CRITICAL ANALYSIS OF DEEP NEURAL NETWORK ON MALAYSIA LICENSE PLATE RECOGNITION

LEE WEI XIANG

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

DESIGN OF AN AUTOMATED ILLUMINATED EMERGENCY EXIT DOOR SYSTEM

LEE WEI XIANG

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

Faculty of Electronic and Computer Engineering
Universiti Teknikal Malaysia Melaka

2021

DECLARATION

I declare that this report entitled "Critical analysis of deep neural network on Malaysia license plate recognition" is the result of my own work except for quotes as cited in the references.

Signature :

Author : LEE WEI XIANG

Date : 23 JUNE 2021

APPROVAL

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

Signature : ……………………………………

Supervisor Name : LIM KIM CHUAN

Date : 24 JUNE 2021

DEDICATION

Special dedication to my senior, Tan Kien Leong and Tay Choon Kiat; my supervisor, Prof Madya Dr Lim Kim Chuan and to anyone who has helped me directly or indirectly in the process of producing this thesis.

ABSTRACT

Nowadays, traffic congestion in Malaysia is getting serious. To optimize the traffic flow, creating an automatic license plate recognition system that can collect the traffic flow parameter will be an ideal solution. To create a good automatic license plate recognition model, the effect of variety prefix on neural network needs to be study as there is variety prefix in Malaysia's license plate. Hence, the aim of the project is to analyse the deep neural network on the variety of prefix in Malaysia license plate recognition. To obtain the results, three Malaysia license plate recognition model was train with three different datasets, together with three different distributions of prefix of Malaysia license plate. Besides that, a license plate detector (LPD) was also being train to automate the process of cropping the license plate from the source image captured from overhead poles camera at traffic light junctions. The LPD was trained and achieved 75.8% mAP and 99.8% accuracy in automatic detecting the license plate given a vehicle image. All the LPR models (Model A, B, and C) manage to have 79.58%, 81.25% and 80.42% in mean test full sequence accuracy after training.

ABSTRAK

Pada masa kini, kesesakan lalu lintas di Malaysia semakin serius. Untuk mengoptimumkan aliran lalu lintas, sistem pengenalan plat kenderaan automatik yang dapat mengumpulkan parameter aliran lalu lintas akan menjadi penyelesaian yang ideal. Untuk menyediakan model pengenalan plat lesen automatik yang baik, kesan awalan pelbagai pada rangkaian saraf perlu dikaji kerana terdapat pelbagai awalan dalam plat nombor Malaysia. Oleh itu, tujuan projek ini adalah untuk menganalisis kesan pelbagai awalan dalam plat nombor Malaysia kepada neural mendalam dalam pegenalan. Untuk mendapatkan hasilnya, tiga model pengenalan plat nombor Malaysia dilatih dengan tiga set data yang berbeza, bersama dengan tiga taburan awalan plat nombor Malaysia yang berbeza. Selain itu, pengesan plat nombor juga dilatih untuk mengotomatisasi proses memotong plat nombor dari gambar sumber yang diambil dari kamera di persimpangan lampu isyarat. Pengesan plat nombor dilatih untuk memiliki 75.8% peta dan ketepatan 99.8% dalam mengesan plat nombor gambar kenderaan terpotong. Semua model pengenalan plat kenderaan (Model A, B, dan C) berjaya memperoleh 79.58%, 81.25% dan 80.42% dalam ujian ketepatan urutan penuh selepas dilatih.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Prof Madya Dr Soo Yew Guan and Prof Madya Dr Lim Kim Chuan, they gave a lot of advice throughout the process of training of the model, guided me in doing the analysis of model and suggested ways to improve it. I really appreciate that I was given the opportunity to do this project. Besides my supervisor, I would also like to pay gratitude to my senior Mr. Tan Kien Leong and Mr. Tay Choon Kiat, they spent alot of time giving me knowledge inputs when I have trouble understanding the neural network. Finally, my sincere appreciation to Led Vision Sdn Bhd for providing pre-trained vehicle detector and raw dataset for the project. Without these supports, the project will not be able to complete in time.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

CNN	:	Convolutional Neural Network
RNN	:	Recurrent Neural Network
CRNN	:	Convolutional Recurrent Neural Network
mAP	:	Mean Average Precision
LPD	:	License Plate Detector
LPR	:	License Plate Recognition

CHAPTER 1

INTRODUCTION

This project focus on the analysis of 3 trained model in recognizing Malaysia vehicle license plate and train a license plate detector to crop out the license plate in a cropped vehicle image. This chapter will talk about the project background, problem statement, objectives, scope of work and thesis outline.

1.1 Project Background

License plate of a car is same as the identity card of every people, it is unique for every vehicle therefore it can be used at parking management system, toll management and highway monitoring where tracking cars with different identity is needed. The license plate recognition will usually be implemented by using camera and smart model, the smart model is being created by using artificial neural network which is similar to the thinking method and learning method of human being.

By referring to road transport department Malaysia [1], 3 different form of Malaysia license plate can be found in Malaysia. First, white letter and number are affixed or bounce over black frame, second, white letters and numbers are affixed or embossed over red frames for embassy vehicles, third, black letters and numbers are bounced over white frame for taxi and rental car. Besides that, the preferred font used for License Plate is Arial Bold, Calisto MT Italic and Franklin Gothic Bold, the spacing ~~of~~ between border and font will be following the standard stated by JPJ as shown in Figure 1.1.

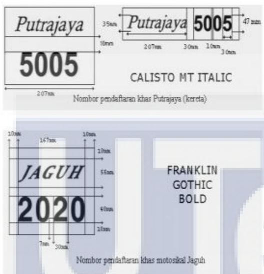


Figure 1.1: JPJ's standard format for Malaysia license plate

Figure 1.2: License Plate of Malaysia with white letter and number are affixed or bounce over black frame

Figure 1.3: License Plate of Malaysia with white letters and numbers are affixed or embossed over red frames for embassy vehicles

Figure 1.4: License Plate of Malaysia with white letters and numbers are affixed or embossed over red frames

In addition, the Malaysia license plate will always begin with one or more alphabet letter. The alphabet in front of the license plate represent the state of the license plate registered, all the alphabets will be used to form the license plate except O, I and Z to prevent confusion in recognition. Table 2 will be formed by using example of license plate which is "**A**BC 1234" and the alphabet letter bolded is the key alphabet to represent the state of license plate registered. The example of this format of license plate is as shown as Figure 1.2.

Table 1.1: License Plate Example by Registered States

Example of License Plate	State Registered
ABC 1234	Perak
BBC 1234	Selangor
CBC 1234	Pahang
DBC 1234	Kelantan
FBC 1234	Putrajaya
JBC 1234	Johor
KBC 1234	Kedah
MBC 1234	Melaka
NBC 1234	Negeri Sembilan
PBC 1234	Penang
QBC 1234	Sarawak
RBC 1234	Perlis
SA 1234 B	Sabah
TBC 1234	Terengganu
LB 1234	Labuan
VBC 1234**, W**BC 1234 **W**A 1234 B	Kuala Lumpur

For taxi and rental car, its license plate is slightly different with the example stated in table 2, the license plate for them will always start with alphabet "H" to represent their identity of taxi and rental car. The second or third prefix will be the key prefix to represent the registered state. The example of this format of license plate is as shown as Figure 1.3.

Table 1.2: License Plate Example by Registered States for Taxi and Rental Cars

Example of License Plate	State Registered
HAC 1234	Perak
HBC 1234 **HB**C 1234 SA	Selangor
HCC 1234	Pahang
HDC 1234	Kelantan
HJC 1234	Johor
HKC 1234	Kedah
HMC 1234	Melaka
HNC 1234	Negeri Sembilan
HPC 1234	Penang
HQC 1234	Sarawak
HRC 1234	Perlis
HSA 1234	Sabah
HTC 1234	Terengganu
HLB 1234	Labuan
HWC 1234	Kuala Lumpur

License plate for embassy vehicles have different pattern compared to the license plate above, it is using 1C-2C-AB format [2]. 1C is first code, 2C is second code and AB is constant suffix. The example of this format of license plate is as shown as Figure 1.4.

Table 1.3: Foreign Mission Format

Foreign Mission	**1C**	**2C**	**AB**
Diplomatic Corps	Nationality	Rank	DC
Consular Corps	Nationality	Rank	CC
United Nations	Organisation	Rank	UN

Development of an automatic license plate recognition system will divide into 3 stages, which are license plate detection, license plate segmentation and license plate recognition [3]. License plate detection is the first stage in the system which is very important and requires high accuracy because false detection of the first stage will affect inappropriate output of the following stages. License plate detection is to localize the license plate of vehicle in the given input image, factors such as congested traffic with multiple plates, ambiguous signs, and advertisements, tilting plates, as well as obscure images taken in bad weather and nighttime [4], all these environmental factors is the first challenge face in the LPR. The method ~~is~~ proposed to solve this environmental factor is to use a vehicle detector to crop the vehicle in the image to reduce the false detection of license plate[4].

Figure 1.5: Result of license plate detection which affected by environmental factor [4]

For character segmentation, it is a process to localize the text in the image. According to “An efficient license plate recognition system using convolution neural networks” [4], the method used for segmentation is first gray scale and binarize the detected license plate to eleminate noise, then a horizontal projection to determine the position of character in the image, finally a vertical projection to divide the character into single character. After the segmentation is done, a convolutional neural network was used to detect the single character extract from the segmentation where the output layer containe 34 neuron to recognize the correspond 34 character, 0 to 9 , A to Z exclue “I” and “O” [4].

Figure 1.6: Eliminate the upper and lower boundary of the detected license plate [4]

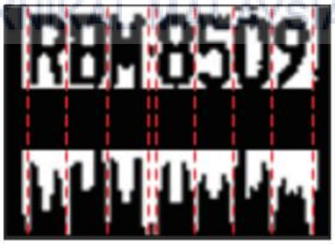


Figure 1.7: Separate character into single character [4]

Besides that, a text recognition without segmentation was also proposed in “An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition” [5]. In real world, most of the text was appear in a sequence such as license plate, house number and score, which tend to appear in a sequence form therefore the ALPR system need a segmentation method to identify single character for text recognition. The principle of segmentation free method proposed is to recognize the sequence like object, using a neural model with DCNN to extract the feature of image then predict sequence of text in the image by RNN. Another reason of this method being proposed is due to some of the text recognition system used text detector to localize the position of text in the image and the text detector will require a huge amount of dataset for training which will make the text recognition system being difficult to complete [5].

Besides that, based on “Segmentation-free Handwritten Chinese Text Recognition with LSTM-RNN” [6], it states that in 2015, most of the text recognition competition was being won by system using Multi Directional Long Short-Term Memory Recurrent Neural Network [6], it is using a segmentation free method. It also states that, segment each line into character is quite costly and increase the chance of errors [6], because the text is in sequence and need use an algorithm to detect the position of word for segmentation, this will increase the time in processing and the error in segmentation will cost the error in recognition. The statement above, stated that, segmentation free method with RNN is better method in recognize sequence like object (license plate) compared to segmented based method.

1.2 Problem Statement

Nowadays in Malaysia, traffic jam become an often problem occur in our daily life, this is because of the concentration of population in economic potential area. The amount of vehicle in Malaysia is around 31.2 million [7] which is near to our population which is 32.7 million [8], for those cities in Malaysia like Subang Jaya, Kuala Lumpur, Klang and Johor Bahru which have concentrate population, the problem of traffic congestion is even serious compared to others place. Hence, automatic license plate recognition system is needed in Malaysia to collect the traffic flow parameter for optimize traffic control [9], where license plate recognition with segmentation-free is a suitable method in real time application but the effect of variety prefix in Malaysia license plate to this method have not being study.

From the source image supply LED VISON Sdn Bhd, it is a cropped vehicle image, hence, to conduct the study on Automatic Malaysia License Plate recognition system, a license plated detector is needed to automate the process.

1.3 Objectives

- To critically investigate the recurrent neural network used in segmentation free Malaysia license plate recognition system to provide a guide in development of Malaysia license plate recognition system.
- To develop a license plate detector to automate the process for crop out the license plate in the cropped vehicle image.

1.4 Scope of project

The scope of this project is to study on trained license plate recognition models with segmentation free method by using dataset get from the 3 junction overhead poles which installed with camera. The processor of a computer will be use as the processing unit to run the training and testing of the license plate recognition model after train. The license plate model will be train by using TensorFlow as the framework, TensorFlow is an end-to-end open-source platform for machine learning. A CRNN model was used to develop 3 of the LPR model and the programming language is Python. Then, the trained license plate recognition model will cooperate with trained license plate detector model to run the accuracy test with random input from the 3 junctions. In addition, the LPR model will be train by using 3 different of dataset, first dataset with a ratio same as the collected dataset (Usual ratio of collected license plate), the second dataset is a balance dataset with a balanced amount of prefix, the third dataset will contain only 1 prefix. All the datasets will have the same amount of image to achieve a fair condition for comparing. In addition, a license plate detector was being train using SSD mobilenet v3 as the model, TensorFlow as the framework and develop using Python programming language. The LPD needs to be trained to have mAP 75% for automate the process of cropping.

1.5 Project Significance

The project gives significance for those who want to train a Malaysia license plate recognition system a good guidance in collecting the respective dataset for license plate recognition training and guide in license plate recognition training. In addition, all the datasets collect is collect in the Malaysia junction and all testing and training is conduct with real license plate hence the reliability is very high. Besides that, a guide for training Malaysia license plate detector will also provide.

CHAPTER 2

BACKGROUND STUDY

2.1 General Theory

2.1.1 Artificial Neural Network

Many tasks involving intelligence or pattern recognition are extremely difficult to automate but appear to be performed very easily by humans [10]. Humans receive a lot of visual information from surrounding therefore it is easier for a human to execute some tasks like calculating how many humans is moving, what is the number of the house lot or where is the cat inside the image. Therefore, artificial neural network was being introduced to perform task with a similar way like human brain.

Human brain is made up of a network of neurons that are coupled with receptors and effectors [10]. Dendrite of neuron will collect signal from others neuron in a limited area, example like a fingertips, tongue, or eyes, then cell body will process the input signal and decide whether a neuron should send an output signal, myelinated axon is to transmit the processed signal to axon terminal and axon terminal is to connect with others neuron dendrites for transmitting data.

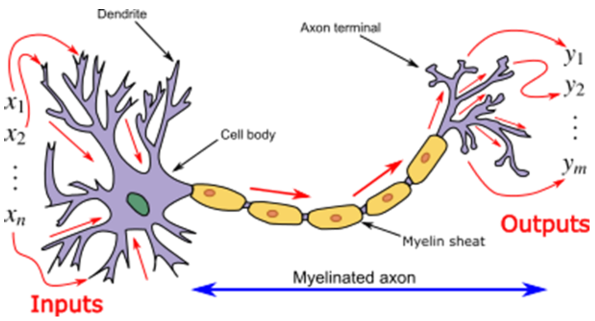


Figure 2.1: Biological Neuron (Image retrieved from https://en.wikipedia.org/wiki/Biological_neuron_model#/media/File:Neuron3.png)

Artificial neural networks are mainly comprised of a high number of interconnected computational nodes (referred to as neurons), of which work entwine in a distributed fashion to collectively learn from the input to optimize its final output [11]. The interconnected computational nodes can be known as perceptron, it can be divided into 4 part, first part is the input layer which use to receive input, second part is weight and bias, this part will be execute by using the equation shown in Figure 2.3, C is the final output value, x1 and x2 is the input value, w1 and w2 is the weight and b is bias value, third part is activation function which decided which output should be send and the last part is output later which pass the final output value to the others neuron in the network or take it as final output value. In summarize, perceptron is a mathematical model that receive the input from input layer, then weighs them separately and sums them up and pass the sum through to a nonlinear function to produce output.

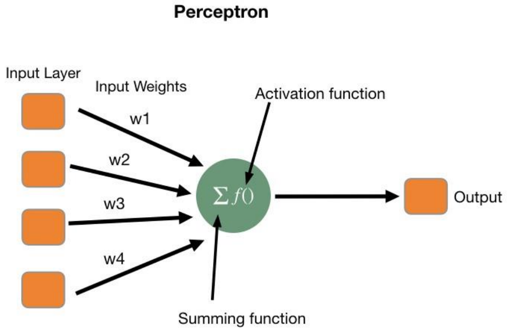


Figure 2.2: Artificial Neuron (Image retrieved from https://towardsdatascience.com/what-is-the-relation-between-artificial-and-biological-neuron-18b05831036)

$$C = w1 * x1 + w2 * x2 + b$$

Figure 2.3: Equation for weight and bias (Image retrieved from https://towardsdatascience.com/what-is-the-relation-between-artificial-and-biological-neuron-18b05831036)

2.1.1.1 Convolutional Neural Network

Convolutional neural networks are analogous to traditional ANNs in that they are comprised of neurons that self-optimise through learning [11]. CNN will still receive input and perform as the formula shown in Figure 2.3, then pass the output value to the activation function to decide for which outputs should be send. The different between is CNN and traditional ANN is that CNN is primarily used in the field of pattern recognition within images [11].

CNN allow encode image-specific features into the architecture, making the network more suited for image-focused tasks [11]. CNN is made up of input layer, convolutional layer, pooling layer, and fully connected layer. Input layers hold the pixel of the input image as same as traditional ANN. Convolutional layer play an important role in CNN, the input data will be stored in an array form and inside this layer will have filter or smaller array to perform multiplication with the input data as shown in Figure 2.5, from the figure it show that, the input image was being pooled into a 3x3 vector then the scalar product will calculated for each value in the kernel and place the answer in the destination pixel. This filter or kernel can lower down the connection of perceptron with input image data. For example, an input colour image with 64x64 pixel, will need to have 64x64x3 connection because of each pixel have 3 bytes for RGB colour then the required connection will have 12888, if convolutional layer is applying and the kernel was being set as 6x6 with a depth of 3 (depth 3 is for colour image regard to RGB bytes), the connection only needs to have 108 hence the computing power can be used more efficient and the model can be train effectively too.

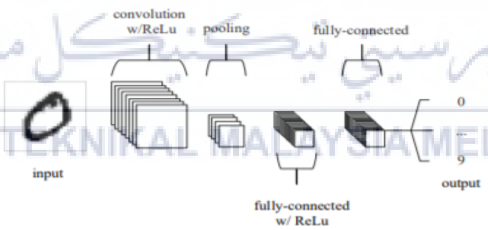


Figure 2.4: Architecture of CNN [11]

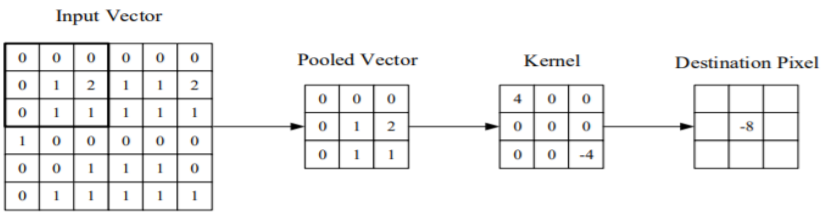


Figure 2.5: Operation in Convolutional Layer [11]

`After the convolution layer, activation maps were being produced each activation maps are regarding to1 features extract from the input image, by refer to Figure 2.6 there is 6 activation maps after the input image go through the convolution layer, means that there are 6 type of kernel with same size to extract 6 different type of feature [12].

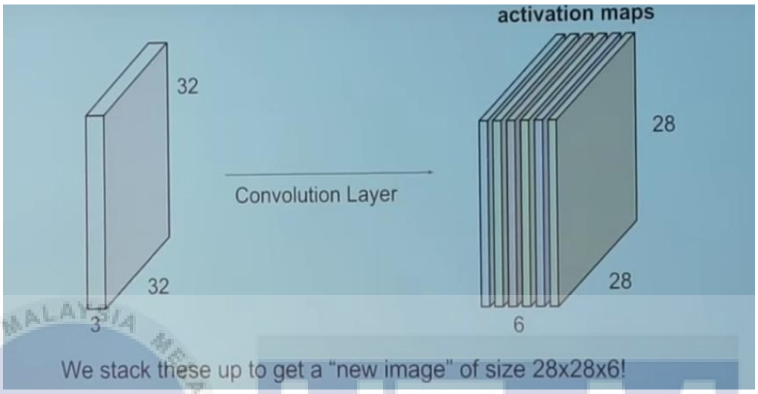


Figure 2.6: Activation Maps After Convolution Layers (Image retrieved from https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8)

For the pooling layers, gradually reduce the dimensionality of the representation, and thus further reduce the number of parameters and the computational complexity of the model [11]. After the features maps being produced, it will be send to the pooling layer, in the pooling layer, there are filter to filter the features map one by one. There are few types of pooling techniques which is average pooling, max pooling, and min pooling, among these 3 types of pooling max pooling is the most common used techniques. Figure 2.7 show the filter with 2x2 filter with maximum pooling operation, the 2x2 filter first take the highest number inside the red part (2x2 size) of the 4x4 input data, then it results 6 inside the 2x2 output data, the stride 2 means that the filter will move 2 pixels once it move for the next filter [13].

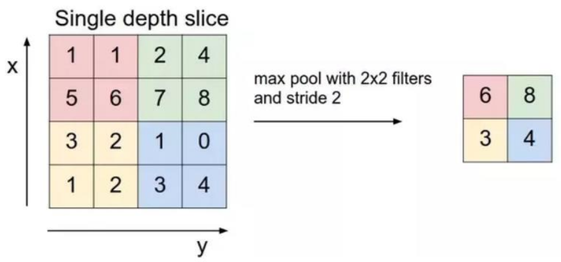


Figure 2.7: Operation of Pooling Layer [13]

In the fully connected layer, the features map will be converted to matrix and become the input of the fully connected network, this fully connected network is working as same as the traditional neural network. Refer to Figure 2.8, each features map will become one of the inputs x1, x2, x3, x4 then go through the neural network will the weight and bias calculation and produced the output [13].

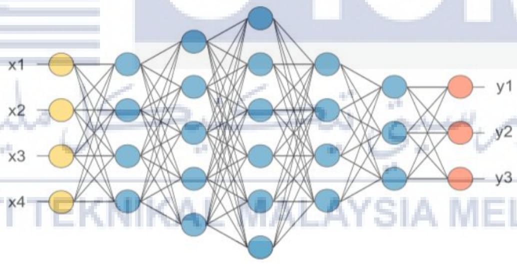


Figure 2.8: Fully connected layer [13]

2.1.1.2 Recurrent Neural Network

Recurrent neural network is invented because the convolutional neural network needs to have a fixed size of input and output which is not suitable to use in predict sequential data [14] like machine translation and sentiment analysis hence recurrent neural network was invented with variable length of input and output data [15].

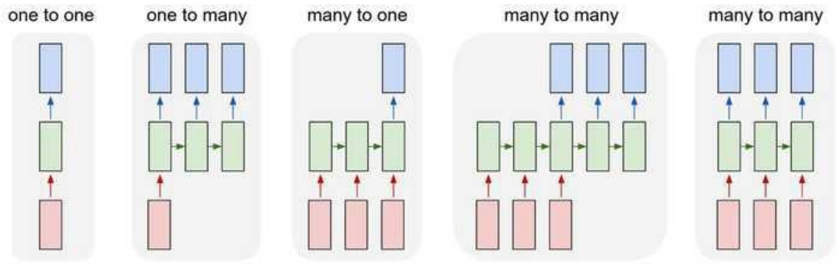


Figure 2.9: Example of recurrent neural network [15]

In the Figure 2.9 show four type of RNN architecture, which is one to one, one to many, many to one and many to many, one to one RNN usually used general machine learning problem which have 1 input and output, one to many RNN usually used in image captioning which 1 image is given and the output is a caption, for many to one RNN, sentiment analysis is one of the example which it require a sequence data like sentence, then the RNN will provide an output to predict it is positive sentiment or negative, many to many RNN normally used in language translation, which receive a sentence of word in Chinese then the RNN translate and give a sentence with the same meaning in English [16].

The working principle of recurrent neural network is it will save the output of a particular layer and feedback to the next input to predict the final output as shown as Figure 2.10 [16]. As refer to the figure 2.10, h(t-1) will feedback to h(t) to predict the output y(t) where h(t) will also feedback to h(t+1) to predict y(t+1) therefore it shows that the old state, h(t-1) at the previous time will always feedback to the time after it, which is the current state, h(t) to improve the current prediction.

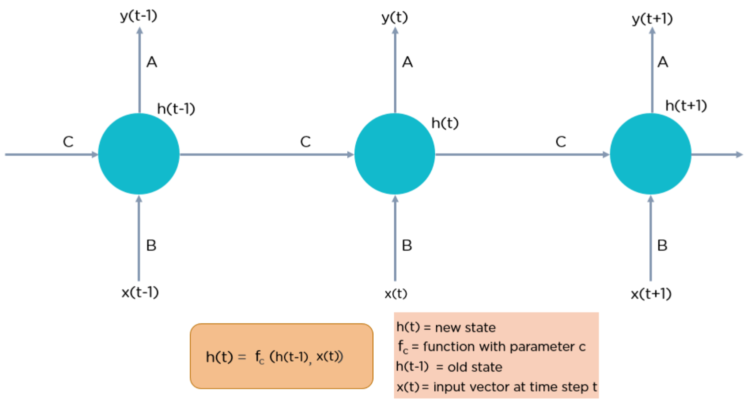
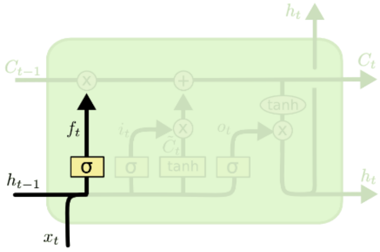


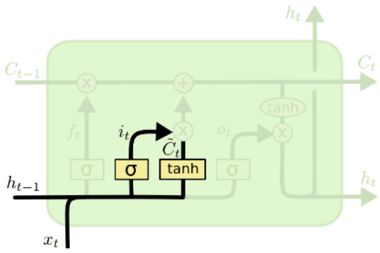
Figure 2.10: Sample structure of recurrent neural network [16]

Due to the vanishing gradient problem in a usual RNN which will cause the network bias to the nearer time data sequence, hence a long short-term memory network is introduced to solve the problem. LSTM work in four steps, first step is to decide how much pass data should be store, by refer to figure 2.11, the sigmoid function will used to determine the pass data, h_{t-1} based on the current x_t. The weight of the h_{t-1} and x_t will be the parameter to adjust how much the pass and current data should be keep when the data went through the forget gate, f_t. Second step is to decide how much the pass and current data to be add into the current state. The first part of second step is the sigmoid function called input gate layer will determine how much the pass data, h_{t-1} and current data, x(t) should update to get the input, i_t then the tanh function will give the weightage to both of data between -1 to 1 to get the candidate value, $\hat{C}_t$. Third step is to update the old cell state, C_{t-1} to new cell state, C_t. The new cell state will update based on the equation in Figure 2.13. The last step is to decide the output, first the output from the sigmoid will be calculate then the current cell state will be filter by a tanh function and multiply with the output from sigmoid gate to get the final output, h_t.



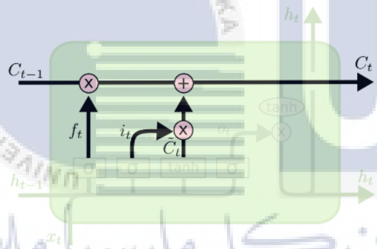
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Figure 2.11: First step of working of LSTM [17]



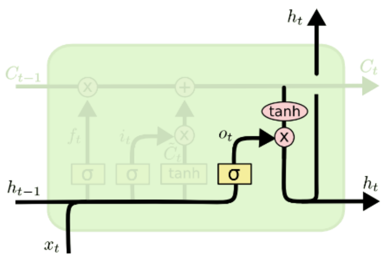
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Figure 2.12: Second step of working of LSTM [17]



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 2.13: Third step of working of LSTM [17]



$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 2.14: Last step of working of LSTM [17]

2.1.2 Neural learning

As Herbert Simon says, Machine Learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population more efficiently and more effectively the next time [14]. After the model go through the neural learning, it will gain the ability to do a specific task or improve the performance on the related task. In artificial neural networks, learning refers to the method of modifying the weights of connections between the nodes of a specified network [10]. The purpose of neural learning for ANN is to modify the weight of connection to the perceptron, by refer to the equation in Figure 2.3 the weight is the important parameter for the perceptron to doing calculation for the final output because the weight of each features needs to be adjusted to get an accurate output therefore a good method to modify the parameter is a need. ANN learning paradigm have supervised learning, unsupervised learning and reinforce learning, the 2 most common paradigm is supervised and unsupervised learning.

2.1.2.1 Supervised Learning

Supervised learning is based on training a data sample from data source with correct classification already assigned [20]. It is using labeled data to train the model; the labeled data will become the ground truth which is a reference to check whether the model predicted result same as the true output. Supervised learning can be divided to 2 type of algorithm which is Regression and Classification.

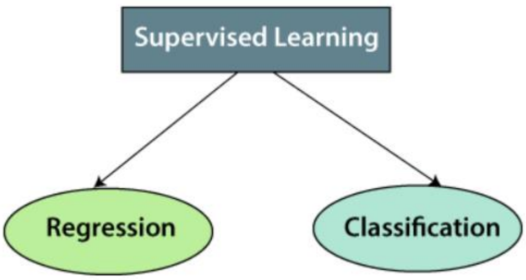


Figure 2.15: Type of Algorithm in Supervised Learning (Image retrieved from https://www.javatpoint.com/supervised-machine-learning)

Regression algorithm is suitable to use for predicting a dependent variable by refer to the independent variable where one of the most common used regression algorithms is Linear Regression. Linear Regression will used to show or analyses the relationship between the target variable and independent variable. For example, by refer to Table 2.1 size and price of house for first 4 column was given as the training data then linear regression algorithm was able to predict the value of X which is the price of house after training. The equation for making prediction will be use in this linear regression is $H(\theta) = \theta_0 + \theta_1 x$, where $H(\theta)$ is the predicted output (Price of House), x is the independent variable or features (Size of House), θ_1 is the weight of features and θ_0 is the bias.

Table 2.1: Example of Linear Regression

Size of House, m^2	Price of House, RM
10	100000
20	200000
30	300000
40	400000
50	X

Classification algorithm is suitable to used when the target output is categorical, 0 or 1, high or low. One of the most common classification algorithms used is Logistic Regression, it is a predicted analysis which work on the concept of probability. For example, it can be used to predict the tumor is malignant or not based on the input tumor size, as refer to the Figure 2.16, the red dot is the input data, and the blue line is hypothesis function of logistic regression or the predicted output, if the input tumor size has less probability than 0.5 the output will become 0 which is not malignant tumor and vice versa. The equation of this logistic regression is $H(\theta) =$

$\frac{1}{1+e^{-\theta^T x}}$, where $H(\theta)$ is the predicted output (Probability of Malignant Tumor), θ^T is the weight and x is the independent variable (Tumor Size).

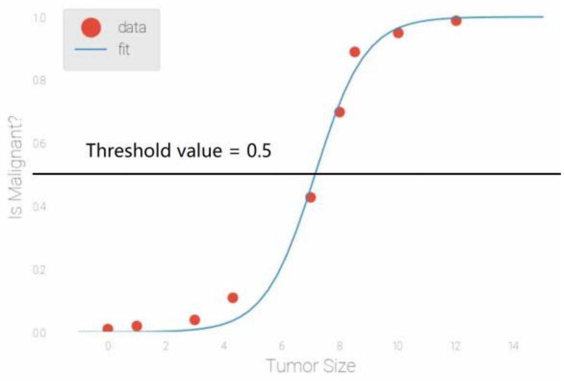


Figure 2.16: Example of Logistic Regression (Image retrieved from https://www.cloudera.com/content/dam/www/marketing/tutorials/breast-cancer-analysis-using-a-logistic-regression-model/assets/logistic-sigmoid.jpg)

2.1.2.2 Unsupervised Learning

Self-Organizing neural networks learn using unsupervised learning algorithm to identify hidden patterns in unlabeled input data [20]. Unsupervised learning is used to find the underlying structure of the input data, group the input data by similarity, and represent the data in a compressed format without any labeled input. Unsupervised learning can be categorized into Clustering and Association algorithm.

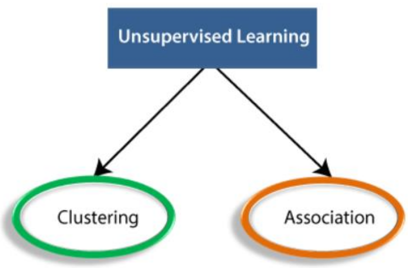


Figure 2.17: Type of Algorithm in Supervised Learning (Image retrieved from https://www.cloudera.com/content/dam/www/marketing/tutorials/breast-cancer-analysis-using-a-logistic-regression-model/assets/logistic-sigmoid.jpg**)**

Clustering Algorithm is a algorithm used to find a similar object in the input data and group them into a cluster, for example the input data is images of cows and sheep after go through this algorithm, the input images was being cluster to 2 groups which one of the group content only cows and the other one content only sheep.

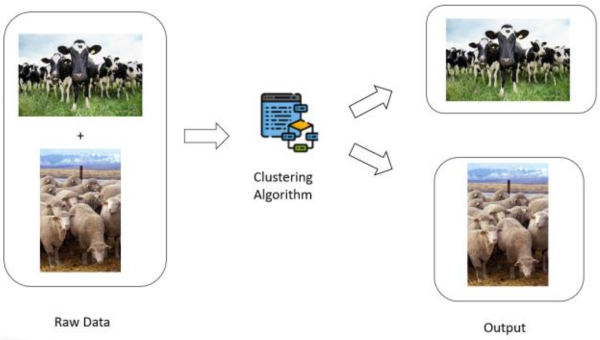


Figure 2.18: Example of Clustering Algorithm

Association algorithm is used to discover the relationship between the input datasets. For example, by refer to Figure 2.19 the market item bought by Customer 1,2,3 was being given as the input data then the output should predict was the next item will Customer n probably buy based on the item took and the list of items bought of Customer 1,2,3.

Figure 2.19: Example of Association Algorithm (Image retrieved from https://www.javatpoint.com/association-rule-learning

2.1.3 Training Dataset Format

The training dataset is the information needed for the model to learn, and the training dataset was mark as the ground truth, it will be use as the reference to judge whether the model was being train and work as we expected. In object detection, there are 2 popular data format which is COCO and Pascal VOC [19], both data format has the same purpose which is annotated the object inside an image that need to be learn by the model.

2.1.3.1 COCO

COCO is large scale images with Common Objects in Context (COCO) for object detection, segmentation, and captioning data set [19]. There are five type of annotation in COCO which is object detection, keypoint detection, stuff segmentation, panoptic segmentation, and image captioning, all the annotation file is JSON. The JSON file usually with include the collection of “info”, “license”, “images”, “annotation”, “categories” (all cases except Captions annotations) and “segment info” (only in Panoptic annotations) [18].


```
{
    "info": {...},
    "licenses": [...],
    "images": [...],
    "annotations": [...],
    "categories": [...], <-- Not in Captions annotations
    "segment_info": [...] <-- Only in Panoptic annotations
}
```

Figure 2.20: Sample of COCO [18]

2.1.3.2 Pascal VOC

Pascal VOC is one of the popular data formats, it is used for annotating object found in a data set used for computer vision [19]. Pascal VOC is an xml file, that have the bounding box include xmin, ymin, xmax and ymax for the label object. In addition, the bounding box is an axis-aligned rectangle, it bound the target object of the image and given a name to the target.

For the various tag inside the XML file, <folder> represent the folder content the images, <filename> represent the name of the image, <path> represent the directory to the image, <size> content the width and height of the bounding box and the depth have 2 variables which is 1 and 3, 1 is for black and white image whereas 3 is for colour image. For those sub tag inside <object>, <name> content the name that the model needs to identify, <truncated> is to indicate whether the bounded target is completely framed or not, 1 represent that the object extends beyond the bounding box whereas 0 represent the object is totally inside the bounding box, <difficult> have 2 possible value which is 1 and 0, 1 represent the bounded object is hard to recognize whereas 0 represent the bounded object is easy to recognize and <bndbox> content the position of the bounding box.


```xml
<annotation>
        <folder>Kangaroo</folder>
        <filename>00001.jpg</filename>
        <path>./Kangaroo/stock-12.jpg</path>
        <source>
                <database>Kangaroo</database>
        </source>
        <size>
                <width>450</width>
                <height>319</height>
                <depth>3</depth>
        </size>
        <segmented>0</segmented>
        <object>
                <name>kangaroo</name>
                <pose>Unspecified</pose>
                <truncated>0</truncated>
                <difficult>0</difficult>
                <bndbox>
                        <xmin>233</xmin>
                        <ymin>89</ymin>
                        <xmax>386</xmax>
                        <ymax>262</ymax>
                </bndbox>
        </object>
</annotation>
```

Figure 2.21: Sample of Pascal VOC [19]

2.2 Related Works

Throughout the research of paper regard to this project the first paper needs to talk about is “An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition” by Shi, Baoguang, Xiang Bai, and Cong Yao [5] because the whole study is based on the neural network propsed in this paper. As refer to figure 2.23, the proposed network can be divide into 3 layers which is convolutional layer, recurrent layers and transcription layer.

Table 2.2 is showing the struture of CRNN where k showing the kernel size, s mean stride and p is represent the padding. Convolutional layer in CRNN is used for sequence feature extraction [5], before the image feed into the CRNN, the height of the input image need to be resize to 32 which can refer to table 2.2 which is same as an usual CNN but due to the CRNN model convey deep features into sequential representations hence the width of the input image was not being a fixed size.. Then convolutional layer will first extract a feature map then it will extract the feature sequence from the left to right on the features map by column hence the i-th feature vector is the concantenation of i-th column of all the extract features map.

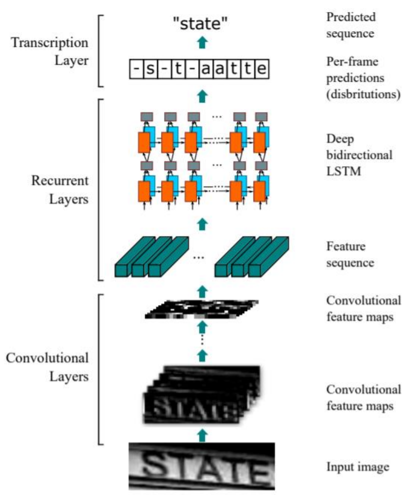
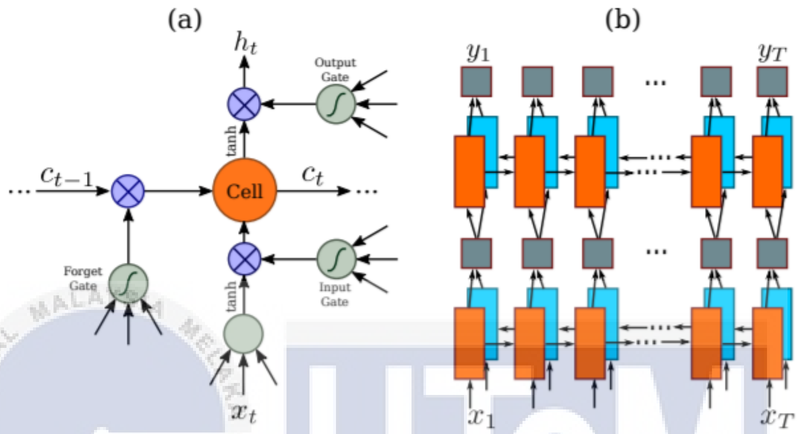


Figure 2.22: Proposed network of CRNN [5]

Table 2.2: CRNN Configuration Table [5]

Type	Configurations
Transcription	-
Bidirectional-LSTM	#hidden units:256
Bidirectional-LSTM	#hidden units:256
Map-to-Sequence	-
Convolution	#maps:512, k:2 × 2, s:1, p:0
MaxPooling	Window:1 × 2, s:2
BatchNormalization	-
Convolution	#maps:512, k:3 × 3, s:1, p:1
BatchNormalization	-
Convolution	#maps:512, k:3 × 3, s:1, p:1
MaxPooling	Window:1 × 2, s:2
Convolution	#maps:256, k:3 × 3, s:1, p:1
Convolution	#maps:256, k:3 × 3, s:1, p:1
MaxPooling	Window:2 × 2, s:2
Convolution	#maps:128, k:3 × 3, s:1, p:1
MaxPooling	Window:2 × 2, s:2
Convolution	#maps:64, k:3 × 3, s:1, p:1
Input	W × 32 gray-scale image

The recurrent layer in CRNN is for output the per frame prediction to the transcription layer, in this recurrent layer a deep bidirectional LSTM unit was used. The long short-term memory unit can solve the gradient vanishing problem which limit the range of context can store, LSTM have 3 sigmoid function which is input gate, output gate, and forget gate, input and output gate allow the cell to store the context for long period where forget gate was used to forget the memory. With these 3 gates, the LSTM manage to capture long-range dependencies. LSTM is directional, by 2 LSTM read from 2 directions, from left to right and from right to left, it will form a bidirectional LSTM. Bidirectional LSTM are useful and complementary for each other and by stacking multiple bidirectional LSTM, a deep bidirectional LSTM will be formed as shown in Figure 2.23(b). A deep structure allows higher level of abstractions than a shallow one and has achieved significant performance improvements in the task of speech recognition [5].



(a) (b)

Figure 2.23: (a) The structure of a basic LSTM unit.

(b) The structure of deep bidirectional LSTM used in the paper

The transcription layer is to convert the per frame prediction output from the recurrent layer into a label sequence. A Connectionist Temporal Classification decoder was used in this structure, it will look through the per frame prediction, then ignore the same output with the consecutive time step, for example " hel-loo—wwor-lrd", it is an example with 17 time steps, the CTC decoder end up will give a output "helloworld", the output "oo" with consecutive time step will be ignore but and the "-" will be use to prevent consecutive character to be ignore. For transcription it is two modes available in this model, first is a lexicon based which each of the training sample is associate with a lexicon and the prediction is make based on the lexicon, second is a lexicon free which prediction made without any lexicon.

By refer to figure 2.24, the accuracy test of CRNN achieve a very good accuracy result in various of word dataset, IIIT5K, Street View Test (SVT), IC03 and IC13. Except the label None is lexicon-free prediction in the figure, the others are lexicon-based prediction with 50, 50k or full. For lexicon-based prediction CRNN have at least 94.4% of accuracy and it also able to achieve 78.2% and above accuracy on lexicon-free prediction, the only proposed model to win the CRNN in this comparison is “Jaderberg et al. [23]” , from the paper they point the reason is because of the model have a very large trainable parameter which 490 million compare to CRNN model size which only have 8.3 million parameter, CRNN model is really outstanding and very suitable for real time application.

	IIIT5k			SVT		IC03				IC13
	50	1k	None	50	None	50	Full	50k	None	None
ABBYY [34]	24.3	-	-	35.0	-	56.0	55.0	-	-	-
Wang *et al.* [34]	-	-	-	57.0	-	76.0	62.0	-	-	-
Mishra *et al.* [28]	64.1	57.5	-	73.2	-	81.8	67.8	-	-	-
Wang *et al.* [35]	-	-	-	70.0	-	90.0	84.0	-	-	-
Goel *et al.* [13]	-	-	-	77.3	-	89.7	-	-	-	-
Bissacco *et al.* [8]	-	-	-	90.4	78.0	-	-	-	-	87.6
Alsharif and Pineau [6]	-	-	-	74.3	-	93.1	88.6	85.1	-	-
Almazán *et al.* [5]	91.2	82.1	-	89.2	-	-	-	-	-	-
Yao *et al.* [36]	80.2	69.3	-	75.9	-	88.5	80.3	-	-	-
Rodrguez-Serrano *et al.* [30]	76.1	57.4	-	70.0	-	-	-	-	-	-
Jaderberg *et al.* [23]	-	-	-	86.1	-	96.2	91.5	-	-	-
Su and Lu [33]	-	-	-	83.0	-	92.0	82.0	-	-	-
Gordo [14]	93.3	86.6	-	91.8	-	-	-	-	-	-
Jaderberg *et al.* [22]	97.1	92.7	-	95.4	80.7*	**98.7**	**98.6**	93.3	**93.1***	**90.8***
Jaderberg *et al.* [21]	95.5	89.6	-	93.2	71.7	97.8	97.0	93.4	89.6	81.8
CRNN	**97.6**	**94.4**	**78.2**	**96.4**	**80.8**	**98.7**	97.6	**95.5**	89.4	86.7

Figure 2.24: Accuracy result of CRNN compared to others [5]

The next paper review is “A robust real-time automatic license plate recognition based on the YOLO detector”. This paper presents a robust and efficient

ALPR system based on the state-of-art YOLO object detector, the dataset was collect inside a vehicle driving through regular traffic in an urban environment, they record the video with resolution 1920x1080 in PNG format by 3 camera which is Go Pro Hero4 Silver, Huawei P9 Lite and iPhone 7 Plus. Each camera was record 50 videos with 1 second, the image was extract from video with 30 frames per second, hence each camera will have 1500 images, the total dataset collected is 4500 images. For training session, the image of cropped license plate was duplicate with negative image and the dataset was split into 40% training, 40% testing and 20% for validation.

The proposed method in the paper was shown in figure 2.25, first the vehicle detector will crop the vehicle out from the image by using vehicle detector which train from YOLO model with the backbone of ImageNet. Then the cropped vehicle image pass to the license plate detector which also train using the YOLO model with ImageNet as backbone. After the license plate was cropped, a proposed CNN which modified based on CR-NET was used for text detector to segmentize the character in the license plate. After the segmentation, the character was cropped then go for the single frame character recognition, in the recognition stage 2 networks was used, one for digit recognition and the others for letter recognition. The reason of using 2 networks in recognition is for tuning 2 different network input size for digit and letter to achieve higher accuracy of recognition. The process of single frame character recognition, multiple frames from same vehicle will be gather to explore temporal redundancy information, the final recognition result will based on the most frequently predicted character at the respective position in license plate.

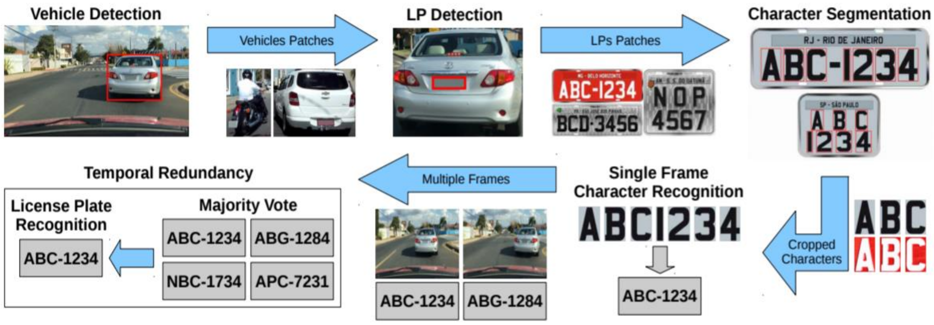


Figure 2.25: Flow of the ALPR proposed [3]

In the paper "An efficient license plate recognition system using convolution neural networks", it also using the vehicle detector to reduce the noise from environment factor before license plate detection. After the image cropped into a single vehicle image, it will pass to the license plate detector and cropped out the license plate, the algorithm used in this paper is SVM OAR (one against rest) architecture with the HOG value. After that, for the character segmentation the proposed method was first binarize and grayscale the cropped license plate to reduce the noise shown as figure 2.26, then a horizontal projection was used to remove the lower and upper boundary as figure 2.27, after that, a vertical projection will used to determine the position of each character in the license plate and divide it into single character as shown in figure 2.28. After segmentation, a proposed CNN model in the paper was used for recognition of license plate.

Figure 2.26: (a) An example of a detected vehicle license plate, (b) a binarized plate [4]

Figure 2.27: Eliminating the upper and lower borders by horizontal projection [4]

Figure 2.28: Separating characters by vertical projection [4]

CHAPTER 3

METHODOLOGY

3.1 Method of Data Collection

3.1.1 Data Collection of License Plate Detector

The raw data (vehicle cropped images) was provided by LED Vision Sdn Bhd. The provided cropped vehicle images are collected from three junctions (Public Bank, Sirim and Infineon) at Batu Berendam, Melaka. LED Vision Sdn Bhd preinstalled cameras with a resolution of 1920x1080 pixel at the overhead poles of the junctions and an image based vehicle detector was used to assist the camera in capturing the vehicles appear at the junction.

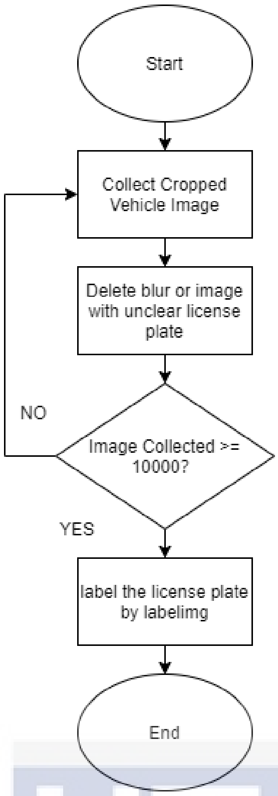


Figure 3.1: Flowchart of prepare LPD dataset

By referring to Figure 3.1, fo

Besides that, another testing dataset was prepared. This testing dataset will consist of 1000 cropped vehicle that is not included in the LPD training to evaluate the accuracy of LPD in cropping the license plate.

3.1.2 Data Collection of License Plate Recognizer

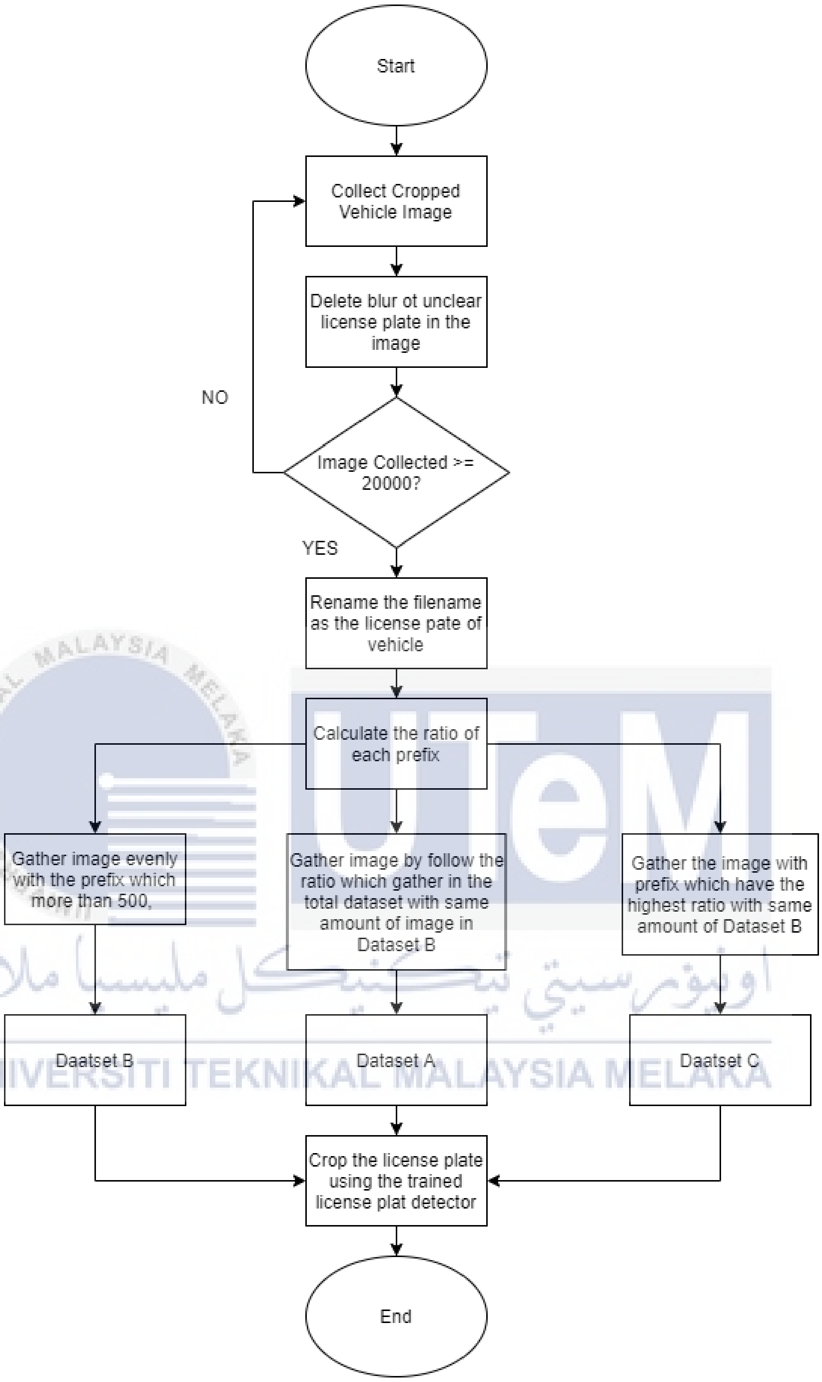


Figure 3.2: Flowchart of preparation of LPR dataset

For the license plate recognition dataset, with reference to Figure 3.2, the target amount is 20000. Blurred or unclear license plate shown in image will be deleted.

When the dataset reaches or more than 20000, the collection will stop. After the collection, the process of renaming the filename as the license plate in the image will start. The rename process is for the use of the filename as the ground truth of LPR training. After the rename process, the amount of image of each prefix will be calculated using a python script. The types of prefix were being assigned into 7 categories: A1234, A1234B, AB1234, AB1234C, ABC1234, ABC1234D and Special, where A, B, C, D represent any of alphabet in the license plate with the respective position; 1234 represent the amount of any number in the position from amount 1 to 4; Special represent the special license plate like UTEM 1, LIMAU 2 or SATRIA 3. After calculating the amount of each prefix categories, the images will be used to form 3 different datasets with equal amount of images each: Dataset A, a ratio of distribution of prefix which is similar to the collection of 20000 images but with the limit of image which same as Dataset B and C. Dataset B is a set of data assembled from prefixes that have more than 500 images in the collected dataset, with a limitation of having the same amount of images for each prefix. Dataset C, only the highest amount of prefix will assign to this dataset with the same amount of Dataset A, B and C. All the vehicle license plates in the datasets will be cropped by using a trained license plate detector.

In addition, a testing dataset was prepared for running the evaluation on the accuracy of the license plate recognition model. The testing dataset is formed with all the prefixes that were not included in the training of the model and was limited to the amount of 1000 for each prefix. Besides that, a set of synthetic license plate was also being prepared [24], with 1000 images for each prefix with total amount of 7000 images.

3.2 Method of Training

3.2.1 Method of Training on License Plate Detector

A ConvNET model SSD Mobile Net v3 from TensorFlow model zoo was used as the license plate detector as its higher accuracy in object detection as compared to YOLOv3 model [21]. The framework used for training is TensorFlow with version 1.15, the dataset prepared was being assign to training and testing with a ratio of 9:1. The training of License Plate Detector was monitor by using tensorboard, it was setup to show several parameters such as mean average precision, total loss, and result image for the model to localize the license plate.

The method to monitor the training is to ensure the regularization loss was decrease during the training, if the loss keep rising until the middle session of training step most probably is because of the learning rate was too high. Then observe the result image and mean average precision, observe the result image is because of some time there is some mistake during labeling, if the location of license plate draw by the model was not the exact location, then need to stop training and check the training dataset. MAP and AR is the way to check whether the training achieve the result wanted which is more than 75% mAP.

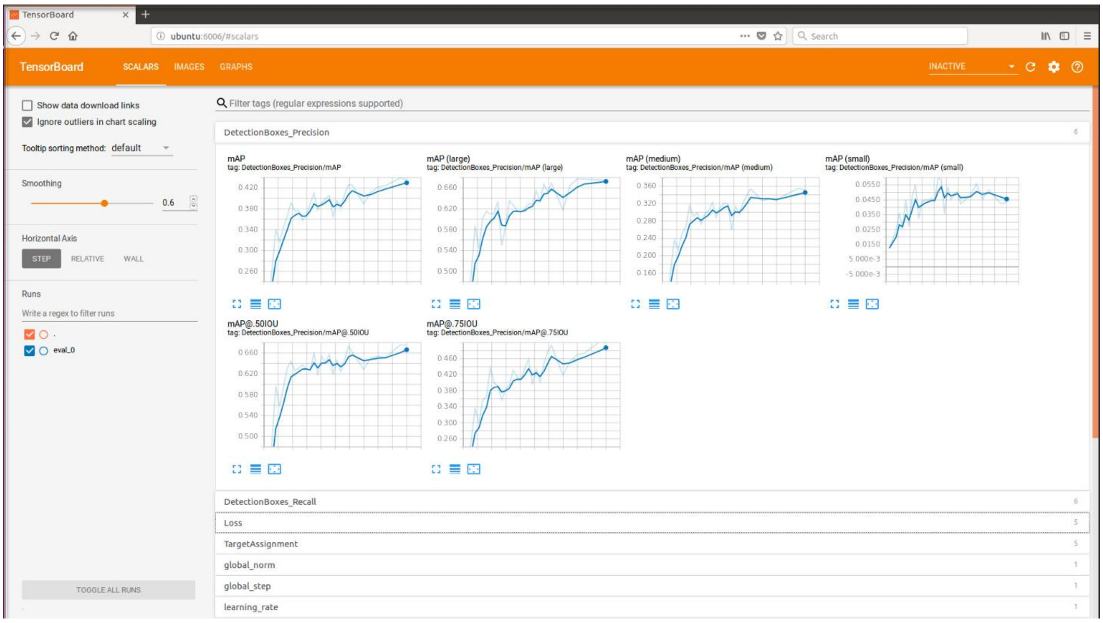


Figure 3.3: Tensorboard configure to monitor the training

3.2.2 Method of Training License Plate Recognizer

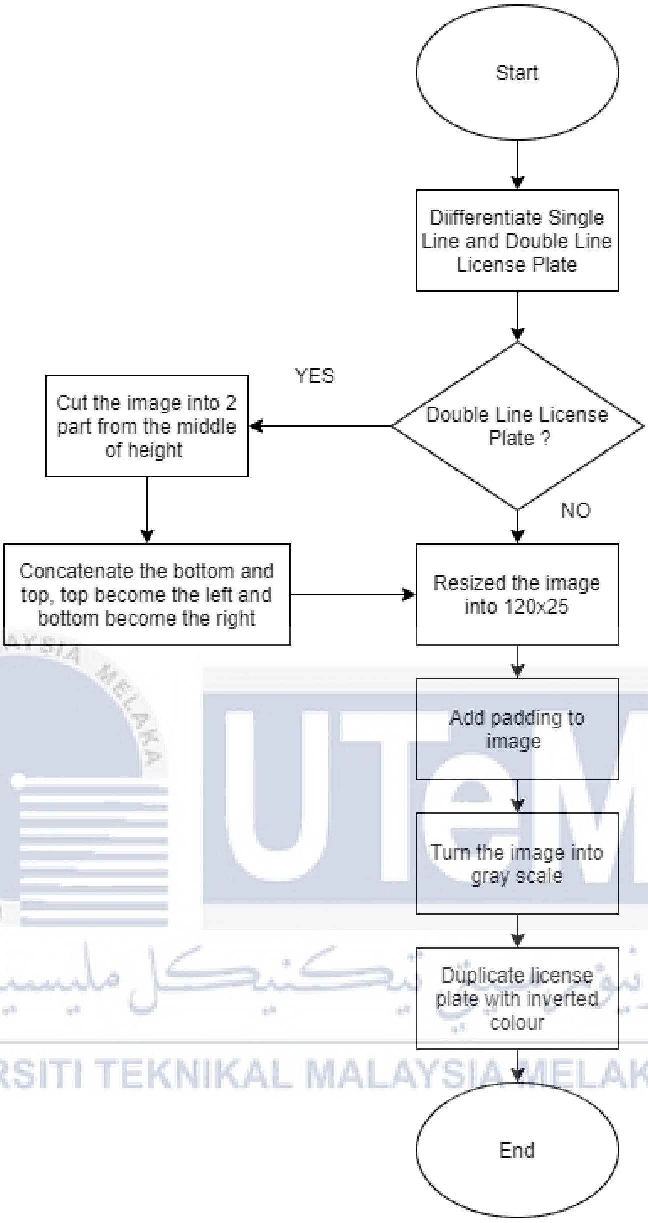


Figure 3.4: Preprocessing of Dataset for LPR

Before the training of LPR, a preprocessing process was run for all the LPR dataset. As refer to Figure 3.4, the license plate will be differentiated into double line and single line license plate, then double line license plate will being cut into half (top and bottom) from the mid of the height, then concatenate the image by assign the top to left and bottom to the right for training the Convolutional Recurrent Model effectively, this is because during feature sequence extraction the CRNN model will start from the left to the right and generate the feature vector [5].After the concatenate, all the license plate was being resize to 120x25 due to the size of image of license plate some is smaller than the CRNN model input size. Then the padding needed to be added because the image after resizing will not be necessary to become the exact resolution hence the padding is needed for achieve the exact the resolution. The last step is changing the image into grayscale image because the RGB colour is not important in the recognition and after grayscale will sharpen the image and make the training become more effective. The last step is duplicate the image in the dataset with invert colour, the purpose is to create the even amount of image for black background white text and white background black text license plate, then the dataset will be process into the format as Synth90k dataset [22].

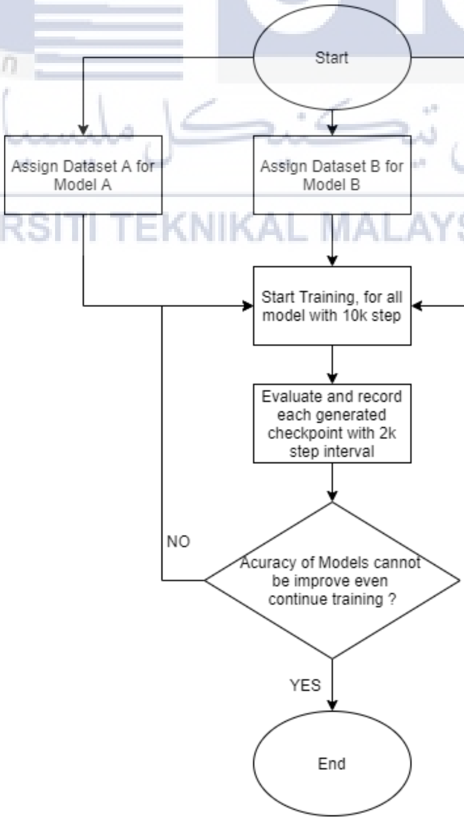


Figure 3.5: Flowchart of training LPR

For the LPR training as refer to Figure 3.5, 3 datasets will used to train 3 different model by using Convolutional Recurrent Neural Network and Tensorflow as the framework. Dataset A will use to train Model A, Dataset B will use to train Model B and Dataset C will use to train Model C. Each dataset will assign 90% of dataset for training, 5% for testing and 5% for confusion matrix, all the models will being train will configure to train with 10k of step and will run evaluation for every 2k step of checkpoint of each model, if the full sequence mean test accuracy of model have the possibility to increase, the models will train 10k and repeat the evaluation. The checkpoint chosen for each model to be export will be the same to ensure the fairness of analysis to each model, hence the best checkpoint will be chosen based on the performance on 3 model.

3.2.3 Method of Selection on Checkpoint of License Plate Recognizer

To select the suitable checkpoint, rank of accuracy was used. Rank of accuracy represent the mean test full sequence accuracy of the respective checkpoint, if the respective checkpoint has the highest accuracy, rank is 1, if the accuracy is the second highest the rank is 2. The checkpoint model will be select by compare the rank of each model on respective training with a accepted of difference in rank with 1, if the selected checkpoint and it respective rank on 3 of the model do not have difference in rank more than 1, it is the final checkpoint, if throughout the comparison cannot select any checkpoint, the accepted of difference in rank will be increase to 2. Besides that, the selection of checkpoint will come from the highest rank to lowest to ensure the accuracy of the selected model is good.

3.3 Method of Evaluation

3.3.1 Method of Evaluation on License Plate Detector

The method uses to evaluate the license plate detector is based on the mean average precision on the evaluation of testing dataset. Average precision (AP) is calculated as the "area under the precision-recall-curve" [23], the precision was calculated by using formula (1) and recall was calculate using formula (2). For the true positive, false positive and false negative, for example if a dataset content A and B, the true positive is the model predicted result is A and the ground truth is A, false positive is predicted result is B but the ground truth is A and false negative is predicted result is A but the ground truth is B. The precision-recall curves will form by using precision and recall then the average precision will calculate based on the formula (3) where n is the number in discrete form of precision and recall, and the mean average precision is the sum of average precision of each class divide by the total number of class. Hence, mAP is actually showing the model precision and recall.

$$\text{Precision} = \frac{TruePositive}{TruePositive + FalsePositive} \tag{1}$$

$$\text{Recall} = \frac{TruePositive}{TruePositive + FalseNegative} \tag{2}$$

$$AP = \sum_n (R_n - R_{n-1}) P_n \tag{3}$$

The evaluation of license plate detector will be run, during the training process and the result of mAP was recorded after every training. Besides that, the other evaluation will also being run based on the testing dataset prepared, the license plate detector will run to localize the license plate in the cropped vehicle images, the result of localize will be visualize in the image and the result of accuracy will be record through observation with 0.5 and 0.3 of confidence threshold.

3.3.2 Method of Evaluation and Comparison on License Plate Recognizer

The license plate recognition model will used 2 evaluation metrics (mean test per char accuracy and mean test full sequence accuracy) to evaluate the performance of the trained model. Per Char Accuracy is the model predicted the character on the license plate correctly, calculate by using equation (4) where Full Sequence Accuracy is the model predict the full sequence of license plate correctly, calculate by using equation (5). The confusion matrix was also being used to show the probabilities of the model to predict character respect to the labeled character as shown in Figure 3.6.

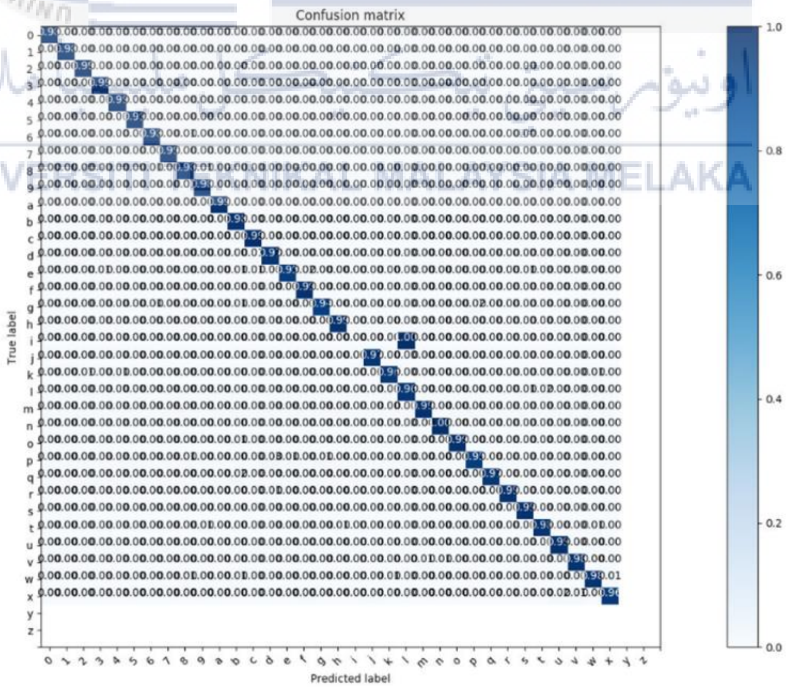


Figure 3.6: Sample of confusion matrix

$$\text{Mean Test Per Char Accuracy} = \frac{\text{Per Char Accuracy}}{\text{Number of Testing Image}} \tag{4}$$

$$\text{Mean Test Full Sequence Accuracy} = \frac{\text{Full Sequence Accuracy}}{\text{Number of Testing Image}} \tag{5}$$

The license plate recognition models will be evaluated after each of the training, the result of every checkpoint of every model will be recorded to decide which checkpoint to be used for a fair comparison. For the comparison, another testing dataset and a synthetic license plate testing dataset was used for evaluation of 3 of the models, the models will be compare based on the mean test full sequence accuracy, and the result will be discussing.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Result of Data Collection

4.1.1 Result of Data Collection for License Plate Detector

For license plate detector, the dataset consists of 10019 images and all the images were labelled using the labelImg application as shown as Figure 4.1. After labelling the position of license plate in the image, a .xml file was generated as shown in Figure 4.2. The .xml file consists of the respective .jpg image name as shown in the <filename> and the position of the license plate in the image as shown in <bndbox>. This information is important as it would be used as the ground truth for the license plate detection and training.

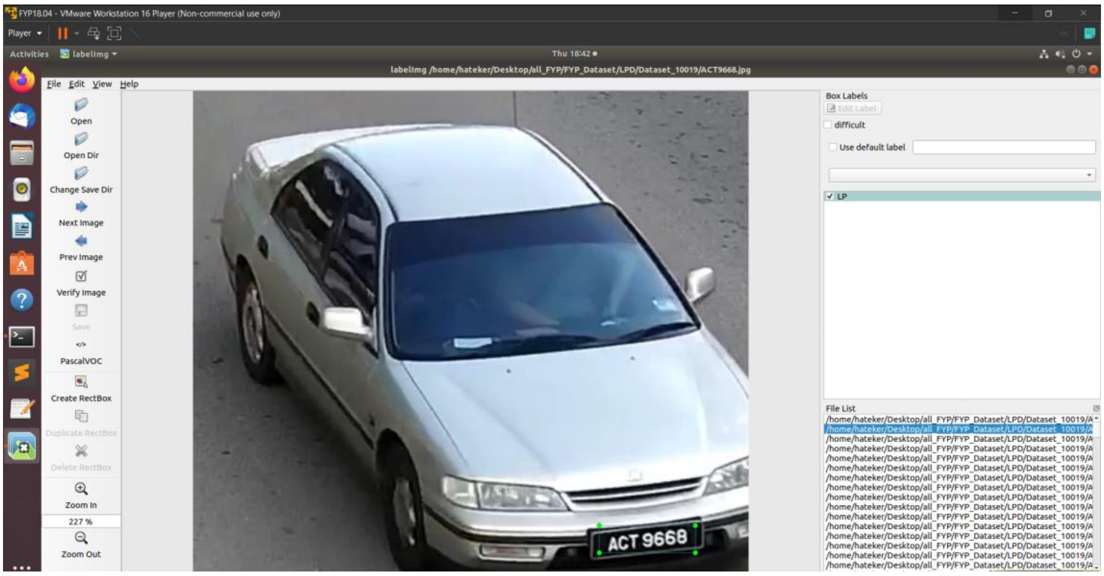
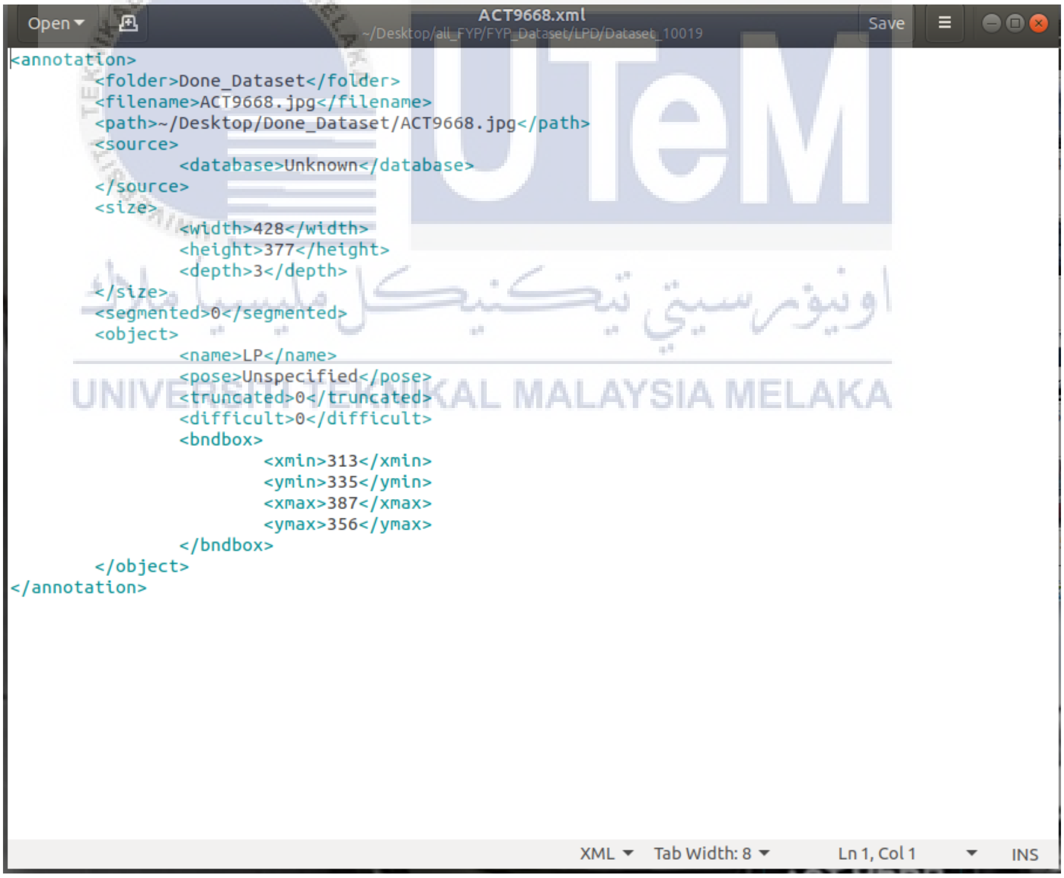


Figure 4.1: Sample of Labelled Image



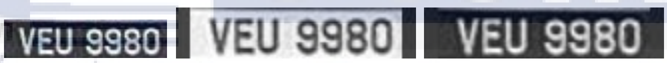

```xml
<annotation>
	<folder>Done_Dataset</folder>
	<filename>ACT9668.jpg</filename>
	<path>~/Desktop/Done_Dataset/ACT9668.jpg</path>
	<source>
		<database>Unknown</database>
	</source>
	<size>
		<width>428</width>
		<height>377</height>
		<depth>3</depth>
	</size>
	<segmented>0</segmented>
	<object>
		<name>LP</name>
		<pose>Unspecified</pose>
		<truncated>0</truncated>
		<difficult>0</difficult>
		<bndbox>
			<xmin>313</xmin>
			<ymin>335</ymin>
			<xmax>387</xmax>
			<ymax>356</ymax>
		</bndbox>
	</object>
</annotation>
```


Figure 4.2: Sample of .XML File Generated after Labeled

4.1.2 Result of Data Collection for License Plate Recognizer

For the collected license plate recognition dataset, 20947 images were collected and categorized according to prefix as shown in Table 4.1. Inside these 20947 images, 18690 images belong to prefix ABC1234 which occupy 89.23% of the collected dataset (skewed dataset). Other than prefix AB1234 and AB1234C have occupy 4.78% and 4.44% of the dataset. The rest of the prefixes is not more than 2% in this dataset.

The LPR dataset is then undergo preprocessing (resize, add padding, gray scale the image and duplicate with invert color), as shown in Figure 4.3. For double line license plate, additional step is needed which is to cut the image into top and bottom (divide by half) and then concatenate the top part as left and bottom part as right (see Figure 4.4).



(a) (b) (c)

Figure 4.3: (a) Raw license plate image, (b) license plate after preprocessing (resize, add padding, grayscale and invert color), (c) license plate after resize, add padding and grayscale

(a) (b) (c)

Figure 4.4: (a) Raw double line cropped license plate, (b) double line license plate after resize, add padding, grayscale and invert color, (c) double line license plate after resize, add padding and grayscale

Table 4.1: Dataset Collected for License Plate Recognition Categorized by Prefix

Type of prefix	Number of Image	Percentage of image in Dataset, %
A1234	68	0.32
A1234B	247	1.18
AB1234	1001	4.78
AB1234C	931	4.44
ABC1234	18690	89.23
ABC1234D	4	0.02
Special	6	0.03
Total	**20947**	**100**

Based on the collected dataset, three datasets were formed (Dataset A, B and C). The first dataset, Dataset B (see Table 4.3), with three prefixes (AB1234, AB1234C and ABC1234) is created (captured license plate with prefix of AB1234, AB1234C and ABC1234 have more than 500 images and plan to reserve 100 images to form the testing dataset). The second dataset (Dataset A) was created with 2400 images and follow the ratio in the collected dataset as shown in Table 4.2. Dataset C was created using only 1 prefix which is ABC1234 with 2400 images as shown in Table 4.3.

Table 4.2: Dataset A

Type of prefix	**Number of Image**	**Percentage of image in Dataset, %**
A1234	8	0.32
A1234B	28	1.18
AB1234	115	4.78
AB1234C	107	4.44
ABC1234	2141	89.23
ABC1234D	0	0.02
Special	1	0.03
Total	**2400**	**100**

Table 4.3: Dataset B

Type of prefix	**Number of Image**	**Percentage of image in Dataset, %**
AB1234	800	33.33
AB1234C	800	33.33
ABC1234	800	33.33
Total	**2400**	**100**

Table 4.4: Dataset C

Type of prefix	**Number of Image**	**Percentage of image in Dataset, %**
ABC1234	2400	100
Total	**2400**	**100**

Making use of the created datasets, a testing dataset for the evaluation of license plate recognition model was also form by using the rest of the unused dataset. A total of 800 images from Dataset B with prefix of A1234B and AB1234 were used for training therefore the testing dataset for both of the prefix was left with 201 and 131 images. For prefix of ABC1234 since it has a large amount of image in the prefix which is 18690 therefore the amount of testing dataset was still up to 1000 images even though 2400 images of ABC1234 was used for the training. For Dataset C, the remains images were used as the testing dataset to create a fair comparative result. In addition, 1000 synthetic images were generated for each prefix as the supportive data for testing.

Table 4.5: Testing Dataset

Type of prefix	**Number of Image**
A1234	60
A1234B	219
AB1234	201
AB1234C	131
ABC1234	1000
ABC1234D	4
Special	5
Total	**1620**

Table 4.6: Testing Dataset form by synthetic license plate

Type of prefix	**Number of Image**
A1234	1000
A1234B	1000
AB1234	1000
AB1234C	1000
ABC1234	1000
ABC1234D	1000
Special	1000
Total	**7000**

4.2 Result of Training

4.2.1 Result of Training for License Plate Detector (LPD)

For the license plate detector, mean average precision of the test accuracy was targeted to be 75% or above with confidence threshold used of 0.5. The collected License Plate Detector (LPD) dataset consists of 10019 images and 10% were used for testing (1001 images). Batch size of 16 and learning rate 0.04 were applied during the LPD training.

Table 4.7 shows the training result of the license plate detector. Large detection bounding box was excluded as the size of the license plate in the collected dataset is always small. Due to the transfer learning (ImageNet pretrained weight is used), the model easily reach mAP of 70.83% (checkpoint 40000) at the end of the training. Since it has not reached the targeted mAP of 75%, second training was conducted with the same number of steps (40000 steps).

For the second training, in the beginning of the training the mAP was slightly reduced and training loss slight raised as the learning rate used in the beginning is high and it will decrease when the step increase. As observed, mAP raised slower in the second training as compared to the first training. At the end of second training both mAP medium and mAP small able to reach 75% and above The training halted and the model was exported for evaluation.

Table 4.7: Training Result of LPD

Number of Training	Checkpoint	Training loss	Mean average precision	mAP, medium bounding box	mAP, small bounding box
1	3000	1.3590	0.5209	0.5220	0.5230
	9000	0.9848	0.6004	0.6053	0.5705
	15000	0.9404	0.5892	0.5909	0.5866
	21000	0.7354	0.6661	0.6673	0.6704
	27000	0.7722	0.6984	0.7022	0.6840
	33000	0.6600	0.7057	0.7093	0.6908
	40000	0.6733	0.7083	0.7121	0.6938
2	3000	0.7300	0.6311	0.6291	0.6571
	9000	0.6235	0.7038	0.7059	0.7036
	15000	0.6675	0.6866	0.6841	0.7083
	21000	0.4918	0.7237	0.7252	0.7238
	27000	0.4015	0.7313	0.7305	0.7504
	33000	0.4458	0.7495	0.7483	0.7684
	40000	0.4294	0.7508	0.7511	0.7683

4.2.2 Result of Training for License Plate Recognition (LPR)

For the LPR dataset used, the total size of the dataset will increased by double after the invert of color (original size 2400 and become 4800 images after the process of color inversion). 240 images (5%) of the images were used for testing. The LPR training was conducted with batch size of 64 and learning rate of 0.001. As the LPR model needs to be trained from scratch and dataset for training is only 2400 images therefore a dataset with 50000 synthetic license plate images was created by using OCR generator [24]. The 50000 images synthetic license plate was being trained with 10000 steps and the resulting weight file is used as the pretrained weight for each of the training of three models (Model A, B, and C). Hence, hence the checkpoint showed in Table 4.8, 4.9, and 4.10, with the reporting interval of 2000 steps, begin from 12000.

Model B and C show the mean test full sequence accuracy start dropping from checkpoint 16000 but Model A seem to have the possibility to rise the accuracy hence second training with another 10000 steps was conducted to ensure the most suitable model can be chosen for the evaluation.

After the second training, accuracy of Model A start dropping from checkpoint 24000, Model B reach the highest accuracy at checkpoint 22000 where Model C highest accuracy was still at checkpoint 16000, therefore the training is stop and the suitable checkpoint was identified.

Table 4.8: Training Result of Model A

Checkpoint	**Training loss**	**Mean test per char accuracy**	**Mean test full sequence accuracy**	**Rank of Accuracy**
12000	0.031387	0.920436	0.770833	4
14000	**0.018905**	**0.934385**	**0.795833**	**2**
16000	0.010694	0.921766	0.779167	3
18000	0.007370	0.929306	0.791667	3
20000	0.006129	0.928313	0.791667	3
22000	0.008989	0.933968	0.791667	3
24000	0.005512	0.931091	0.800000	1
26000	0.003353	0.929206	0.791667	3
28000	0.003327	0.932877	0.791667	3
30000	0.003831	0.930496	0.787500	5

Table 4.9: Training Result of Model B

Checkpoint	Training loss	Mean test per char accuracy	Mean test full sequence accuracy	Rank of Accuracy
12000	0.024291	0.928145	0.808333	3
14000	**0.018355**	**0.926002**	**0.812500**	**2**
16000	0.010819	0.928085	0.812500	2
18000	0.008196	0.926796	0.808333	3
20000	0.007052	0.926895	0.804167	4
22000	0.006897	0.934931	0.816667	1
24000	0.005314	0.924712	0.808333	3
26000	0.003892	0.930069	0.812500	2
28000	0.004043	0.927014	0.812500	2
30000	0.002760	0.927808	0.812500	2

Table 4.10: Training Result of Model C

Checkpoint	Training loss	Mean test per char accuracy	Mean test full sequence accuracy	Rank of Accuracy
12000	0.023267	0.928145	0.800000	3
14000	**0.017499**	**0.931925**	**0.804167**	**2**
16000	0.010745	0.928085	0.808333	1
18000	0.008056	0.926796	0.795833	4
20000	0.007143	0.926895	0.795833	4
22000	0.006522	0.934931	0.770833	5
24000	0.005111	0.931091	0.770833	5
26000	0.003468	0.929206	0.795833	4
28000	0.004125	0.932877	0.770833	5
30000	0.002356	0.930496	0.770833	5

4.2.3 Model Selection on Checkpoint of License Plate Recognition Training

Making use of the obtained mean test full sequence accuracy for the License Plate Recognition (LPR) training from checkpoint 12000 to 30000, rank is assigned accordingly to Model A, B, and C as shown in the right most column of Table 4.8, Table 4.9, and Table 4.10, respectively. Model A at checkpoint 24000 is ranked at #1 with highest mean test full sequence accuracy of 0.8. Model B obtained highest mean test full sequence accuracy at checkpoint 22000. Model C managed to obtain highest mean test full sequence accuracy at checkpoint 16000. Checkpoint 14000 for all three models were as having the same rank which is 2 at the respective checkpoint.

The confusion matrix of checkpoint 14000 for Model A, B and C is as shown in Figure 4.6, 4.7 and 4.8, respectively. The darker is the blue color, the higher probability of the model successfully make the correct prediction. For Model B and C, the confusion matrix shows that both models always able to predict the respective label correctly (diagonal blue line from the top left to the bottom right). For model A, it has one predicted label “I” not having the highest probability to the true label. This is due to the one and only one of the special prefix images include in the dataset A (see Figure 4.6)

Figure 4.5: The only license plate content “I” label in Dataset A

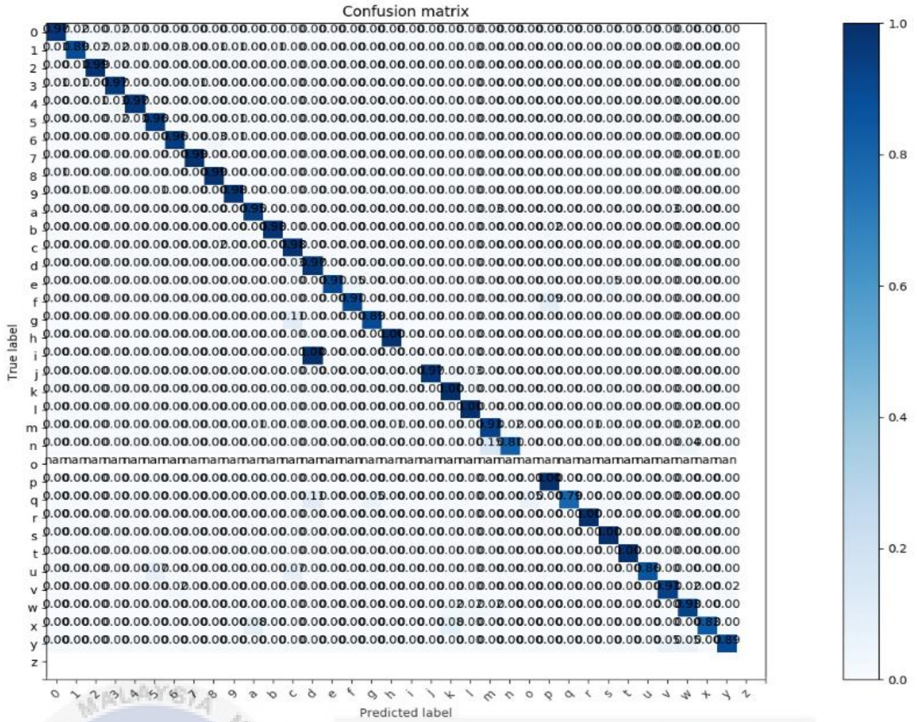


Figure 4.6: Confusion Matrix of Model A

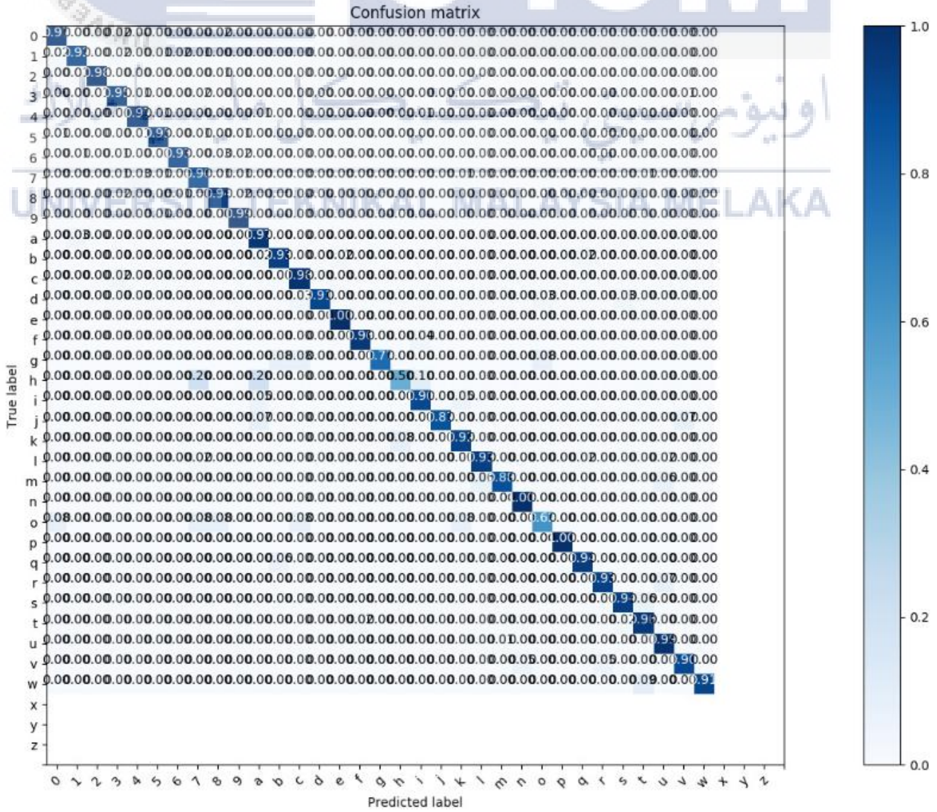


Figure 4.7: Confusion Matrix of Model B

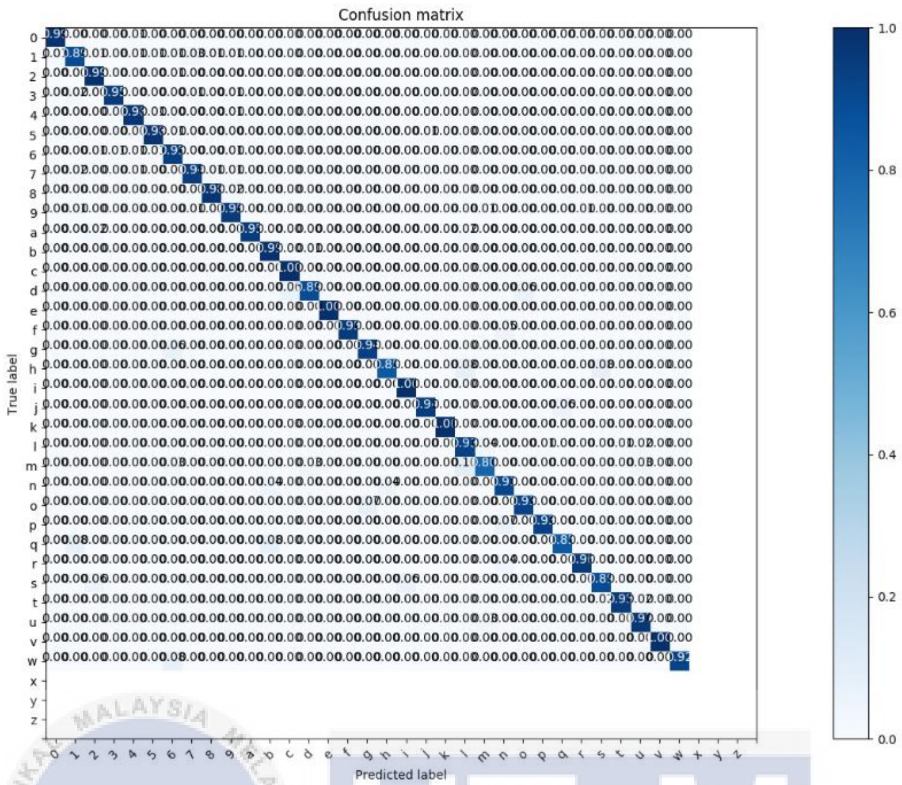


Figure 4.8: Confusion Matrix of Model C

4.3 Result of Evaluation

4.3.1 Result of Evaluation on License Plate Detector (LPD)

The evaluation of LPD uses two different confidence threshold values (0.3 and 0.5) with the collected vehicle cropped image. During the evaluation, only the detected object with the highest confidence will be shown and the result will be visualize on the cropped vehicle images (see Figure 4.9) and save to a folder. After that, the result images will be observed one by one, and the accuracy was being recorded for both thresholds. For threshold value of 0.5, location of license plate in 998 out of 1000 images is found correctly detected within the detection bounding box. For threshold value of 0.3, the accuracy is 100%. Therefore this threshold was used to crop the license plate out from the collected vehicle cropped images for subsequent license plate recognition (LPR) model evaluation.

Table 4.11: Evaluation Result of LPD

Number of Images	Accuracy,%	
	0.5 Threshold	0.3 Threshold
1000	99.80	100

Figure 4.9: Result of LPD

4.3.2 Result of Evaluation and Comparison on License Plate Recognizer

The result of evaluation of all LPR model was tabulate into Table 4.12. As shown in the table, only four test images with prefix of ABC1234D and "Special" are available therefore both prefixes will not be emphasizing in this analysis. For easier analysis, the table was being export to a graph which shown in Figure 4.10.

From Figure 4.10, first observation is Model C does not have the ability to recognize any of the image with prefix A1234B and AB1234C as the recognition rate is zero. This proves that without include a combination of Aphabet-Digit-Alphabet in the dataset, the model will not have the ability to recognize this combination. Similar result is observed on the synthetic testing dataset (see Figure 4.11). Secondly, Model C also shows that even with single prefix of ABC1234 include in the training dataset, Model C is still able to recognize license plate image with prefix of AB1234 and A1234. This prove that the with multiple alphabets in the combination Alphabet-Digit can train the LPR model to have the ability to recognize the respective combination. In addition, it also proves that the license plate can be combined into two categories which are Alphabet-Digit and Alphabet-Digit-Alphabet.

Table 4.12: Evaluation Result of All LPR Model on Testing Dataset

Prefix	**Number of Testing Image**	**Mean Test Full Sequence Accuracy**		
		Model A	**Model B**	**Model C**
A1234	60	60.00	65.00	47.67
A1234B	219	56.16	49.77	0.00
AB1234	201	78.61	89.05	75.62
AB1234C	131	64.12	86.26	0.00
ABC1234	1000	77.00	68.90	77.00
ABC1234D	4	25.00	25.00	0.00
Special	4	0.00	0.00	0.00

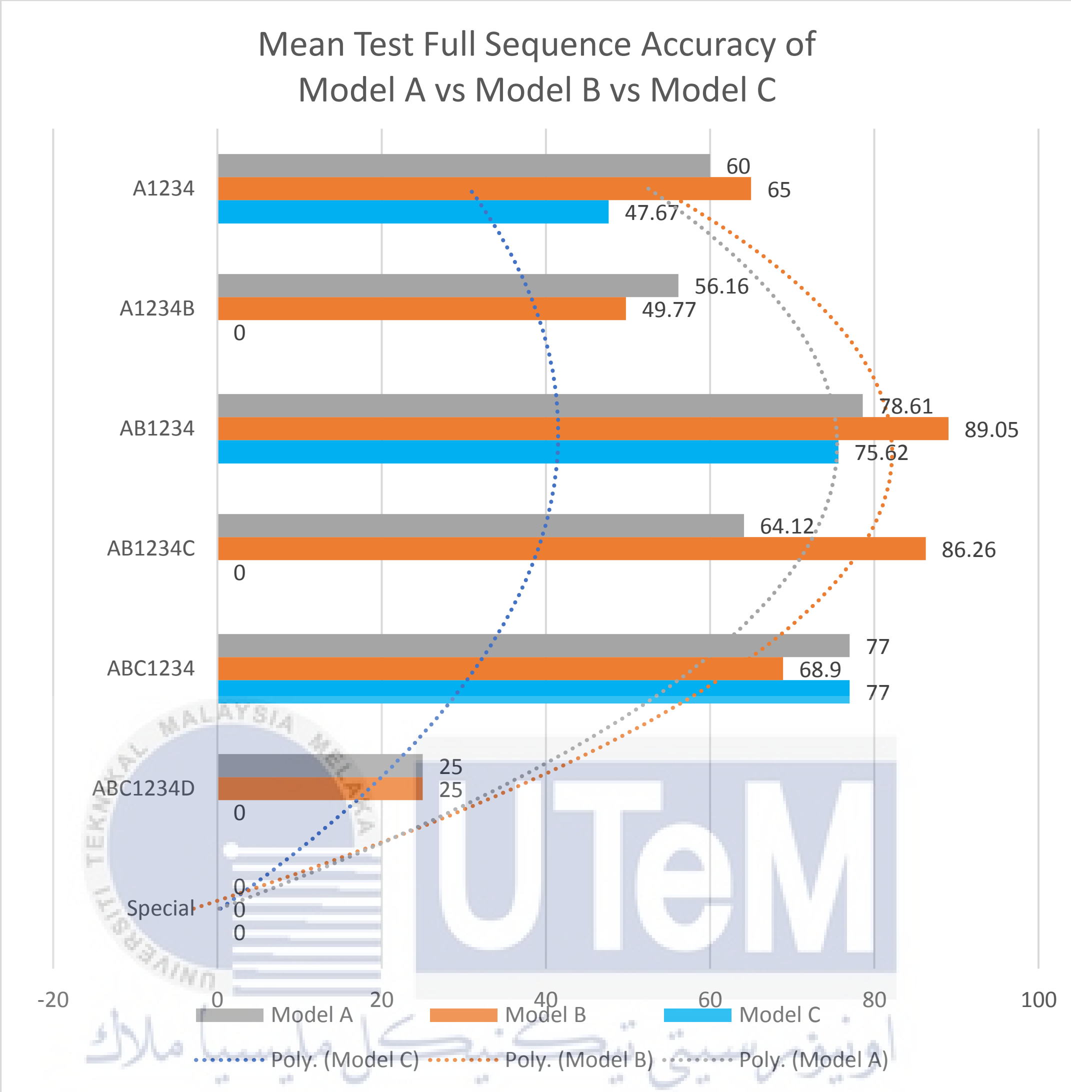


Figure 4.10: Mean Test Full Sequency Accuracy of All Models on Testing Dataset

For Model B, it shows good recognition accuracy for license plate image with prefixes ABC1234, AB1234C, and AB1234. This is because it is trained with a balanced dataset which consists of 800 images for ABC1234 (68.9%), AB1234C (86.26%) and AB1234 (89.05%). Besides that, by comparing the accuracy of Model B and C on prefix A1234, Model B have a higher accuracy compare to Model C. It show that include prefix of AB1234 in the dataset will get higher accuracy on A1234 because Model C do not have any prefix of AB1234 in its dataset where Model B have

800 images included in the dataset. This prove that, in the combination of Alphabet-Digit license plate, the nearer amount of alphabet in the combination, the higher the accuracy rise as compared to the lesser amount of alphabet in the combination.

For Model A, the accuracy on prefix A1234B is higher than Model B and the amount of AB1234C in Dataset B is 800 which is higher than Dataset A with 107 images of AB1234C and 28 images of AB1234. This show that with a little number of images on the respective prefix will rise the accuracy more than the higher amount of image in similar prefix. By compare the accuracy of Model A and B on prefix AB1234C, Model A have an accuracy of 64.12% and Model B 86.26% there is roughly 22% different, this show that the relationship of accuracy and amount of image is not proportional, to increase accuracy on high accuracy will require more images to increase. This also being shown on the accuracy on prefix ABC1234 of 3 models, Model A and C have 2141 and 2400 images on the respective prefix and the accuracy for both models is 77% where Model B have only 800 images and the accuracy is 68.9%.

In addition, based on the inference above and from the observation of accuracy on Model B on prefix AB1234 and AB1234C, it shows that with amount of 800 images of AB1234 and ABC1234 can achieve 89.05% of accuracy where 800 images of AB1234C will be able to achieve 86.26% of accuracy which show that, with a pre trained weight of synthetic license plate and duplication of license plate with inverted colour, a less amount of license plates needed to achieve high accuracy. Besides that, by compare trendline of figure 4.10 and 4.11, it is showing 2 different characteristics, hence synthetic license plate was different with real license plate even the license plate has been processed to have similar characteristic as synthetic license plate.

Table 4.13: Evaluation Result of All Model on Testing Dataset (Synthetic)

Prefix	Number of Testing Image	Mean Test Full Sequence Accuracy		
		Model B	Model B	Model C
A1234	1000	65.60	64.00	48.40
A1234B	1000	53.30	55.10	0.00
AB1234	1000	52.00	55.90	49.3
AB1234C	1000	42.70	44.50	0.00
ABC1234	1000	43.50	44.80	42.6
ABC1234D	1000	30.20	32.30	0.00
Special	1000	0.05	0.03	0.02

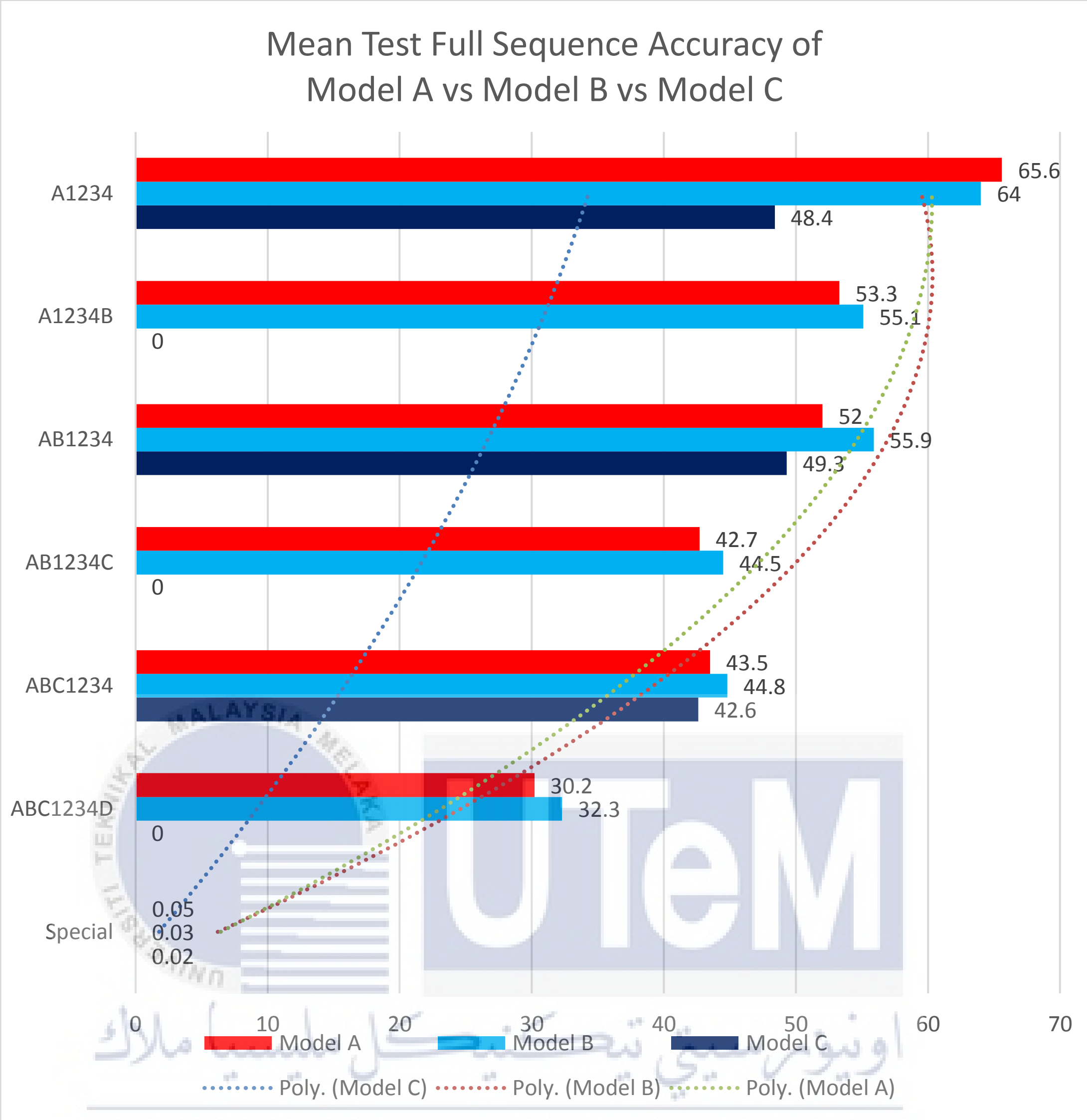


Figure 4.11: Mean Test Full Sequency Accuracy of All Models on Testing Dataset (Synthetic)

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1 Conclusion

Throughout the analysis there are few points can be concluded. First, preparing Malaysia license plate image for automatic license plate recognition can be just two categories, which are Alphabet-Digit and Alphabet-Digit-Alphabet. Secondly, the difficulty of recognize the 7 prefixes from the highest to the lowest is Special > ABC1234D > ABC1234 > AB1234C > AB1234 > A1234B > A1234.

After using a pre trained weight which train with synthetic license plate and duplication of license plate with inverted colour, less amount of dataset is needed to achieve higher accuracy. Last but not least, synthetic license plate has a different characteristic with real license plate, it is not suitable to include in training for recognize real license plate.

From this work, license plate images with prefix of A1234, A1234B, AB1234 and AB1234C can achieve recognition accuracy with at least 80% by firstly pre trained it with synthetic license plate images and subsequently continue train with 800 images duplicated with color inverted.

In addition, a license plate detector is encouraged to be prepared before the training of automatic license plate recognition as the size of license plate cropped by the license plate detector will affect the performance of recognition.

5.2 Future Works

This analysis will be able to provide a good guidance in development of automatic Malaysia license plate recognition system with segmentation free convolutional recurrent neural network.

Besides that, the view of the license plate still needs to be further analysis since it is not concern in this analysis but it might affect the license plate recognition accuracy.

REFERENCES

[1] Jabatan Pengangkutan Jalan Malaysia, “Plate Number Specification” [Online]. Available: https://www.jpj.gov.my/en/web/main-site/kenderaan1-en/-/knowledge_base/vehicle/plate-number-registration. [Accessed: 24-June.-2021].

[2] World Heritage Encyclopedia, “Vehicle Registration of Malaysia” [Online]. Available: http://self.gutenberg.org/articles/eng/Vehicle_registration_plates_of_Malaysia. [Accessed: 24-June.-2021].

[3] R. Laroca et al., “A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector,” in Proceedings of the International Joint Conference on Neural Networks, 2018, vol. 2018-July, doi: 10.1109/IJCNN.2018.8489629.

[4] C. H. Lin, Y. S. Lin, and W. C. Liu, “An efficient license plate recognition system using convolution neural networks,” 2018, doi: 10.1109/ICASI.2018.8394573.

[5] B. Shi, X. Bai, and C. Yao, “An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 11, 2017,

doi: 10.1109/TPAMI.2016.2646371.

[6] R. Messina and J. Louradour, “Segmentation-free handwritten Chinese text recognition with LSTM-RNN,” in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 2015, vol. 2015-November, doi: 10.1109/ICDAR.2015.7333746.

[7] A. Lim, “Vehicles Registrations in Malaysia – 31.2 mil as of 2019,” paultan.org, 2020.

[8] D. of S. Malaysia, “Current Population Estimates, Malaysia, 2020,” Dep. Stat. Malaysia, no. July, 2020.

[9] C. Nie, H. Wei, J. Shi, and M. Zhang, “Optimizing actuated traffic signal control using license plate recognition data: Methods for modeling and algorithm development,” Transp. Res. Interdiscip. Perspect., vol. 9, 2021, doi: 10.1016/j.trip.2021.100319.

[10] Dongare, A. D., R. R. Kharde, and Amit D. Kachare. "Introduction to artificial neural network." International Journal of Engineering and Innovative Technology (IJEIT) 2.1 (2012): 189-194.

[11] O'Shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint arXiv:1511.08458 (2015).

[12] Prabhu. "Understanding of Convolutional Neural Network (CNN) — Deep Learning." (2018). [Online]. Available: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148 [Accessed: 24-June.-2021].

[13] Rana, K. "Pooling Layer — Short and Simple." (2020). [Online]. Available: https://ai.plainenglish.io/pooling-layer-beginner-to-intermediate-fa0dbdce80eb [Accessed: 24-June.-2021].

[14] Pranoy Radhakrishnan. " Introduction to Recurrent Neural Network." (2017) [Online]. Available: https://towardsdatascience.com/introduction-to-recurrent-neural-network-27202c3945f3 [Accessed: 24-June.-2021].

[15] Victor Zhou. "An Introduction to Recurrent Neural Networks for Beginners." (2019) [Online]. Available: https://towardsdatascience.com/an-introduction-to-recurrent-neural-networks-for-beginners-664d717adbd [Accessed: 24-June.-2021].

[16] Avijeet Biswal. " Recurrent Neural Network (RNN) Tutorial for Beginners." (2021) [Online]. Available: https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn [Accessed: 24-June.-2021].

[17] C. Olah, “Understanding LSTM Networks [Blog],” Web Page, 2015.

[18] Immersivelimit. " Create COCO Annotations From Scratch." (2019) [Online]. Available: https://www.immersivelimit.com/tutorials/create-coco-annotations-from-scratch. [Accessed: 24-June.-2021].

[19] Khandelwal, R. "COCO and Pascal VOC data format for Object detection." (2019) [Online]. Available: https://towardsdatascience.com/coco-data-format-for-object-detection-a4c5eaf518c5 [Accessed: 24-June.-2021].

[20] R. Sathya and A. Abraham, “Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification,” Int. J. Adv. Res. Artif. Intell.,

vol. 2, no. 2, 2013, doi: 10.14569/ijarai.2013.020206.

[21] W. Kurdthongmee, “A comparative study of the effectiveness of using popular DNN object detection algorithms for pith detection in cross-sectional images of parawood,” Heliyon, vol. 6, no. 2, 2020, doi: 10.1016/j.heliyon.2020.e03480.

[22] C. Y. Lee and S. Osindero, “Recursive Recurrent Nets with Attention Modeling for OCR in the Wild,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016, vol. 2016-December, doi: 10.1109/CVPR.2016.245.

[23] Mustamo, Pirkko. "Object detection in sports: TensorFlow Object Detection API case study." University of Oulu (2018).

[24] Tay Choon Kiat. “EasyOCRGen.” (2019) [Online]. Available: https://github.com/choonkiattay/EasyOCRGen [Accessed: 24-June.-2021].