SELVADURAI



**BACHELOR OF MECHATRONICS ENGINEERING WITH
HONOURS
UNIVERSITI TEKNIKAL MALAYSIA MELAKA**

2024

Real-Time Automatic Number Plate Recognition System Using Deep Learning Approach”

VIGNESH A/L SELVADURAI

**A report submitted
in partial fulfilment of the requirements for the degree of
Bachelor of Mechatronics Engineering with Honors**



UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

DECLARATION

I declare that this thesis entitled "Real-Time Automatic Number Plate Recognition System Using Deep Learning Approach" is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in the candidature of any other degree.

Signature

:



Name

:

Vignesh A/L Selvadurai

Date

:

15/06/2024




اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

APPROVAL

I hereby declare that I have checked this report entitled "Real-Time Automatic Number Plate Recognition System Using Deep Learning Approach", and in my opinion, this thesis fulfils the partial requirement to be awarded the degree of Bachelor of Mechatronics Engineering with Honours

Signature :  : _____
Supervisor Name : Dr. Loh Ser Lee : _____
Date : 16/06/2024 : _____

اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

DEDICATIONS

I would like to express my dedication for this project to God Almighty, who has been my rock and the wellspring of wisdom and inspiration. I also want to dedicate my dissertation work to my family and the wonderful friends who have supported me. A heartfelt thank you goes to my loving parents for their special role in this journey.



ACKNOWLEDGEMENTS

I want to start by sincerely thanking University Technical Malaysia Melaka (UTeM), especially the Faculty of Electrical Engineering (FKE), for giving me the chance to learn and forging relationships with my amazing instructors and friends. A huge thank you to Dr. Lor Ser Lee, my primary project supervisor, for her wonderful advice, support, and suggestions that greatly influenced the project as a whole.

Finally, my family and friends deserve my sincerest thanks. I will always be grateful for their unfailing support, assistance, and selflessness throughout this difficult period in my studies and final year project.



ABSTRACT

Automatic Number Plate Recognition (ANPR) technology is important and widely used in parking control, traffic monitoring and security surveillance. The current existing ANPR camera faces issues of higher costing, and the process of deployment and maintenance is complex since it requires more advanced technology and higher computational power. Hence, there is a need to develop a simpler but efficient ANPR system using simple devices where it can be applied easily. This project aims to develop an automatic number plate recognition system using a webcam on Predator Helios 300, analyze model accuracy through YOLOv7 algorithms, and examine the accuracy of the ANPR model under different sun ray conditions. This study focuses on the practical challenges of recognizing characters on number plates in the context of developing a real-time automatic number plate recognition (ANPR) system using a deep learning approach. Localization of number plates is the initial and essential step in automatic number plate recognition, which is then followed by character recognition. In this study, the number plate is first identified using a preprocessing technique to extract its features. It then goes through fully connected layers to classify and predict grid and cell division. A post-processing phase called non-maximum suppression (NMS) is used to reduce the number of overlapping candidate regions of interest (ROIs) and replace them with a single, more accurate detection. Combining YOLOv7, a powerful convolutional neural network, with Tesseract-OCR, a highly advanced optical character recognition engine, is crucial for achieving precise and instantaneous ANPR results. The study highlights the plate localization and character recognition, demonstrating the exceptional performance metrics of the YOLOv7 model: mean average precision of 0.99405, precision of 0.994168, and recall of 0.988764. This output is expected to be beneficial to smart parking systems, stolen vehicle identification, and automatic enforcement systems. It provides a reliable and efficient method for accurately recognizing license plates in real-time situations.

ABSTRAK

Teknologi Pengesanan Nombor Plat Automatik (ANPR) adalah penting dan digunakan secara meluas dalam kawalan tempat letak kereta, pemantauan trafik dan pengawasan keselamatan. Kamera ANPR sedia ada menghadapi isu kos yang tinggi, dan proses pengeluaran serta penyelenggaraan adalah kompleks kerana memerlukan teknologi yang lebih canggih dan kuasa pengkomputeran yang lebih tinggi. Oleh itu, terdapat keperluan untuk membangunkan sistem ANPR yang lebih mudah tetapi cekap menggunakan peranti mudah yang boleh digunakan dengan mudah. Projek ini bertujuan untuk membangunkan sistem pengesanan nombor plat automatik menggunakan kamera web pada Predator Helios 300, menganalisis ketepatan model melalui algoritma YOLOv7, dan mengkaji ketepatan model ANPR di bawah keadaan cahaya matahari yang berbeza. Kajian ini memberi tumpuan kepada cabaran praktikal untuk mengenal pasti aksara pada nombor plat dalam konteks pembangunan sistem pengesanan nombor plat automatik (ANPR) masa nyata menggunakan pendekatan pembelajaran mendalam. Penempatan nombor plat adalah langkah awal dan penting dalam pengesanan nombor plat automatik, yang kemudian diikuti oleh pengesanan aksara. Dalam kajian ini, nombor plat mula-mula dikenal pasti menggunakan teknik prapemprosesan untuk mengekstrak ciri-cirinya. Ia kemudian melalui lapisan bersambung sepenuhnya untuk mengelaskan dan meramalkan pembahagian grid dan sel. Fasa pasca pemprosesan yang dipanggil penekanan bukan maksimum (NMS) digunakan untuk mengurangkan bilangan kawasan calon yang bertindih dan menggantikannya dengan satu pengesanan yang lebih tepat. Menggabungkan YOLOv7, rangkaian neural konvolusi yang berkuasa, dengan Tesseract-OCR, enjin pengesanan aksara optik yang sangat maju, adalah penting untuk mencapai hasil ANPR yang tepat dan segera. Kajian ini menyoroti penempatan plat dan pengesanan aksara, menunjukkan metrik prestasi yang cemerlang model YOLOv7: ketepatan purata min sebanyak 0.99405, ketepatan sebanyak 0.994168, dan ingatan sebanyak 0.988764. Hasil ini dijangka memberi manfaat kepada sistem tempat letak kereta pintar, pengenalpastian kenderaan yang dicuri, dan sistem penguatkuasaan automatik. Ia menyediakan kaedah yang boleh dipercayai dan cekap untuk mengenal pasti nombor plat dengan tepat dalam situasi masa nyata.

TABLE OF CONTENTS

	PAGE
DECLARATION	
APPROVAL	
DEDICATIONS	
ACKNOWLEDGEMENTS	2
ABSTRACT	3
ABSTRAK	4
TABLE OF CONTENTS	5
LIST OF TABLES	7
LIST OF FIGURES	8
LIST OF SYMBOLS AND ABBREVIATIONS	10
LIST OF APPENDICES	11
CHAPTER 1 INTRODUCTION	12
1.1 Background and Motivation	12
1.2 Problem Statement	13
1.3 Objectives of the Study	14
1.4 Scope and Limitations	15
CHAPTER 2 LITERATURE REVIEW	16
2.1 Number Plate Classification	16
2.2 Image Processing	18
2.3 Methods for Number Plate Detection	18
2.4 Method for Alphanumeric Character Recognition	19
2.5 YOLO (You Only Look Once)	19
2.5.1 Architecture of YOLO	22
2.6 YOLOv7	23
2.6.1 Architectural Reforms	24
2.6.2 Trainable Bag of Freebies in YOLOv7	25
2.6.3 Focal Loss	27
2.6.4 YOLOv7 Experiments and Results	28
2.7 YOLOv7 and Its Application in Object Detection	29
2.8 Overview of Automatic Number Plate Recognition (ANPR)	30
2.9 Previous Works and Research in ANPR	31
2.10 Summary	37
CHAPTER 3 METHODOLOGY	39
3.1 Introduction	39
3.2 Project Flow	40

3.3	Data Selection	43
3.4	Data Labelling	44
3.5	Convolution Neural Network (Deep Learning)	45
	3.5.1 YOLOv7-tiny	46
	3.5.2 YOLOv7-tiny Training	46
	3.5.3 Tuning of the Hyperparameter	49
3.6	Detection or Recognition of Number Plate	49
3.7	Alphanumeric Character Recognition	50
3.8	Evaluation of Models and Results Collect	51
3.9	Components of the ANPR System	53
	3.9.1 Logitech C270 720p Widescreen Video Webcam Camera	53
3.10	Integration with Predator Helios 300	54
	3.10.1 Software Configuration	54
3.11	Ethics and Safety of the Method	55
3.12	Gantt Chart	56
3.13	Summary	57
CHAPTER 4 RESULTS AND DISCUSSIONS		58
4.1	Introduction	58
4.2	Model Train Accuracy	58
	4.2.1 YOLOv7 Tiny Model	59
	4.2.2 Comparison of Poor Trained Model versus Good Trained Model	60
4.3	Real Time Automatic Number Plate Text Extraction System using Predator Helios 300	62
	4.3.1 Real Time Simulation Using Predator Helios 300	62
	4.3.2 Discussion on Pass and Fail Ratio of Real-Time Simulation Using Predator Helios 300	63
4.4	Real Time Automatic Number Plate Localization Based on Accuracy	65
	4.4.1 Real Time Simulation Using Predator Helios 300	65
	4.4.2 Discussion Based on The Accuracy	73
4.5	Number Plate Localization and Extraction based on Confusion Matrix	75
	4.5.1 Real Time Simulation using Predator Helios 300	75
	4.5.2 Discussion Based on the Confusion Matrix	80
CHAPTER 5 CONCLUSION AND RECOMMENDATIONS		82
5.1	Conclusion	82
5.2	Future Works	83
REFERENCES		84
APPENDICES		87

LIST OF TABLES

Table 2.1: Format of Number Plate Currently Used in Malaysia	16
Table 2.2: Malaysia State Alphabet Represent	17
Table 2.3: Type of Image Processing	18
Table 2.4: Modified of YOLOv7	23
Table 2.5: Comparison of Various ANPR Systems	35
Table 3.1: Shows The Parts and The Amount Needed for The Project	53
Table 3.2: ISO/IEC Relate to AI and ML	55
Table 3.3: Gantt Chart of Project Implementation	56
Table 4.1: Result of YOLOv7-tiny Model	60
Table 4.2: Results of Previous Poor Trained YOLOv7-tiny Model	61
Table 4.3: Image Set Tested on Poor and Good Trained Model	61
Table 4.4: License Plate Extraction (%)	63
Table 4.5: License Plate Localization	66
Table 4.6: Confusion Matrix based on Morning	76
Table 4.7: Confusion Matrix based on Noon	78
Table 4.8: Confusion Matrix based on Afternoon	79

LIST OF FIGURES

Figure 2.1: Image with equal dimensions or grid cells	20
Figure 2.2: Example of Bounding Box	20
Figure 2.3: Image represents how IOU works	21
Figure 2.4: Process flow of the three techniques applied	21
Figure 2.5: YOLO Architecture	22
Figure 2.6: E-ELAN and Previous Work on Maximal Layer Efficiency	24
Figure 2.7: YOLOv7 compound scaling	25
Figure 2.8: Re-parameterization Trials	26
Figure 2.9: Average Precision (AP) and Speed of YOLOv7 vs other YOLO	28
Figure 2.10: YOLOv7 vs Other Baseline Models	28
Figure 2.11: YOLOv7 Models FPS Comparison	29
Figure 2.12: General Architecture for the ANPR System using KNN Method	32
Figure 2.13: CT5L Method Block Diagram	33
Figure 3.1: Project Flow	41
Figure 3.2: Automatic Number Plate Recognition System Flow	42
Figure 3.3: The Separation of the Dataset	43
Figure 3.4: Image of the Vehicles with Number Plate	44
Figure 3.5: Theos Labelling Tool	44
Figure 3.6: Image with Number Plate Labelling	45
Figure 3.7: Layers of CNN	46
Figure 3.8: Workflow on YOLOv7 Training	48
Figure 3.9: Epochs and Batch Size	49
Figure 3.10: Tesseract OCR Process	51
Figure 3.11: Diagram of IoU	51
Figure 3.12: Logitech C270 720p Widescreen Video Webcam Camera	53
Figure 4.1: mAP of the model	59
Figure 4.2: AP of the model	59
Figure 4.3: Recall of the model	60
Figure 4.4: Confidence Levels Across Detected Cars (SET A)	66
Figure 4.5: Confidence Levels Across Detected Cars (SET B)	68
Figure 4.6: Confidence Levels Across Detected Cars (SET C)	70

Figure 4.7: Graph Morning vs Noon	71
Figure 4.8: Graph Noon vs Afternoon	72
Figure 4.9: Graph Morning vs Noon vs Afternoon	72
Figure 4.10: Morning Performance Using Confusion Matrix	75
Figure 4.11: Noon Performance Using Confusion Matrix	77
Figure 4.12: Afternoon Performance Using Confusion Matrix	78



LIST OF SYMBOLS AND ABBREVIATIONS

ANPR	-	Automatic Number Plate Recognition System
NMS	-	Non-Maximum Suppression
ROI	-	Region of Interest
AP	-	Average Precision
mAP	-	Mean Average Precision



LIST OF APPENDICES

APPENDIX A	Coding in PyCharm (Image.py)	83
APPENDIX B	Coding in PyCharm (Webcam.py)	83
APPENDIX C	Coding in PyCharm (Real-Time)	84



CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

The daily traffic congestion, rush hour, and safety concerns have greatly affected our life. Traffic control and law enforcement require smart solutions as cities develop and road networks become increasingly complicated. Due of these issues, Automatic Number Plate Recognition (ANPR) technology is necessary for tracking and identifying cars. Automatic Number Plate Recognition (ANPR) systems were originally intended for security, but they are now crucial to smart city initiatives and affect traffic, urban planning, and law enforcement.

ANPR technology has advanced due to deep learning, particularly computer vision and machine learning. Computer vision and machine learning systems can read license plates and adapt to varied environments. Convolutional Neural Networks (CNNs) improve accuracy and decode complicated visual patterns in this method. ANPR in metropolitan locations is problematic, especially when processing real-time data on low-power devices. This study examines the problems of constructing a deep learning-based real-time automatic license plate detection system for the Predator Helios 300.

This research aims to simplify real-time ANPR processing, notably on devices like the Predator Helios 300. The research decodes this complexity to develop ANPR technology and create safer, more dynamic, and intelligent cities. Parking control, access management, traffic monitoring, and security surveillance use ANPR.

Pattern recognition and image processing-based ANPR systems can have accuracy and computational challenges, especially in unfavorable conditions. As urbanization and automobile traffic rise, new solutions are needed. Due of its flexibility and growing importance to smart city programs, ANPR may solve urban issues.

For real-time ANPR processing, this study uses the Predator Helios 300 as a computer platform to investigate a compact yet powerful solution for smart cities. The research introduces a Tesseract-OCR and YOLOv7-based technique to increase ANPR system functionality and performance. Deep learning, character recognition, and number plate location might create an accurate ANPR system and advance smart city technologies.

1.2 Problem Statement

Automatic License Plate Recognition (ALPR) technology has improved security, shared area, parking, and acceleration management. However, the rapid growth of cars and transit networks makes public oversight difficult. Thus, automated car number plate number recognition is essential to our daily lives. Real-time image processing for license plate recognition provides substantial computational challenges despite the development of many approaches, methodologies, and algorithms. The computational power required for real-time picture processing and license plate identification is a major obstacle. Effective Automatic Number Plate Recognition (ANPR) solutions, especially those for Predator Helios 300, are also in demand. Despite their adaptability, these devices have little computing power.

Deploying a deep learning-based real-time automatic license plate recognition system on Predator Helios 300 requires understanding and overcoming the challenges. Optimizing ANPR systems' efficiency and speed is crucial, especially when processing enormous volumes of video data. Working with high-resolution photographs makes this task considerably harder, requiring more sophisticated computational tools. This study examines Tesseract-OCR and YOLOv7 for Predator Helios 300 ANPR to handle these complicated challenges. The key difficulty is improving ANPR systems' real-time processing while preserving accuracy and flexibility in dynamic metropolitan environments. These challenges must be resolved for ANPR to integrate smoothly into urban life, especially as cities grow and incorporate smart technology.

1.3 Objectives of the Study

The project aim is to propose a number plate recognition system using contour method. To achieve the goal, the objectives of this project are:

1. To develop automatic number plate recognition system using Convolutional Neural Network
2. To analyze the accuracy of model based on mean average precision (mAP), average precision (AP) and recall using YOLOv7 algorithms
3. To evaluate the accuracy of the proposed model under different sun ray conditions



1.4 Scope and Limitations

The scope and limitations of the project are:

1. The format of the number plate is only limit for Malaysia number plate.
2. The number plate spacing format must follow the rules given by JPJ (Road Transport Department Malaysia).
3. Number classification is divided into 10 classes from 0 to 9.
4. Capital alphabet classification is divided into 26 classes from A to Z.
5. Only vehicles are tested with the trained model.
6. The system's effectiveness may vary due to changing environments conditions, such as different lighting, weather, or camera angles, and diverse license plate formats.
7. The morning range is defined as 10 AM to 11 AM, the noon range is around 12 PM, and the afternoon range is defined as 5 PM to 6 PM



CHAPTER 2

LITERATURE REVIEW

2.1 Number Plate Classification

Using number plate classification, the alphanumeric characters on the plate are categorized. There are 36 classifications for the alphanumeric character. The classes used the capital letters A through Z and the digits 0 through 9. There are three federal territories and thirteen states in Malaysia. Every state and federal jurisdiction presents the alphabet in a different way. The formats of the number plates that are now in use are shown in Table 2.1 below.

Table 2.1: Format of Number Plate Currently Used in Malaysia

Type of Number Plate	Layout
Private and commercial vehicles	ABC 1234 or WA 1234 B or QAB 1234 C or SAB 1234 C or KV 1234 C
Taxi	HAB 1234 or as used private \$ commercial vehicles
Military	ZA 1234
Temporary	A 1234 A (W / TP 1234 for Kuala Lumpur)
Diplomatic corps	12-34-DC
Royals and government	(Full Title)

Table 2.2: Malaysia State Alphabet Represent

State	Alphabet
Perak	A
Selangor	B
Pahang	C
Kelantan	D
Putrajaya	F
Johor	J
Kedah	K
Melaka	M
Negeri Sembilan	N
Penang	P
Perlis	R
Terengganu	T
Kuala Lumpur	W / V
Sarawak	QA* * Depend on the district Example: Kuching – QK / QA Sibu - QS
Sabah	SA* * Depend on the district Example: Sandakan – SS / SM West Coast – SA

Furthermore, Malaysia has a few unique series of license plates. For instance:

Langkawi - KV 1234 B

Putrajaya – Putrajaya 1234

Military – ZB* 1234 (B* is branch prefix. Ex: ZD – Malaysia Army, ZU – Royal Malaysian Air Force)

Malaysia, PROTON, PERODUA, WAJA. Chancellor, Putra, Persona, Satria, Tiara, Perdana, LOTUS, KRISS, Jaguh, NAZA, SUKOM, BAMbee, XIINAM, XOIC, XXVIASEAN, XXXIDB, 1M4U, PATRIOT, PERFECT, TTB, NAAM, VIP, RIMAU, IQ, G, GG, G_G, GT, 15 GTR, G1M, GP, A1M, T1M, K1M, NBOS, Q,

SAS, SAM, SMS, E, X, XX, Y, YY, YA, U, UU, UUU, US, UP, UA, UT, UMT, UM, UTM, UKM, UUM, USM, UiTM, UPM, IIUM, UMK, UTEM, UR and UC.

2.2 Image Processing

The process of performing numerous operations to a picture to produce an enhanced image or extract some important information from it is known as image processing [1]. This type of signal processing takes a picture as input, and the result could be another image, or it could be a set of features or attributes associated with the original image. With the daily rapid advancement of technology, image processing is becoming increasingly crucial. Global society's primary focus of research is gradually shifting to image processing. The five main categories of image processing are shown in Table 2.3 below.

Table 2.3: Type of Image Processing

Type of Image Processing	Description
Visualization	Find objects that are not visible in the image
Recognition	Distinguish or detect object in the image
Sharpening and Restoration	Create an enhanced image form the original image
Pattern Recognition	Measure the various patterns around the objects in the image
Retrieval	Browse and search image from a large database of digital images that are similar to the original image

2.3 Methods for Number Plate Detection

Using cutting-edge technology, number plate identification can identify or detect a car's license plate and read the alphanumeric characters on it. It is crucial to the creation of intelligent transportation systems as well. There are various techniques for identifying license plates. These days, the number plate recognition system uses the most recent techniques, including computer vision-powered algorithms, deep learning, machine learning, and artificial intelligence (AI) to scan the number plate without the need for human intervention [2]. The number plate can be found and detected from an image using a technique called morphological operators, as reported in journal [3]. The number plate will first perform pre-processing, which involves adjusting the image's RGB color, then enhancement, which involves eliminating noise

from the image, and lastly contour detection. According to the journal [4], another technique is YOLO. They have employed YOLOv3 to locate the location of a license plate by taking pictures, preprocessing them, and identifying the region of interest (ROI).

2.4 Method for Alphanumeric Character Recognition

Character recognition is the next step in number plate alphanumeric identification. Nowadays, OCR is utilized to recognize alphanumeric characters. Pattern matching and feature extraction are OCR algorithms' main categories. OCR produces character output from optical images [5]. Computers can recognize and extract text from scanned documents and photographs using Google's open-source OCR engine, Tesseract-OCR. It also processes scanned documents and supports several languages.

2.5 YOLO (You Only Look Once)

The abbreviation YOLO, which translates to "You Only Look Once," denotes an algorithm that has been purposefully developed to efficiently identify and categorize objects within images. By addressing object detection as a regression problem, this algorithm generates class probabilities for the objects that are identified. Convolutional neural networks (CNNs) are employed by the YOLO algorithm in order to accomplish object detection in real time. As its name implies, object detection with this algorithm requires only a solitary forward transit through the neural network. Three techniques comprise the YOLO algorithm: intersection over union (IOU), bounding box regression, and residual blocks.

Residual blocks are generated during the initial phase, wherein the image is partitioned into several grids, with each grid having the dimensions $S \times S$. As shown in the figure below, the image contains numerous grid cells of equal size; each grid cell will identify objects that occur within it.

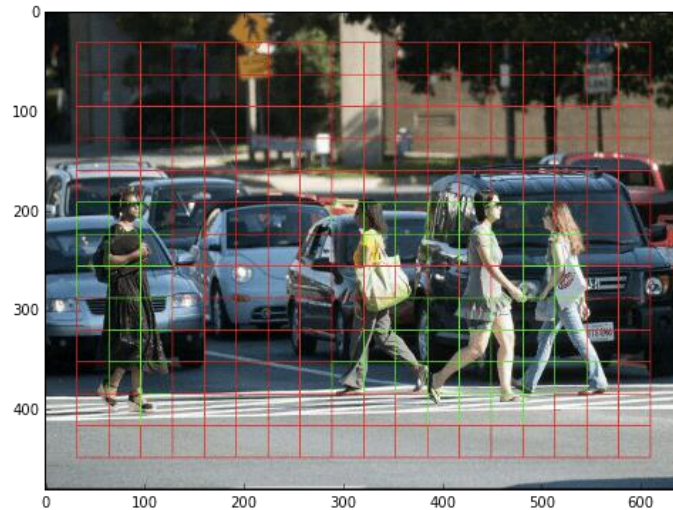


Figure 2.1: Image with equal dimensions or grid cells

Bounding box regression, the second approach, operates by predicting the properties of a bounding box, a contour designed to highlight an object within an image. The specified values for the width (b_w), height (b_h), class (represented by c as human, automotive, or traffic signal), and centre coordinates (b_x , b_y) correspond to each bounding box in the image. To illustrate this, consider a bounding box example from Figure 2.2. The red outline denotes the bounding box, and YOLO calculates the size, center coordinates, and type of items contained within the box using a singular bounding box regression. Indicated is the probability that an object will be present within the bounding box.

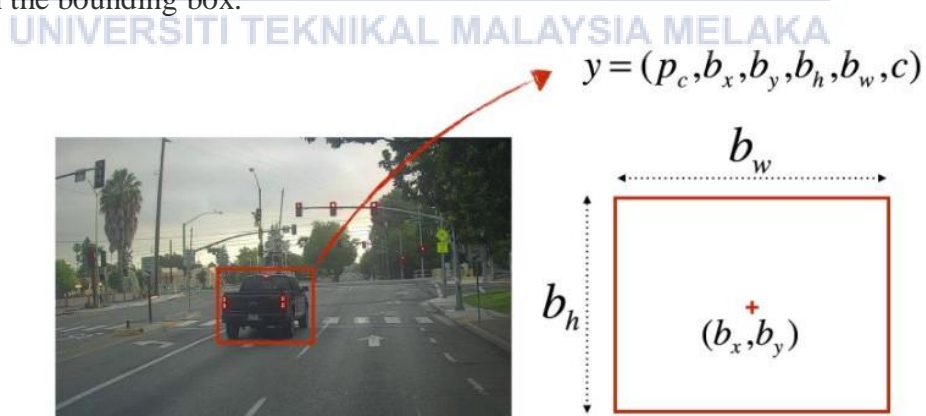


Figure 2.2: Example of Bounding Box

Intersection Over Union (IOU) represents the concluding practice. IOU is a crucial concept in object detection due to the fact that it quantifies the overlap between bounding boxes. IOU is utilized by the YOLO algorithm to generate precise bounding boxes that enclose the objects securely. Each grid cell in YOLO is tasked with the prediction of bounding boxes along with their corresponding confidence scores. IOU value becomes 1 when the predicted bounding box precisely matches the actual box. This mechanism eliminates any bounding boxes that fail to correspond with the actual ground truth box. This concept is visually represented in Figure 2.3, which depicts two bounding boxes: the ground truth box is denoted by the green box, and the predicted box is denoted by the red box. YOLO verifies the correspondence between these two bounding areas.

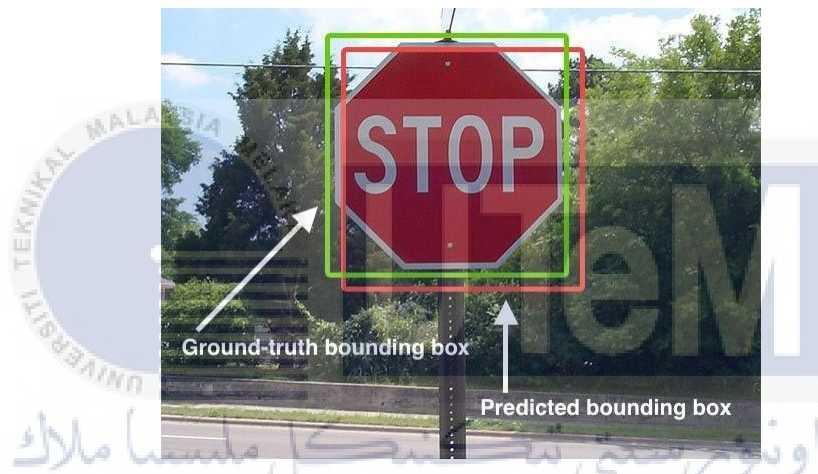


Figure 2.3: Image represents how IOU works

The figure below illustrates the integration of these three methodologies and the process flow, as well as the resulting detection outcomes.

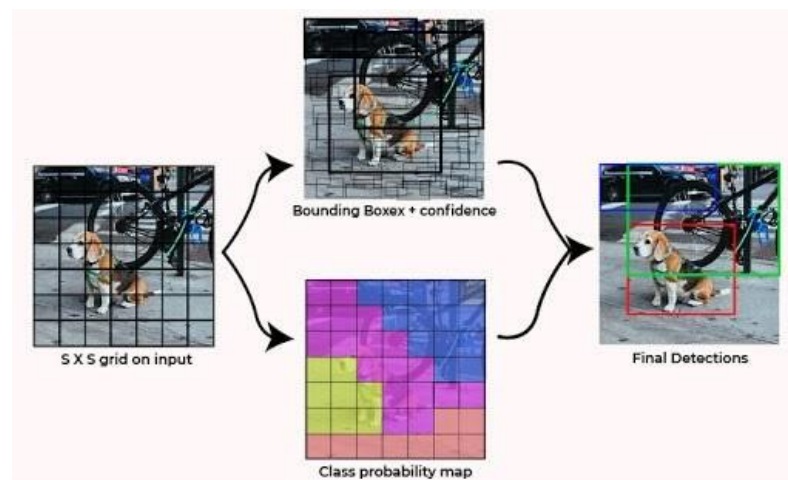


Figure 2.4: Process flow of the three techniques applied

2.5.1 Architecture of YOLO

Figure 2.5 illustrates the YOLO architecture [6], which is used for image classification and is based on the GooLeNet model. It has 24 convolutional layers and 2 fully connected layers, making it nearly five times larger than the ZF-5 model utilized in the SPP-net and Faster-RCNN architecture. First, the image is separated into $S \times S$ grids ($S=7$, meaning that each 64×64 sub-image corresponds to a single grid) and scaled to a $L \times L$ dimension, which is typically 448×448 . Every grid proposes B boundary boxes, where B usually equals 2. Each bounding box is represented by four coordinates (x_{center} , y_{center} , width, and height) together with a confidence level. The dimensions of the enclosed box are height and breadth, and its center coordinates are its x_{center} and y_{center} coordinates.

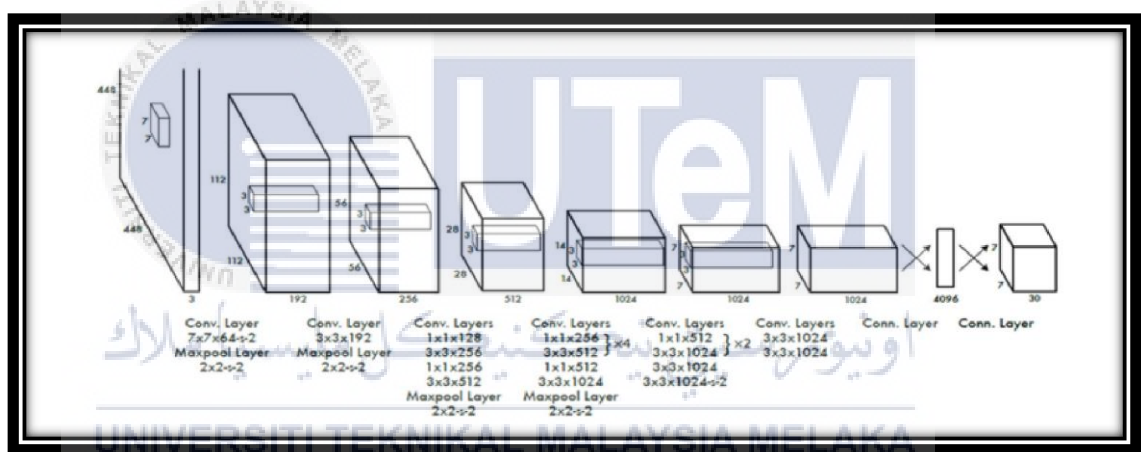


Figure 2.5: YOLO Architecture

The confidence value is computed using the following equation:

$$Confidence = Pr(Object) \times IOU \frac{truth}{pre}$$

If a ground truth box exists within the grid cell, the value of $Pr(Object)$ is 1; otherwise, it is 0. The term "IOU" stands for Intersection over Union, which represents the value obtained by calculating the overlap between the actual ground truth box and the predicted bounding box. Furthermore, each grid generates conditional class probabilities, denoted as $Pr(Class_{ii}|Object)$, with C usually being equal to 20. Consequently, an image produces $S \times S (B \times 5 + C)$ outputs, where S

represents the size of the image, B represents a certain value, and C represents another value. In this case, the calculation is $7 \times 7(2 \times 5 + 20)$, resulting in a total of 1,470 outputs. During testing, YOLO combines the conditional class probabilities with the individual box confidence forecasts to generate class-specific confidence ratings for each bounding box.

$$\Pr(\text{Class}_i | \text{Object}) \times \Pr(\text{Object}) \times IOU_{pre}^{truth} = \Pr(\text{Class}_i) \times IOU_{pre}^{truth}$$

2.6 YOLOv7

The publication of YOLOv7 occurred in 2022, and it incorporates numerous architectural modifications aimed at enhancing both accuracy and speed [7]. YOLOv7 does not utilize ImageNet pre-trained backbones for its backbones, like to Scaled YOLOv4. Exclusively, the models are trained using only the COCO dataset. This proximity is anticipated due to the fact that Scaled YOLOv4, an expansion of YOLOv4, and YOLOv7 were authored by the same individuals. The subsequent notable alterations for YOLOv7 are mentioned in table below.

Table 2.4: Modified of YOLOv7

Architectural Reforms	Trainable BoF (Bag of Freebies)
E-ELAN (Extended Efficient Layer Aggregation Network)	Planned re-parameterized convolution
Model Scaling for Concatenation-based Models	Coarse for auxiliary and Fine for lead loss

2.6.1 Architectural Reforms

YOLOv7 was developed as a result of combining YOLOv4, Scaled YOLOv4, and YOLO-R [7]. The YOLOv7 models, which have undergone more research and refinement, function as the underlying structure. The computational component of the YOLOv7 backbone, known as the E-ELAN, builds upon previous studies to enhance the efficiency of the network. Factors such as memory access cost, I/O channel ratio, element-wise operations, activations, and gradient path were meticulously considered to enhance both the speed and accuracy. The suggested E-ELAN algorithm enhances the learning capacity of the network by including techniques such as expand, shuffle, and merge cardinality, while ensuring the accuracy of the original gradient route. In other words, the E-ELAN architecture, which is built upon the ELAN computing block, improves the learning capabilities of the framework.

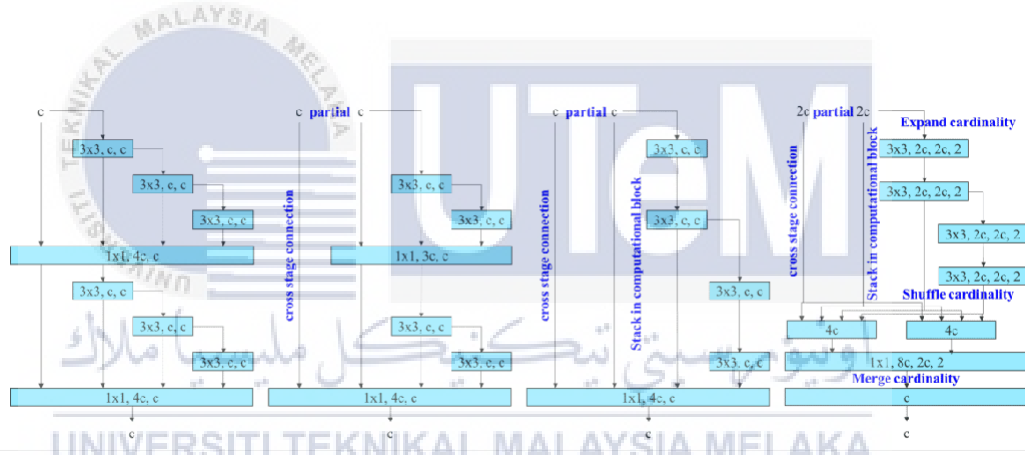


Figure 2.6: E-ELAN and Previous Work on Maximal Layer Efficiency

YOLOv7 uses compound model scaling to meet different application needs. This technique allows models to be prioritized for high speed or accuracy, making them suitable for application on a range of computing platforms. When scaling a model, several factors are taken into account. Resolution denotes the size of the input image; width denotes the number of channels; depth denotes the number of layers; and stage denotes the number of feature pyramids. These are some of the parameters. Network Architecture Search (NAS) techniques are widely used by researchers to scale models successfully. The optimal scaling factors in NAS are discovered by iterative testing of various parameter combinations. However, it's crucial to remember

that methods such as NAS often employ parameter-specific scaling, in which the scaling factors are unrelated to each other.

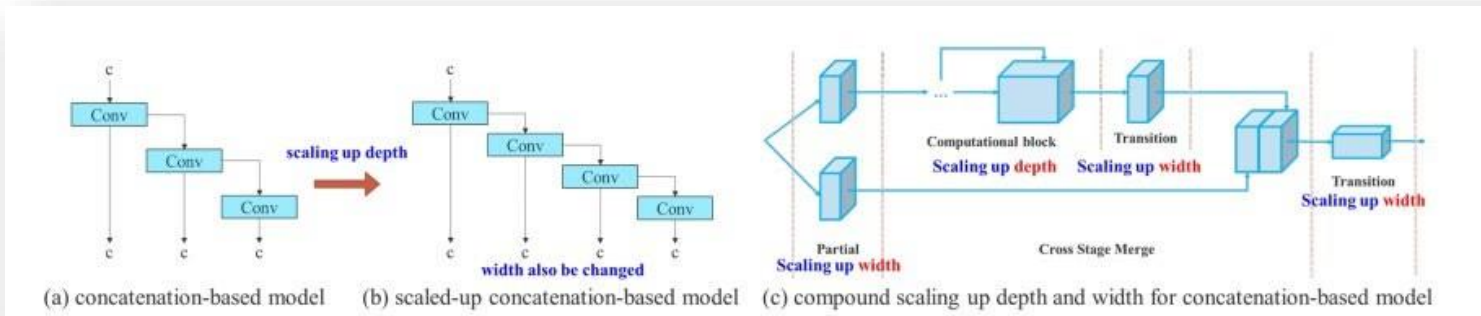


Figure 7: YOLOv7 compound scaling

2.6.2 Trainable Bag of Freebies in YOLOv7

The strategies used to increase a model's performance without raising training expenses are known as trainable BoF. In YOLOv7, the BoF technique is presented. One of the methods employed in BoF is planned re-parameterized convolution, which improves a model's performance post-training. The results of inference are superior even with the extended training period. Two re-parameterization techniques are employed to finish the models: model level and module level ensemble. Re-parameterizing models at the model level can be done in two ways. First off, the same settings and training data can be used to train numerous models. The weights of the several models are averaged to form the final model. Alternatively, the weights of the models at several epochs might be averaged to produce the desired results. Recently, module level reparameterization has garnered a lot of attention from researchers. This approach divides the model-training procedure into multiple modules, the output of which is integrated to generate the final model.

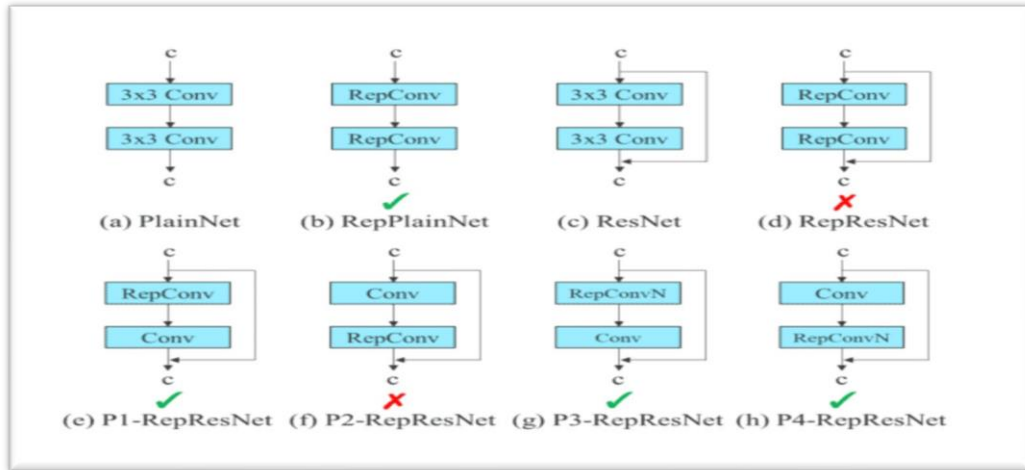


Figure 2.8: Re-parameterization Trials

In the Bag-of-Features (BoF) technique, a coarse loss function is utilized for the major work, while a coarse loss function is employed for the auxiliary duty. The three fundamental components of the YOLO architecture are the head, neck, and backbone. It is the head's responsibility to deliver the necessary outcomes. Compared to earlier versions, YOLOv7 has many heads, which increases its functional versatility. Recall that the concept of multiple heads in a framework has already been discussed, particularly with regard to deep supervision methods applied in deep learning models. An Auxiliary Head is utilized to support training in the early stages of YOLOv7, while the Lead Head is crucial to producing the finished result. Deep supervision and improved model learning are made possible by the employment of an auxiliary loss to update the weights of the auxiliary heads. The Lead Head and the Label Assigner are the sources of these concepts. The Label Assigner method is important in YOLOv7 since it assigns soft labels instead of hard labels and considers both the ground truth and the prediction output of the network. This approach increases training flexibility and improves model performance by providing coarse and soft labels.

2.6.3 Focal Loss

New loss function "focal loss" improves YOLOv7. Convolutional neural networks (CNNs) employ focus loss to find objects. Lin's 2017 study "Focal Loss for Dense Object Detection" discussed it [8]. The focused loss solves the class imbalance problem that often comes up in object recognition tasks when there are a lot more background (negative) examples than foreground (positive) examples. This difference in class sizes could bring bias into the training process, which could affect how well the model works. The targeted loss changes the common cross-entropy loss by adding a changing part that makes cases that were correctly classified less important and cases that were wrongly classified more important. By adding more weight to samples that were wrongly classified, it lets the model focus more on tough cases during training. By adding the focused loss to the training process, the model can learn to focus on tough cases and do better, especially when the class distributions aren't balanced.

Formula for focal loss:

$$Focal(p_t) = -(1 - p_t)^\gamma \log(p_t)$$

p_t = predicted probability of the correct class,

γ = Tunable parameter that controls the rate at which the loss is down-weighted for well-classified examples. When gamma is set to 0, the focal loss reduces to the standard cross-entropy loss.

2.6.4 YOLOv7 Experiments and Results

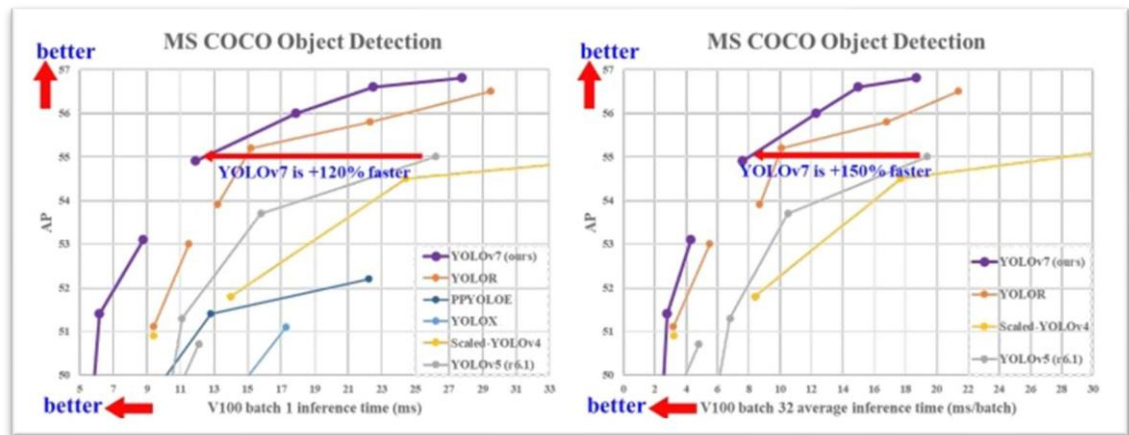


Figure 2.9: Average Precision (AP) and Speed of YOLOv7 vs other YOLO

Based to the data presented in Figure 2.9, we can observe that the speed and accuracy of the YOLOv7 models have surpassed those of the earlier object detectors. Furthermore, it provides a good AP and speed that is 120% faster than the other.

Model	#Param.	FLOPs	Size	AP^{val}	AP_{50}^{val}	AP_{75}^{val}	AP_S^{val}	AP_M^{val}	AP_L^{val}
YOLOv4	64.4M	142.8G	640	49.7%	68.2%	54.3%	32.9%	54.8%	63.7%
YOLOR-u5 (r6.1)	46.5M	109.1G	640	50.2%	68.7%	54.6%	33.2%	55.5%	63.7%
YOLOv4-CSP	52.9M	120.4G	640	50.3%	68.6%	54.9%	34.2%	55.6%	65.1%
YOLOR-CSP	52.9M	120.4G	640	50.8%	69.5%	55.3%	33.7%	56.0%	65.4%
YOLOv7	36.9M	104.7G	640	51.2%	69.7%	55.5%	35.2%	56.0%	66.7%
improvement	-43%	-15%	-	+0.4	+0.2	+0.2	+1.5	=	+1.3
YOLOR-CSP-X	96.9M	226.8G	640	52.7%	71.3%	57.4%	36.3%	57.5%	68.3%
YOLOv7-X	71.3M	189.9G	640	52.9%	71.1%	57.5%	36.9%	57.7%	68.6%
improvement	-36%	-19%	-	+0.2	-0.2	+0.1	+0.6	+0.2	+0.3
YOLOv4-tiny	6.1	6.9	416	24.9%	42.1%	25.7%	8.7%	28.4%	39.2%
YOLOv7-tiny	6.2	5.8	416	35.2%	52.8%	37.3%	15.7%	38.0%	53.4%
improvement	+2%	-19%	-	+10.3	+10.7	+11.6	+7.0	+9.6	+14.2
YOLOv4-tiny-3l	8.7	5.2	320	30.8%	47.3%	32.2%	10.9%	31.9%	51.5%
YOLOv7-tiny	6.2	3.5	320	30.8%	47.3%	32.2%	10.0%	31.9%	52.2%
improvement	-39%	-49%	-	=	=	=	-0.9	=	+0.7
YOLOR-E6	115.8M	683.2G	1280	55.7%	73.2%	60.7%	40.1%	60.4%	69.2%
YOLOv7-E6	97.2M	515.2G	1280	55.9%	73.5%	61.1%	40.6%	60.3%	70.0%
improvement	-19%	-33%	-	+0.2	+0.3	+0.4	+0.5	-0.1	+0.8
YOLOR-D6	151.7M	935.6G	1280	56.1%	73.9%	61.2%	42.4%	60.5%	69.9%
YOLOv7-D6	154.7M	806.8G	1280	56.3%	73.8%	61.4%	41.3%	60.6%	70.1%
YOLOv7-E6E	151.7M	843.2G	1280	56.8%	74.4%	62.1%	40.8%	62.1%	70.6%
improvement	=	-11%	-	+0.7	+0.5	+0.9	-1.6	+1.6	+0.7

Figure 2.10: YOLOv7 vs Other Baseline Models

When compared to YOLOv4, YOLOv7 achieves 1.5% higher average precision (AP) and 75% fewer parameters with 36% less processing. In comparison to YOLOR, YOLOv7 achieves 0.4% greater AP, 15% less computing required, and a 43% reduction in parameters. On AP, YOLOv7-tiny outperforms YOLOv5-N by a whopping 127 frames per second and 10.7% in accuracy. In comparison to the similar YOLOv5-L with 99 FPS, the version YOLOv7-X achieves 114 FPS inference speed; nevertheless, YOLOv7 achieves greater accuracy (higher AP by 3.9%) which is shown in Figure 2.10.

Model	Parameters (million)	FPS	AP test (%)
YOLO7-Tiny	6.2	286	38.7
YOLOv7	36.9	161	51.4
YOLOv7-X	71.3	114	53.1
YOLOv7-W6	70.04	84	54.9
YOLOv7-E6	97.2	56	56.0
YOLOv7-D6	154.7	44	56.6
YOLOv7-E6E	151.7	36	56.8

Figure 2.11: YOLOv7 Models FPS Comparison

A detailed comparison of YOLOv7's FPS with other models is given in Figure 2.11 of the YOLOv7 study. It also has the comparison of COCO with mAP.

2.7 YOLOv7 and Its Application in Object Detection

YOLOv7 has been successfully applied in various object detection scenarios, including underwater sea cucumber detection [9], object detection in low-light foggy traffic environments [10], real-time weed detection in agriculture [11], urban vehicle detection [12], and mask detection for COVID-19 monitoring [13]. These applications demonstrate the versatility and real-time capabilities of YOLOv7, which are essential for ANPR systems. The study by [9] demonstrated the effectiveness of YOLOv7 in detecting sea cucumbers in underwater scenes, showcasing its potential for real-time object detection in challenging environments [9]. Additionally, the work by [10] focused on using YOLOv7 for image dehazing to improve object detection in low-

light foggy traffic environments, highlighting its adaptability to varying environmental conditions [10].

Furthermore, Narayana & Ramana [11] presented an efficient real-time weed detection technique using YOLOv7, indicating its applicability in agricultural settings [11]. Moreover, Elhanashi et al. [13] demonstrated the use of YOLOv7 for mask detection and social distancing monitoring, showcasing its potential for real-time applications in public safety and security [13]. Additionally, it has been applied in the efficient detection of license plates, which is relevant to the Automatic Number Plate Recognition (ANPR) system [14].

The literature review indicates that YOLOv7 has been successfully employed in various object detection applications, demonstrating its effectiveness in different domains. These applications provide valuable insights into the potential use of YOLOv7 in the development of an Automatic Number Plate Recognition (ANPR) system, showcasing its capabilities in real-time detection and recognition tasks. Overall, the literature supports the potential of YOLOv7 in object detection, including its application in the development of an Automatic Number Plate Recognition (ANPR) system. The diverse range of applications demonstrates the algorithm's adaptability and effectiveness in different contexts, providing a strong foundation for its potential utilization in ANPR systems.

2.8 Overview of Automatic Number Plate Recognition (ANPR)

Automatic Number Plate Recognition (ANPR) systems have gained significant attention due to their wide range of applications. ANPR is computationally intensive and involves high system cost with memory and processing time constraints [15]. These systems have been beneficial in numerous applications such as automatic toll collection, criminal pursuit, traffic law enforcement, traffic monitoring, automatic ticketing of vehicles at parking areas, tracking vehicles during signal violation, automatic toll collection at toll plazas, access control in parking areas and buildings, border control, stolen car detection, journey time measurement, marketing research, and in many other applications with huge savings of human effort and cost [16][17]. ANPR systems are widely used for traffic detection, electronic toll collection, and

vehicle identification, in terms of ‘non-intrusive’ technology, which does not need any installation or equipment in the vehicle [18].

The ANPR process typically involves three stages: license plate localization, character segmentation, and optical character recognition [19]. These systems are actively being researched, especially alongside the rising interests in autonomous driving and intelligent transportation systems, where the need for human intervention is minimized by the system to determine vehicle identity [20]. ANPR or License Plate Recognition (LPR) has many applications such as traffic surveillance, stolen car detection, speed limit enforcements, parking lot control, or even border management [21]. The main aim of ANPR systems is to replace manual systems with an automated system for proper identification and localization of number plate information [22]. In the last stage, characters are segmented from the number plate so that only useful information is retained for recognition where the image format will be converted into characters [23].

In conclusion, ANPR systems have a wide range of applications and are actively being researched for their potential in various fields such as traffic management, law enforcement, and automation. Despite the computational intensity and high system cost, the benefits in terms of efficiency, cost savings, and automation make ANPR systems a valuable area of research and development.

2.9 Previous Works and Research in ANPR

These days, a lot of automatic number plate recognition (ANPR) systems have been created. Convolutional Neural Networks (CNNs) based on deep learning [24] as a means of identifying Iraqi car number plates. CNN was trained using two datasets: the first, which included 2000 images of Arabic numerals and characters, and the second, which included 1200 images of the Arabic names of Iraqi governorates. They also employed image processing techniques. They were effective at every level, attaining a 97% success rate overall.

YOLOv5 was utilized in the creation of the ANPR system [25]. They contend that the YOLOv5 ANPR system is an advanced, quick, and memory-efficient

automatic number plate recognition system suitable for Internet of Things (IoT) devices with constrained memory and processing capacity. They were using an LSTM-based OCR engine and a custom transfer-learned model to identify and detect license plates. The data utilized in the study came from the Google Open Images dataset and the collection of Indian number plates. Their proposed ANPR system has a final mean average precision of 87.2%. They did, however, also highlight some of the system's shortcomings, such as its inability to function in low light or at night and its lack of real-world performance monitoring.

Principal Component Analysis and the K-Nearest Neighbor Algorithm (PCA) were used to construct an ANPR system for Indonesian number plates [26]. Using the ANPR method they recommended, the number plate in the surveillance camera footage is identified. They collected 125 number plate photos, split them into 25 testing and 100 training photos, and attained a 92.86% success rate. The problems that have been encountered—such as license plate modifications, characters that are printed closely together or similarly—as well as how the location of the number plate, its condition, and its intensity affect the outcomes—have been described. Manual license plate retrieval is still necessary. The general design of the ANPR system that they proposed is seen in the diagram below.

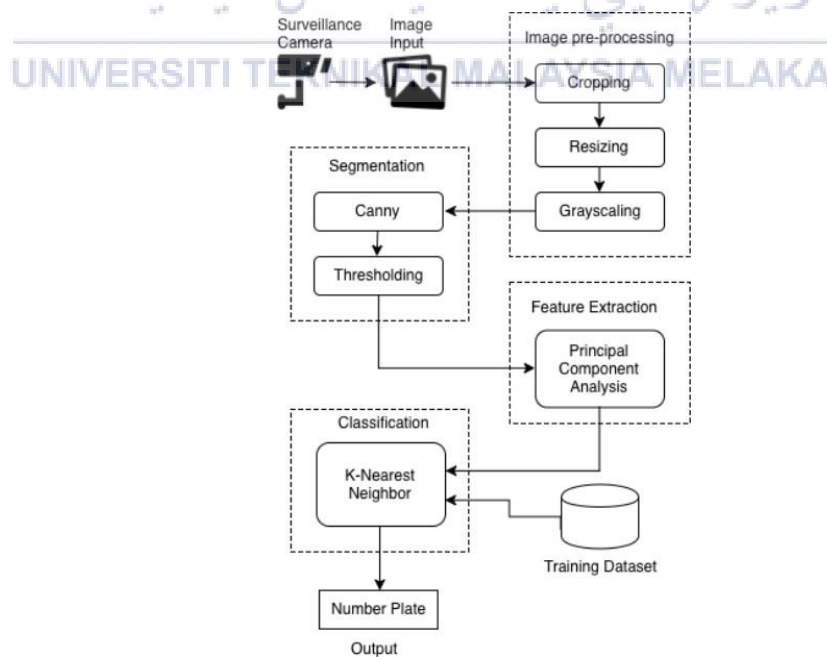


Figure 2.12 : General Architecture for the ANPR System using KNN Method

An ANPR system with machine learning in place of number of recognitions was created using CT5L approach [27]. According to them, CT5L is a cutting-edge technique that uses techniques to preserve the continuity of black-and-white pixel values inside number plates in both the vertical and horizontal directions. The Hough transform only finds vertical and horizontal lines. The system performed better than the conventional fixed threshold technique, Otsu's method, and the conventional JavaANPR method. Compared to the fixed case, the detection rate increased by 20% to 74.24%. Figure 2.13 depicts the essential stages of the CT5L technique.

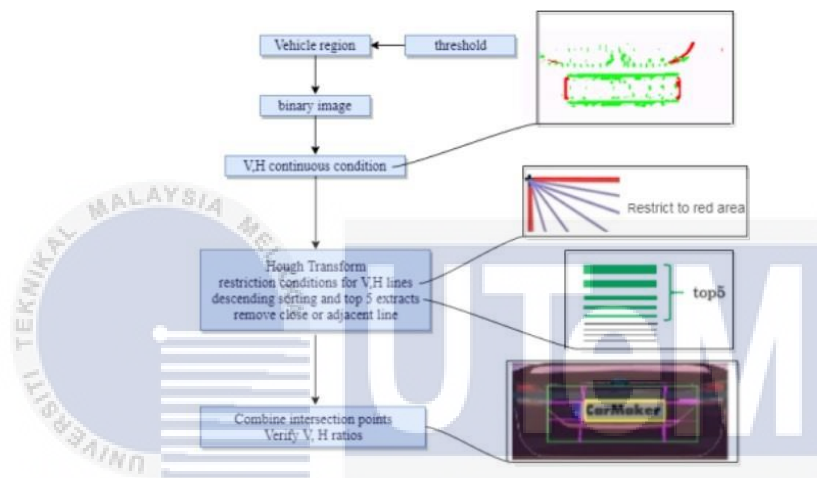


Figure 2.13: CT5L Method Block Diagram

An exceptionally precise Automatic Number Plate Recognition (ANPR) system has been developed using Depthwise Separable Convolution Networks [28]. The most recent deep learning techniques and innovative network architecture make up their suggested network model. In order to improve accuracy while using less processing resources, they examined three different datasets. The system produced results that were over 70 frames per second and had an accuracy of over 99.60%. The inability to read number plates at night due to back lights' propensity to overexpose cameras, as well as the distance and skewed angles between the camera and the car, were among the difficulties they encountered, they did remark.

The suggested ANPR system uses computer vision techniques for character segmentation (CS) and number plate localization (NPL) [29]. It also proposed an approach to optical character recognition (OCR) based on deeper learning (DL). Following the NPL and CS stages, the detected number plate number is identified using the CNN algorithm. They used the Jetson TX2 Nvidia target to train the model on a range of image datasets, and the accuracy results were 95.84%.

A convolutional neural network (CNN) was utilized for optical character recognition (OCR) in an ANPR system in which the YOLO v3 model was employed to identify the area of interest (ROI) [4]. They created a dataset called "Indian Number Plate" typeface, which included 6459 pictures of different alphanumeric characters, in order to train the OCR model. In addition, a separate dataset including 1500 images was produced specifically for the YOLO model training. The strategy was to first locate the ROIs, then apply preprocessing techniques to improve the images, and lastly run the images through the CNN model to recognize optical characters. The OCR model's accuracy was 91.5 percent, while the YOLO model's accuracy was 96%.

YOLO v5 and the Easy-OCR method are combined to create an automatic number plate recognition system [30]. Two fundamental concepts are used in their approach for both object detection and optical character recognition. These models were created with 500 training epochs, an 8-person batch size, and 640 x 480 pixels of images. YOLO v5 and Easy-OCR worked together to produce a real-time ANPR solution. Significantly, the Easy-OCR approach's character identification accuracy of 95% was significant.

YOLOv5 model and transfer learning technique have demonstrated a real-time automobile license plate recognition system [31]. They demonstrated how to identify and detect license plates using their developed YOLOv5 model, which was trained on a dataset of 300 images. The authors noted that even with a limited sample, the model could be trained effectively. Their method demonstrated rapid object detection and efficient high frame rate video processing. They were able to achieve an astounding 99.4% accuracy rate overall with their model. Table 2.5 presents a comparison of different automatic number plate recognition (ANPR) systems.

Table 2.5: Comparison of Various ANPR Systems

Method	Features Used	Pros	Cons	Classification Accuracy
Convolutional Neural Network (CNN)	Image-processing techniques Contours and Morphological operations (Edge detection and final contours), Deep learning techniques	Better feature extraction than other techniques, good results obtained even tested tilted	Two databases built and train to identifying	97.00%
YOLOv5	Using a custom transfer learned model for license plate detection, an LSTM-based OCR engine for recognition	Approach ideal for IoT devices that usually less memory and processing power, Novel memory and time-efficient	The ability to work under low visibility and night conditions, the lack of real- world performance measurement	87.2%
K-Nearest Neighbor (KNN) – by MATLAB	Principal Component Analysis (PCA) used to extract features from the images	Need only low specification of device: CPU: 1.80GHz Hard disk: 500GB RAM: 2GB Operating System: Microsoft Windows 8.1 Pro (or above)	Condition of the license plate and intensity will influence the results, required to obtain the license plate manually	92.86%
CT5L	Techniques of the continuity of vertical and horizontal black- and- white pixel values inside the plate, unique Hough	No need to apply deep learning, achieved the best performance among the conventional fixed	Mitigation for over detection	74.24%

	transform	threshold method, Otsu's method and conventional method of JavaANPR		
Depthwise Separable Convolution Networks	Novel segmentation-free framework Xception and Inception-ResNet v2, Segmentation-free LPR method	Able to achieve high accuracy even in an average of 70 FPS (highly efficient), can run in cheaper hardware	Entire running time and errors caused by incorrect tilting correction	99.60%
Convolution Neural Network (CNN)	Computer Vision Algorithm and Characters Segmentation Algorithm and trained by image database by Jetson TX2 NVIDIA	An improved method in Optical Character Recognition (OCR) based on DL techniques	Quite large amount of image training is needed	95.84%
YOLO v3 and CNN for OCR	Image Deblur, Pre- Processing, Number Plate Recognition (YOLO), Character Recognition (CNN)	Can deblur even the image given is blur, perform better than existing methods	Model is not testing with the video	YOLO v3 model (96%) and CNN OCR model (91.5%)
YOLO v5 and Easy-OCR	Open-CV for recognize images, Pre-Processing, Number Plate Detection (object localization and classification)	Speed for object detection and character recognition is improved and is suitable for real-time applications	A good GPU is required	95% for Easy- OCR

YOLO v5 and CNN for transfer learning	Frame extraction, annotation and augmentation, detection and recognition of number plate	Training effectively with limited dataset, perform well in high FPS (140FPS)	Extremely bright and dark regions is difficult to determine.	99.4%
---------------------------------------	--	--	--	-------

2.10 Summary

Based on the study projects that have already been looked at, there are a number of approaches and strategies that could be strengthened and made better for this one. First and foremost, the approach taken in creating the automatic number plate recognition system must be taken into account for this project. Although the YOLO approach is one prominent example of the gains made in the industry, it is vital to take into account CNN's historical preference for accuracy and feature extraction skills. Compared to previous methods, the ANPR system can achieve greater accuracy by utilizing YOLO, which can also improve speed, memory efficiency, and performance.

Additionally, accuracy is a crucial concern, particularly for the creation of deep learning-based systems. High accuracy rates and promising real-time capabilities were demonstrated by the YOLO-based ANPR systems. To sum up, the YOLO approach—more especially, YOLOv5—is a critical component of ANPR system development when considering alternative methods. As the next version of the YOLO approach, YOLOv7 offers considerable improvements in speed and accuracy over earlier YOLO models, as stated in Section 2.6.4. The Predator Helios 300 is an optimal choice for this project due to its compact design, cost-effectiveness, and energy-efficient features. This gaming laptop offers a balance between performance and portability, making it suitable for projects that require a powerful yet relatively small device. Additionally, its affordability makes it a practical choice for those working within budget constraints. The energy-efficient characteristics further enhance its suitability for projects that prioritize power consumption. Due to its general-purpose computing devices characteristics, it is highly suitable for real-time applications that require fast and efficient processing, such as automatic number plate recognition (ANPR).

Furthermore, the YOLOv7 model, specifically selected for Automatic Number Plate Recognition (ANPR), can be fine-tuned to operate with optimal efficiency on Predator Helios 300, guaranteeing precise and swift identification of license plates. The Predator Helios 300's portability allows it to be easily used in many settings, making it a useful option for applications such as vehicle surveillance and monitoring.

To summarize, utilizing Predator Helios 300 for the vehicle number plate recognition system guarantees a cost-efficient, energy-conserving, and highly versatile solution for real-time automatic number plate identification activities.



CHAPTER 3

METHODOLOGY

3.1 Introduction

The purpose of this chapter is to provide an overview of the project's step-by-step procedure, with the major emphasis being placed on the development of the Automatic Number Plate Recognition (ANPR) system that makes use of YOLOv7 on the Predator Helios 300. The purpose of this discussion is to offer a full knowledge of the required actions that must be taken prior to beginning the simulation. An in-depth analysis of the strategies and techniques used in the ANPR system's construction will be conducted. Furthermore, a comprehensive investigation of the particular measures that are essential for the project's effective execution will be carried out. In order to do this, it is necessary to go into the specifics of the ANPR development process, place an emphasis on the most important approaches, and handle any possible issues that may come up. Implementation of a project Gantt chart as a visual timetable will ensure that the project is completed in a timely manner and with proper organization. This timetable will serve as a guide for us as we go through each step, ensuring that we stick to the plan and complete all of the tasks within the time limit that has been established. Bringing to completion a Real-Time Automatic Number Plate Recognition System on the Predator Helios 300 utilizing YOLOv7 is our overall aim, and the Gantt chart will play a crucial role in monitoring milestones and maintaining project momentum. This will come in perfect alignment with our overarching purpose.

3.2 Project Flow

The flowchart in Figure 14 shows the successive processes to achieve the project's goals. The first step of the procedure is data selection, which makes use of a dataset of cars with license plates that Roboflow provided. Theos annotation tool is then used to annotate the dataset with car number plate labels. Then, three sets are created from the number plate detection dataset: a train set, a verified set, and a test set. After the data preparation phase is over, the process moves on to Google Colab for model training. A Predator Helios 300 and Logitech C270 720p Widescreen Video Webcam Camera are among the hardware components used for this project. Pytesseract is used to interpret the alphanumeric characters as soon as the trained model detects a number plate.

Next, a prepared dataset with the designated Predator Helios 300 hardware components is used to test the trained model. An examination of training performance is carried out following the collection of findings. This thorough process guarantees that the Automatic Number Plate Recognition (ANPR) system is effectively trained and tested. It also assures a smooth integration of the produced system with the designated Predator Helios 300 and related components for real-time deployment and analysis. Figure 3.1 illustrates the workflow of the automatic number plate recognition system project.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

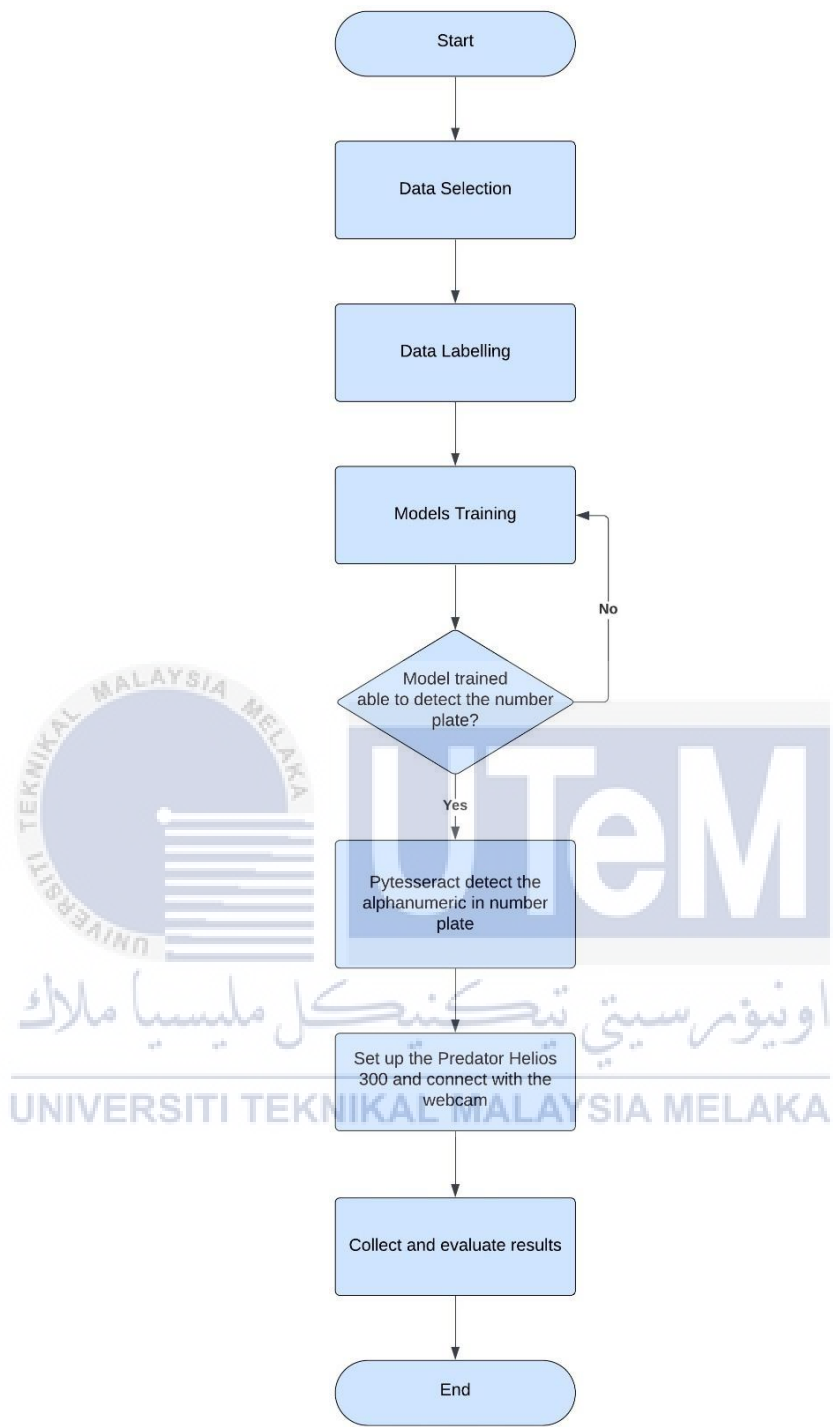


Figure 3.1: Project Flow

The system that uses a Predator Helios 300 with a camera module to take real-time pictures of cars is started, and this initiates the Automatic Number Plate Recognition (ANPR) procedure. After being taken, these pictures go through a process called plate detection, which uses a deep learning algorithm like YOLOv7 to find and identify the license plate of the car. As soon as the license plate is identified, the technology isolates the particular area that the number plate is located in from the surrounding context.

The extracted number plate region next goes through a crucial process called character segmentation, which separates the individual characters on the plate for additional examination. To convert the segmented characters into a readable format, a character recognition system is given the characters. OCR (optical character recognition) techniques are frequently used in this process. The system displays the recognized alphanumeric data that was taken from the license plate of the car to complete the ANPR process. With the help of an all-inclusive procedure, information from license plates can be automatically identified and retrieved, enabling uses like smart parking, access control, and effective traffic monitoring. The process for the automatic number plate recognition system is shown in Figure 3.2.

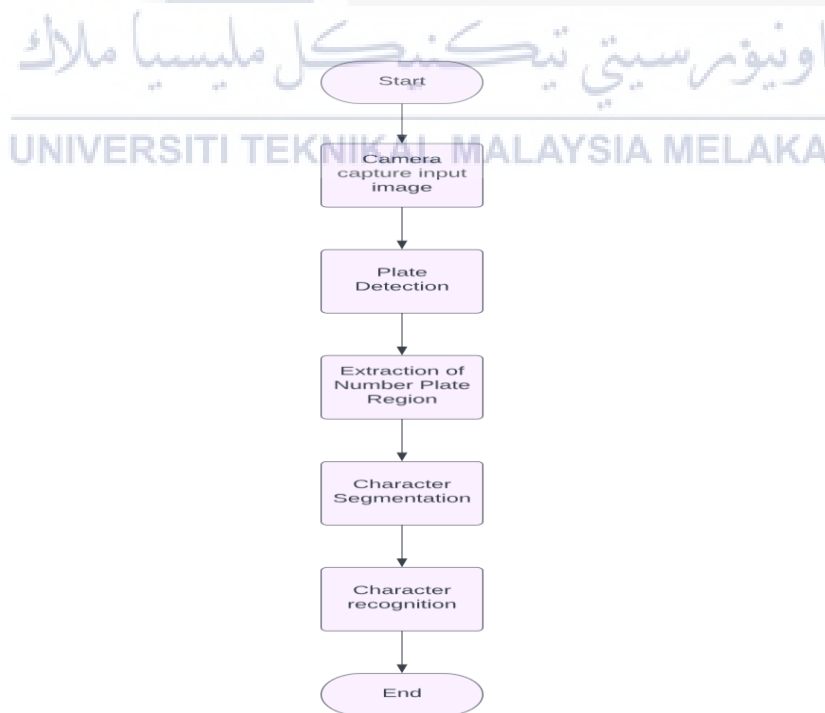


Figure 3.2: Automatic Number Plate Recognition System Flow

3.3 Data Selection

Data selection is the first step in this project. A set of vehicle images consisting of number plates provided by Theos is used. There are a total of 866 images in the dataset. The dataset is namely as license-plate. 69.7% (604 images) of the number plate dataset is for training purposes, while 20.4% (177 images) is for a valid set, and the rest, 9.8% (85 images), is for a test set. Figure 3.3 shows the percentage of the image for train, validate, and test, while Figure 3.4 shows an example of the vehicle dataset.

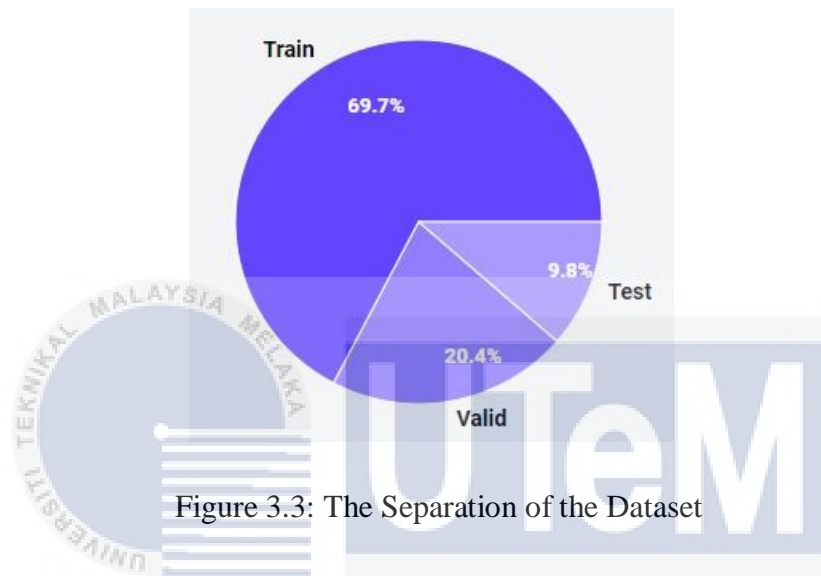


Figure 3.3: The Separation of the Dataset



Figure 3.4: Image of the Vehicle with Number Plate

3.4 Data Labelling

The next step after data collection is data annotation. Data annotation is a process that uses the Theos labelling tool to label where the number plate is in the image. All 866 images are uploaded to the Theos websites and labelled one by one, as shown in the figure below. There are total of 872 labels were obtained for all the images, and all the labels are named as number plates. Now the dataset is ready for training models using Theos AI.

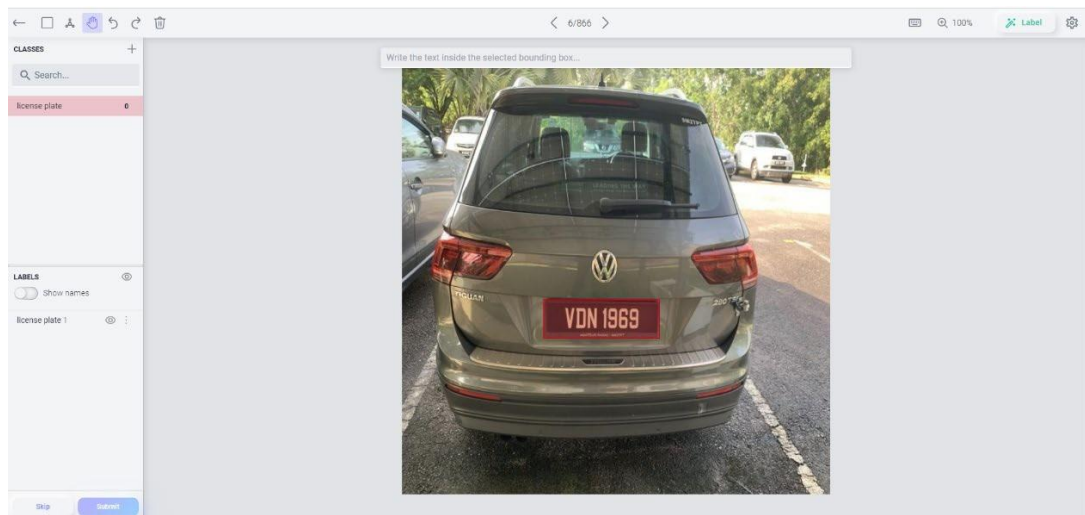


Figure 3.5: Theos Labelling Tool



Figure 3.6: Image with Number Plate Labelling

3.5 Convolution Neural Network (Deep Learning)

The CNN-based model that is being used in this research is trained by YOLOv7, which is one sort of CNN. A convolutional neural network (CNN) is a type of neural network that allows for the processing of inputs that have a grid-like design, such as images. Currently, CNN is mostly utilized for tasks such as the classification of photographs, the localization of objects, and the recognition of faces. CNN takes the data and processes it via a number of different layers of filters. A variety of characteristics inside an image can be identified by each layer. In neural networks, there are three different types of layers, which are referred to as the input layer, the hidden layer, and the layer of output. Figure 3.7 can be found under this heading. In the case of the input layer, this will result in the division of the total number of pixels in the data when the picture data is utilized for additional operations. Following the input layer comes the hidden layer, which receives the data that was generated as an input value. The term "hidden layer" refers to the layers that are responsible for producing an output by employing an activation function and taking in a collection of weighted inputs. Hidden layers are built of numerous layers. This means that the output of the hidden layers has arrived at the output layer at this point. The output layer is a completely linked layer that is responsible for flattening and delivering the input from the layer below it in order to divide it into the required number of classes according to the requirements.

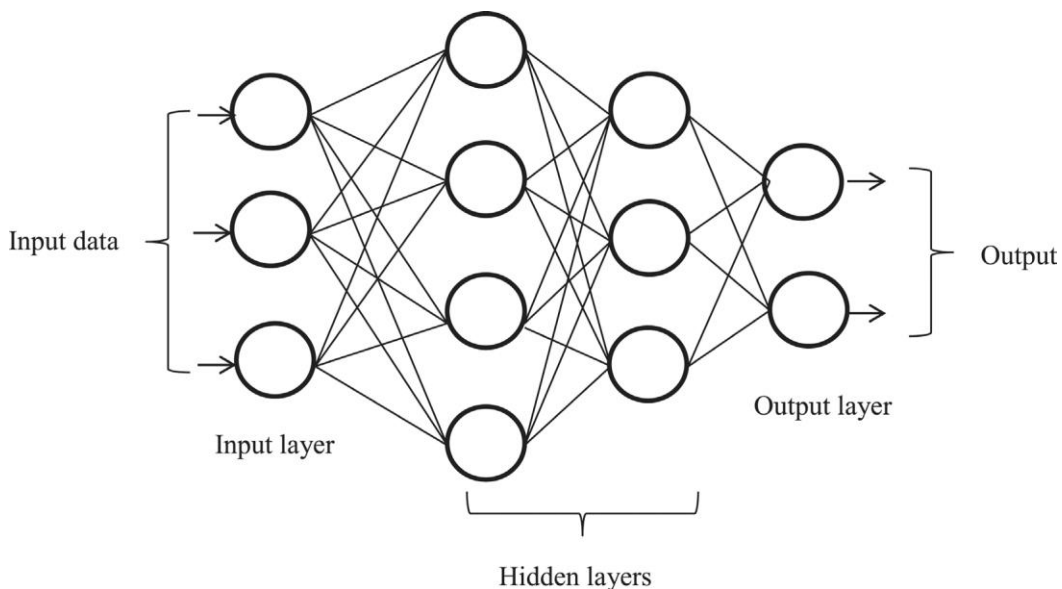


Figure 3.7: Layers of CNN

3.5.1 YOLOv7-tiny

A few versions of YOLOv7 have been released, and the algorithm used to train the model is YOLOv7 Tiny. A straightforward model that is tailored for edge GPUs is called YOLOv7-tiny. Prefix "tiny" computer vision models make machine learning (ML) easier to implement on mobile devices or dispersed edge servers and devices. Additionally, they are optimized for deep learning and edge AI applications. This strategy is essential for networked real-world computer vision applications. The edge-optimized YOLOv7-tiny varies from the other versions in that it employs leaky ReLU as the activation function as opposed to SiLU in the other models. In this research, YOLOv7-tiny is used because, in comparison to YOLOv7, it can be identified more quickly and with less resource consumption at the expense of accuracy, particularly when ANPR is used. Thus, YOLOv7-tiny was chosen.

3.5.2 YOLOv7-tiny Training

For training, validation, and testing purposes, the dataset has been prepared and partitioned into the proportions 7:2:1. The subsequent step involves the installation of the Theos Python and YOLOv7 small dependencies in Google Colab. Users are need to log in to both Google Colab and Theos AI. Following the completion of all of the setup, the model is now prepared to undergo training. For the purpose of ensuring that the training process is both smooth and effective, the epoch and batch size of the model train have been changed to a total of 300 epochs and 32 batches respectively. During the process of training the model, both the validated accuracy and the accuracy of the

training are monitored. In the event that both the accuracy and the results are not attained after a few epochs, the hyperparameter should be tuned in order to guarantee that the results are received at the conclusion. The model is then downloaded and stored after the training has been completed. When it comes to testing, the model is now ready to be used. Nevertheless, if the accuracy of the prediction is not satisfactory, the model ought to be retrained once more by either including new data or modifying the existing dataset. All of the processes are going to be displayed in Figure 3.8 down below.



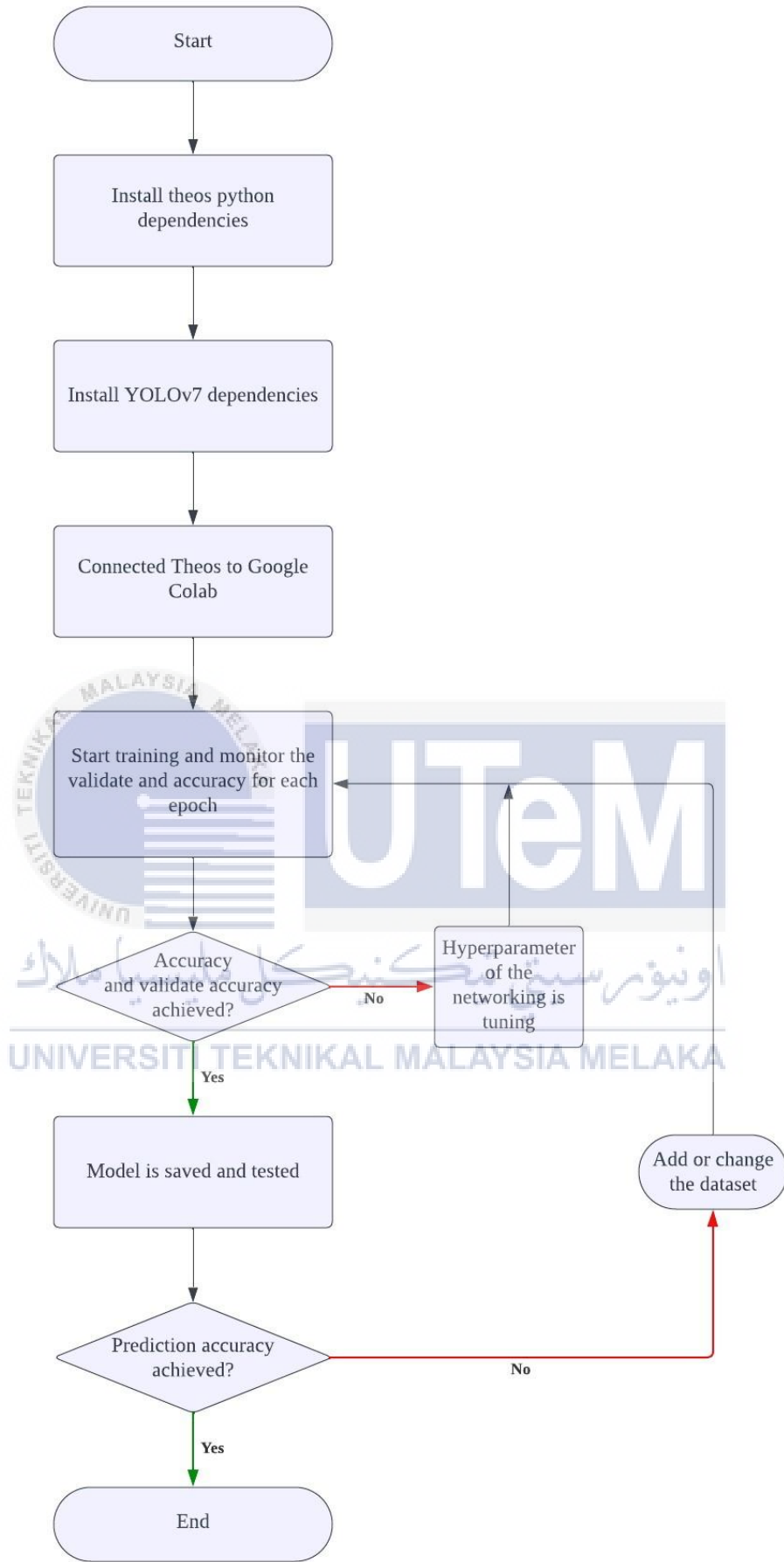


Figure 3.8: Workflow on YOLOv7 Training

3.5.3 Tuning of the Hyperparameter

There have been some adjustments made to the parameters while the model was being trained. In addition, the batch size has been reduced from 64 to 32 in order to guarantee that the mAP and AP of the model after the train are within acceptable limits is shown in Figure 3.9. Additionally, the epochs are set to 300, which indicates that the train and validate sessions will be repeated a total of 300 times. This is done to guarantee that the model will train with a greater level of accuracy and precision, resulting in improved number plate identification.

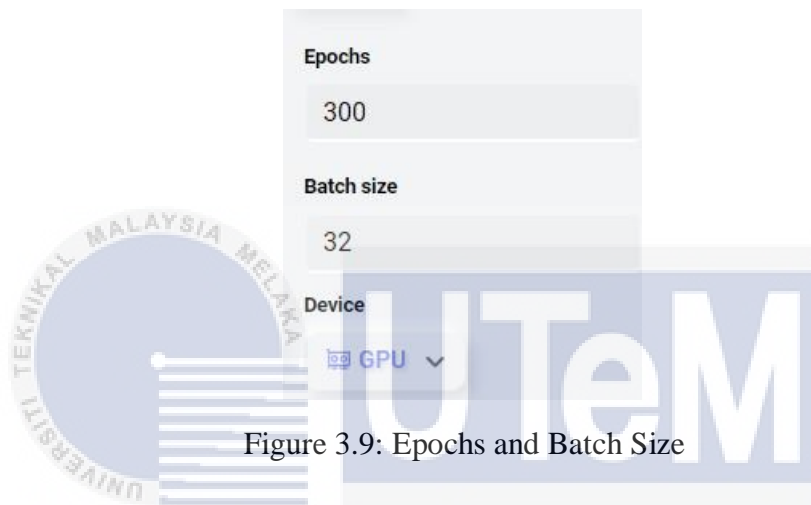


Figure 3.9: Epochs and Batch Size

3.6 Detection or Recognition of Number Plate

As soon as the input picture is identified, Preprocessing is the process of resizing and normalizing the input image to make it 640 by 640 by default, a size appropriate for the YOLOv7 model. The features from the preprocessed picture will then be extracted using CNN. Multiple convolutional and pooling layers make up CNN, which determines how to extract different levels of abstraction from the image. The transmission of the extracted features across this layer is necessary to move into the fully connected layers. In order to help anticipate the bounding box coordinates and class probability of the number plate, these layers will function as a classifier and regression. The next step is to divide the picture into a grid of cells using a technique called grid and cell division. A set of bounding boxes and class probabilities are anticipated for each cell's contents. The networks generate a set of bounding boxes and class probabilities, or likely number plates inside the picture, following grid and cell division. There will be predictions made for every cell. The bounding box predictions

will be improved by utilizing the Non-Max Suppression post-processing technique. After removing the overlapping boxes, the box with the best chance of each object will be selected. This reduces the frequency of duplicate or redundant forecasts. For every license plate in the picture, a set of predicted bounding boxes and class labels are generated using non-max suppression. These show the altered image with the license plate labeled and localized.

In general, a CNN is used to extract features from the input picture before fully connected layers are applied for regression and classification. The network predicts bounding boxes and class probabilities for every cell in a grid. After that, the predictions are improved using post-processing methods like non-max suppression to create the final, modified image that has the license plate on it.

3.7 Alphanumeric Character Recognition

An interface for utilizing the Tesseract OCR (Optical Character Recognition) engine is provided by the Python package Pytesseract. Pytesseract may be used to extract text from photos. After the number plate is localized by the YOLOv7, Pytesseract is responsible for extracting the alphanumeric from the plate in this project. Installing Tesseract OCR in Google Colab is the first step in the process. For improved image processing, the Python Imaging Library (PIL) or cv2 (OpenCV) is loaded together with the pytesseract modules.

The first step in the procedure is sending an API request to the Tesseract OCR engine from the solution. After that, an API request for text extraction is submitted together with the input image. In order to guarantee that the picture quality is as good as possible to produce reliable data extraction results, the image will now be pre-processed. This may involve scaling, converting to grayscale, or adding filters to increase the text visibility. Tesseract and OpenCV are now combined to improve picture quality prior to data extraction. Following processing by the Tesseract OCR engine using OpenCV and training data sets, the data is extracted. The number plate box will display the available alphanumeric characters when the text has been extracted from the input, and the API will respond when the output is prepared. The technique is fully depicted in the figure below.

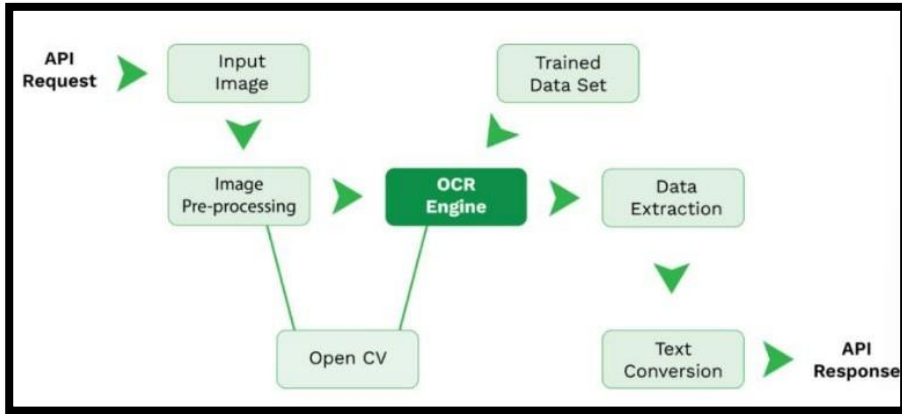


Figure 3.10: Tesseract OCR Process

3.8 Evaluation of Models and Results Collect

Metrics for Object Detection Evaluation Mean Average Precision (mAP), the most popular evaluation metric, is one of several metrics frequently used to assess the effectiveness of object identification models. The subsequent calculation is considered while calculating mAP.

Calculate the intersection over union (IoU) between the ground truth and forecasted bounding boxes. IoU is the proportion of the union area of the two boxes to their junction area. It calculates the amount of overlap between the expected and actual boxes as shown in Figure 3.11

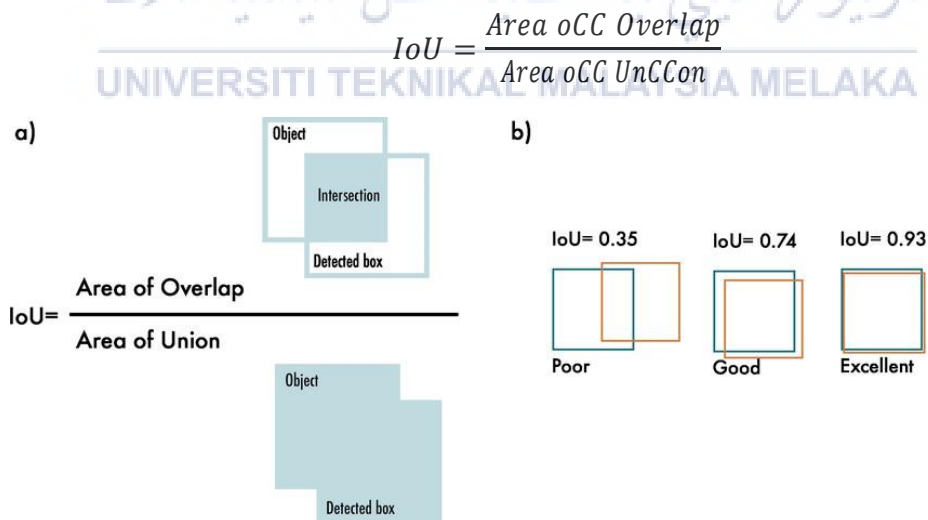


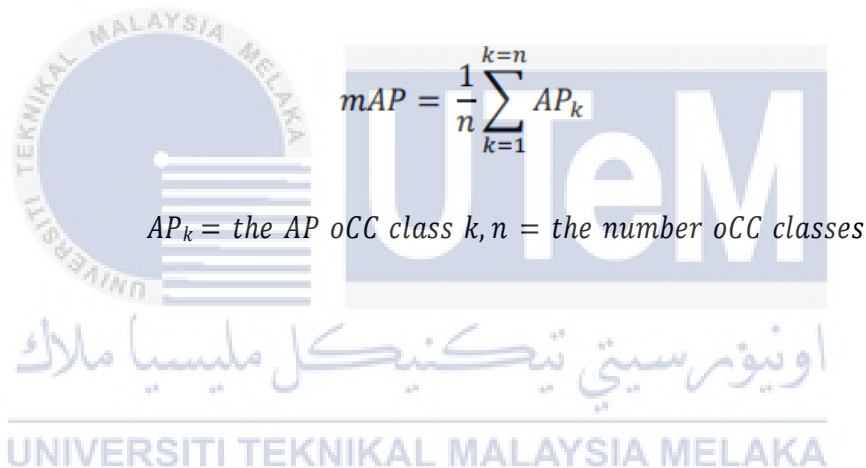
Figure 3.11: Diagram of IoU

Calculate the area under the precision-recall curve to determine the average precision for each class. AP stands for the typical accuracy for a given class. The equation for calculating AP involves computing the average precision for each class and then taking their mean.

$$AP = \sum_{k=0}^{k=n-1} [Recalls(k) - Recalls(k - 1)] * Precisions(k)$$

$Recalls(n) = 0, Precision(n) = 1$ where $n = \text{Number of thresholds}$

The precision and recall values are obtained from the precision-recall curve. Finally, mAP is computed by averaging the AP values across all classes. To get the mAP score, take the mean of the AP scores for all classes. The object detection model's total performance is measured by mAP.



$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k$

$AP_k = \text{the AP of class } k, n = \text{the number of classes}$

3.9 Components of the ANPR System

Parts	Quantity
Logitech C270 720p Widescreen Video Webcam Camera	1

Table 3.1: Shows The Parts and The Amount Needed for The Project

3.9.1 Logitech C270 720p Widescreen Video Webcam Camera



Figure 3.12: Logitech C270 720p Widescreen Video Webcam Camera

The Logitech C270 720p Widescreen Video Webcam Camera offers users a reliable solution for high-definition video calling and conferencing as shown in Figure 3.12. It features a 720p resolution, delivering crisp and clear video at up to 30 frames per second, ensuring smooth and detailed visuals during calls. Equipped with Logitech Right Sound technology, its built-in microphone enhances audio clarity by reducing background noise, making conversations more natural and focused. The webcam provides a 60-degree diagonal field of view, suitable for single-person use or small group settings, ensuring everyone remains visible without the need for constant adjustments. Designed for convenience, it connects effortlessly to computers via USB 2.0, compatible with a range of operating systems including Windows, macOS, Chrome OS, and Android. The webcam comes with a versatile universal clip that securely attaches to laptops or monitors, allowing for easy setup and positioning. Ideal for various applications such as video conferencing, online teaching, and live streaming, the Logitech C270 also includes features like automatic light correction to adapt to different lighting conditions, ensuring consistent image quality in any environment.

3.10 Integration with Predator Helios 300

The process begins by initializing the system, ensuring the laptop and webcam are properly connected and functional. The webcam captures real-time video feed, which is then processed by the YOLOv7 algorithm for area of interest (ROI) detection. The detected ROIs, representing potential license plates, undergo further analysis by the PyTesseract optical character recognition (OCR) engine for character extraction and recognition. Post-processing steps, such as non-maximum suppression, are applied to refine the results and eliminate overlapping candidate regions. The recognized license plate information is then stored or displayed as output. Throughout the entire process, the system continuously checks for real-time updates, creating a loop that allows for continuous ANPR functionality. This flow ensures that the Predator Helios 300, equipped with the necessary software and algorithms, effectively performs real-time license plate recognition using the connected webcam.

3.10.1 Software Configuration

In the software configuration phase, the primary objective is to establish and optimize the essential software components crucial for the Real-Time Automatic Number Plate Recognition (ANPR) system on the Predator Helios 300. This involves a meticulous focus on the integration of YOLOv7, Tesseract-OCR, Pytesseract, and the incorporation of PyCharm for streamlined development.

To begin, YOLOv7 and Tesseract-OCR are installed on the Predator Helios 300, facilitating efficient object detection and character recognition, respectively. Concurrently, Pytesseract is seamlessly integrated with Tesseract-OCR to enhance the extraction of alphanumeric characters from license plates. Moreover, the software configuration encompasses the installation of pertinent Python libraries, encompassing image processing, system communication, and algorithm integration. The inclusion of PyCharm as the integrated development environment (IDE) adds an additional layer of sophistication, providing a robust platform for coding, testing, and debugging throughout the software development process. This comprehensive approach ensures a well-optimized and cohesive software foundation for the Real-Time ANPR system, aligning with the specific hardware specifications and project goals.

3.11 Ethics and Safety of the Method

Standardization is essential in the field of machine learning and artificial intelligence. The worldwide International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) is a well-known worldwide standardization organization that develops and publishes standards for a range of industries and fields, including machine learning and artificial intelligence. Table 3.2 lists a few of the most important undertakings.



<p>ISO/IEC JTC 1/SC 42</p> 	<p>This is a division of the ISO/IEC Joint Technical Committee 1 (JTC 1), which is devoted to AI. AI-related standards, such as those for AI systems and their applications, are created by SC 42. They focus on language, reference designs, data, the reliability of AI, and ethical issues.</p>
<p>ISO/IEC 40500</p> 	<p>This specification, which focuses on web accessibility, is also known as the "Web Content Accessibility Guidelines (WCAG) 2.0." Despite not being a specific AI standard, it is important because to maintain inclusion, AI systems and applications should adhere to accessibility standards</p>

Table 3.2: ISO/IEC Relate to AI and ML

3.12 Gantt Chart

Table 3.3: Gantt Chart of Project Implementation

Activity	Week													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Purchase Hardware and Software	■													
Assembly planning Hardware and Software		■												
Python Coding Building			■	■										
Testing Python Coding					■	■	■	■						
Testing Model								■	■					
Chapter 3 : Methodology									■	■	■			
Chapter 4 : Result											■	■		
Chapter 5 : Conclusion													■	
Report Submission FYP 2 Seminar													■	■



3.13 Summary

In a nutshell, the project's primary objective is to create a Real-Time Automatic Number Plate Recognition (ANPR) system by utilizing YOLOv7 on the Predator Helios 300. The Predator Helios 300 and Logitech C270 720p Widescreen Video Webcam Camera are all included in the compilation of the hardware components. Data collection, annotation, CNN-based model training with YOLOv7-tiny, and character recognition with Pysesearct are all components of the project, which follows a methodical flow. The Gantt chart assures that the project will be finished on time, and it places an emphasis on the system's ethics and safety. When combined with YOLOv7 and Tesseract-OCR integration, the powerful hardware of the Predator Helios 300 serves as the foundation for effective automatic number plate recognition (ANPR). An emphasis is placed on PyCharm for development as part of the software configuration, which includes the installation of Python libraries and dependencies that are required. With regard to the Predator Helios 300, the project's overarching objective is to develop a real-time ANPR system that is both efficient and easily integrated.



CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

This section will address and summarize the model's simulation as well as the outcomes that were attained. This made it possible to research the creation of automatic systems for recognizing license plates. The performance of the models using the metrics mean average precision (mAP), average precision, and recall value will be the main topics of this chapter.

4.2 Model Train Accuracy

Additionally, Theos AI and Google Colab are utilized in order to train the model, and the YOLOv7 algorithm is utilized in this project for the purpose of network training. The Google Colab platform is a cloud-based platform that was developed by Google. It offers customers free access to sophisticated hardware accelerators, such as Graphical Processing Units (GPUs) and Tensor Processing Units (TPUs), in order to accelerate computations and make them appropriate for machine learning models. It is going to be described how effective the models are based on the mean average precision (mAP), the average precision (AP), and the recall value. There are a total of 866 images that are utilized in the model train, and these images are divided into three different sets respectively: Out of the total number of images, 69.7% (604 images) are allocated to the training set, 20.4% (177 images) are allocated to the valid set, and the remaining 9.8% (85 images) are allocated to the test set.

4.2.1 YOLOv7 Tiny Model

Following the completion of the training process, the results for the model's mAP, AP, and recall values can be acquired. The results are shown in the graph that can be found below. In the course of the training, a total of 300 epochs were utilized, and the batch size was 32. It was determined that the following values were acquired by analyzing the graphs for map0.5/epoch @val, precision/epoch @val, and recall/epoch @val: a mean average precision value of 0.994405, a precision value of 0.994168, and a recall value of 0.988764 which is shown in Figure 4.1,4.2 and 4.3.

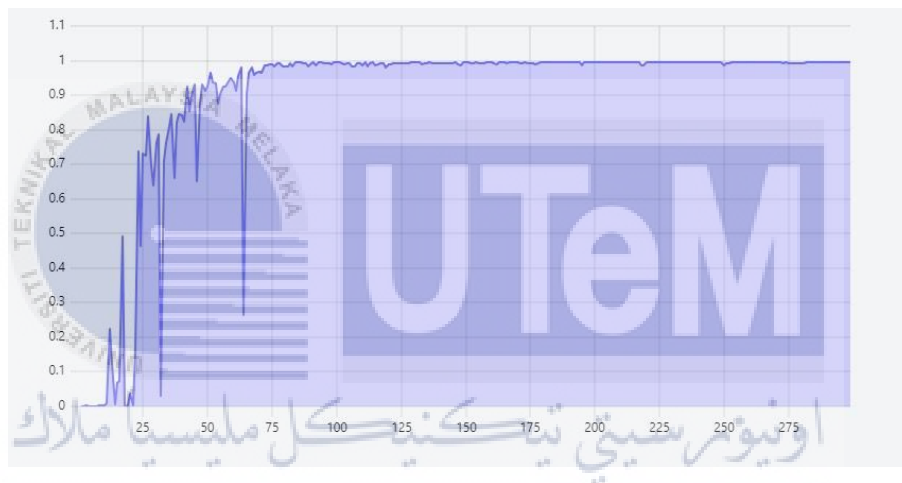


Figure 4.1: mAP of the model

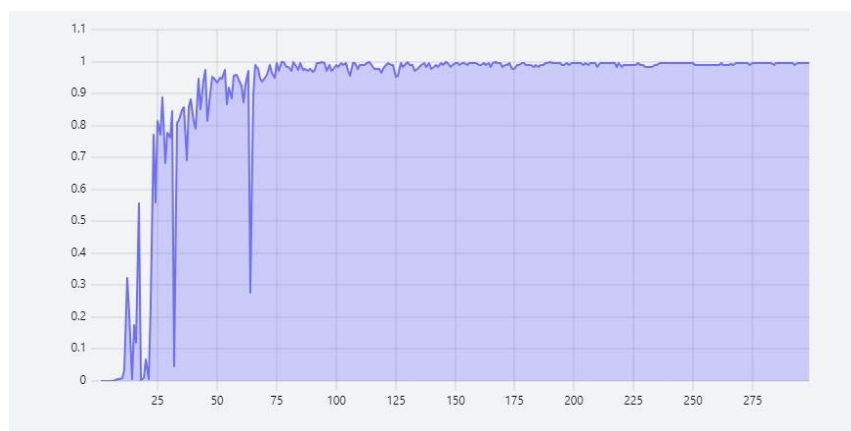


Figure 4.2: AP of the model

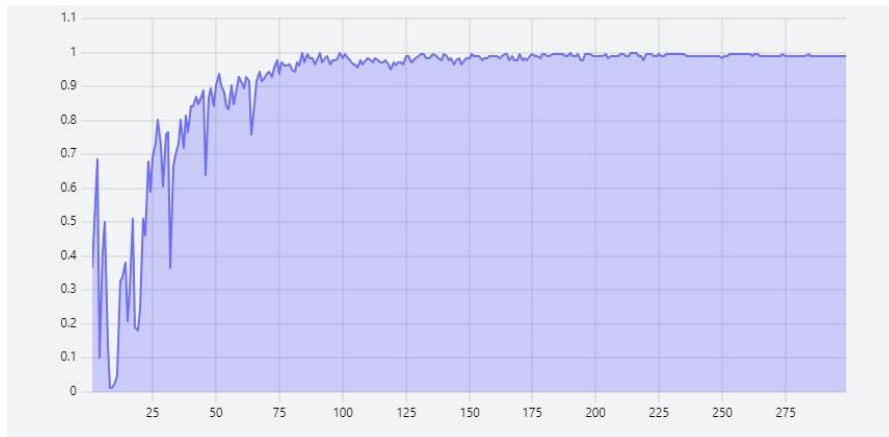


Figure 4.3: Recall of the model

Table 4.1: Result of YOLOv7-tiny Model

Algorithm	Mean Average Precision	Average Precision	Recall
YOLOv7-tiny	0.994405	0.994168	0.988764

4.2.2 Comparison of Poor Trained Model versus Good Trained Model

The previous model was trained using a dataset that consisted of only 737 photos. The dataset was partitioned into 70.4% for the training set, 20.6% for validation, and 9.0% for testing. The training process involved 300 epochs, with each epoch consisting of a batch size of 32. The model's evaluation metrics indicate a mean average precision (mAP) of 0.976464, a precision of 0.980044, and a recall of 0.967320. Nevertheless, the model lacks sufficient training, resulting in unsatisfactory outcomes for the image dataset. A model with a low mean Average Precision (mAP) will result in inadequate localization of the number plate, as demonstrated in Table 4.2. Upon comparing the metrics of the prior inadequately trained model with the latest model, it is clear that the mean average precision (mAP) and recall of the former were inferior. The table presented below displays the mAP, average precision (AP), and recall values for the previously trained model. Additionally, a comparative table was generated to assess the performance of both trained models on the image dataset. The current model has clearly demonstrated enhancements in the process of recognizing number plates. According to the comparison, the most recent model has shown improved ability in reliably identifying number plates.

Table 4.2: Results of Previous Poor Trained YOLOv7-tiny Model

Algorithm	Mean Average Precision (mAP)	Average Precision (AP)	Recall
YOLOv7-tiny	0.976464	0.980044	0.967320

Table 4.3: Image Set Tested on Poor and Good Trained Model

Poor Trained Model	Good Trained Model
 <p>BOB 9648 – 66% JLX 5752 – 92%</p>	 <p>BQB 9648 – 70% JLX 5752 – 96%</p>
 <p>1 – 70% JWD 7675 – 93%</p>	 <p>JWD 7675 – 92%</p>
 <p>TAX – 38% JSJ 8715 – 96%</p>	 <p>JSJ 8715 – 95%</p>

4.3 Real Time Automatic Number Plate Text Extraction System using PredatorHelios 300

In this section, we will look at the findings of an experiment employing a general-purpose computing device, specifically 3 different scenarios linked with an automatic license plate text extraction system. The primary goal of this investigation is to assess model performance by measuring pass and fail percentages in the context of license plate text extraction. These metrics provide a quantitative assessment of the model's accuracy and efficiency when applied to real-world scenarios. It is critical to do this thorough review to determine the model's capacity to work optimally in resource-constrained and realistic circumstances. The findings of this evaluation not only help to refine the model for greater accuracy, but they also provide critical information for the proper deployment of the license plate text extraction system in real-world operating circumstances. This includes applications for traffic control, security, and law enforcement, which require excellent performance within the limits of such circumstances.

4.3.1 Real Time Simulation Using Predator Helios 300

In a real-time situation, the automatic number plate recognition system will be tested using a dataset of 30 car license plates to identify and recognize characters on each plate. This test is aimed to evaluate the system's accuracy and performance in real-time license plate recognition using webcam. The goal is to assess how successfully the system identifies characters on the plates, providing useful information about its practical functionality. The findings will be noted in the table below, demonstrating the system's performance by identifying when it correctly detected characters (pass) and when it did not (fail). This systematic investigation quantifies the system's efficiency in real-world scenarios and provides a full grasp of its character recognition capabilities on license plates. The pass/fail results will help

to refine the system and optimize its accuracy for better performance in future applications such as traffic management, security, and law enforcement.

Table 4.4: License Plate Extraction (%)

Scenario	Number of Cars	Pass Rate (%)	Fail Rate (%)
Morning	10	90	10
Noon	10	70	30
Afternoon	10	80	20

4.3.2 Discussion on Pass and Fail Ratio of Real-Time Simulation Using Predator Helios 300

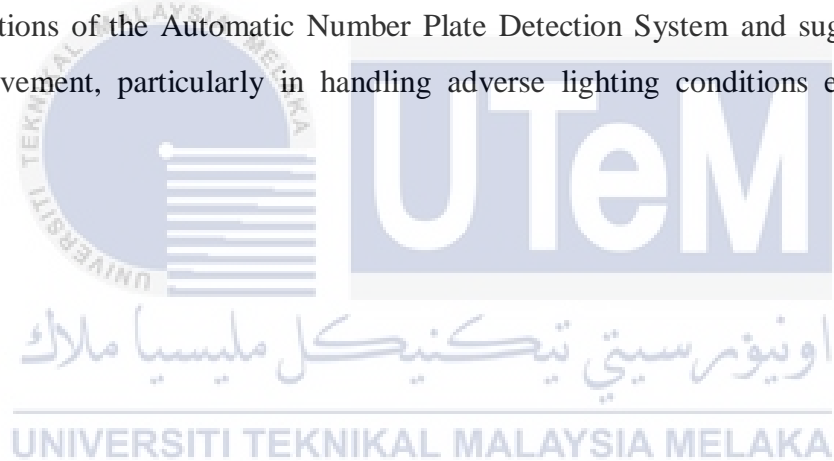
Section 4.3.2 will provide a detailed study of the pass and fail ratios observed during the real-time simulation of the automatic number plate recognition system using the Predator Helios 300. The findings of this experiment reveal both the limits imposed by the Predator Helios 300 and the system's performance in real-world circumstances, providing useful insights into these parameters. For each scenario, we will analyze data from 10 cars, making a total of 30 cars for the study. The performance of the system will be measured in terms of the percentage of successful detections.

During the morning, when the sunlight intensity is low to moderate, the camera is likely to capture clear images of license plates with minimal glare and reflections. We expect the system to perform optimally under these conditions, providing a high percentage of successful detections. This period offers relatively ideal lighting, which should allow the system to detect and extract license plate information with greater accuracy and consistency. At noon, the sunlight intensity reaches its peak, presenting significant challenges for the detection system. The high intensity can cause overexposure, glare, and reflections, which may obscure the details of the license plates. This scenario will test the system's robustness under adverse lighting conditions, and we anticipate a lower success rate compared to the morning scenario. The percentage of successful detections during this period will indicate the system's ability to handle high-glare environments.

In the afternoon, as the sunlight intensity decreases, the issues of glare and

reflections become less pronounced. However, the changing angle of the sun might still pose some challenges. This scenario evaluates the system's performance as the light conditions transition from high to moderate. We expect a moderate success rate, reflecting the system's adaptability to varying light intensities throughout the day. The overall performance may still be influenced by the suboptimal quality of the Logitech C270 webcam and the potential impact of the vehicle's speed on image clarity.

By calculating the percentage of successful detections for each set of 10 cars under different sunlight conditions, we will gain valuable insights into the optimal and suboptimal conditions for the system's performance. The morning is expected to show a high success rate due to favorable lighting; noon is anticipated to have a lower success rate due to high glare, and the afternoon is expected to exhibit a moderate success rate as lighting conditions improve. This study will highlight the strengths and limitations of the Automatic Number Plate Detection System and suggest areas for improvement, particularly in handling adverse lighting conditions encountered at noon.



4.4 Real Time Automatic Number Plate Localization Based on Accuracy

A Real-Time Automatic Number Plate Recognition (ANPR) System is an advanced technology designed to quickly and precisely recognize license plates from a live video stream. It focuses specifically on accuracy. The system's core component is an enhanced ANPR algorithm that can process video frames with high precision. One important parameter to assess the system's effectiveness in identifying license plates is its accuracy. The system is optimized for accurate real-time performance on a platform such as the Predator Helios 300, leveraging hardware features to ensure precise processing. The live video feed, usually recorded by a webcam, replicates actual situations, allowing the system to continuously monitor and verify its functionality.

Applications for this real-time ANPR system include security, parking management, law enforcement, and toll collecting, among other fields where accurate license plate identification is essential. The system provides insights into its efficacy through thorough data logging and analysis of accuracy for every frame. This allows for continuous enhancements and optimizations for improved real-time license plate identification. By focusing on accuracy, the ANPR system ensures reliable performance in diverse and dynamic environments, making it a critical tool for applications requiring high precision and dependability.

4.4.1 Real Time Simulation Using Predator Helios 300

The goal of real-time simulation utilizing the Predator Helios 300 for license plate detection is to evaluate how well the system can swiftly and precisely identify license plates from a camera stream under various conditions. The detection time, which quantifies how long it takes the Automatic Number Plate Recognition (ANPR) system to recognize and process license plates in real-time, is the main parameter of interest in this situation. Table below shows the detection time analysis based on real-time simulation.

Table 4.5: License Plate Localization

Time of Day	Sunlight Intensity	Data Source	Average Accuracy
Morning	Low to Moderate	Set A	75%
Noon	High	Set B	62%
Afternoon	Decreasing	Set C	70%

SET A (30 CARS)

$$\text{Average Accuracy} = \frac{1}{30} \left(\begin{aligned} &0.75 + 0.72 + 0.76 + 0.82 + 0.72 + 0.73 + \\ &0.70 + 0.70 + 0.73 + 0.76 + 0.70 + 0.74 + \\ &0.70 + 0.75 + 0.80 + 0.73 + 0.71 + 0.71 + \\ &0.79 + 0.78 + 0.72 + 0.76 + 0.82 + 0.83 + \\ &+ 0.72 + 0.77 + 0.80 + 0.76 + 0.83 + 0.79 \end{aligned} \right)$$

$$= \frac{22.60}{30} = 0.7533 \approx 0.75 \Rightarrow 75\%$$

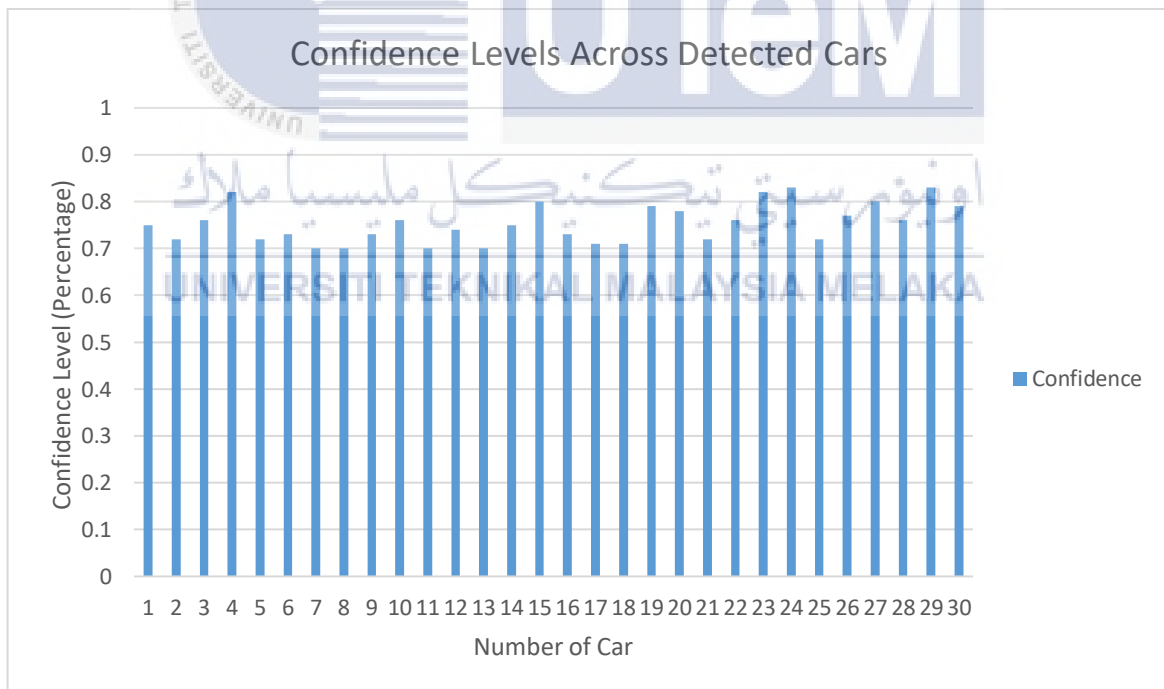


Figure 4.4: Confidence Levels Across Detected Cars (SET A)

During the morning, when the sunlight intensity is low to moderate, the system achieved an average accuracy of 75%. This suggests that the ANPR system performs optimally under such lighting conditions. The moderate sunlight in the morning provides balanced lighting without causing excessive glare or harsh shadows, which facilitates clearer image capture for the camera and more accurate detection and reading of license plates.

The graph titled "Confidence Levels Across Detected Cars" illustrates the confidence level (in percentage) of the detected license plates for 30 cars in the morning. The confidence levels are relatively consistent, ranging from 0.70 to 0.83, with an average confidence level of 0.75 (75%). This consistency indicates that the detection algorithm is robust and reliable under the given lighting conditions. The minor fluctuations in confidence levels can be attributed to slight variations in lighting and other environmental factors, but overall, the system demonstrates stable performance.

The high accuracy in the morning can be attributed to the optimal lighting conditions that reduce the chances of glare and reflections, which are more prevalent during other times of the day with higher sunlight intensity. The low to moderate sunlight in the morning creates fewer shadows and ensures that the license plates are well-lit and easily recognizable by the camera and detection algorithm.

SET B (30 CARS)

$$\text{Average Accuracy} = \frac{1}{30} \left(\begin{array}{l} 0.71 + 0.68 + 0.72 + 0.62 + 0.66 + 0.63 + \\ 0.66 + 0.56 + 0.55 + 0.67 + 0.67 + 0.67 + \\ 0.54 + 0.61 + 0.64 + 0.57 + 0.58 + 0.54 + \\ 0.61 + 0.64 + 0.56 + 0.50 + 0.51 + 0.67 + \\ 0.61 + 0.55 + 0.64 + 0.66 + 0.55 + 0.68 \end{array} \right)$$

$$= \frac{18.46}{30} = 0.6153 \cong 0.62 \Rightarrow 62\%$$

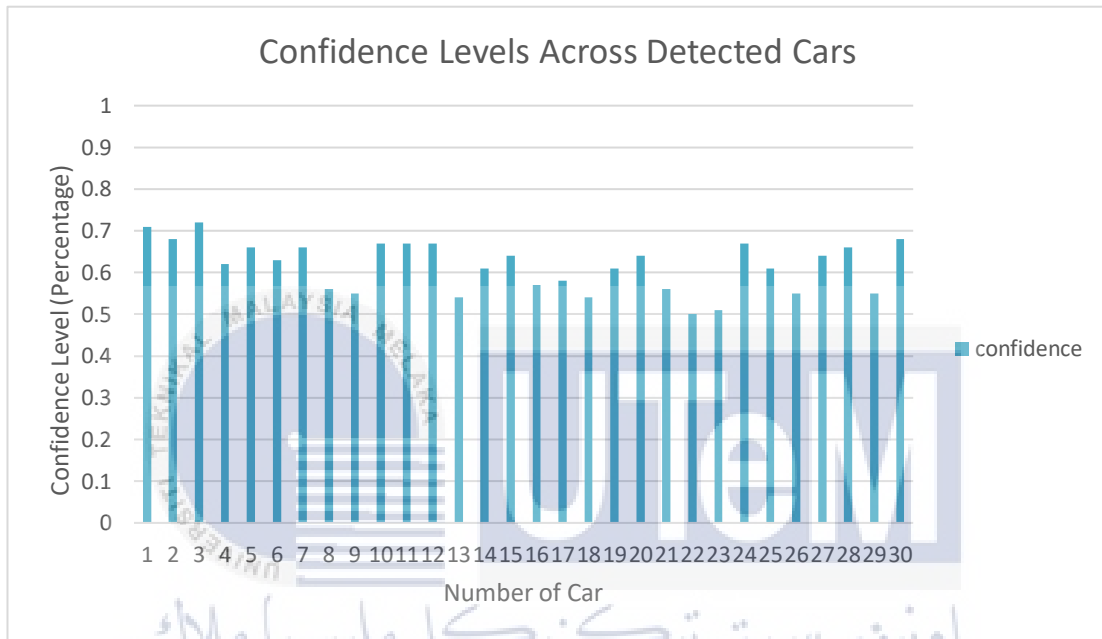


Figure 4.5: Confidence Levels Across Detected Cars (SET B)

The noon period, represented by Set B, highlights significant challenges for license plate detection under high sunlight intensity. During noon, the sunlight is at its peak, introducing severe glare and reflections on vehicle surfaces, including license plates. These environmental conditions lead to a noticeable drop in the system's accuracy, which averages 62%.

The data for Set B consists of confidence levels from 30 cars, with individual confidence scores ranging from 0.50 to 0.72. Despite some scores nearing the upper limit, the overall average confidence level remains lower due to the adverse effects of intense sunlight. High sunlight intensity can cause overexposure in the images

captured by the camera, leading to washed-out details and reduced contrast. This makes it difficult for the recognition algorithm to accurately detect and interpret the characters on the license plates.

The variability in confidence scores within Set B indicates that while some detections are relatively reliable, others are significantly compromised. This inconsistency can be attributed to factors such as the angle of sunlight, the position of the car relative to the camera, and the reflective properties of the license plate material. Cars with more reflective license plates or those positioned at angles that maximize glare are particularly challenging for the detection system.

To mitigate these challenges, potential solutions could include implementing polarizing filters on the camera to reduce glare, using advanced image processing techniques to enhance contrast and detail in overexposed areas, or employing machine learning models specifically trained to handle high-glare conditions. Additionally, real-time adjustments to camera settings based on the detected light intensity could help optimize image quality and improve detection accuracy.

Overall, the noon period's analysis underscores the critical impact of environmental lighting on the performance of the ANPR system. Addressing these challenges is essential for developing a robust and reliable license plate recognition system capable of maintaining high accuracy during peak sunlight hours

SET C (30 CARS)

$$\text{Average Accuracy} = \frac{1}{30} \left(\begin{array}{l} 0.71 + 0.61 + 0.58 + 0.60 + 0.56 + 0.75 + \\ 0.56 + 0.77 + 0.77 + 0.74 + 0.63 + 0.64 + \\ 0.84 + 0.76 + 0.61 + 0.64 + 0.76 + 0.74 + \\ 0.81 + 0.70 + 0.59 + 0.69 + 0.66 + 0.80 + \\ 0.77 + 0.68 + 0.75 + 0.77 + 0.73 + 0.67 \end{array} \right)$$

$$= \frac{20.89}{30} = 0.6963 \approx 0.70 \Rightarrow 70\%$$

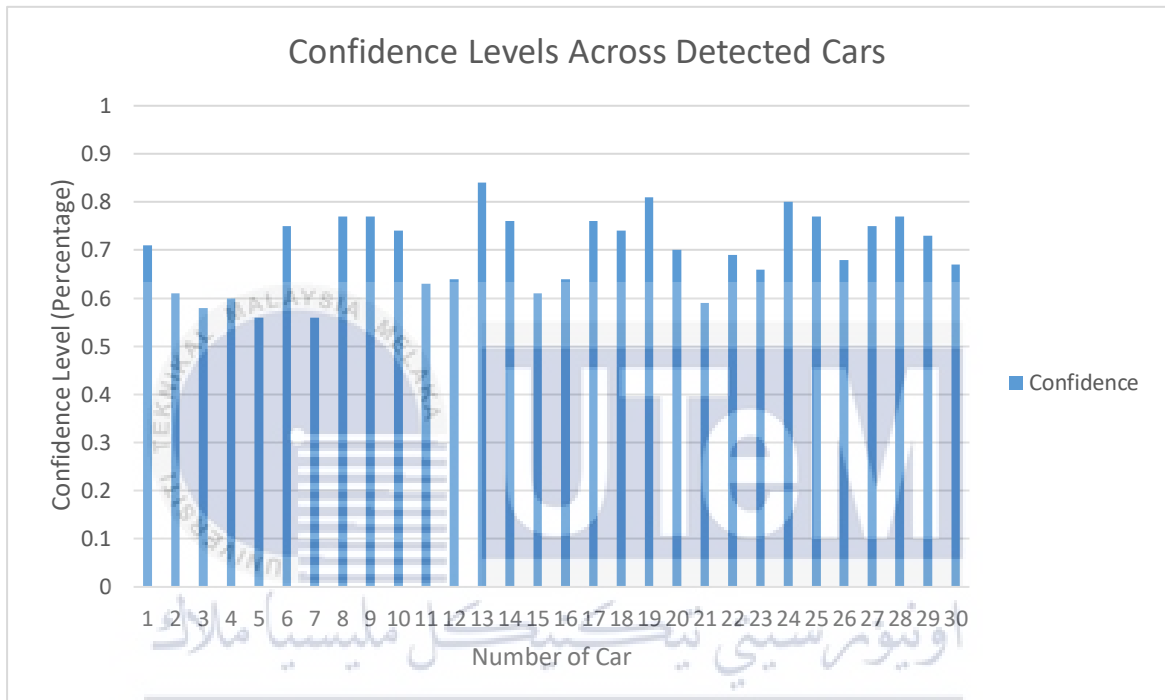


Figure 4.6: Confidence Levels Across Detected Cars (SET C)

The afternoon period, represented by Set C, showcases the system's performance under decreasing sunlight intensity. During this time, the sunlight is less intense compared to noon, which generally creates more favorable conditions for license plate detection. However, the presence of moderate glare and shadows can still pose challenges. The data for Set C includes confidence levels from 30 detected cars, with individual scores ranging from 0.56 to 0.84. The average confidence level, calculated at approximately 0.70 or 70%, aligns with the reported average accuracy for this period.

The variability in the confidence levels indicates that while some detections are reliable, others are affected by factors such as the angle of sunlight, car position, and the reflective properties of the license plates. Vehicles with less reflective plates or those positioned to minimize glare tend to have higher confidence scores. To mitigate these challenges, potential solutions include using polarizing filters on the cameras to reduce glare, employing advanced image processing techniques to enhance contrast and detail, and training machine learning models specifically for moderate glare and shadow conditions. Additionally, real-time adjustments to camera settings based on light intensity could help optimize image quality and improve detection accuracy.

Overall, the afternoon period's analysis underscores the impact of environmental lighting on the ANPR system's performance. Despite the relatively high average accuracy of 70%, addressing the issues of glare and shadows through targeted solutions is essential for developing a robust and reliable license plate recognition system capable of maintaining high accuracy throughout different times of the day.

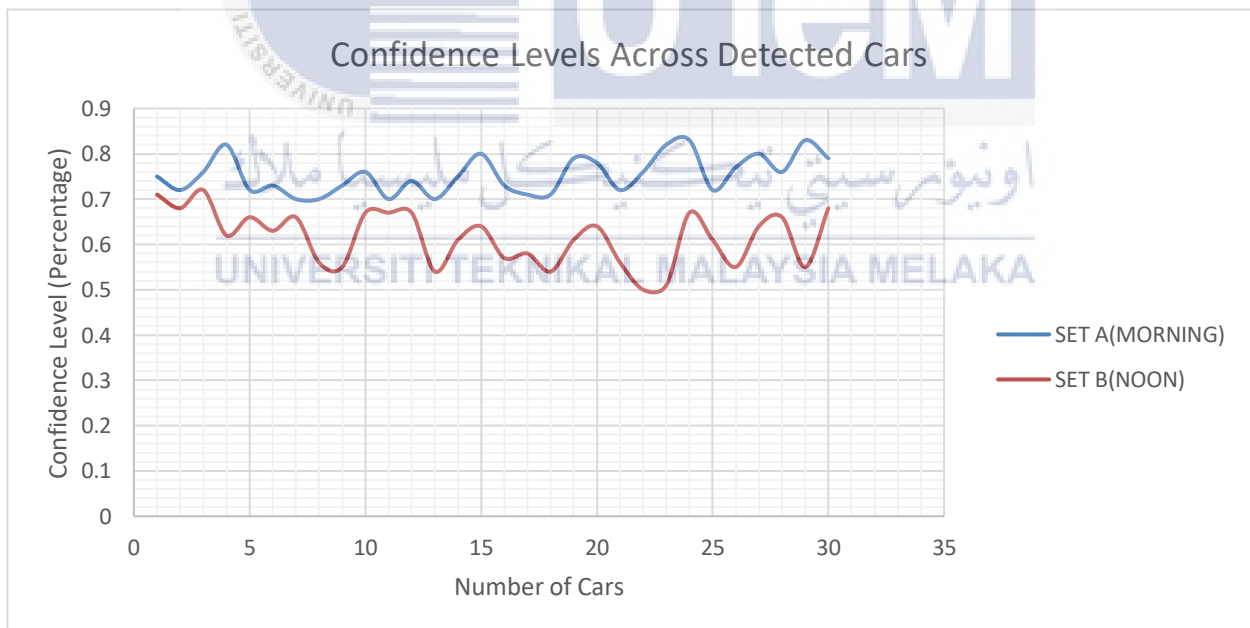


Figure 4.7: Graph Morning vs Noon

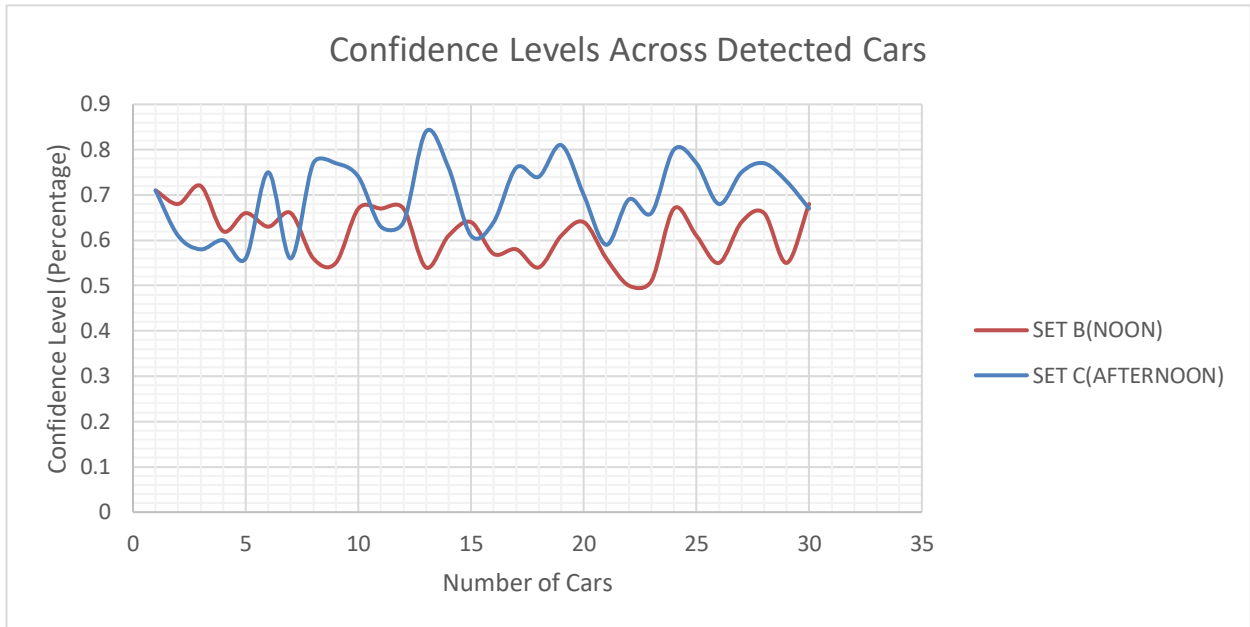


Figure 4.8: Graph Noon vs Afternoon

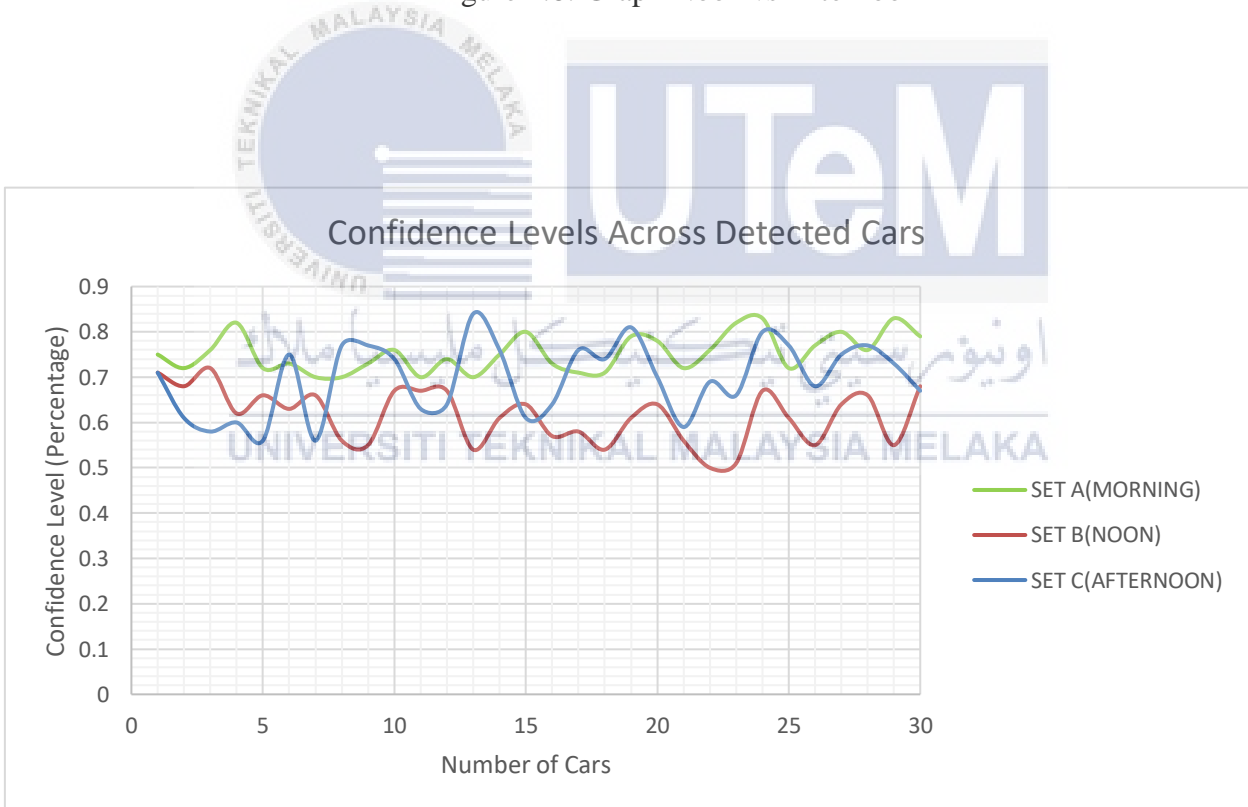


Figure 4.9: Graph Morning vs Noon vs Afternoon

4.4.2 Discussion Based on The Accuracy

The accuracy of an automatic number plate recognition (ANPR) system can be significantly influenced by the varying levels of sunlight intensity during different times of the day—morning, noon, and afternoon. In the morning, the sunlight intensity is low to moderate, with the sun low on the horizon causing longer shadows and softer light. These conditions can reduce glare and reflections, making it easier for the system to capture clear images of number plates. However, longer shadows may obscure parts of the number plate, affecting accuracy depending on the vehicle's orientation relative to the sunlight. Consequently, the accuracy tends to be moderate to high in the morning due to these balanced lighting conditions that minimize glare and harsh shadows.

At noon, the sun is at its highest point, providing the most direct and intense light with minimal shadows. While the high intensity and direct light ensure that the number plate is well-illuminated, reducing issues related to poor lighting, the intensity can also cause significant glare and reflections, potentially obscuring the characters and leading to variable accuracy. The accuracy during this time often ranges from moderate to low because the intense sunlight can introduce challenges with excessive glare. The angle of the camera and potential disruptions from nearby fans affecting the webcam's stability further complicate the localization process. The error analysis for license plate localization also indicates that slow Wi-Fi can cause delays in real-time processing, impacting the synchronization between the captured frames and the detection algorithm. Additionally, license plates that do not adhere to JPJ specifications may present unique challenges in localization.

In the afternoon, the sunlight intensity is decreasing as the sun moves lower in the sky, leading to less harsh light and shorter shadows compared to noon. These conditions generally provide a more stable environment for number plate detection, with reduced glare and reflections compared to noon. However, as the sunlight continues to decrease, shadows can become longer again, which might obscure parts of the number plate depending on the vehicle's orientation and the angle of the sunlight. The accuracy in the afternoon tends to be high to moderate, as the softer light reduces glare but longer shadows might still pose challenges.

Based on this analysis, the morning period might offer better accuracy for number plate detection due to its balanced lighting conditions that minimize glare and harsh shadows, compared to noon and afternoon. Noon, despite having the highest sunlight intensity, can introduce challenges with excessive glare, while the afternoon provides a more stable environment with decreasing sunlight but still has some potential issues with shadows. Therefore, optimizing ANPR systems to handle varying lighting conditions is crucial, incorporating technologies like anti-glare measures for noon and strategies to manage shadows for both morning and afternoon periods to enhance overall performance throughout the day. Understanding these variations helps in optimizing ANPR systems to handle different lighting conditions better, thereby enhancing overall performance throughout the day.

The Predator Helios 300 with a webcam is used to analyze the ANPR system's performance primarily in terms of accuracy. Several factors influence the accuracy, including plate angle, and distance of the webcam to the car. Examining accuracy patterns allows one to identify differences based on these variables. Plate angle is a crucial factor, as the system's ability to accurately detect and recognize number plates can vary significantly depending on whether the plate is viewed from the front, back, or side. Plates that are angled or not perpendicular to the camera can cause distortion or partial occlusion, leading to reduced accuracy. Additionally, the slow Wi-Fi connection may cause latency issues, and the presence of fans or other nearby disturbances can affect the stability of the webcam setup. Addressing these errors and their underlying causes will enhance the performance and reliability of the automatic number plate recognition system in real-time using a webcam.

Distance between the webcam and the car also plays a significant role in accuracy. At closer proximities, the system generally achieves higher accuracy due to clearer and more detailed images. However, as the distance increases, the clarity diminishes, and the system may struggle to accurately detect and read the plates due to reduced image resolution and increased chances of obstructions. Each experiment's consistency and dependability are assessed to identify any instances of inconsistent performance and possible areas for improvement. This comprehensive analysis of accuracy under various conditions provides insights into the ANPR system's strengths and limitations, offering directions for enhancements to improve real-world

applicability in dynamic contexts. To enhance the accuracy, it is recommended to improve the algorithm's handling of varying lighting conditions, particularly intense sunlight. Implementing image preprocessing techniques to mitigate glare and reflections could be beneficial.

4.5 Number Plate Localization and Extraction based on Confusion Matrix

The confusion matrix for the number plate recognition system includes several key components that help evaluate its performance. True Positives (TP) represent the number of times the model correctly predicted the positive class, which in this context refers to correctly identifying a number plate. False Positives (FP) measure the number of times the model incorrectly predicted the positive class, meaning it identified a number plate but included extra characters. False Negatives (FN) count the number of times the model incorrectly predicted the negative class, which refers to failing to detect a number plate that was actually present. True Negatives (TN) refer to the number of times the model correctly predicted the negative class, which means correctly identifying instances where no number plate was present. In this analysis, the performance of the number plate recognition system is evaluated using a single car. The system was tested five times to detect the number plate of this car.

4.5.1 Real Time Simulation using Predator Helios 300

Plate Number	Confidence Level		
RN 4923	0.816038	TP	
RN 4923	0.852513	TP	
RN 4923	0.861783	TP	Morning
RN 4923	0.864049	TP	
RN 4923	0.853162	TP	

Figure 4.10: Morning Performance using Confusion Matrix

Confusion Matrix Analysis:

True Positive (TP): 5

False Positive (FP): 0

False Negative (FN): 0

True Negative (TN): 0

Precision:

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{5}{5+0} = 1.0$$

Recall

$$\text{Recall} = \frac{TP}{TP+FN} = \frac{5}{5+0} = 1.0$$

F1 Score

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \times \frac{1.0 \times 1.0}{1.0 + 1.0} = 1.0$$

Accuracy

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{5}{5+0} = 1.0 = 100\%$$

Table 4.6: Confusion Matrix based on Morning

		TRUE CLASS	
		Positive	Negative
PREDICTED CLASS	Positive	5	0
	Negative	0	0

In this analysis, the performance of the number plate recognition system is evaluated and conducted in the morning using a single car. The system was tested five times to detect the number plate of this car. The number plate analysed is RN 4923. The number plate recognition system demonstrates perfect performance in this morning scenario based on the provided data. All identified plates are true positives with no false positives or false negatives, indicating 100% accuracy, precision, recall, and F1 score.

MDF 2095	0.867636		TP	
MDF 2095 -	0.850682		FP	
MDF 2095	0.871337		TP	Noon
MDF 2095	0.870932		TP	
_MDF 2095	0.851969		FP	

Figure 4.11: Noon Performance using Confusion Matrix

Confusion Matrix Analysis:

True Positive (TP): 3

False Positive (FP): 2

False Negative (FN): 0

True Negative (TN): 0

Precision:

$$Precision = \frac{TP}{TP+FP} = \frac{3}{3+2} = 0.6$$

Recall

$$Recall = \frac{TP}{TP+FN} = \frac{3}{3+0} = 1.0$$

F1 Score

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.6 \times 1.0}{0.6 + 1.0} = 0.75$$

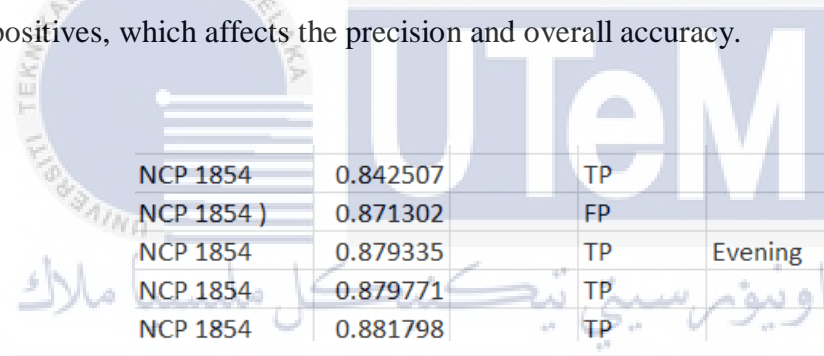
Accuracy

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{3}{5} = 0.6 = 60\%$$

Table 4.7: Confusion Matrix based on Noon

		TRUE CLASS	
		Positive	Negative
PREDICTED CLASS	Positive	3	2
	Negative	0	0

In this analysis, the performance of the number plate recognition system is evaluated and conducted in the noon using a single car. The system was tested five times to detect the number plate of this car. The number plate analysed is MDF 2095. The system shows high recall, correctly identifying all true positives, but with some false positives, which affects the precision and overall accuracy.



NCP 1854	0.842507	TP	
NCP 1854)	0.871302	FP	
NCP 1854	0.879335	TP	Evening
NCP 1854	0.879771	TP	
NCP 1854	0.881798	TP	

Figure 4.12: Afternoon Performance using Confusion Matrix

Confusion Matrix Analysis:

True Positive (TP): 4

False Positive (FP): 1

False Negative (FN): 0

True Negative (TN): 0

Precision:

$$Precision = \frac{TP}{TP+FP} = \frac{4}{4+1} = 0.8$$

Recall

$$Recall = \frac{TP}{TP+FN} = \frac{4}{4+0} = 1.0$$

F1 Score

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.8 \times 1.0}{0.8 + 1.0} = 0.88$$

Accuracy

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{4}{5} = 0.8 = 80\%$$

Table 4.8: Confusion Matrix based on Afternoon

		TRUE CLASS	
		Positive	Negative
PREDICTED CLASS	Positive	4	1
	Negative	0	0

In this analysis, the performance of the number plate recognition system is evaluated and conducted in the afternoon using a single car. The system was tested five times to detect the number plate of this car. The number plate analysed is NCP 1854. These metrics indicate that the system has high precision and perfect recall for this test scenario, but the presence of false positives affects the overall performance slightly, as reflected in the F1 score.

4.5.2 Discussion Based on the Confusion Matrix

For the analysis, the number plate recognition system was tested on a vehicle detected 5 times in the morning, noon, and afternoon. This gives us a total of 15 detection instances across different lighting conditions and environmental factors. In the morning, all five detections resulted in true positives (TP) which is 100%, indicating that the system performed exceptionally well under morning conditions. This high accuracy can be attributed to consistent and optimal lighting conditions in the morning, which provide clear visibility of number plates. Additionally, lower traffic volumes in the morning may reduce occlusions and distractions, contributing to the system's high performance. Moreover, the system might be particularly well-calibrated for morning conditions, leading to enhanced accuracy.

During the noon scenario, the system achieved three true positives (TP) which is 60% and two false positives out of five detections, suggesting some challenges faced under noon conditions. Harsh lighting conditions at noon, characterized by bright and direct sunlight, can cause glare and shadows that obscure number plates or create reflections, confusing the recognition system. Increased traffic during midday can introduce more occlusions and dynamic changes in the environment, complicating the detection process. High temperatures at noon may also cause heat haze or distortions, affecting the clarity of the number plates and contributing to the false positives.

In the afternoon, the system achieved four true positives (TP) which is 80% and one false positive out of five detections, indicating relatively high performance with minor issues. Afternoon light can vary significantly as the sun sets, creating transitional lighting conditions that can sometimes confuse the system. The introduction of artificial lighting, such as street lights and vehicle headlights, can either help or hinder the recognition process, depending on their placement and intensity. Additionally, if the system operates continuously throughout the day, there might be performance degradation due to factors like sensor overheating or software drift.

In summary, the analysis highlights the varying performance of the number plate recognition system at different times of the day. The system performed optimally in the morning with all detections being true positives due to favourable lighting and environmental conditions. Performance dropped slightly at noon due to harsh lighting and increased traffic, leading to some false positives. In the afternoon, the system maintained high performance with minor issues, likely due to varying lighting conditions and the introduction of artificial lights. Understanding these patterns helps in optimizing the system for different conditions by adjusting parameters or incorporating adaptive algorithms to handle environmental variations more effectively.



CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In general, the development of the real-time automatic number plate recognition (ANPR) system on the Predator Helios 300 by utilizing YOLOv7 has been a project that has been both comprehensive and enjoyable. Implementing automated recognition of license plates was the primary objective, with the secondary goal being to expedite and improve the processes involved in vehicle identification.

The project started off with a series of rigorous processes, which included the collecting of datasets, annotation, and the utilization of a YOLOv7-based model for license plate detection that was carried out in an effective manner. The incorporation of Tesseract-based optical character recognition (OCR) considerably enhanced the capabilities of the system by enabling the precise extraction of alphabetic and numeric characters from license plates. The configuration of the software, which included the use of crucial Python libraries and tools such as PyCharm, led to the creation of a system that was both well optimized and integrated. The Predator Helios 300 and Logitech C270 720p Widescreen Video Webcam Camera were the constituents of the hardware that laid the groundwork for the implementation of real-time automatic number plate recognition (ANPR). A Gantt chart was used to illustrate the timetable of the project, which ensured that all milestones were completed on time. Additionally, the timeline ensured that ethical considerations and safety were always a primary priority throughout the development process.

In addition, the study recognized the importance of light interference in the identification of number plates. Changes in illumination can provide difficulties for ANPR systems, and the research emphasized the importance of addressing and minimizing such disruptions to ensure precise outcomes. Ultimately, the utilization of the YOLOv7-tiny model demonstrated its effectiveness in accurately identifying and distinguishing number plates. The assessment of the trained model utilizing mAP, AP, and recall metrics demonstrated its exceptional performance. ANPR systems utilizing

deep learning have significant potential for smart city applications, as they can improve traffic management, security, and general efficiency through additional developments and optimizations. The key of this project is that it was successful in implementing a real-time automatic number plate recognition system that provides efficiency and accuracy. Through the integration of both hardware and software components, as well as the exploitation of YOLOv7 and Tesseract OCR, the system that was developed is positioned to be a promising solution for improving vehicle identification and monitoring in intelligent transportation systems. Through the completion of this initiative, a foundation is laid for future developments and improvements in the field of automated license plate recognition and traffic management.

5.2 Future Works

The ANPR system based on deep learning has various potential areas for extension and enhancement in future work. First, expanding the dataset size could help the model perform better further; adding more varied and sizable datasets to the training set would be advantageous. More diverse license plates, fonts, and backgrounds will help the model become more versatile and capable of handling a greater range of situations. Collecting and annotating more datasets can assist boost the accuracy and robustness of the ANPR system. Future research may also focus on handling light and noise interference. It would be beneficial to look at cutting-edge methods for reducing the effects of light and noise interference on the ANPR system. Pre-processing techniques like picture enhancement, noise reduction algorithms, or adaptive illumination correction techniques may be used in order to increase the precision and dependability of number plate recognition in difficult light environments.

REFERENCES

- [1] G. Anbarjafari, "1. introduction to image processing," Sisu@UT. [Online]. Available: <https://sisu.ut.ee/imageprocessing/book/1#:~:text=Image%20processing%20is%20a%20method,features%20associated%20with%20that%20image.>
- [2] P. Barhate, "An understanding of vehicle number plate detection mechanism using computer vision," Mobisoft Infotech, 29-Nov-2022. [Online]. Available: <https://mobisoftinfotech.com/resources/blog/vehicle-number-plate-detection-mechanism-using-computer-vision/>.
- [3] Adithya M., Sumitha B.S, Rahul K., Nitish Kumar P., and Peamod G. H., Automatic Number Plate Recognition Using Contours and Convolution Neural Networks, vol. 7, no. 4, pp. 9–13, 2021.
- [4] R. Shashidhar, A. S. Manjunath, R. Santhosh Kumar, M. Roopa, and S. B. Puneeth, "Vehicle number plate detection and recognition using yolo- V3 and OCR Method," 2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNBC), 2021. doi:10.1109/icmnbw52512.2021.9688407
- [5] "What is ...," Amazon, 1978. [Online]. Available: <https://aws.amazon.com/whatis/ocr/#:~:text=The%20two%20main%20types%20of,patern%20matching%20and%20feature%20extraction.>
- [6] J. Wu, "Complexity and accuracy analysis of common artificial neural networks on pedestrian detection," MATEC Web of Conferences, vol. 232, p. 01003, 2018. doi:10.1051/mateconf/201823201003
- [7] Kukil Sovit Rath, Kukil, and S. Rath, "Yolov7 paper explanation: Object detection and yolov7 pose," LearnOpenCV, <https://learnopencv.com/yolov7-object-detection-paper-explanation-and-inference/>
- [8] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," 2017 IEEE International Conference on Computer Vision (ICCV), 2017. doi:10.1109/iccv.2017.324
- [9] Y. Wang, B. Fu, L. Fu, and C. Xia, "In Situ Sea Cucumber Detection across Multiple Underwater Scenes Based on Convolutional Neural Networks and Image Enhancements," *Sensors*, vol. 23, no. 4, 2023, doi: 10.3390/s23042037.
- [10] Y. Qiu, Y. Lu, Y. Wang, and H. Jiang, "IDOD-YOLOV7: Image-Dehazing YOLOV7 for Object Detection in Low-Light Foggy Traffic Environments," *Sensors*, vol. 23, no. 3, 2023, doi: 10.3390/s23031347.
- [11] C. L. Narayana and K. V. Ramana, "An Efficient Real-Time Weed Detection Technique using YOLOv7," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 2, 2023, doi: 10.14569/IJACSA.2023.0140265.

- [12] Y. Zhang, Y. Sun, Z. Wang, and Y. Jiang, "YOLOv7-RAR for Urban Vehicle Detection," *Sensors*, vol. 23, no. 4, 2023, doi: 10.3390/s23041801.
- [13] A. Elhanashi, S. Saponara, and Q. Zheng, "An automated AI and video measurement techniques for monitoring social distancing, mask detection, and facial temperature screening for COVID-19," 2023. doi: 10.1117/12.2663754.
- [14] T. V. Devi, V. Satyanarayana, and M. K. Singh, "An Efficient Hybrid Technique for Automatic License Plate Recognitions," in *Advances in Transdisciplinary Engineering*, 2023. doi: 10.3233/ATDE221255.
- [15] Lubna, N. Mufti, and S. A. A. Shah, "Automatic number plate recognition: A detailed survey of relevant algorithms," *Sensors*, vol. 21, no. 9. 2021. doi: 10.3390/s21093028.
- [16] J. Jiao, Q. Ye, and Q. Huang, "A configurable method for multi-style license plate recognition," *Pattern Recognit*, vol. 42, no. 3, 2009, doi: 10.1016/j.patcog.2008.08.016.
- [17] S. Kaur, "An automatic number plate recognition system under image processing," *International Journal of Intelligent Systems and Applications*, vol. 8, no. 3, 2016, doi: 10.5815/ijisa.2016.03.02.
- [18] X. Zhai, F. Bensaali, and R. Sotudeh, "Real-time optical character recognition on field programmable gate array for automatic number plate recognition system," *IET Circuits, Devices and Systems*, vol. 7, no. 6, 2013, doi: 10.1049/iet-cds.2012.0339.
- [19] A. S., J. Yankey, and E. O., "An Automatic Number Plate Recognition System using OpenCV and Tesseract OCR Engine," *Int J Comput Appl*, vol. 180, no. 43, 2018, doi: 10.5120/ijca2018917150.
- [20] A. S. F. Gani *et al.*, "A live-video automatic number plate recognition (anpr) system using convolutional neural network (CNN) with data labelling on an android smartphone," *International Journal of Emerging Technology and Advanced Engineering*, vol. 11, no. 10, 2021, doi: 10.46338/ijetae1021_11.
- [21] M. Rezaei and M. Isehaghi, "An efficient method for license plate localization using multiple statistical features in a multilayer perceptron neural network," in *2018 9th Conference on Artificial Intelligence and Robotics and 2nd Asia-Pacific*
- [22] M. P. B., G. Bade, and V. Patil, "Image Processing Based Toll Automation Technique Using ANPR," *HELIX*, vol. 9, no. 3, 2019, doi: 10.29042/2019-4926-4930.
- [23] A. Puranic, D. K., and U. V., "Vehicle Number Plate Recognition System: A Literature Review and Implementation using Template Matching," *Int J Comput Appl*, vol. 134, no. 1, 2016, doi: 10.5120/ijca2016907652.

- [24] G. Y. Abbass and A. F. Marhoon, "Car License Plate segmentation and recognition system based on Deep Learning," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 4, pp. 1983–1989, 2022.
- [25] P. Batra, I. Hussain, M. A. Ahad, G. Casalino, M. A. Alam, A. Khalique, and S. I. Hassan, "A novel memory and time-efficient ALPR system based on Yolov5," *Sensors*, vol. 22, no. 14, p. 5283, 2022.
- [26] D. Gunawan, W. Rohimah, and R. F. Rahmat, "Automatic number plate recognition for Indonesian license plate by using K-nearest neighbor algorithm," *IOP Conference Series: Materials Science and Engineering*, vol. 648, no. 1, p. 012011, 2019.
- [27] K. Ohzeki, M. Geigis, and S. Schneider, "License plate detection with machine learning without using number recognition," *Proceedings of the 2019 Federated Conference on Computer Science and Information Systems*, 2019.
- [28] W. Song-Ren, S. Hong-Yang, S. Zheng-Yi, and T. Wen-Kai, "End-to-End High Accuracy License Plate Recognition Based on Depthwise Separable Convolution Networks," *Computer Vision and Pattern Recognition*, Feb. 2022. doi: <https://doi.org/10.48550/arXiv.2202.10277>
- [29] Taheni Damak, Ousseme Kriaa, Asma Baccar, Mohamed Ali Ben Ayed, and Nouri Masmoudi, *Automatic Number Plate Recognition System based on Deep Learning*, vol. 14, no. 3, pp. 86–90, 2020.
- [30] R. U SUSHMA, M. RITHIKA DEVI, N. MAHESHWARAM, and BHUKYA DR. SREEDHAR, *Automatic License Plate Recognition with YOLOv5 and Easy-OCR method*, vol. 9, no. 1, pp. 1243–1247, Jun. 2022.
- [31] E. Nabizada and D. Kaur, *Real Time Vehicle's License Plate Recognition System Using YOLOv5 Model and Transfer Learning: A Case Study of Afghanistan*, vol. 3, no. 9, pp. 42–47, Sep. 2022.

APPENDICES

APPENDIX A Coding in PyCharm (Image.py)

```
ocr_image.py x
1 from algorithm.object_detector import YOLOv7
2 from utils.detections import draw
3 import json
4 import cv2
5
6 yolov7 = YOLOv7()
7 yolov7.set(ocr_classes=['cell phone'])
8 yolov7.load(weights_path='coco.weights', classes='coco.yaml', device='cpu') # use 'gpu' for CUDA GPU inference
9 image = cv2.imread('phone.webp')
10 detections = yolov7.detect(image)
11 detected_image = draw(image, detections)
12 cv2.imwrite('detected_ocr.jpg', detected_image)
13 print(json.dumps(detections, indent=4))
```

APPENDIX B Coding in PyCharm (Webcam.py)

```
webcam.py x
4 import cv2
5
6 yolov7 = YOLOv7()
7 yolov7.load(weights_path='best.weights', classes='classes.yaml', device='cpu') # use 'gpu' for CUDA GPU inference
8
9 webcam = cv2.VideoCapture(0)
10
11 if webcam.isOpened() == False:
12     print('[!] error opening the webcam')
13
14 try:
15     while webcam.isOpened():
16         ret, frame = webcam.read()
17         if ret == True:
18             detections = yolov7.detect(frame)
19             detected_frame = draw(frame, detections)
20             print(json.dumps(detections, indent=4))
21             cv2.imshow('webcam', detected_frame)
22             cv2.waitKey(1)
23         else:
24             break
25 except KeyboardInterrupt:
26     pass
27
28 webcam.release()
29 print('[+] webcam closed')
30 yolov7.unload()
```

APPENDIX C Coding in PyCharm (Real-Time)

```
run.py x
1 import cv2
2 import pytesseract
3 import pandas as pd
4 import requests
5 import csv
6 import json
7 import time
8 import os
9
10 # Define your RoboFlow API key and model details
11 ROBOFLOW_API_KEY = "ANAbPfQvcYHcLQ6S2uWP"
12 ROBOFLOW_MODEL_URL = "https://detect.roboflow.com/plate-number-dwkyk/1?api_key=" + ROBOFLOW_API_KEY
13
14
15 # Function to perform inference on an image using RoboFlow
16 usage
17 def perform_inference(frame_path):
18     with open(frame_path, 'rb') as file:
19         files = {'file': file}
20         response = requests.post(ROBOFLOW_MODEL_URL, files=files)
21
22     if response.status_code == 200:
23         try:
24             result = response.json()
25             print("Inference result:", json.dumps(result, indent=4))
26         except json.JSONDecodeError as e:
27             print("JSONDecodeError:", e)
28             result = {}
29     else:
30         print("Error:", response.status_code, response.text)
31         result = {}
32
```

```
run.py x
35 # Function to extract text from image using Tesseract OCR
36 usage
37 def extract_text_from_image(image):
38     try:
39         text = pytesseract.image_to_string(image, config='--psm 8').strip()
40     except pytesseract.TesseractNotFoundError:
41         print("Tesseract OCR is not installed or not found in PATH.")
42         text = ""
43     return text
44
45 # Function to draw bounding boxes and extract text
46 usage
47 def process_image(image_path, detections):
48     image = cv2.imread(image_path)
49     results = []
50
51     if "predictions" in detections:
52         for detection in detections["predictions"]:
53             x, y, width, height = detection["x"], detection["y"], detection["width"], detection["height"]
54             top_left = (int(x - width / 2), int(y - height / 2))
55             bottom_right = (int(x + width / 2), int(y + height / 2))
56
57             # Crop the detected number plate from the image
58             cropped_image = image[top_left[1]:bottom_right[1], top_left[0]:bottom_right[0]]
59
60             # Extract text from the cropped image
61             extracted_text = extract_text_from_image(cropped_image)
62             print("Extracted text:", extracted_text)
63
64     # Save the result
```

```

63         # Save the result
64         results.append(extracted_text)
65
66         # Draw bounding box and label on the image
67         cv2.rectangle(image, top_left, bottom_right, (0, 255, 0), 2)
68         cv2.putText(image, extracted_text, org=(top_left[0], top_left[1] - 10), cv2.FONT_HERSHEY_SIMPLEX, fontScale=0.9,
69                    color=(0, 255, 0), thickness=2)
70
71     else:
72         print("No predictions found in the detections.")
73
74     # Save the annotated image
75     cv2.imwrite(filename="annotated_image.jpg", image)
76     return results
77
78 # Initialize the webcam
79 cap = cv2.VideoCapture(0)
80
81 # List to store results
82 all_results = []
83
84 try:
85     while True:
86         ret, frame = cap.read()
87         if not ret:
88             print("Failed to capture frame from webcam. Exiting...")
89             break
90
91         # Save the current frame as a temporary image file
92         frame_path = "current_frame.jpg"
93         cv2.imwrite(frame_path, frame)

```

```

run.py x
93     cv2.imwrite(frame_path, frame)
94
95     # Perform inference on the current frame
96     detections = perform_inference(frame_path)
97
98     # Process the image and extract text
99     results = process_image(frame_path, detections)
100    all_results.extend(results)
101
102    # Display the frame
103    cv2.imshow(winname='Frame', frame)
104
105    if cv2.waitKey(1) & 0xFF == ord('q'):
106        break
107
108 finally:
109    # Release the webcam and close windows
110    cap.release()
111    cv2.destroyAllWindows()
112
113    # Save results to a CSV file
114    with open('number_plates.csv', 'w', newline='') as file:
115        writer = csv.writer(file)
116        writer.writerow(["Plate Number"])
117        for result in all_results:
118            writer.writerow([result])
119
120    print("Results saved to number_plates.csv")
121
122    # Clean up the temporary image file
123    if os.path.exists("current_frame.jpg"):
124        os.remove("current_frame.jpg")

```