

**DEEP LEARNING-BASED CAR PLATE OPTICAL CHARACTER  
RECOGNITION**

**CHOO ZHEN BO**

**A project report submitted in partial fulfilment of the  
requirements for the award of Bachelor of Engineering (Honours)  
Electrical and Electronic Engineering**

**Lee Kong Chian Faculty of Engineering and Science  
Universiti Tunku Abdul Rahman  
May 2022**

## DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature :   
\_\_\_\_\_

Name : Choo Zhen Bo  
\_\_\_\_\_

ID No. : 1806581  
\_\_\_\_\_

Date : 29 Sep 2022  
\_\_\_\_\_

## APPROVAL FOR SUBMISSION

I certify that this project report entitled “**DEEP LEARNING-BASED CAR PLATE OPTICAL CHARACTER RECOGNITION**” was prepared by **CHOO ZHEN BO** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Honours) Electrical and Electronic Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature

:



---

Supervisor

:

Ir. Ts. Dr. Tham Mau Luen

---

Date

:

29 Sep 2022

---

Signature

:

---

Co-Supervisor

:

---

Date

:

---

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of Universiti Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2022, Choo Zhen Bo. All right reserved.

## **ACKNOWLEDGEMENTS**

I would like to thank everyone who has contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor, Ir. Ts. Dr. Tham Mau Luen for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, I would also like to express my gratefulness toward my parents, course mates, and friends who have shown me unconditional support and encouragement while I have been carrying out this project.

## ABSTRACT

In the field of intelligent transport systems, recent years have witnessed the application of deep learning techniques to both car plate detection and recognition. The latter stage, known as optical character recognition (OCR), is more challenging as it requires an accurate prediction of the entire license numbers. One of the widely used OCR engines is the Tesseract, which uses long short-term memory (LSTM). However, the drawback of this approach is the time-consuming image preprocessing techniques. This project aims to design an accurate yet lightweight OCR solution by exploring the bidirectional LSTM, connectionist temporal classification (CTC) and ResNet. The training datasets comprise two public synthetic datasets and one self-collected dataset, which is specific to the Malaysian car plate format. The trained models are subsequently optimized via OpenVINO for faster inference time. Results show that the proposed solution is 10x faster than the Tesseract OCR while still having more than a 2x increase in accuracy. In a case study of vehicle surveillance, a local webserver is established to host the newly developed OCR solutions in combination with a pre-trained YOLOv4 car plate detection. Results show that the end-to-end solution can process video streams at a rate of 20 frames per second (FPS).

## TABLE OF CONTENTS

<b>ACKNOWLEDGEMENTS</b>		<b>i</b>
<b>TABLE OF CONTENTS</b>		<b>iii</b>
<b>LIST OF TABLES</b>		<b>vi</b>
<b>LIST OF FIGURES</b>		<b>vii</b>
<b>LIST OF SYMBOLS / ABBREVIATIONS</b>		<b>ix</b>
<b>LIST OF APPENDICES</b>		<b>x</b>
<b>CHAPTER</b>		
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	General Introduction	1
1.2	Importance of the Study	2
1.3	Problem Statement	3
1.4	Aim and Objectives	4
1.5	Scope and Limitation of the Study	4
1.6	Contribution of the Study	4
1.7	Outline of the Report	5
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>6</b>
2.1	Introduction	6
2.2	License Plate Recognition	7
2.2.1	Car Plate Detection	7
2.2.2	Optical Character Recognition	9
2.3	License Plate Recognition Products in Malaysia	12
2.3.1	Weakness	14
2.4	Summary	16
<b>3</b>	<b>METHODOLOGY AND WORK PLAN</b>	<b>17</b>
3.1	Introduction	17
3.2	Hardware	17
3.2.1	Intel NUC 10i7 Mini PC Kit	17
3.2.2	Beelink Mini PC BT3 Pro	18

3.3	Software	19
3.3.1	Python	19
3.3.2	OpenCV	19
3.3.3	Intel Distribution of OpenVINO Toolkit	20
3.3.4	Apache HTTP Server	20
3.3.5	Flask	21
3.3.6	MySQL	22
3.3.7	YOLOv4 Object Detection	22
3.3.8	Google Colaboratory	22
3.4	Workplan	22
3.4.1	Data Preparation and Annotation	22
3.4.2	OCR Model Selection and Training	23
3.4.3	LAMP Setup	26
3.4.4	LPR System Setup	26
3.4.5	Integration of OCR and LAMP into LPR system	28
3.4.6	Performance Evaluation	28
3.4.7	Gantt Chart	29
3.5	Summary	31
<b>4</b>	<b>RESULTS AND DISCUSSION</b>	<b>32</b>
4.1	Introduction	32
4.2	Car Plate Detection Model	32
4.3	Database Design	32
4.4	Database Dictionary	33
4.5	Graphical User Interface Design	35
4.6	Data flow	38
4.7	Performance of FPS	39
4.8	Performance of OCR Accuracy	41
4.8.1	Comparisons between models	41
4.9	Summary	44
<b>5</b>	<b>CONCLUSIONS AND RECOMMENDATIONS</b>	<b>45</b>
5.1	Conclusions	45
5.2	Recommendations	46
	<b>REFERENCES</b>	<b>47</b>



**APPENDICES**

**LIST OF TABLES**

Table 2.1:	Feature Comparison between each Product	15
Table 3.1:	Proposed Hardware for the Project	17
Table 4.1:	Dictionary of Public List in MySQL	33
Table 4.2:	Dictionary of Resident List in MySQL	34
Table 4.3:	Dictionary of Auto Log List in MySQL	34
Table 4.4:	OCR accuracy of the models using different dataset	41
Table 4.5:	Comparison of models corresponding to FPS and OCR accuracy	42

**LIST OF FIGURES**

Figure 2.1:	Detected objects with bounding boxes	6
Figure 2.2:	Likelihood of an recognised object	7
Figure 2.3:	Character Segmentation	10
Figure 2.4:	System model	11
Figure 2.5:	Architecture of proposed LPR	11
Figure 3.1:	Intel NUC 10i7 Mini PC Kit	18
Figure 3.2:	Beelink Mini PC BT3 Pro	19
Figure 3.3:	Python Logo	19
Figure 3.4:	OpenCV logo	20
Figure 3.5:	OpenVINO logo	20
Figure 3.6:	How server works	21
Figure 3.7:	Flask logo	21
Figure 3.8:	Details of creating LMDB dataset	23
Figure 3.9:	Comparison of 2 type of trade-offs of various combination models	24
Figure 3.10:	Parameters of 4 different stages of combination model	25
Figure 3.11:	Working flow for training and testing the OCR model in Google Colab	25
Figure 3.12:	Architecture of LAMP	26
Figure 3.13:	LPR system flowchart processes	27
Figure 3.14:	Overall flowchart of LPR system after integrated LAMP and OCR model	28
Figure 3.15:	Project Milestone	30
Figure 4.1:	Database Design	33
Figure 4.2:	Video Streaming GUI	35

Figure 4.3:	Public car plate registration GUI	36
Figure 4.4:	New condominium resident registration GUI	37
Figure 4.5:	Public database GUI	38
Figure 4.6:	Resident database GUI	38
Figure 4.7:	Data flow between GUI and backend Python	39
Figure 4.8:	Graph of FPS against type of models	40
Figure 4.9:	Graph of comparison with and without openvino implementation	40
Figure 4.10:	Stem-Vine framework of ResNet	43
Figure 4.11:	Arcitecture of BiLSTM	43

**LIST OF SYMBOLS / ABBREVIATIONS**

AI	Artificial Intelligence
AP	Average Precision
Attn	Attention
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Networks
CPU	Central Processing Unit
CTC	Connectionist Temporal Classification
CTPN	Connectionist Text Proposal Network
DL	Deep Learning
EAST	Efficient and Accurate Scene Text Detector
FPS	frames per second
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HTML	HyperText Markup Language
IoT	Internet of Things
IP	Internet Protocol
IR	Intermediate Representation
LMDB	Lightning Memory-Mapped Database
LP	License Plate
LPR	License Plate Recognition
LSTM	Long Short-Term Memory
OCR	Optical Character Recognition
R-CNN	Region-based Convolutional Neural Network
ReLU	Rectified Linear Unit
STN	Spatial Transformation Network
TPS	Thin-Plate Transformation
URL	Uniform Resource Locator
WSGI	Web Server Gateway Interface

## LIST OF APPENDICES

**No table of figures entries found.**

## CHAPTER 1

### INTRODUCTION

#### 1.1 General Introduction

In the 1950s, Minsky and McCarthy were the pioneers of Artificial Intelligence, AI. “It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to a similar task by using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable.” (McCarthy, 2007). The scope of AI can be categorised into 6 major sub-branches such as robotics, fuzzy logic, machine learning, expert systems, natural language processing and neural network.

In the 1970s, optical character recognition which is part of computer vision was used to interpret typed and handwritten text. This advancement is widely used in data entry of printed paper data records such as documents, invoices, bank statements etcetera. Hence, it is time-saving for a lot of businesses. In the era of the internet, computer vision becomes significant as there are a lot of images available online. These images can be used for analysis, recognition, image sharpening and so forth. Implicitly, computer vision is implemented in video surveillance systems such as facial recognition to identify trespassers on private land and car plate recognition to identify whether a vehicle was at the scene of a crime.

Convolutional Neural Network (CNN) is chosen to mimic the human vision system as it is a powerful Deep Learning (DL) algorithm for image recognition. CNN architecture has the same analogy as the neurons network in the human brain which consists of numerous layers of artificial neurons with learnable weights. The typical layers of artificial neurons are the convolutional layer, pooling layer and fully connected layer. Firstly, the convolutional layer is used to perform feature extraction from an image where the major computational occurs as undergoing a series of the dot product between the matrix of an image and Kernel to generate an output array called feature map as illustrated in Figure 1.1. The feature map will be further processed in the pooling layer where a reduction of the number of parameters occurs. It is known as

downsampling to reduce the computational power required in the fully connected layer. Lastly, the classification of the fully connected layer is to produce a probability from 0 to 1.

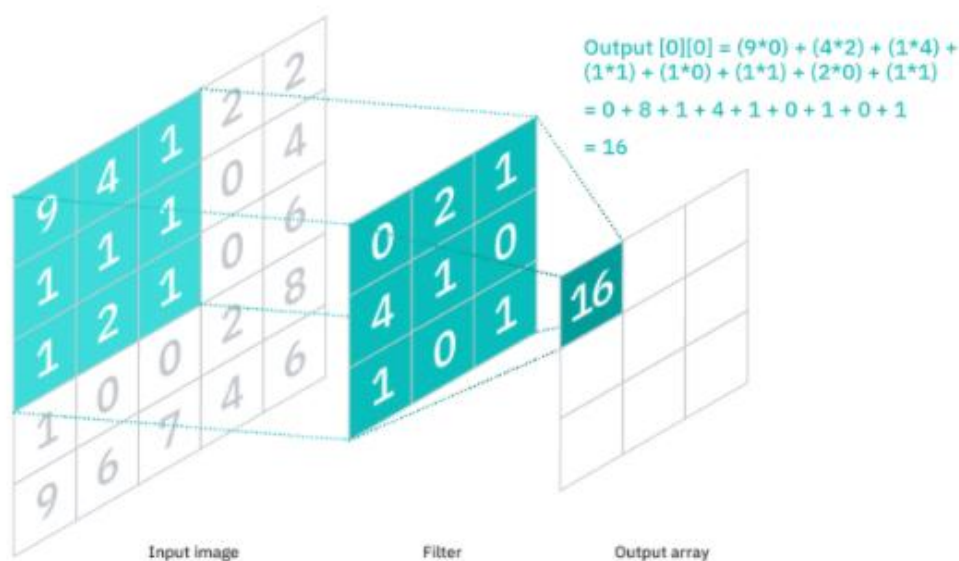


Figure 1.1: Convolution (IBM Cloud Education, 2020)

As the concern of cost and the implementation of IoT devices, the computational time is the constraint of performing real-time applications such as car plate recognition used in parking management of a mall and traffic management. Hence, You Only Look Once, YOLO is a popular algorithm that allows neural networks to perform object detection in real-time and has high accuracy.

## 1.2 Importance of the Study

Nowadays, the popularity of real-time applications has increased which indicates that the amount of computational time should be minimized as much as possible. However, the conventional OCR has a long computational time which is not suitable for real-time applications such as car parking monitoring systems. Besides, the conventional OCR is less effective in reading text in the natural scene due to imperfect conditions of the real world. Hence, a lightweight DL-based OCR is used to automate the extraction and analysis of complex data features with lesser computational time.



This model can be used in a traffic surveillance system to capture the car plate number of a wrongdoer on road and shopping mall car parking monitoring system. For example, Sunway Pyramid Malaysia developed a ticketless and cashless parking experience and residential surveillance system to block the unregistered car plate number. Hence, it has marketing potential to help the public to develop their businesses in a safer sense.

### **1.3 Problem Statement**

In the era of digitalization, the usage of computer vision is more significant to industries and businesses due to the invention of IoT devices. IP camera is one of the IoT devices which is used to capture the images and transmit them to the processing units for further analysis such as optical characters recognition (OCR). There are some applications of OCR in our daily life which are surveillance systems and car park monitoring systems. Hence, a real-time monitoring system is required to sustain the business. However, the high computing power of OCR is required to perform real-time applications which are not cost-effective.

The conventional OCR is based on the light and dark pattern of letters and characters of a captured image to convert it into text form. Besides, conventional OCR requires strict rules to recognise more fonts that are not flexible, increase computational time and high setup cost as a new rule is applied. In reality, the accuracy of OCR to detect characters and letters on car plates is lower than that of handwritten text or printed text due to environmental factors such as light intensity and elevation of geographic location which affect the readability of conventional OCR. Hence, the font of the car plate will look different from a certain degree of angular distance. Therefore, conventional OCR often required more rules for different scenarios to have higher accuracy, which may result in longer computational time.

#### **1.4 Aim and Objectives**

Several conventional OCR solutions exist in the current technology. However, the accuracy and computational time of conventional OCR are the constraints of performing car plate recognition due to environmental factors. Hence, the project aims to develop a DL based OCR. The objectives of the study are:

1. To develop an OCR neural network model based on Malaysia car plates
2. To optimize the model into a lightweight version
3. To integrate it with pre-trained YOLOv4 based license plate detection

#### **1.5 Scope and Limitation of the Study**

Developing a parking area monitoring system that can recognise the car plate from a database using a DL model is the scope of this project. The limitation of this study is the limited processing power of the device used to train the OCR model. Hence, a strong GPU device is recommended for training the DL model. It helps in shortening the training time of the model effectively. On the other hand, Google Colab can overcome the constraint in processing power. However, Google Colab has limited access time for 12 hours of free GPU usage. Even so, the limited access time is sufficient for this project as the small-scale dataset is used.

Another limitation of the study is obtaining a Malaysian car plate dataset to train a model. Although the dataset can be easily found using Google Image, there is no good source for citation and the number of images in it is limited.

Lastly, there are several conversion of ONNX operators is not supported by Pytorch. Therefore, not all the models can be successfully implemented in OpenVINO.

#### **1.6 Contribution of the Study**

This study contributes a huge amount of knowledge in the aspect of license plate recognition and web-based graphical user interface, GUI which are focused on the innovation of industry in terms of technology and usage. Indeed, this study has contributed to the techniques to improve the accuracy of OCR neural network models and reduce excessive computation. These techniques are geofencing and centroid tracking.

## **1.7 Outline of the Report**

The rest of the report is organised as follows:

- Introduction
- Literature Review
- Methodology
- Results and Discussion
- Conclusions and Recommendations

The introduction describes the general knowledge of AI and it is followed by the importance of the study, problem statements, aim and objectives, scope and limitation of the study, and contribution of the study.

The literature review describes the related works which are car plate detection and OCR. In methodology, a proposed solution is presented by integrating the license plate recognition, LPR system and web-based GUI. In the section on results and discussion, the performance evaluation of OCR is analysed and discussed. Lastly, conclusions and recommendations are to summarise the report and recommend several future works to improve the system.

## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction

Generally, object detection is a subset of object recognition that has the combined abilities to recognise and localise an object from an image or video. The localisation of object detection is to draw a rectangular bounding box of detected objects in an image which is indicating it can track the position of an object precisely as illustrated in Figure 2.1. Besides, object detection allows for multiple objects to be identified and located within the same image (MathWorks, n.d.). While the recognition of object detection has a similar technique used in object recognition as the output will be represented in the form of probability. For instance, Figure 2.2 illustrates that the object has 97% predicted as a dog while 3% as a cat. Last but not least, the detected object will be bounded by a bounding box with the respective label.

In some of the applications, it requires the combination of object recognition and object detection such as automatic license plate recognition system implemented car plate detection and OCR instead of doing both at once due to the existing algorithms allowing the single task to be performed are defined as narrow AI.

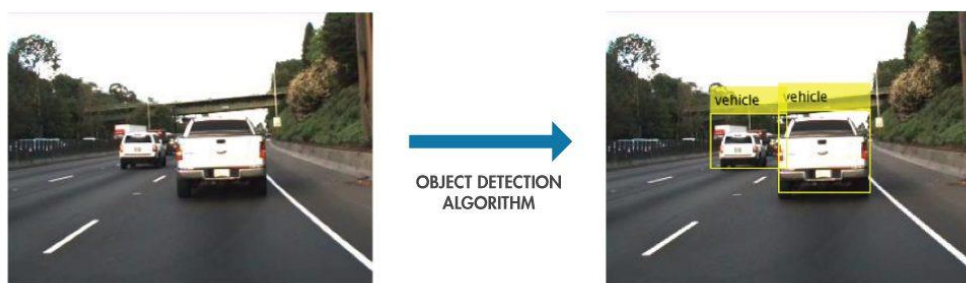


Figure 2.1: Detected objects with bounding boxes (MathWorks, n.d.)



Figure 2.2: Likelihood of an recognised object (GeeksforGeeks, 2020)

## 2.2 License Plate Recognition

The preceding work shows that the proposed architecture of LPR consists of 2 stages, which are OCR by implementing the CNN technique and car plate detection. It states that the traditional LPR system implemented handcrafted features of the machine learning technique as it is simple and fast. As a trade-off, it has low accuracy due to sensitivity to image noise and complex backgrounds. To have better detection accuracy and real-time performance, the work implemented the YOLO model on car plate detection and OCR.

### 2.2.1 Car Plate Detection

Faster Region-based Convolutional Neural Network, R-CNN with the composition of a fully convolutional network and ResNet as the base network is used as the proposed car plate detection model. The fully convolutional network has the advantages of fast processing and is capable of various image resolutions while ResNet has a deeper layer and higher accuracy compared to VGG and ZF base networks (Anusuya and Joseph., 2020). ResNet mainly performs feature extraction with 101 layers. To reduce the computational burden, only 2 pooling layers are used in ResNet which are Max Pooling on the first layer and Average Pooling on the last layer. This model is trained with 880 datasets while tested with 220 datasets. As a result, the detection model has achieved 99.1% of accuracy.

To have a strong car plate detection, 2 different CNN classifiers are used to perform the text string detection and distinguish license plates from the general text (Li et al., 2018). In the first stage of car plate detection, 4 layers of

CNN classifier called CNN-I is used to recognise the existence of characters in an image patch to accelerate the detection process. Besides, the researchers developed a character saliency map to further reduce the processing time of CNN-I. CNN-I is trained with upper-case English letters, numerical digits and a non-character class that has better discriminative and specific learning features. After the first stage, a generated bounding box will further perform false positive elimination and bounding box refining. In the final stage, another CNN classifier called CNN-II will verify the processed bounding box to enhance the classification performance. As a result, the testing is using the Caltech cars dataset, the obtained precision of the entire detection is 97.56%.

To achieve real-time performance, FAST-YOLO is chosen to perform car plate detection as it is one of the fastest networks without relying on image pyramid and sliding windows (Montazzolli Silva, S. and Rosito Jung, n.d.). The original architecture of FAST YOLO has been modified by reducing the number of filters and output from 125 to 35 and 20 to 2 respectively. To train car plate detection, 10 k iterations on the Stochastic Gradient Descent algorithm is used. It has been specified the parameters used in the algorithm are 64 mini-batch sizes, 0.001 learning rate for the first 1 k iterations and 0.0001 learning rate for the rest of the iterations. Average Precision (AP) is used to measure the car plate detection accuracy. A typical 0.5 Intersection over Union (IoU) threshold is used, the obtained AP tested from OpenALPR and SSIG datasets are 90.94% and 96.09% respectively.

Another proposed architecture of car plate detection is using YOLOv2 which has achieved a good compromise between accuracy and speed (Kessentini et al., 2019). Hence, the researchers keep remaining the original architecture of YOLOv2. During the training and testing, the experiments are conducted using NVIDIA GTX 1080 Ti Graphic Processing Unit (GPU). There is 2 set of datasets that were used which are GAP-LP and Radar. The GAP-LP dataset consists of different resolution images, view angles and outdoor lighting conditions while the Radar dataset is the real scene images with a complex background, various lighting intensities, different angles and positions of the car plates. The number of training datasets used in GAP-LP and Radar are 7117 and 3998 respectively while the number of testing datasets used in GAP-LP and

Radar are 1602 and 2000 respectively. As a result, the detection model obtained 100% for precision and recall using the GAP-LP dataset while the precision and recall using Radar are 100% and 99.09% respectively. These results are obtained as the IoU threshold is equal to 0.5.

### **2.2.2 Optical Character Recognition**

According to the journal, License Plate Recognition Using Convolutional Neural Network (Saunshi et al., n.d.), the proposed architecture of OCR uses 2 convolution layers, 2 fully connected layers, 5x5 kernels used in the first convolution layer and followed by a 2x2 size of pooling layer. This architecture is trained with 30000 characters and 6000 characters for testing which has 98% of character recognition accuracy.

One of the approaches to OCR architecture consists of 2 operations which are character segmentation and character recognition (Varad Vinay et al., 2021). In the stage of character segmentation, each of the characters from a car plate will be further separated by a bounding box as illustrated in Figure 2.3. After the characters are isolated with the bounding box, the character recognition starts to perform feature extraction from every character. Furthermore, template matching is the technique is used in character recognition that has bad performance on different font styles used in the car plate. The researchers provided a solution using a neural network able to overcome the various font styles and various colours of the characters.

Another DL architecture used in OCR is Multi-Layer Perceptron Neural Network. It has a single input layer and one hidden layer which consists of 135 input neurons and 40 neurons with log-sigmoid transfer functions respectively (Othman et al., 2007). Lastly, the output stage has 36 neurons which correspond to 26 alphabetic characters and 10 Arabic numerals. Each of the output neurons will be represented in probability corresponding to the input image. The model is trained with the back-propagation algorithm which allows the weights are iteratively converging to the desired output. This model has been tested with 150 images for recognising Malaysian license plates. These testing images has 3 criteria which are complex scene, various environments such as street, different inclined angles and so on. As a result, the performance of the



Figure 2.3: Character Segmentation (Varad Vinay et al., 2021)

character recognition is 93.2%.

CNN has been applied in OCR with another proposed architecture which is using SoftMax in the outer layer of the network, 3 ReLu activations, 2 convolutional layers, 2 fully connected layers and 2 pooling layers (Anusuya and Joseph., 2020). ReLu is preferable to compare to old Sigmoid and Tanh functions due to it having better gradient propagation and being extremely computationally efficient. While SoftMax is commonly used in multiclass classification problems or last activation functions as it can calculate probabilities for every class and the sum of probability of the classes is equal to 1. It shows that the character recognition achieved 99.9 % and 98% accuracies using training and validation datasets respectively.

To recognise Saudi license plate numbers, the researchers developed Artificial Neural Network (ANN) classifier based on OCR to detect and recognise the plate characters from the cleaned image (Alyahya et al., 2017). Figure 2.4 shows the overall architecture of LPR during the training and testing phase. They stated that there is a limited resource of Saudi license plate images. Hence, 22 images of the Saudi license plate which consist of a hybrid letter and number of English and Arabic used as a training dataset. For instance, a mixture of Arabic and English letters or Arabic and Arabic numerals on the license plates. There are 3 different performance metrics are used such as True Positive Rate (TP), False positive (FP) and ROC area. The experiment is conducted using MATLAB for pre-processing, recognition and finally testing. As a result, the proposed system achieved a recognition rate of 92% as it was able to recognise Arabic and English on the same plate.



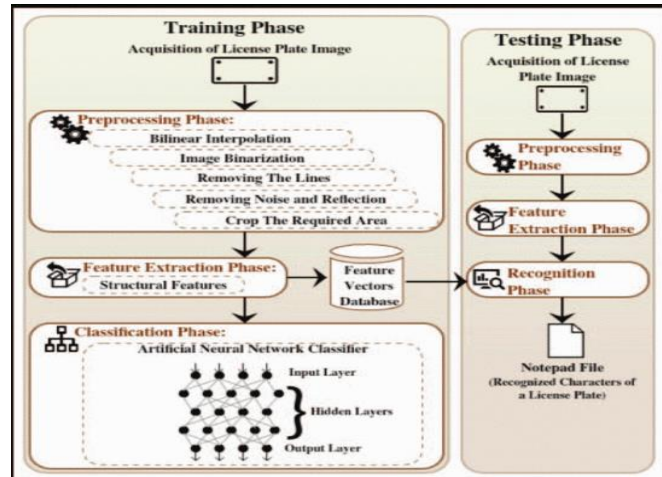


Figure 2.4: System model (Alyahya et al., 2017)

A conventional OCR can be classified into 2 types which are offline recognition and online recognition. The offline recognition is integrated with LPR to recognise Iraqi car plates (Omran and Jarallah, 2017). The proposed LPR architecture is illustrated in Figure 2.5 which a lot of effort had been done in the pre-processing stages. The experiment is conducted in MATLAB, there 35 testing datasets are being used. As a result, the character recognition accuracy of the conventional OCR is achieved at 85.7%.

According to Zheng et al. (2013), they stated the recognition accuracy is significantly reduced as there is no proper character segmentation regardless of the models of OCR used. Therefore, the challenge lies in extracting the boundary of the car plate and segmenting the characters on the car plate. This experiment used 300 vehicle images as a training dataset while 160 images are used as training datasets. These images are taken in various circumstances such as various illumination conditions and view angles. Also, Open source Tesseract OCR is used for recognition. With the proposed segmentation algorithm, the character recognition is achieved at 98.7% and the time taken for detection and recognition of 648 x 486 license plate image is within 0.1 s as using Pentium



Figure 2.5: Architecture of proposed LPR (Omran and Jarallah, 2017)

GHz Central Processing Unit (CPU).

In reality, the conventional OCR has difficulty recognising the natural scene text which has complex background compared to the scanned document images. ResNet is used as feature extraction in a few modifiable networks called Connectionist Text Proposal Network, CTPN and Efficient and Accurate Scene Text Detector, EAST (Yang and Hsieh, 2019). CTPN consists of two sequence networks which are CNN to perform spatial analysis from the receptive field and LSTM to recognise the sequence of input. Hence, the CNN architecture in CTPN is modified from VGG-16 to ResNet. While the feature extraction of EAST architecture is modified from PVANET to ResNet. As a result, the accuracy of modified CTPN and EAST have higher accuracy than the original architecture by using the ICDAR dataset.

From the existing OCR models, the accuracies are considered good. However, most of them do not emphasize the processing time of each frame. Even the processing time of the character recognition is about 0.1 s, which is considered slow when it is required to integrate into a complex system which is computational consuming and ends up the overall latency to process each frame is more than 0.1 s.

### **2.3 License Plate Recognition Products in Malaysia**

M4U Smart License Plate Recognition is one of the LPR products in Malaysia which is developed by Manage4U Sdn Bhd (Manage4U, n.d). The LPR solution of the product uses OCR to recognise the character on the car plate without performing the conventional LPR architecture car plate detection and character segmentation. It is recommended that the product can be integrated into the visitor management system to recognise guest vehicles from the database and allow the automation of vehicle access control to reduce congestion. According to the Manage4U website, the product does not provide the details on accuracy while the hardware specifications do. The hardware consists of 2 LPR cameras that support infrared illuminator and the frame rate per second up to 30 while the mini pc uses an Intel Core i7-6500 processor, 16 GB RAM, 480 GB hard disc and Windows 10 pro operating system.

Wiicontrol Information Technology Co., Ltd has developed another LPR product in Malaysia (Anon., 2022a). The product can detect the vehicle, and recognise the car plate details such as English letters, Arabic numerals and car plate colours. The product has a recognition accuracy of at least 95%, able to recognise 150 car signs, high-speed detection of the vehicle up to 40 km per hour and recognition of 10 colours of the vehicle. It also has a built-in database of the registered car plate that suits the recommended applications such as parking area management, index measurement of traffic flow control, car theft detection, highway speeding tracker and so forth.

Tan Jee Yee (2019) reported that the parking lot of the Bukit Jalil City building in Kuala Lumpur runs kiplePark which was developed by Green Packet Bhd. It is the pioneer in the implementation of LPR technology in Malaysia which created a ticketless, contactless and cashless system. The overview of kiplePark architecture uses camera-embedded LPR and centralised management systems such as payment systems. It is claimed that the accuracy of recognising characters is achieved at least 99% regardless of bad illuminating background on the characters and partially damaged on the car plate.

ZKTeco License Plate Recognition is another LPR technology implemented in a parking management system (ZKTeco, n.d). It has greater colour detection of up to 16 colours, and recognise the manufacturer logo of over 124 vehicle manufacturer such as Audi, BWM, Ford etcetera. Besides, character recognition can recognise the car plate fonts over 27 countries and vehicle type recognition such as sedan and truck. This allows multi-factor authentication to become much more reliable. In addition, it has implemented a DL-based anti-counterfeiting license plate system to prevent fake license plates from consolidating security. It is also claimed that the long-distance and ultra-speed license plate detection is up to 3m long and maximum speed up to 40 km per hour respectively which enable the enhancement of car park traffic capacity.

Moreover, IPSEC Engineering invented an LPR solution using a customised DL algorithm for Malaysian car plates. The accuracy of the algorithm can reach up to 98.5% (IPSEC, n.d). Also, there are supported features such as vehicle colour, brand and type of the recognition algorithm as well for

multiple-factor authentication. However, the overall system relies on a vehicle detector instead of using DL car plate detection in other Malaysia LPR solutions.

Last but not least, Sunway Pyramid is one of the largest shopping malls in Malaysia. The mall smart parking system has implemented the LPR technology of JIESHUN, a China company with 6 years of experience in automated car park management. Felix Zheng Ya Fei, a representative from JIESHUN mentioned that the recognition rate of the system is achieved at 95% as tested with Malaysian car plates (Chips, Y. 2019). The smart parking system has integrated various mobile payment methods such as GrabPay, Boost, Touch'NGo and so on to enable a cashless and ticketless experience. It has an anti-theft feature that enables car owners to "LOCK" their cars in the parking area. Hence, the cars with "LOCK" status are not allowed to exit the car park.

### **2.3.1 Weakness**

Table 2.1 summarized all the features of each product introduced in topic 2.3. Most of the products does not implement LP detection algorithm with OCR which is indicating the computational burden of OCR is questionable as non of the product stated how fast is the processing of the monitoring system. In contrast to the ZKTeco product, the computational of OCR is relatively light as it implemented LP detection with an anti-counterfeiting DL algorithm and other features as well but the accuracy of OCR is not being stated. Furthermore, the IPSEC Engineering product had stated the DL architecture is being used in OCR but the accuracy is up to 98.5% which is undesirable. Hence, the OCR of the products has a trade-off between speed and accuracy.

Table 2.1: Feature Comparison between each Product

Features	M4U	Wiiicontrol	kiplePark	ZKTeco	IPSEC Engin- eering	JIESHUN
LP detection	X	X	X	√	X	X
OCR	√	√	√	√	√ DL	√
Colour detection	X	√ Up to 10 colours	X	√ Up to 16 colours	√	X
Logo detection	X	√ 150 logo	X	√ 124 logo	√	X
High speed sensitivity	X	√ Up to 40 km/h	X	√ Up to 40 km/h	X	X
Anti- counterfei ting	X	X	X	√ DL	X	X
LP detection accuracy	X	X	X	?	X	X
OCR accuracy	?	>= 95%	>= 99%	?	Up to 98.5%	Achieved 95%

where

X = Not applicable

√ = Applicable

? = Not stated

## 2.4 Summary

In this chapter, the various architecture of car plate detection, OCR, and LPR products of Malaysia has been discussed. In general, YOLO architecture has achieved good performance on accuracy and speed in car plate detection OCR. Although car plate detection has better accuracy, the challenges lie in OCR as implementing character segmentation for higher accuracy. Besides, most of the products in Malaysia are considered usable for low power-hungry applications as none of them does not emphasize how much the processing time of each frame is required. In reality, it might not apply to surveillance systems as no implementation of anti-counterfeit features an LPR system. Hence, there are a lot of gaps to improve the Malaysia LPR system, especially in OCR.

## CHAPTER 3

### METHODOLOGY AND WORK PLAN

#### 3.1 Introduction

The whole project will be divided into 3 phases for development. For the first phase, a web server will be developed to monitor and interact with the LPR system. Once the first phase is completed successfully, the second phase will be carried out, which is to carry out model development. During this phase, a Malaysian vehicle license plate dataset was collected from an online source and Sungai Long, Selangor area. This dataset is used to train, validate and test the selected model. Lastly, the web server and the model are integrated into the LPR system.

#### 3.2 Hardware

The proposed hardware is shown in Table 3.1, which consists of a single development board and an implementation board. Both hardware is booted with Ubuntu OS to assist the execution of the software.

Table 3.1: Proposed Hardware for the Project

Hardware	Model	Quantity
Development Board	Intel NUC 10i7 Mini PC Kit	1
Implementation Board	Beelink Mini PC BT3 Pro	1

##### 3.2.1 Intel NUC 10i7 Mini PC Kit

In this project, Intel NUC 10i7 Mini PC Kit is chosen for the development stage. Intel NUC 10i7 Mini PC Kit is a full-fledged desktop in a small or mini form factor, which means that it takes up less space and is extremely portable. It has an Intel Core i7-10710U Processor, where the processor base frequency is 1.10 GHz, with a max turbo frequency of 4.70 GHz. It also has 12 MB of Intel Smart Cache. This PC kit supports up to 64 GB of RAM, and it also supports both M.2 form-factored SSD and HDD for internal storage. The CPU also has integrated Intel UHD Graphics for 10th Generation Intel Processors. This NUC kit has a



Figure 3.1: Intel NUC 10i7 Mini PC Kit (Anon, 2022b)

Thermal Design Power (TDP) of 25 W, which is the average power dissipated by the processor when it is operating at the base frequency with all cores active under load. A TDP of 25 W is relatively low compared to most desktops or laptops. As this kit comes without memory and storage, 64 GB of RAM and 2 TB of SSD were added to enhance its performance.

### 3.2.2 Beelink Mini PC BT3 Pro

Beelink Mini PC BT3 is another full-fledged desktop in a small or mini form factor as well. It uses an Intel Atom Z8350 Processor, where the processor base frequency is 1.44 GHz, with a burst frequency of 1.92 GHz. This Mini PC supports up to 4 GB of DDR3 RAM, and it also supports MicroSD (TF) for internal storage. The CPU also has integrated Intel Atom Z8350. There is an internal GPU that uses Intel HD Graphic. This Mini PC has all the necessary functions such as 2.4 GHz and 5.8 GHz wireless connectivity, Bluetooth 4.0, 4 standard USB ports and Gigabit Ethernet. Although, Beelink Mini PC BT3 has a better processing power compared to Raspberry Pi board computer, the processing power of Beelink Mini PC BT3 is slower compared to Intel NUC 10i7 Mini PC Kit in terms of burst frequency. However, it is used for implementation as the price is relatively low compared to Intel 10i7 Mini PC Kit.





Figure 3.2: Beelink Mini PC BT3 Pro (Anon, 2022c)

### 3.3 Software

In this project, the chosen software are Python, OpenCV, Intel Distribution of OpenVINO toolkit, Apache HTTP Server, Flask, MySQL, YOLOv4 Object Detection and Google Colaboratory in both stages.

#### 3.3.1 Python

Python as shown in Figure 3.3 will be used as the core programming language to combine and run the whole system because OpenVINO and MySQL both support Python. Also, there are various libraries such as OpenCV and Flask that can be used with Python which simplifies the development process and speeds up the development time. Therefore, Python is preferred over other languages for its simplicity and readily available modules.



Figure 3.3: Python Logo (Anon, 2022d)

#### 3.3.2 OpenCV

OpenCV as shown in Figure 3.4 is an open-source library for computer vision and machine learning software. It is used to provide common facilities for real-time operation and to speed up the capability of the machine to interpret data in

commercial products. It can be used free of charge under the Berkeley Software Distribution (BSD) license. The usage of OpenCV is essential in this project as it is used to perform detection and visualisation on video frames.



Figure 3.4: OpenCV logo (Anon, 2022e)

### 3.3.3 Intel Distribution of OpenVINO Toolkit

The Intel Distribution of OpenVINO Toolkit as shown in Figure 3.5, which stands for Open Visual Inference and Neural Network Optimisation, is a toolkit that is provided by Intel to ease the inferencing process of DL computer vision models. OpenVINO is used to optimise the OCR DL trained model to the Intermediate Representation (IR) format.



Figure 3.5: OpenVINO logo (Ligade, M. 2020)

### 3.3.4 Apache HTTP Server

The fundamental job of a web server is to receive all the incoming requests from the client and inform the server to send the corresponding outputs to the client as illustrated in Figure 3.6. Apache is a well-known free and open-source web server that allows a high degree of customisation for commercial use. Also, Apache is responsible for secure communication between 2 machines that utilise the HTTP protocol. In this project, Apache is used to develop a user-friendly

interface to ease the monitoring of the LPR system where it is codeless in a high-level perspective.



Figure 3.6: How server works (Richard, B., n.d.)

### 3.3.5 Flask

To develop a web applications framework using python, Flask as shown in Figure 3.7 is recommended as it is implemented based on the Werkzeug Web Server Gateway Interface (WSGI) toolkit and the Jinja2 template engine. WSGI is a standard protocol for python which has a common interface between web servers and web applications whereas Werkzeug is a WSGI toolkit to perform data requests, object responses and other utilities as well. Hence, Apache uses WSGI files to interact with Python. Besides, the Jinja2 template engine allows the python logics or variables to pass into HyperText Markup Language (HTML) templates to format dynamic content.



Figure 3.7: Flask logo (Anon, 2022f)

### **3.3.6 MySQL**

MySQL is a free relational database management system. This relational database management system offers flexible data management by storing data in different tables instead of lump-sum it into a table. It also can handle large databases much faster than the existing solution. MySQL supports Python written connection and execution to MySQL server, this allows the interface of Apache and MySQL. Throughout the project, all the detected car plates are stored in the MySQL database and displayed on the stored data at the webserver.

### **3.3.7 YOLOv4 Object Detection**

YOLOv4 is the fourth version of the YOLO algorithm which is one of the DL-based of object detection models. It has achieved 43.5% AP based on the MS COCO dataset with a speed of about 65 frames per second (FPS) which is compatible with Tesla V100 (Bochkovskiy et al., 2020). These advantages support the real-time application of the LPR system. Hence, this project proposes to implement a YOLOv4-tiny network pre-trained model from IoT Based License Plate Recognition System Using DL and OpenVINO (Tham and Tan, 2021) which has achieved 96.64% accuracy and is able to perform the real-time implementation of the project.

### **3.3.8 Google Colaboratory**

Google Colaboratory or Google Colab is an online browser-based platform that allows any user to execute an arbitrary python code. It has provided access to GPU computing resources which are suitable for machine learning development. Other than hardware supports, it also supports most of the available python libraries which are related to machine learning such as TensorFlow, PyTorch, Keras and so forth. Therefore, it assists in this project to develop the OCR DL model.

## **3.4 Workplan**

### **3.4.1 Data Preparation and Annotation**

Plate Portal is a website where various country car plates are captured. It includes Malaysian car plates which are used as a part of the car plate dataset in

this project. While capturing the car plate around Sungai Long is the largest contribution to the car plate dataset. Hence, the total amount of images in the dataset is 1500 which is 80% of it for training, 10% for validation and 10% for testing. Besides, MJSynth and SynthText datasets are synthetically generated and open source datasets which are used as artificial real-world scene text to increase the limited amount of training dataset. Hence, these additional datasets are used to further increase the accuracy of the model. The published work “deep-text-recognition-benchmark” of Clova AI Research a GitHub community provide a simple way to perform data annotation by downloading and running their python code for the desired data folder structure and the arguments. The data folder structure, respective content of the file and the command to create the annotated Lightning Memory-Mapped Database (LMDB) dataset as shown in Figure 3.8.

1. Create your own lmbd dataset.

```
pip3 install fire
python3 create_lmbd_dataset.py --inputPath data/ --gtFile data/gt.txt --outputPath result/
```

The structure of data folder as below.

```
data
├── gt.txt
└── test
    ├── word_1.png
    ├── word_2.png
    ├── word_3.png
    └── ...
```

At this time, `gt.txt` should be `{imagepath}\t{label}\n`  
For example

```
test/word_1.png Tiredness
test/word_2.png kills
test/word_3.png A
...
```

Figure 3.8: Details of creating LMDB dataset (Baek et al, 2019)

### 3.4.2 OCR Model Selection and Training

The OCR neural network model can be divided into transformation, feature extraction, sequence modelling and prediction. Natural scene texts usually appear in a curved and tilted manner, transformation is required to normalise the text region into a predefined region (Baek et al, 2019). While Feature extraction assists in suppressing the amount of redundant image information

such as font, colour, size and background. Hence, it can extract the relevant attribute of the characters. And lastly, Sequence modelling will extract the information within the sequence of characters which is to consolidate the prediction stage.

Since there are constraints of the low processing time of CPU and unsupported ONNX conversion version, the chosen combination models are ResNet, Bidirectional Long Short-Term Memory (BiLSTM) and Connectionist Temporal Classification (CTC). For instance, the operator called grid sample in Thin-Plate Transformation, TPS is unsupported in the current version of PyTorch to perform the exporting to ONNX format. In terms of processing time, the Attention, Attn required more computational time by comparing the TPS-ResNet-BiLSTM-CTC and TPS-ResNet-BiLSTM-Attn. Figure 3.9.a shows that chosen combination models have the best trade-off between speed and accuracy. While Figure 3.9.b shows the combination models consume more memory about 49.6 mega floating-point parameters due to ResNet contributing 44.3 mega floating-point parameters to the combination as shown in Figure 3.10.

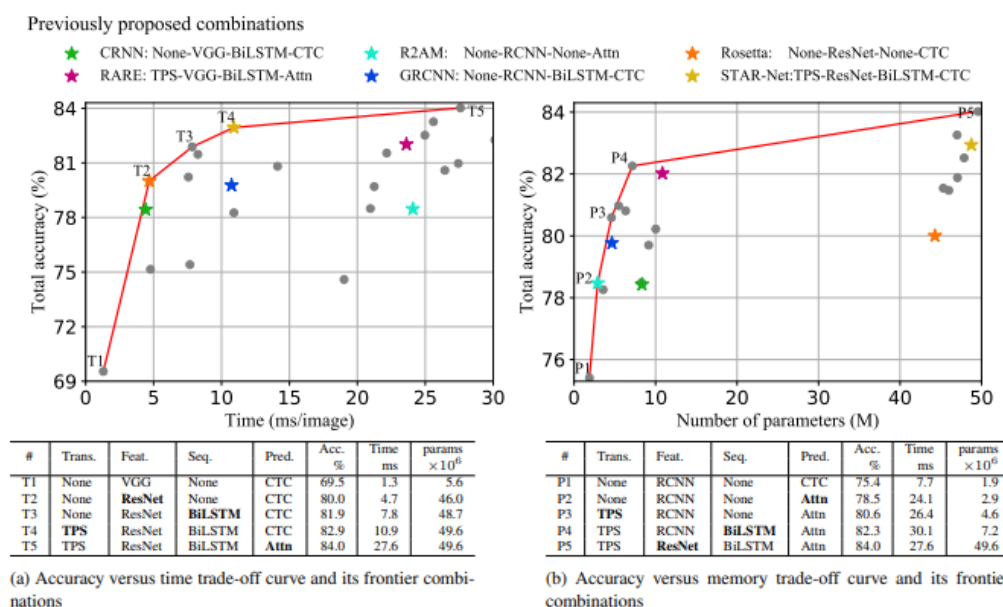


Figure 3.9: Comparison of 2 type of trade-offs of various combination models (Baek et al, 2019)

Stage	Module	Accuracy		Time ms/image	params $\times 10^6$
		Regular (%)	Irregular (%)		
<b>Trans.</b>	None	85.6	65.7	N/A	N/A
	TPS	<b>86.7(+1.1)</b>	<b>69.1(+3.4)</b>	<b>3.6</b>	<b>1.7</b>
<b>Feat.</b>	VGG	84.5	63.9	1.0	5.6
	RCNN	<b>86.2(+1.7)</b>	<b>67.3(+3.4)</b>	<b>6.9</b>	1.8
	ResNet	<b>88.3(+3.8)</b>	<b>71.0(+7.1)</b>	4.1	<b>44.3</b>
<b>Seq.</b>	None	85.1	65.2	N/A	N/A
	BiLSTM	<b>87.6(+2.5)</b>	<b>69.7(+4.5)</b>	<b>3.1</b>	<b>2.7</b>
<b>Pred.</b>	CTC	85.5	66.1	0.1	0.0
	Attn	<b>87.2(+1.7)</b>	<b>68.7(+2.6)</b>	<b>17.1</b>	<b>0.9</b>

Figure 3.10: Parameters of 4 different stages of combination model (Baek et al, 2019)

Figure 3.11 illustrates the workflow in Google Colab to perform training and testing of the OCR model. At the beginning of the workflow, enables free GPU in Google Colab. Then download all the dependency and python files from the GitHub “deep-text-recognition-benchmark”. All the files required for training the OCR model are configured such as dataset and model paths. Then move image datasets to the cloud storage. Lastly, train and test the combination models.

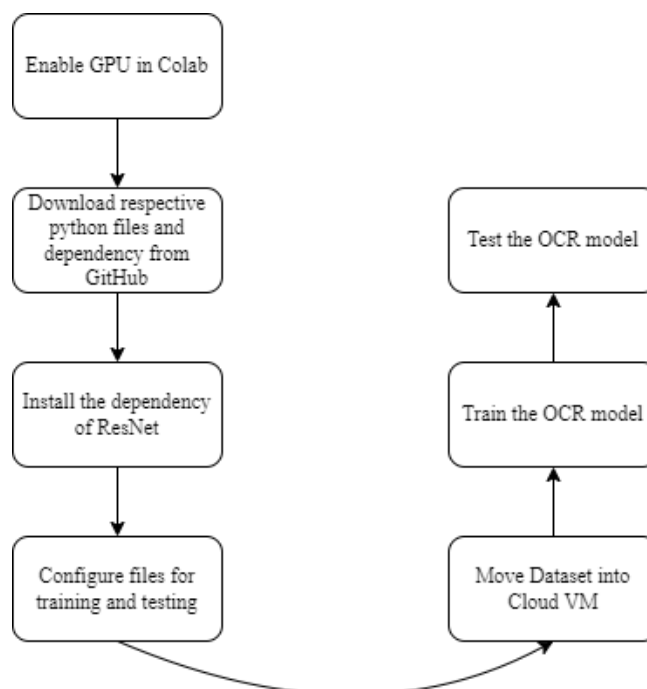


Figure 3.11: Working flow for training and testing the OCR model in Google Colab

### 3.4.3 LAMP Setup

LAMP consists of 4 open-source components which are to develop flexible web server applications. In conventional LAMP, it is stood for Linux, Apache, MySQL and PHP. However, this project utilises Python as the core programming language instead of PHP due to Python being able to perform the dynamic process with the assistance of Apache by Flask and AI inferencing using OpenVINO within the same script. Linux is the backbone of these components to provide more flexibility and configuration options. This project implement LAMP architecture as illustrated in Figure 3.12. This allows users able to interface in the browser by deleting, adding and sorting the databases while streaming video will update it automatically without any user interference.

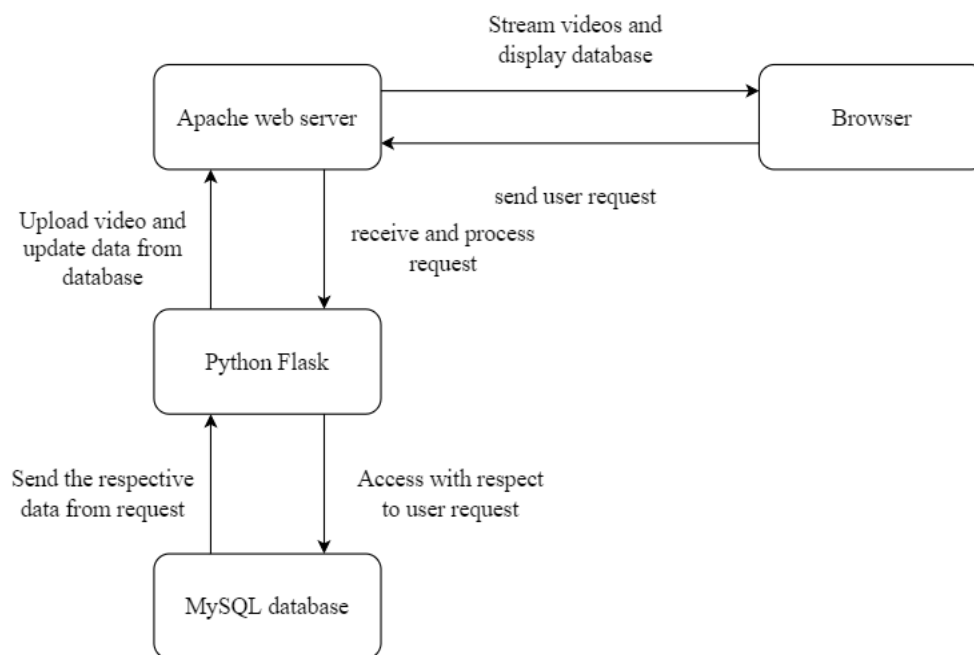


Figure 3.12: Architecture of LAMP

### 3.4.4 LPR System Setup

Figure 3.13 shows the overall flowchart processes of the LPR system. The LPR system consists car plate detection model as mentioned in subchapter 3.3.7 to detect a car plate with the implementation of geofencing to reduce the computation burden by restricting the detecting region and centroid tracking as well to predict the next position of the moving object or in the other word is to identify both objects are the same from previous and current frames. Once the



car plate is detected in the predefined region from geofencing, the car plate will be grayscaled to match the dimension of the input model and reduce the colour dimension structure that implicitly further lowering the computing time.

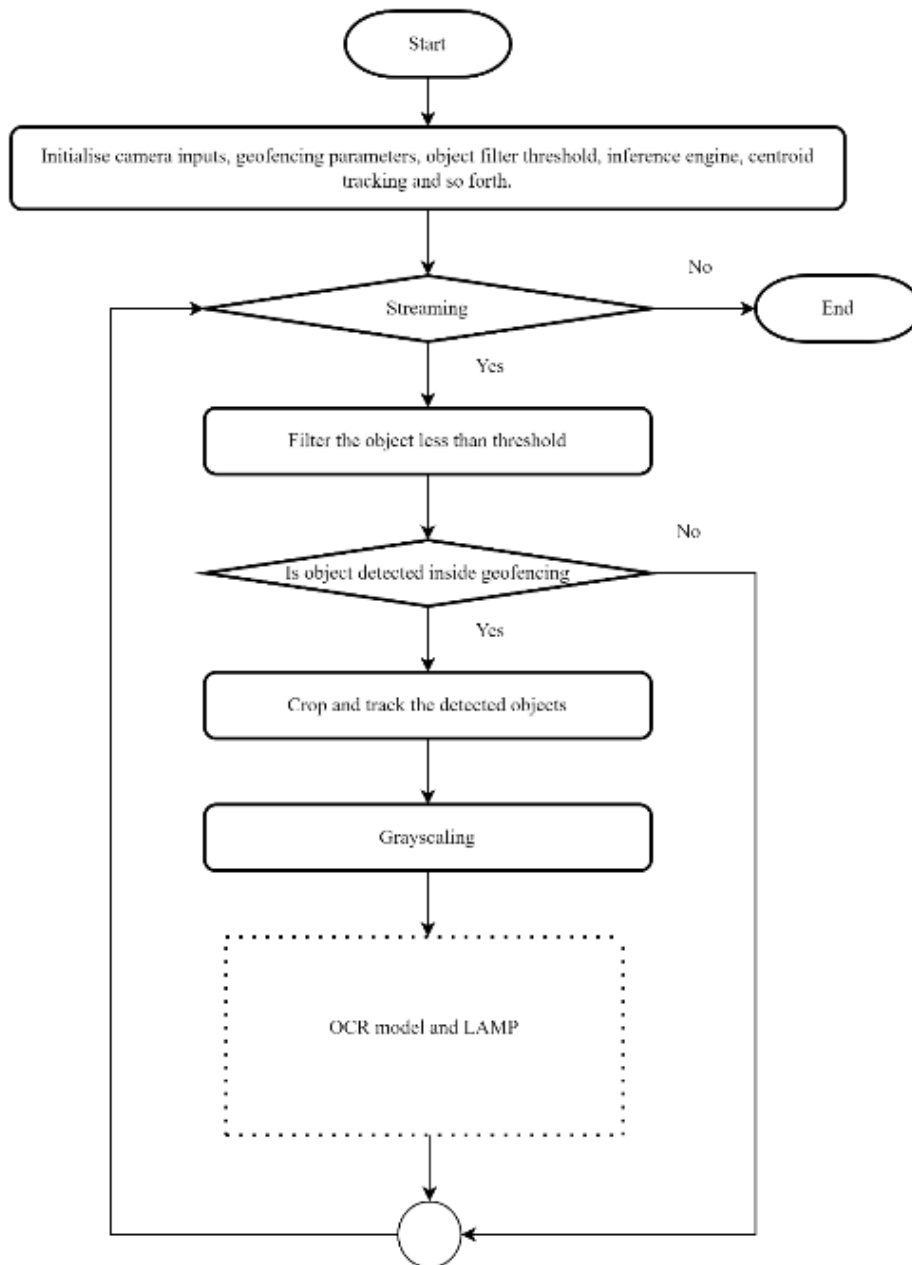


Figure 3.13: LPR system flowchart processes

### 3.4.5 Integration of OCR and LAMP into LPR system

Firstly, the OCR model must convert into IR format before the integration. Then LPR system can integrate the OCR model after the grayscaling of the image to generate the recognised car plate number as output while the LAMP is integrated after the OCR model is implemented which to find the existence of the output in the database and then performs the corresponding action for the existence of the output, for instance, display warning text on the frame as the output not found in the database. The completed LPR system is illustrated in Figure 3.14 after being integrated with the LAMP and OCR model.

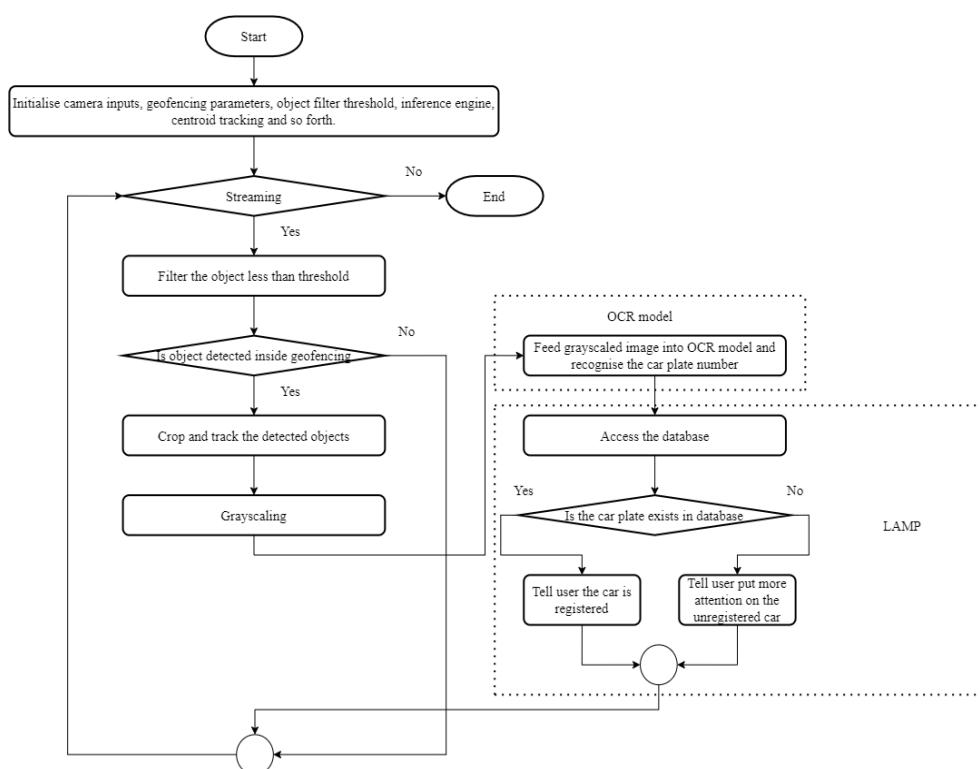


Figure 3.14: Overall flowchart of LPR system after integrated LAMP and OCR model

### 3.4.6 Performance Evaluation

The performance of the system can be divided into which are OCR accuracy and FPS. OCR accuracy is to calculate the percentage of the correct recognised characters by the model while FPS is to measure how fast a system can capture, process and display it as output. The formula of FPS and OCR accuracy is shown in equations (3.1) and (3.2) respectively.

$$FPS = \frac{\text{Input Frames}}{\sum \text{inferencing time of each frame}} \quad (3.1)$$

$$A_{\text{recognition}} = \frac{R_{LP}}{F_i} \times 100\% \quad (3.2)$$

where

FPS = average frames per second, FPS

$A_{\text{recognition}}$  = OCR accuracy, %

$R_{LP}$  = number of license plates is correctly recognised

$F_i$  = number of the input license plates



The performance evaluation consists of two sections. The detailed descriptions are below:

1. Performance of OCR accuracy
  - Comparison of several OCR models
2. Performance of FPS
  - Comparison of several OCR models in the LPR system using OpenVINO
  - Comparison of with and without OpenVINO implementation

### 3.4.7 Gantt Chart

Throughout two Trimester, the project is conducted by following the Gantt chart as illustrated in Figure 3.15. The First Trimester, it is focusing on the model development to reduce the computational power required and web server development to monitor the overall system while the second Trimester will implement both into a complete system.

## Deep Learning Based Car Plate Optical Character Recognition

Completed  Remaining 

Project Supervisor:

Ir. Ts. Dr Tham Mau Luen

Project Start: 7-Feb-22

MILESTONE DESCRIPTION	PROGRESS	START	END	Week	Trimester 1														Trimester 2												
					Jan	Feb				Mar				Apr				May	Jun				Jul				Aug				Sept
					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
<b>Phase 1 Proposal Report</b>																															
Task 1 Introduction	100%	1	2		1	2																									
Task 2 Literature Review	100%	3	4				3	4																							
Task 3 Methodology	100%	5	6					5	6																						
Task 4 Finalisation	100%	11	12										11	12																	
<b>Phase 2 Web server development</b>																															
Task 1 HTML and CSS	100%	5	6					5	6																						
Task 2 Python Backend	100%	5	6					5	6																						
Task 3 Testing and Improvement	100%	7	8						7	8																					
<b>Phase 3 OCR Model Development</b>																															
Task 1 Data Preparation	100%	7	8						7	8																					
Task 2 Data Annotation	100%	8	9							8	9																				
Task 3 Model Developing and Training	100%	9	10								9	10																			
Task 4 Evaluation and Testing	100%	10	11									10	11																		
<b>Phase 4 LPR System Integration and Final Report</b>																															
Task 1 Models Integration	100%	15	18																												
Task 2 Web server Integration	100%	17	20																												
Task 3 Testing and Improvement	100%	19	22																												
Task 4 Poster Design	100%	23	25																												
Task 5 Result and Discussion	100%	24	26																												
Task 6 Conclusion and Finalisation	100%	27	28																												

Figure 3.15: Project Milestone

### **3.5 Summary**

In this chapter, the proposed hardwares are Intel NUC 10i7 Mini PC Kit for development and Beelink Mini PC BT3 Pro for implementation. These hardwares are further installed with certain software to develop the webserver and LPR system. Flask, MySQL, and Apache are the core software to build interactive web servers while OpenCV, OpenVINO and pre-trained YOLOv4 object detection are used as the mainframe of the LPR system. Furthermore, the combination models ResNet, BiLSTM and CTC are selected to train and test in Google Colab as the speed and accuracy are relative optima. Subsequently, the trained model and webserver will be integrated into the LPR system. Finally, the evaluation of the complete LPR system is a comparison different of OCR models for the recognition accuracy and the FPS.

## CHAPTER 4

### RESULTS AND DISCUSSION

#### 4.1 Introduction

The evaluation of this project utilises the CPU of Intel NUC 10i7 mini PC kit and a captured video dataset which consists of a real scene of cars with 30 FPS and a duration length of 26 minutes and 43 seconds. The intention of using this video dataset which has environmental constraints such as light intensity, background complexity and different captured angles as the camera is moving. There are two major evaluations in this chapter which are on the performance of FPS and the performance of OCR accuracy. Performance evaluations will use several models which include ResNet-BiLSTM-CTC, ResNet-CTC, ResNet-FC and Tesseract OCR. As mentioned in the previous chapter, ResNet-BiLSTM-CTC is the proposed model to benchmark with other models. For the sake of completeness, the evaluation is done by implementing the car plate detection model, car plate recognition model and the Graphical User Interface, GUI.

#### 4.2 Car Plate Detection Model

Tham and Tan (2021) have developed a YOLOv4-tiny architecture-based pre-trained model that is able to detect the car plate image with an accuracy of 99%. YOLOv4-tiny is a well-known lightweight model, that has the capability to infer an image with a short computational time. Hence, this car plate detection model is used in these evaluations to filter the unwanted background from the car plate image before it is fed into the recognition models.

#### 4.3 Database Design

In the LPR system, the databases play the part of a major role to identify the car plate. The project utilises 3 databases which can be classified as Public List, Resident List and Auto Log List as shown in Figure 4.1. The design of databases in such a way that records the car plate number, time and permissibility. Permissibility in the LPR system is representing List\_type for instance, the non-register car plate number corresponding to the residential area is forbidden to

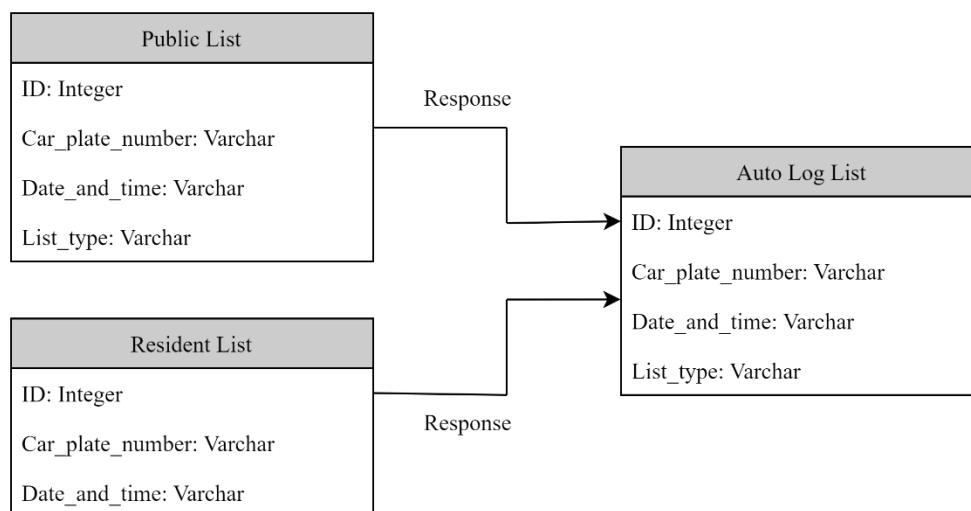


Figure 4.1: Database Design

enter the area or classify it as a black list. The difference between the Resident List compare to Public List and the Auto Log list is the lack of List\_type due to the car plate numbers in the Resident List representing the resident who permits to enter the area. Furthermore, Public List uses as a temporary car plate number registration for classification use such as visitors of a certain condominium to allow them to enter the area within a period called “white list” while a wrongdoer is a visitor who violates the particular rule called “black list”. Lastly, Auto Log List is to automate the recording of a car plate number, time and classification for every entry by checking the existence in Resident List and Public List which serve as backup tracking.

#### 4.4 Database Dictionary

In this section, each of the databases has the details of the information of the fields such as field name, data type, data format, field size, description and example where the shown in Table 4.1, Table 4.2 and Table 4.3.

Table 4.1: Dictionary of Public List in MySQL

Field Name	Data Type	Data Format	Field Size	Description	Example

ID	Integer	N	No limit	Unique ID of detected car plate number	1
Car_plate_number	Varchar	String	20	The recorded car plate number	PNF 7580
Date_and_time	Varchar	%d_%m_%Y - %X	100	Date and time of the recorded car plate	07/03/2022 - 13:04:03
List_type	Varchar	String	40	Indicator of resident and non-resident	white

Table 4.2: Dictionary of Resident List in MySQL

Field Name	Data Type	Data Format	Field Size	Description
ID	Integer	N	No limit	Unique ID of detected car plate number
Car_plate_number	Varchar	String	20	The recorded car plate number
Date_and_time	Varchar	%d_%m_%Y - %X	100	Date and time of the recorded car plate

Table 4.3: Dictionary of Auto Log List in MySQL

Field Name	Data Type	Data Format	Field Size	Description	Example
ID	Integer	N	No limit	Unique ID of detected car plate number	1
Car_plate_number	Varchar	String	20	The recorded car plate number	PNF 7580



Date_and_time	Varchar	%d_%m_%Y - %X	100	Date and time of the recorded car plate	07/03/2022 - 13:04:03
List_type	Varchar	String	40	Indicator of resident and non-resident	white

#### 4.5 Graphical User Interface Design

In this project, GUI is to ease a user to monitor the LPR system. From Figure 4.2, the GUI provide video streaming for the operator such as a guard to monitor the permissibility of the car based on the car plate number. When the car plate number is not existing in the Resident List and Public List databases, the system displays a warning message such as “Alert! The car is not registered” in the streaming video to notify the operator to do action. At the same time, the car plate number is displayed on the video streaming regardless it is within the databases or not.

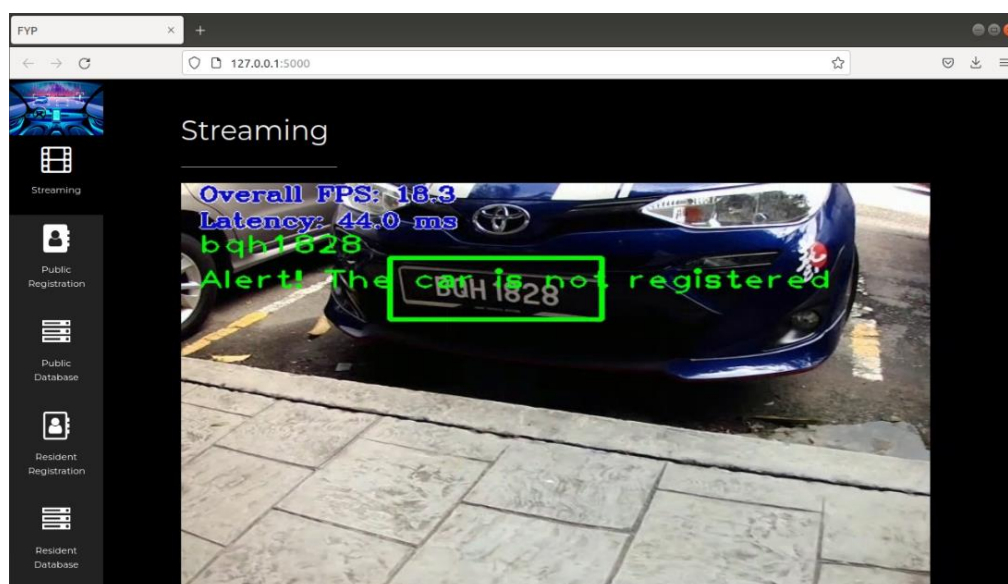


Figure 4.2: Video Streaming GUI

Another GUI design is the Public registration feature, which allows the operator to key in the car plate number and permissibility of the car. For instance, the permissibility of the car of “white” type is within 1 day. Hence, the visitor

can access the area without the blocking of the operator. While permissibility of the car of the “black” type is to block it from entering the area. Figure 4.3 shows there is a space for the operator to key in the car plate number and select the permissibility of list type while there is an automatic mechanism for the registered time of the car plate number. Also, the Resident Registration is the GUI feature as well which has a slightly different working principle compared to Public registration. From Figure 4.4, the difference in Resident Registration is without the list type. Whenever the car plate number registers in the Resident Registration, there is no blocking of the operator for the car entering the area. In addition, every entry of the car plate number will update the database in GUI respectively.

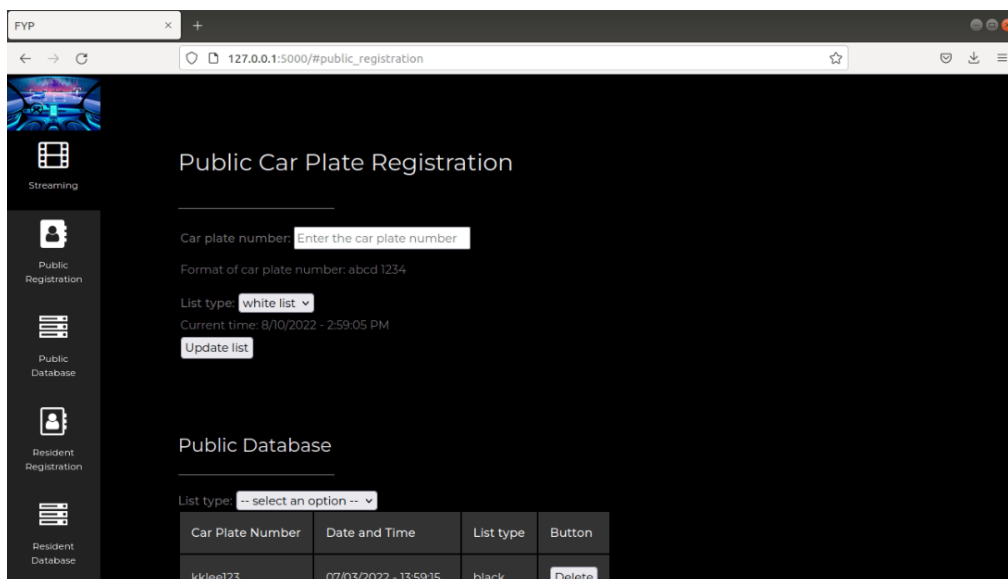


Figure 4.3: Public car plate registration GUI

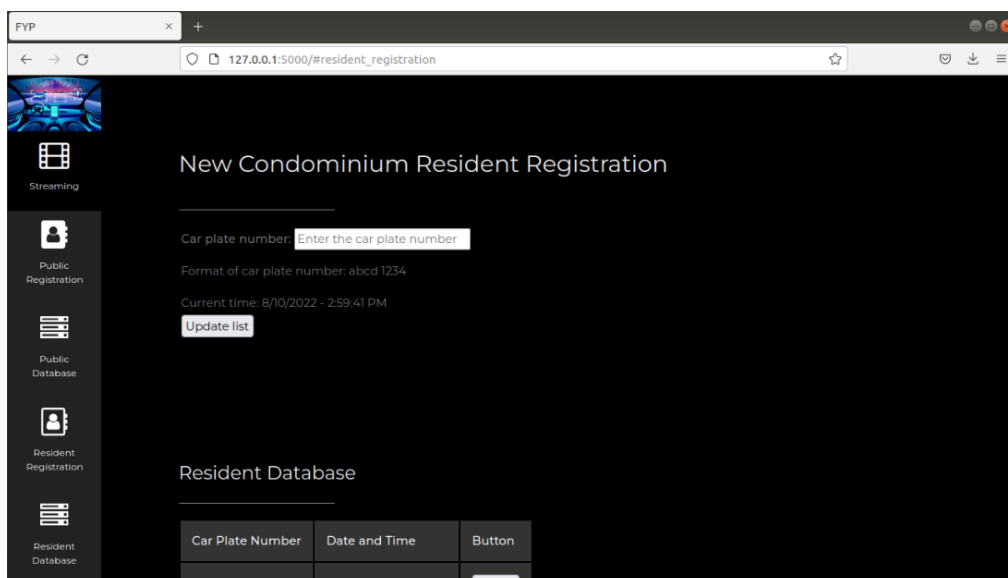


Figure 4.4: New condominium resident registration GUI

Lastly, the Public database and Resident database in the GUI are to visualise and delete car plate numbers corresponding to other fields. In Public database enable the selective type list of the car plate number to ease the operator for tracking whether the car plate number is black or white type or yet registered. The delete button in this GUI feature is to remove the registered car plate number when there is an erratum of entry of the car plate number or change the list type of the car plate number. The resident database has the same feature as the Public database in GUI excluding the selective type list when comparing Figure 4.5 and Figure 4.6. The purpose of the delete button in the Resident database is to remove the resident who no longer lives in the area.

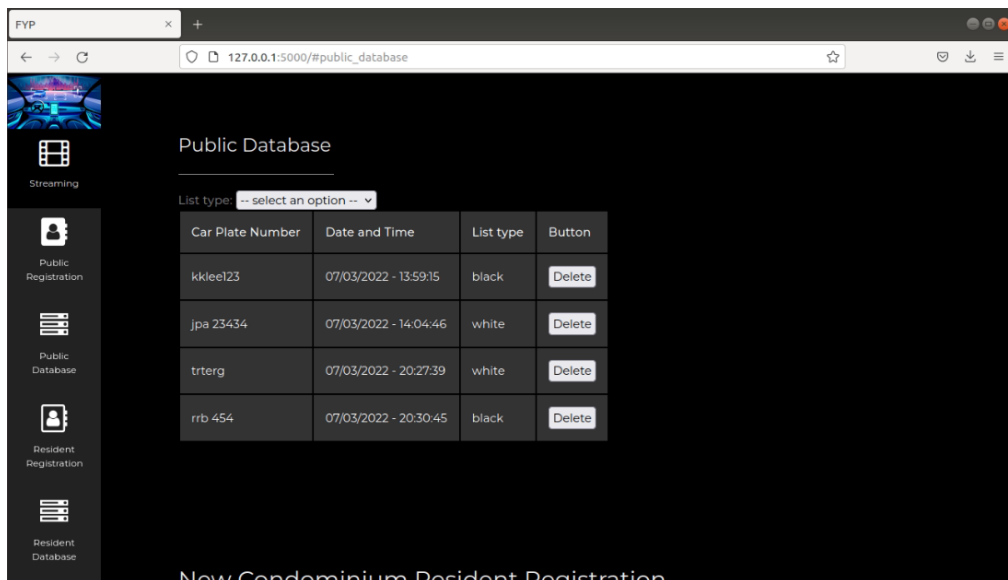


Figure 4.5: Public database GUI

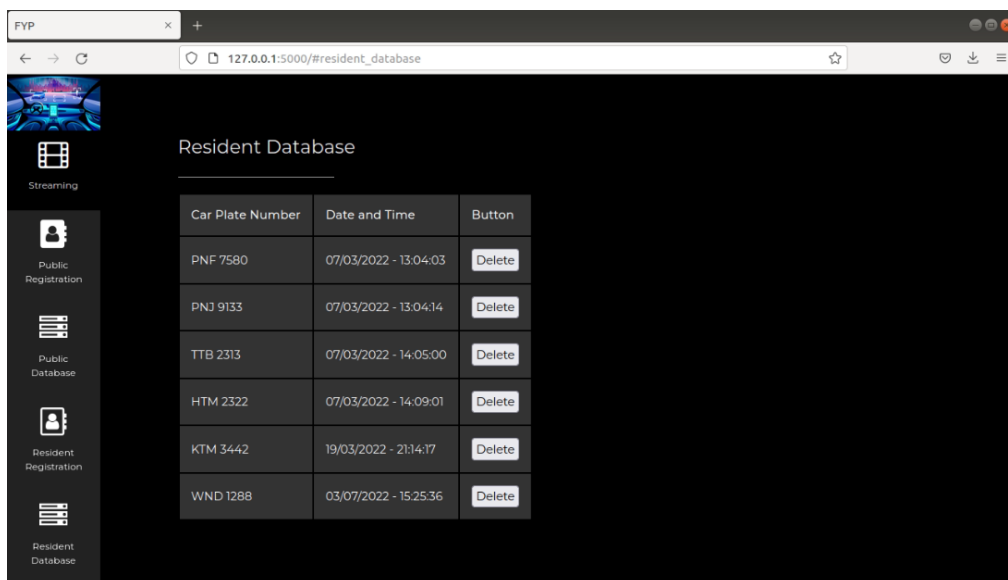


Figure 4.6: Resident database GUI

## 4.6 Data flow

Figure 4.7 show the data flow from browser to backend Python. In the browser, there are several requests from a user. Firstly, a user request to view video streaming will notify the video streaming GUI to request from the python backend process through Python Flask. Secondly, a user sends a new car plate in either the Public database (Public List) or Resident database (Resident List) through the corresponding registration GUI will update the database and intermediately return the updated database to the user via the “view and delete

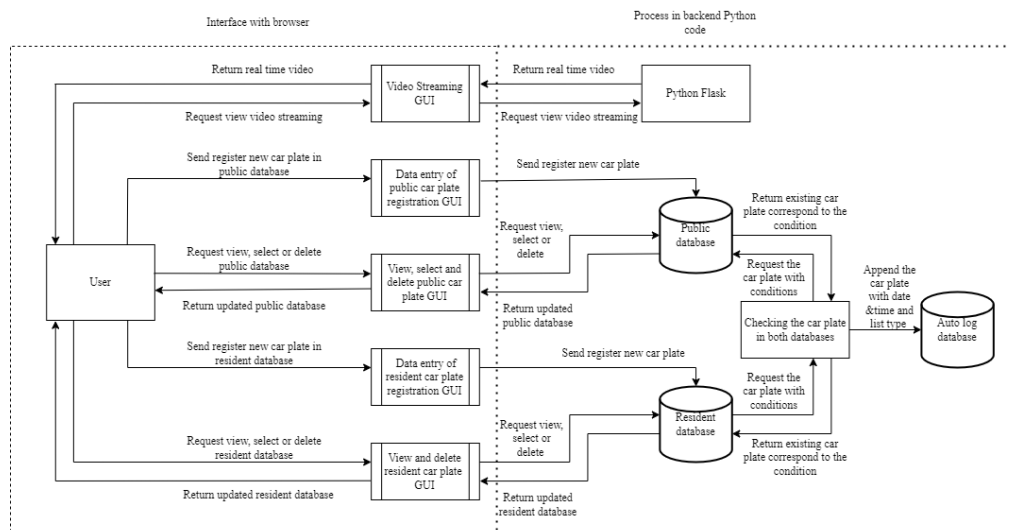


Figure 4.7: Data flow between GUI and backend Python

public/resident car plate” GUI or Public/Resident database GUI. Also, there is no interference of the user to the Auto log database (Auto Log List) as it is used for backend Python code to check the existing recognised car plate number and return it to the display.

#### 4.7 Performance of FPS

In this section, FPS is computed by dividing 1 by the average inferencing time of every input frame that is obtained from the car plate detection model. Based on Figure 4.8, shows that the model ResNet-CTC achieves the fastest FPS among the rest while Tesseract OCR is the slowest. The combination models between ResNet-CTC and ResNet-BiLSTM-CTC, indicate the additional stage will reduce the FPS. Secondly, the FC layer is more computational intensive compared to CTC as it is connected to every input neuron from the previous layer and that explains the significant drop in FPS of ResNet-FC. Hence, CTC is relatively lighter compared to the FC layer. Lastly, Tesseract OCR is used as a conventional OCR to benchmark with these combination models. This measurement was conducted when the models were optimised using OpenVINO.

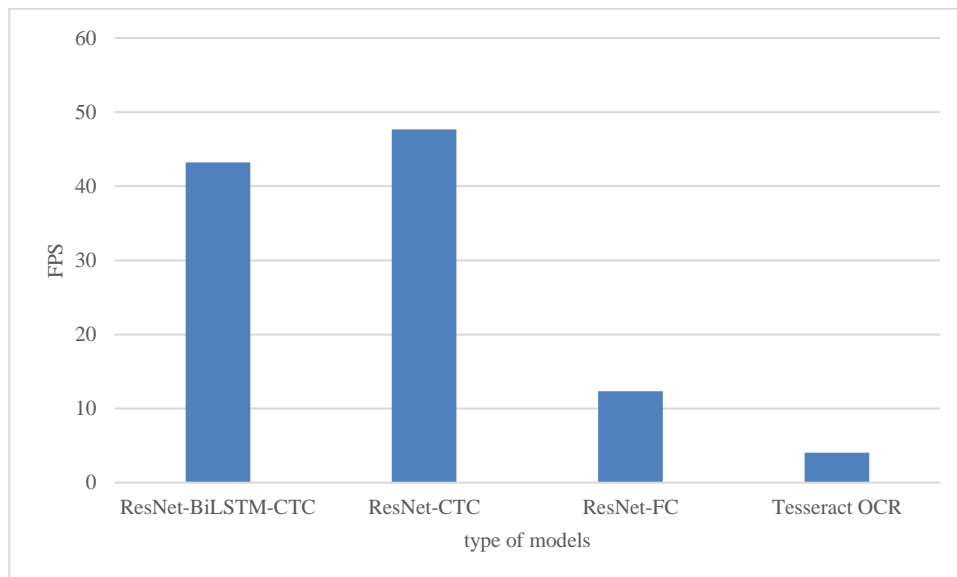


Figure 4.8: Graph of FPS against type of models

Next, Figure 4.9 illustrates the comparison of FPS with and without OpenVINO implementation on these models. With OpenVINO implementation, the models ResNet-BiLSTM-CTC and ResNet-CTC have improved by 3 FPS and 7 FPS respectively. This shows that the implementation of OpenVINO is effective on models with lighter weight.

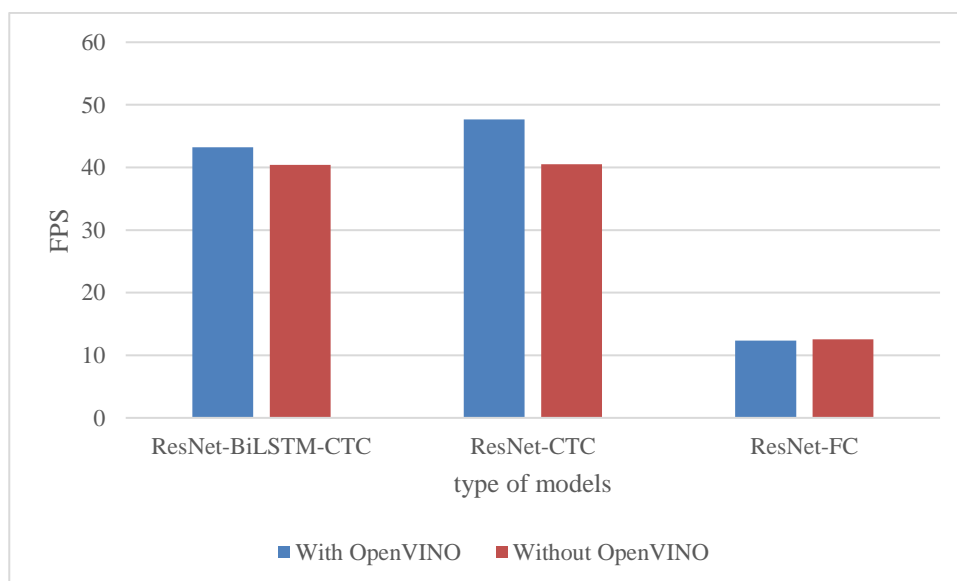


Figure 4.9: Graph of comparison with and without openvino implementation

## 4.8 Performance of OCR Accuracy

With the implementation of the car plate detection model in this evaluation, there is a possibility of missing actual car plates and non-car plates input feed into the OCR models which results in the uneven number of input car plates to each of the models as shown in Table 4.4. Based on the definition of OCR accuracy in chapter 3.4.6, those non-car plate inputs are eliminated in the OCR accuracy calculation. From Table 4.4, the model ResNet-BiLSTM-CTC achieves the highest accuracy while Tesseract OCR has the lowest accuracy.

Table 4.4: OCR accuracy of the models using different dataset

Models	Type of dataset	Total number of input car plate	Number of correct recognises car plate	OCR accuracy, %
ResNet-BiLSTM-CTC	Video dataset	348	278	79.80 %
ResNet-CTC	Video dataset	354	277	78.25 %
ResNet-FC	Video dataset	365	287	78.32 %
Tesseract OCR	Video dataset	260	102	39.23 %

### 4.8.1 Comparisons between models

Based on Table 4.5, the ResNet-FC has an accuracy higher than ResNet-CTC by about 0.07% while the FPS of ResNet-FC is much lower than ResNet-CTC with a FPS value of 35.33. To obtain greater accuracy, ResNet-BiLSTM-CTC has achieved 79.89% accuracy with a drop of 4.45 FPS compared to ResNet-CTC. It can be deduced that ResNet-BiLSTM-CTC is the optimum model while Tesseract OCR shows the worst performance among the rest of the models.

Table 4.5: Comparison of models corresponding to FPS and OCR accuracy

Models	FPS	OCR accuracy, %
ResNet-BiLSTM-CTC	43.22	79.89
ResNet-CTC	47.67	78.25
ResNet-FC	12.34	78.32
Tesseract OCR	4.05	39.23

Figure 4.10 shows ResNet has a structure of non-neighbored layers as chain-like neural networks by implementing the residual connection which has a different convention of stacking layer. The residual connection in ResNet significantly reduce the training difficulty and keeps promising in term of generalization ability (He, Liu and Tao, 2020). By keeping the ResNet as the main frame of the combination model, CTC is lighter than FC which can be observed from FPS. In terms of OCR accuracy, CTC is lower than FC as there is a trade-off for better FPS and accuracy. A fully connected layer is an output feature map of the previous layer and each of the weights is associated with every single input which is well known as a dense layer (Yamashita et al., 2018) that implicitly required a lot of computation time. Whereas CTC is a temporal classifier which has indirect segmentation of input sequence prediction as well as multi-label classification (Wigington, Price and Cohen, 2019). Therefore, FC has higher accuracy but takes longer computational time while CTC has lower accuracy but takes shorter computational time.

BiLSTM is an extension of the LSTM model that has the additional ability to reverse the order of the input to predict input to output and output to input during training as illustrated in Figure 4.11 (Siami-Namini, Tavakoli and Namin, 2019). By comparing ResNet-BiLSTM-CTC and ResNet-CTC, the additional stage which is BiLSTM allows the accuracy can be further improved and it creates alternatives for different combinations which is to achieve the best trade-off between FPS and accuracy. Therefore, ResNet-BiLSTM-CTC is the optimum model for LPR system implementation. Tesseract OCR showed the worst result on both performances as the pre-processing and post-processing are not effective in a real scene at all times which makes the model unable to perform well. For instance, the segmentation of the series of characters which



has insufficient spacing will end up treated as 1 word, especially if the captured car plate has a slanted view.

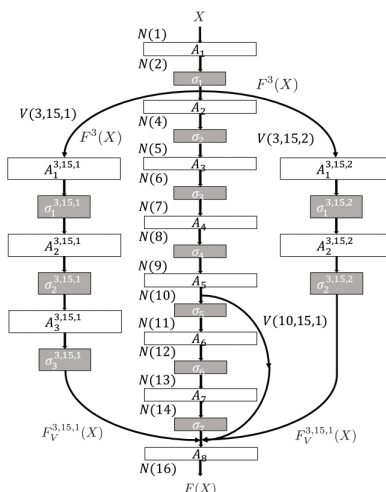


Figure 4.10: Stem-Vine framework of ResNet (He et al., 2020)

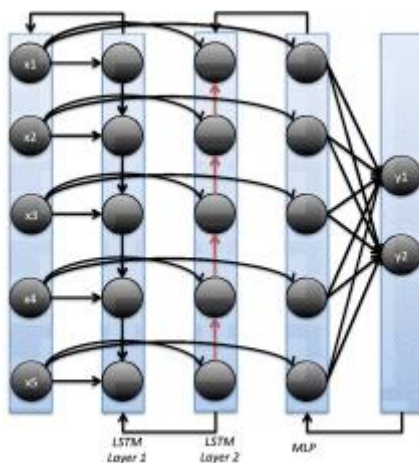


Figure 4.11: Architecture of BiLSTM (Siami-Namini, Tavakoli and Namin, 2019)

Besides, the current Tesseract OCR implements Long Short-Term Memory, LSTM is part of the engines to perform the line detection and Tesseract 3 is used to recognise the character pattern (Anon, 2022g). The Tesseract OCR recognition process utilises a static character classifier, adaptive classifier and linguistic analysis (Smith, n.d.). The static character classifier can distinguish various characters and non-characters while the adaptive classifier uses the isotropic baseline/x-height normalization to differentiate the upper and low cases of the alphabet. Linguistic analysis is used in a new segmentation to

select the best available word string based on several criteria such as Top frequent word, Top dictionary word, Top classifier choice word etcetera. These recognition utilities in Tesseract OCR cause inflexibility and low accuracy of the recognition rate compared to the combination models.

Although the combination models have better performance than Tesseract OCR, the accuracy of the combination models is lower than the models emphasize in the literature review chapter. One of the reasons the causes for the low accuracy of the combination model is the biased training dataset. The car plate in Malaysia consists of 2 type which is single row and double row car plate. In this dataset, they are an extremely uneven number of both type of car plate which result in a single-row car plate being overtrained while a double row is undertrained. Hence, most of the false recognised car plate type is double row. Furthermore, the video dataset has a dynamic scene in which the environmental factors change from one frame to another frame. The sudden change of light intensity from the indoor car park to the outdoor car park causes the car plate cannot to be recognised.

#### **4.9 Summary**

The evaluation uses a video dataset to benchmark the performance of trained models. The combination model ResNet-BiLSTM-CTC has an OCR accuracy of 79.89% which is the highest among the models while ResNet-CTC has an FPS of 47.67 FPS which is the fastest among. However, ResNet-BiLSTM-CTC is the final choice as the optimum model due to it having the best trade-off between accuracy and speed. Hence, the OpenVINO implementation will further improve the FPS. Furthermore, the introduction of GUI with several features is aimed at the feasibility of implementing the model corresponding to the application.

## CHAPTER 5

### CONCLUSIONS AND RECOMMENDATIONS

#### 5.1 Conclusions

The hardware limitation is a significant issue for industries and businesses as computer vision is introduced into the IoT era. OCR as part of computer vision which is used in an integrated system, is a real-time performance required in certain applications especially car park monitoring systems in Malaysia. For the sake of improving the accuracy, the conventional OCR is heavily relying on pre-processing and post-processing which imply a high degree of customisation, less flexibility and the performance is highly affected by environmental factors.

Therefore, an OCR neural network model based on Malaysian car plates was developed to address these issues. The combination models were trained, validated and tested with 1200 images of the car plate dataset, MJSynth and SynthText datasets for the preliminary evaluation before being implemented into the LPR system. The video dataset was used to perform the evaluation, all the combination models achieved at least 78% OCR accuracy while the Tesseract OCR has bad OCR accuracy of about 39.23%. The optimum model was deduced among the combination models by selecting the best trade-off between OCR accuracy and FPS which is ResNet-BiLSTM-CTC. This model is fast enough to process 43.22 frames within a second. Furthermore, OpenVINO is a good optimizer for lightweight models which has increased the FPS of ResNet-CTC and ResNet-BiLSTM-CTC.

In addition, the GUI is developed to ease the user to monitor the model and provide interactive components such as a delete button, data entry and navigation button. Through the GUI, users can access the databases for further monitoring purposes without knowing how the backend of the system works. Lastly, all the objectives have been successfully achieved.

## 5.2 Recommendations

There is a limited Malaysian car plate dataset in the project which is not sufficient for the model to further increase the performance of OCR accuracy. The process of collecting and labelling huge data is tiring, costly and time-consuming. Within the constraints of this project, the data augmentation technique is the sustainable choice to increase the amount of training, testing and validating images as well as to overcome the issue of overfitting and deploy a more robust model.

Besides, there are two general types of Malaysian car plates which are single row and double row car plates. The current car plate dataset has an imbalance ratio of both types. As a result, it is part of the root cause of low OCR accuracy. Hence, the collected dataset should have a ratio close to 1:1 on both types.

Last but not least, the current web-based GUI is hosted within the local machine. To monitor the GUI through the internet, a Python socket module is recommended. This allows the user able to access the GUI through the Uniform Resource Locator, URL which has a similar approach to IoT.

## REFERENCES

Agbeyangi, A.O., Alashiri, O.A. and Otunuga, A.E., 2020. Automatic Identification of Vehicle Plate Number using Raspberry Pi. *2020 International Conference in Mathematics, Computer Engineering and Computer Science, ICMCECS 2020*. <https://doi.org/10.1109/ICMCECS47690.2020.246983>.

Alyahya, H.M., Alharthi, M.K., Alattas, A.M. and Thayananthan, V., 2017. Saudi License Plate Recognition System Using Artificial Neural Network Classifier. *2017 International Conference on Computer and Applications, ICCA 2017*, pp.220–226. <https://doi.org/10.1109/COMAPP.2017.8079759>.

Anon. 2022a. *Parking Revenue Control Systems, Parking System Solutions, On-street Parking Manufacturers and Suppliers China - Factory Price - Wiicontrol*. [online] Available at: <<https://www.wiiparking.com/>> [Accessed 2 September 2022].

Anon, 2022b. *Intel NUC 10 Performance NUC10i7FNH Barebone System Mini PC*. [image online] Available at: < <https://www.amazon.com/Intel-Performance-NUC10i7FNH-Barebone-System/dp/B083SJCJSV>> [Accessed 10 March 2022].

Anon, 2022c. *Beelink BT3 Pro EU Plug Windows 4GB +64GB Mini PC Sale, Price & Reviews/ Gearbest*. [image online] Available at: < [https://www.gearbest.com/tv-box-mini-pc/pp\\_635379.html](https://www.gearbest.com/tv-box-mini-pc/pp_635379.html)> [Accessed 10 March 2022].

Anon, 2022d. *The Python Logo*. [image online] Available at: <<https://www.python.org/community/logos/>> [Accessed 10 March 2022].

Anon, 2022e. *Media Kit*. [image online] Available at: <<https://opencv.org/resources/media-kit/>> [Accessed 10 March 2022].

Anon, 2022f. *Welcome to Flask*. [image online] Available at: <<https://flask.palletsprojects.com/en/2.0.x/>> [Accessed 2 March 2022].

Anon. 2022g. *GitHub - tesseract-ocr/tesseract: Tesseract Open Source OCR Engine (main repository)*. [online] Available at: <<https://github.com/tesseract-ocr/tesseract>> [Accessed 15 August 2022].

Anusuya, R. and Joseph, W., 2020. Deep Learning Based Ethiopian Car's License Plate Detection and Recognition. *International Journal of Recent Technology and Engineering*, 8(6), pp.5730–5737. <https://doi.org/10.35940/ijrte.f9857.038620>.

Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J. and L, H., 2019. *What is Wrong With Scene Text Recognition Model Comparisons? Dataset and*

*Model Analysis*. [image online] Available at: < <https://github.com/clovaai/deep-text-recognition-benchmark#download-lmdb-dataset-for-training-and-evaluation-from-here>> [Accessed 10 March 2022].

Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J. and L, H., 2019. *What is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis*. [online] Available at: <<https://github.com/clovaai/>> [Accessed 13 March 2022].

Bochkovskiy, A., Wang, C.Y. and Mark Liao, H.Y., 2020. *YOLOv4: Optimal Speed and Accuracy of Object Detection*. [online] Available at: < [https://www.researchgate.net/publication/340883401\\_YOLOv4\\_Optimal\\_Speed\\_and\\_Accuracy\\_of\\_Object\\_Detection](https://www.researchgate.net/publication/340883401_YOLOv4_Optimal_Speed_and_Accuracy_of_Object_Detection)> [Accessed 4 March 2022]

Chips, Y., 2019. *Sunway to introduce Smart Parking system with Licence Plate Recognition in early 2020 - News and reviews on Malaysian cars, motorcycles and automotive lifestyle*. [online] Available at: <<https://www.piston.my/2019/10/30/sunway-to-introduce-smart-parking-system-with-licence-plate-recognition-in-early-2020/>> [Accessed 16 February 2022].

GeeksforGeeks., n.d. *Object Detection vs Object Recognition vs Image Segmentation - GeeksforGeeks*. [online] Available at: <<https://www.geeksforgeeks.org/object-detection-vs-object-recognition-vs-image-segmentation/>> [Accessed 16 February 2022].

He, F., Liu, T. and Tao, D., 2020. Why ResNet Works? Residuals Generalize. *IEEE Transactions on Neural Networks and Learning Systems*, 31(12), pp.5349–5362. <https://doi.org/10.1109/TNNLS.2020.2966319>.

He, F., Liu, T. and Tao, D., 2020. *Deep neural network with residual connections under the stem-vine framework*. [online image] Available at: < <https://doi.org/10.1109/TNNLS.2020.2966319>> [Accessed 2 September 2022].

Ibitoye, O., Ejidokun, T., Omitola, O. and Dada, O., n.d. Convolutional Neural Network-Based License Plate Recognition Techniques: A Short Overview. <https://doi.org/10.1109/CSCI51800.2020.00283>.

IBM Cloud Education, 2020. *Neural Networks*. [online] Available at : <<https://www.ibm.com/cloud/learn/neural-networks>> [Access 15 February 2022].

Intel., n.d. *Intel® NUC 10 Performance kit - NUC10i7FNH*. [online] Available at : < <https://ark.intel.com/content/www/us/en/ark/products/188811/intel-nuc-10-performance-kit-nuc10i7fnh.html>> [Access 26 February 2022].

IPSEC., n.d. *License Plate Recognition - IPSEC ENGINEERING SDN BHD*. [online] Available at: <<https://www.ipsec.com.my/products/security/license-plate-recognition/>> [Accessed 16 February 2022].

Kessentini, Y., Besbes, M.D., Ammar, S. and Chabbouh, A., 2019. A two-stage deep neural network for multi-norm license plate detection and recognition. *Expert Systems with Applications*, 136, pp.159–170. <https://doi.org/10.1016/J.ESWA.2019.06.036>.

Li, H., Wang, P., You, M. and Shen, C., 2018. Reading car license plates using deep neural networks. *Image and Vision Computing*, [online] 72, pp.14–23. <https://doi.org/10.1016/j.imavis.2018.02.002>.

Ligade, M., 2020. *Introduction of Intel OpenVINO Toolkit*. [image online] Available at: <<https://medium.com/techwasti/introduction-of-intel-openvino-toolkit-bc235a9c5b6d>> [Accessed 10 March 2022]

Manage4U, n.d.. M4U Smart License Plate Recognition. Available through: Kuala Lumpur in Malaysia, *Manage4U Sdn. Bhd.* [online] Available at: <https://www.manage4u.com.my/lpr> [Accessed 12 February 2022].

MathWorks., n.d. *What Is Object Detection? - MATLAB & Simulink*. [online] Available at: <<https://www.mathworks.com/discovery/object-detection.html>> [Accessed 16 February 2022].

McCarthy, J., 2007. *What is AI? / Basic Questions*. [online] Available at: <<http://jmc.stanford.edu/artificial-intelligence/what-is-ai/index.html>> [Accessed 15 February 2022].

Montazzolli Silva, S. and Rosito Jung, C., n.d. Real-time license plate detection and recognition using deep convolutional neural networks q. [online] <https://doi.org/10.1016/j.jvcir.2020.102773>.

Omran, S.S. and Jarallah, J.A., 2017. Iraqi car license plate recognition using OCR. *2017 Annual Conference on New Trends in Information and Communications Technology Applications, NTICT 2017*, pp.298–303. <https://doi.org/10.1109/NTICT.2017.7976127>.

Othman, K., Sheroz, K., Rafiqul, I. and Ahmad, S., 2007. *Malaysia Vehicle License Plate Recognition*. [online] Available at: <<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.443.9961&rep=rep1&type=pdf>> [Accessed 13 February 2022].

Rayson, L., Luiz, A, Z., Gabriel, R, G., Eduardo, T., William, R, S. and David, M., 2019. *An Efficient and Layout-Independent Automatic License Plate Recognition System Based on the YOLO detector*. [online] Available at: <<https://arxiv.org/pdf/1909.01754v4.pdf>> [Accessed 16 April 2022].

Richard, B., n.d. *How-Server-Work*. [image online] Available at: <<https://www.hostinger.com/tutorials/what-is-apache>> [Accessed 2 March 2022].

Siarni-Namini, S., Tavakoli, N. and Namin, A.S., 2019. *BiLSTM*. [online image] Available at: <<https://doi.org/10.1109/BIGDATA47090.2019.9005997>> [Accessed 2 September 2022].

Siarni-Namini, S., Tavakoli, N. and Namin, A.S., 2019. The Performance of LSTM and BiLSTM in Forecasting Time Series. *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, pp.3285–3292. <https://doi.org/10.1109/BIGDATA47090.2019.9005997>.

Saunshi, S., Sahani, V., Patil, J., Yadav, A. and Rathi, S., n.d. *License Plate Recognition Using Convolutional Neural Network*. [online] Available at: <<https://www.iosrjournals.org/iosr-jce/papers/Conf.17014-2017/Volume-1/6.%2028-33.pdf?id=755>> [Accessed 15 August 2022].

Smith, R., n.d. *An Overview of the Tesseract OCR Engine*. [online] Available at: <<https://ieeexplore.ieee.org/abstract/document/4376991>> [Accessed 15 August 2022].

Tan, J., Y., 2019. *kiplePark makes parking frictionless with Malaysia's first license plate recognition system | Digital News Asia*. [online] Available at: <<https://www.digitalnewsasia.com/business/kiplepark-makes-parking-frictionless-malysias-first-licence-plate-recognition-system>> [Accessed 15 February 2022].

Tham, M.-L. and Tan, W.K., 2021. IoT Based License Plate Recognition System Using Deep Learning and OpenVINO. *2021 4th International Conference on Sensors, Signal and Image Processing*, [online] pp.7–14. <https://doi.org/10.1145/3502814.3502816>.

Varad Vinay, K., Omkar Balaobaiiah, I., Sohail Mahiboob, M. and Dinesh Nagnath, S., 2021. Automatic Number Plate Recognition For Different Fonts and Non-Roman Script. *International Journal of Engineering Applied Sciences and Technology*, [online] 5(11), pp.314–317. Available at: <<http://www.ijeast.com>> [Accessed 15 February 2022].

Wigington, C., Price, B. and Cohen, S., 2019. Multi-label connectionist temporal classification. *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pp.979–986. <https://doi.org/10.1109/ICDAR.2019.00161>.

Yamashita, R., Nishio, M., Do, R.K.G. and Togashi, K., 2018. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, [online] 9(4), pp.611–629. <https://doi.org/10.1007/S13244-018-0639-9/FIGURES/15>.

Yang, C.S. and Hsieh, C.C., 2019. High Accuracy Text Detection using ResNet as Feature Extractor. *2019 IEEE Eurasia Conference on IOT, Communication and Engineering, ECICE 2019*, pp.92–95. <https://doi.org/10.1109/ECICE4748.2019.8942666>.



Yousri, K., Mohamed, D.B., Sourour, A. and Achraf, C., 2019. *A two-stage deep neural network for multi-norm license plate detection and recognition*. [online] Available at: <https://www-sciencedirect-com.libezp2.utar.edu.my/science/article/pii/S0957417419304361#bib0007> [Accessed 2 February 2022].

Zheng, L., He, X., Samali, B. and Yang, L.T., 2013. An algorithm for accuracy enhancement of license plate recognition. *Journal of Computer and System Sciences*, 79(2), pp.245–255. <https://doi.org/10.1016/J.JCSS.2012.05.006>.

ZKTeco., n.d. *ZKTeco Malaysia*. [online] Available at: <https://www.zkteco.com.my/Advanced-LPR-Solution-with-Vehicle-Recognition> [Accessed 16 February 2022].

## APPENDICES