

**RECOGNISING MALAYSIA
TRAFFIC SIGN WITH
RAINDROP DISTURBANCE
USING DEEP LEARNING**

PANG WAN CHEE

**BACHELOR OF SCIENCE
(HONS) STATISTICAL
COMPUTING AND
OPERATIONS RESEARCH**

**FACULTY OF SCIENCE
UNIVERSITI TUNKU
ABDUL RAHMAN
MAY 2022**

PANG WAN CHEE
B.Sc. (Hons) Statistical Computing
and Operations Research
2022

**RECOGNISING MALAYSIA TRAFFIC SIGN WITH RAINDROP
DISTURBANCE USING DEEP LEARNING**

By

PANG WAN CHEE

A project report submitted to the Department of Physical and Mathematical
Science

Faculty of Science

Universiti Tunku Abdul Rahman

in partial fulfilment of the requirements for the degree of
Bachelor of Science (Hons) Statistical Computing and Operations Research

May 2022

ABSTRACT

RECOGNISING MALAYSIA TRAFFIC SIGN WITH RAINDROP DISTURBANCE USING DEEP LEARNING

Pang Wan Chee

Traffic sign recognition is part of the driving assistance systems that can monitor driving by automatically recognising the traffic signs on the road and alerting drivers. In Malaysia, road accidents have remained one of the top principal causes of death in recent years. Other than the driver's carelessness, weather such as rain might challenge the visibility and recognition of the traffic signs. Hence, we develop a road sign recognition system based on a deep learning algorithm that can identify the Malaysia traffic sign with raindrop disturbance. In this study, a convolutional neural network (CNN) is applied as it is a deep neural network that is well-known for effective image recognition. The algorithms are developed in Python language using Anaconda Spyder as the software. There are 18 different combinations of the epochs, dropout rates, and colour of the image input in the first convolutional layer (either colour or greyscale version) used in this study. In conclusion, we proposed an image recognition system with a predictive accuracy of 99.52% for Malaysia traffic signs with raindrop disturbance.

ACKNOWLEDGEMENT

I am pleased to undertake the final year project entitled “Recognising Malaysia traffic sign with raindrop disturbance using Deep Learning”. I would like to take this opportunity to thank every individual involved in this project. High appreciation is given to my supervisor, Mr. Looi Sing Yan, a lecturer at the Faculty of Science in Universiti Tunku Abdul Rahman, for his strong and timely support in developing my project and writing this report. I would like to express my deepest gratitude to my external consultant, Ts. Dr. Lee How Chinh, a senior lecturer at the School of Business in Monash University Malaysia, for providing me with his invaluable guidance throughout this project. Most importantly, I am grateful to Universiti Tunku Abdul Rahman, for allowing me to study for the programme Bachelor of Science (Hons) Statistical Computing and Operations Research. Lastly, special thanks to my family members and friends who encourage me to complete this project.

DECLARATION

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



Pang Wan Chee

APPROVAL SHEET

This final year project report entitled “**RECOGNISING MALAYSIA TRAFFIC SIGN WITH RAINDROP DISTURBANCE USING DEEP LEARNING**” was prepared by PANG WAN CHEE and submitted as partial fulfilment of the requirements for the degree of Bachelor of Science (Hons) Statistical Computing and Operations Research at Universiti Tunku Abdul Rahman.

Approved by:



(Mr. Looi Sing Yan)

Date: ...30.3.2022.....

Supervisor

Department of Physical and Mathematical Science

Faculty of Science

Universiti Tunku Abdul Rahman

FACULTY OF SCIENCE
UNIVERSITI TUNKU ABDUL RAHMAN

Date: 13 April 2022

PERMISSION SHEET

It is hereby certified that **PANG WAN CHEE** (ID No: **18ADB03015**) has completed this final year project report entitled “RECOGNISING MALAYSIA TRAFFIC SIGN WITH RAINDROP DISTURBANCE USING DEEP LEARNING” under the supervision of Mr Looi Sing Yan (Supervisor) from the Department of Physical and Mathematical Science.

I hereby give permission to the University to upload the softcopy of my final year project report in pdf format into the UTAR Institutional Repository, which may be made accessible to the UTAR community and public.

Yours truly,



(PANG WAN CHEE)

TABLE OF CONTENTS

	Page
ABSTRACT	ii
ACKNOWLEDGEMENT	iii
DECLARATION	iv
APPROVAL SHEET	v
PERMISSION SHEET	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	ix
LIST OF FIGURES	x

CHAPTER

1	INTRODUCTION	1
2	LITERATURE REVIEW	3
3	CONCEPTS OF DEEP LEARNING AND CONVOLUTIONAL NEURAL NETWORK (CNN)	7
	3.1 Introduction to Machine Learning and Deep Learning	7
	3.2 Introduction to Neural Network	8
	3.2.1 Activation function	8
	3.2.2 Loss function	9
	3.2.3 Train a neural network	9
	3.3 Convolutional Neural Network (CNN)	10
	3.3.1 Input	11
	3.3.2 Convolutional layer	11
	3.3.3 Layers	13
	3.4 Techniques to prevent overfitting	14
	3.4.1 Data augmentation	14
	3.4.2 Dropout layer	15

4	RESEARCH METHODOLOGY	16
4.1	Data exploration	16
4.2	Data preparation	17
4.3	Convolutional Neural Network (CNN)	19
5	RESULTS AND DISCUSSION	22
5.1	Results	22
5.2	Discussion	25
5.3	Model generalisation	26
6	CONCLUSION AND FUTURE WORK	28
6.1	Conclusion	28
6.2	Future research	28
	REFERENCES	30
	APPENDICES	34

LIST OF TABLES

Table		Page
1	Traffic sign class ID and name in the dataset	17
2	Steps of data pre-processing	18
3	CNN architecture	20
4	Output shape and number of trainable parameters in each layer of the CNN model	22
5	Results of training CNN with 18 different combinations of epochs and dropout rates with (a) colour input and (b) greyscale input	24
6	The macro and weighted precision, recall and F1 score	26

LIST OF FIGURES

Figure		Page
1	Examples of the traffic sign scenes in EMTD	16
2	Distribution of traffic sign images extracted from EMTD before elimination (left) and after elimination with dataset expansion (right)	18
3	Examples of the augmented traffic sign images in training set for (a) greyscale version and (b) colour version	20
4	Accuracy (left) and loss (right) over the epoch of the selected model	25

CHAPTER 1

INTRODUCTION

Traffic sign recognition is the ability in recognising the traffic sign on the road. Nowadays, there are many studies on traffic sign recognition due to its importance in the development of the Advanced Driver Assistance System (ADAS) as well as autonomous driving. Generally, there are two stages in a traffic sign recognition system, which are detection and recognition. During driving, the presence of a traffic sign would be detected in real-time, and its class would be identified. Based on the traffic sign class, the traffic sign recognition system assists in monitoring driving and alerting drivers of traffic rules. While both detection and recognition form an integral part of a traffic sign recognition system, this study mainly focuses on traffic sign recognition.

Recent research on traffic sign recognition is conducted based on deep learning algorithms, a subset of machine learning that can automatically learn features from training data through multiple-layer neural networks. This is because deep neural networks do not require handcrafted feature engineering but are proven to provide high accuracy. Nonetheless, traffic sign recognition is still a challenging computer vision task as traffic signs in different countries might have a certain degree of differences, although the Vienna Convention and Manual on Uniform Traffic Control Devices are the global major standards (Madani and Yusof, 2016). For example, a slippery road sign in Malaysia is yellow in background colour and is diamond-shaped with a black border and a

black pictogram at the centre. Whereas in German, a slippery road sign contains a similar pictogram, but the road sign background is white, and it is triangular shaped with a red border. Besides, traffic sign recognition in Malaysia is challenging due to the unavailability of large Malaysia traffic sign datasets.

In Malaysia, transport accidents remain the 4th principal cause of death from 2016 to 2020 (Department of Statistics Malaysia, 2017; 2018; 2019; 2020; 2021). Malaysia traffic sign recognition is, therefore, an important component of ADAS to reduce the accident rate. To the best of our knowledge, there is no existing research that investigates Malaysia traffic sign recognition in poor weather conditions, such as rainy conditions which might challenge the visibility and recognition of the traffic signs.

To fill the mentioned research gap, the objectives of this study are as follows:

- i. to develop a deep learning algorithm that can recognise Malaysia traffic signs with raindrop disturbance using a Convolutional Neural Network (CNN), and
- ii. to achieve a classification accuracy above 0.95 on the testing dataset.

This proposed traffic sign recognition will give impetus to the development of the autonomous vehicles industry. The limitation of this study is the limited dataset used to train the CNN models due to the unavailability of publicly available and large Malaysia traffic sign datasets.

CHAPTER 2

LITERATURE REVIEW

Numerous research has been carried out on traffic sign recognition. One of the famous datasets used in these studies is the German Traffic Sign Recognition Benchmark (GTSRB), a publicly available dataset with colour German traffic signs which was provided by a competition namely International Joint Conference Neural Network (IJCNN) in the year 2011 (GTSRB, 2019). Belghaouti, Handouzi, and Tabaa (2020) trained the LeNet model, a classical Convolutional Neural Network (CNN) architecture developed by Lecun, et al. (1998) to recognise the GTSRB. An accuracy of 95.80% was achieved on the testing set. The authors then modified the LeNet model architecture by adding convolutional layers and reported an improved testing accuracy of 98.02%. Yet, there was an overfitting issue on the proposed model as its training accuracy was higher than the testing accuracy, despite applying data augmentation during training to solve the problem that GTSRB with an imbalanced class distribution.

Cao, et al. (2019) used a similar LeNet model compared to that used by Belghaouti, Handouzi, and Tabaa (2020), but modified the initial convolutional kernel and applied batch normalisation. Besides, the authors expanded the GTSRB dataset by performing random sampling to reduce the impact of class imbalance on the model performance. The results showed an average recognition rate of 99.75% based on six traffic sign types.

Also, Velamati and Gopichand (2021) proposed a traffic sign recognition system based on a modified LeNet model similar to the one suggested by Belghaoui, Handouzi, and Tabaa (2020) but without the use of dropout, achieving a testing accuracy of 96% on GTSRB testing dataset. While most of the studies on recognizing GTSRB in our literature review had suggested that the colour traffic sign images were converted to greyscale in data pre-processing to improve the efficiency of the model's learning process, this study was insisted to retain the images in the colour version so that most of the properties of the images were preserved for the model's learning. Next, Quan and Xiong (2019) proposed a convolution-based neural network called ENet to recognise the GTSRB. Besides achieving an accuracy of 98.6% on the testing set of GTSRB, the proposed model is well generalised as it correctly predicted the classes of another five unseen German traffic sign images. This study also highlighted that the model with a smaller number of parameters can identify the testing set faster.

There are a few related works on Malaysia traffic sign recognition. According to Lau, Lim and Gopalai (2015), the performance of recognising 100 classes of greyscale Malaysia traffic signs using a CNN with LeNet model architecture was compared against that of a radial basis function neural network (RBFNN). From the simulation results, the CNN as a deep architecture neural network had an accuracy of 99% on the testing set which was comparatively higher than that of the RBFNN, which is a shallow architecture neural network. Furthermore, the same accuracy was still achieved by CNN when it was tested with Malaysia traffic signs that were added with Gaussian white noise, unlike the RBFNN which showed a large decrease in the testing accuracy. This showed that CNN

had better adaptability than RBFNN. Neoh, et al. (2019), on the other hand, utilised 13 different pre-trained CNN models and retrained their top layers to recognise five classes of Malaysia traffic signs. It was reported that the DenseNet169 received colour traffic sign images as input achieved the highest accuracy of 98.33% on the testing set.

Besides, Islam and Raj (2017) proposed a road sign detection and recognition system. In the recognition system, an artificial neural network (ANN) based on custom feature extraction was used for recognising ten classes of greyscale traffic signs. The ANN contained a hidden layer with ten neurons. Although they achieved an accuracy of 100%, a small testing dataset with only ten traffic signs is not convincing enough to argue the consistency of ANN.

Other than that, Madani and Yusof (2016) proposed a Malaysia traffic sign dataset consisting of two categories: the detection dataset and the recognition dataset. The detection set contained 1000 different traffic sign scenes captured during the day, night, and rainy weather. The traffic signs were then extracted and resized into 32 x 32 pixels to form the recognition dataset. This research further sparks our interest as only 9.5%, which was only a small portion of the traffic sign scenes in the proposed detection dataset were taken during rain. This study only focused on creating ground truth files that contained the traffic sign attributes including the sign type and class ID. No detection and recognition techniques were applied to the proposed traffic sign dataset.

Studies on Malaysia traffic sign recognition using Convolutional Neural Network (CNN) are lacking in the literature review. On top of that, In Malaysia, road accidents have remained one of the top principal causes of death in recent years. Lau, Lim and Gopalai (2015) suggested that the environmental condition could challenge the accuracy of the traffic sign recognition system. Poor weather conditions will affect the traffic signs recognition by the human as well as computer vision systems. Furthermore, Malaysia is a country with a tropical climate where rain is a common weather condition. To our best knowledge, there is no existing research investigating the recognition of Malaysia traffic signs in poor weather conditions, specifically in rainy conditions. Our objective is therefore to develop a deep learning algorithm to recognise Malaysia traffic signs with raindrop disturbance using CNN.

CHAPTER 3

CONCEPTS OF DEEP LEARNING AND CONVOLUTIONAL NEURAL NETWORK (CNN)

3.1 Introduction to Machine Learning and Deep Learning

A machine learning algorithm can automatically build an analytical model based on the training data consisting of input and output pairs to perform associated tasks such as prediction (Janiesch, Zschech and Heinrich, 2021). For example, in an image classification task, a machine learning algorithm is trained with many images together with their tags. Based on the relationship between input (images) and output (tags), the algorithm searched and learned suitable representations for the images such that these data are transformed to predict the tags of the images. The distance between the actual and predicted output such as loss is often used as a feedback signal to adjust the algorithm (Chollet, 2018). This is how the algorithm can “learn” from the data.

According to Chollet (2018), deep learning is a subset of machine learning, that describes the ability to learn meaningful representations from training data via neural networks. According to Janiesch, Zschech and Heinrich (2021) and Mostafa, et al. (2020), unlike traditional machine learning, a deep neural network does not require handcrafted feature engineering that is time-consuming, instead, it can automatically learn feature representations from training data. The advantage of automated feature learning makes deep neural networks effective in many applications.

3.2 Introduction to Neural Network

The concept of a simple neural network is discussed before one of the deep neural networks, Convolutional Neural Network (CNN), is introduced. A neural network consists of artificial neurons connected. When an input vector $\mathbf{x} = x_1, x_2, \dots, x_n$ connected to a neuron is weighted \mathbf{w} , the transformation below is performed using an activation function σ to obtain the output a :

$$a = \sigma(\mathbf{w}\mathbf{x} + b)$$

where b is the bias value (Maggiori, et al., 2017). Both the \mathbf{w} and b are the weights of the layer, which are the trainable parameters.

3.2.1 Activation function

To introduce nonlinearity in the neural network, Rectified Linear Unit (ReLU) activation function is commonly used in hidden layers since it has less gradient diffusion problem that exists when using other traditional activation functions such as Hyperbolic Tangent and Sigmoid functions (Wang, et al., 2020). The equation of ReLU is as follows:

$$f(c) = \max(0, c),$$

where c is referred to the $\mathbf{w}\mathbf{x} + b$ introduced earlier. In multiclass, single-label classification problems, the Softmax activation function is generally used in the output layer to obtain the predicted output \hat{y} . Suppose in a \mathcal{L} -class problem, the probability of an input X belonging to class k can be calculated by the Softmax function using the equation below:

$$\hat{y}_k = p(X = k) = \frac{e^{z_k}}{\sum_{i=1}^{\mathcal{L}} e^{z_i}}, k = 1, 2, \dots, \mathcal{L}$$

where z denotes the value received at a node of the output layer (Kotu and Deshpande, 2019). Then, the input X will be predicted as the class with the highest probability (Fernando and Tsokos, 2021).

3.2.2 Loss function

Loss is the distance between the predicted output $\hat{y}^{(i)}$ and actual output $y^{(i)}$ for the given n inputs. The choice of loss function depends on the application. It is common to use categorical cross-entropy loss function in a \mathcal{L} -class problem:

$$L = -\frac{1}{n} \sum_{i=1}^n \sum_{k=1}^{\mathcal{L}} y_k^{(i)} \log \hat{y}_k^{(i)}, i = 1, 2, \dots, n$$

where $y_k^{(i)} = 1$ if the sample i belongs to class k , and 0 otherwise (Fernando and Tsokos, 2021; Maggiori, et al., 2017).

3.2.3 Train a neural network

According to Blanchard (2020), Chollet (2018), and Maggiori, et al. (2017), the weights of the layers are initialized with random values and are learned through the following steps iteratively:

1. In the forward phase, a batch of training data is fed into successive layers to obtain the prediction.
2. The loss of the network on the batch is calculated.

3. The backpropagation algorithm is used to compute the derivative of the loss with respect to all the weights, $\frac{\partial L}{\partial w_i}$ to identify the gradient of the loss. With that, in the backward phase, gradient descent can be performed, i.e., the weights are slightly moved in the opposite direction of the gradient with a learning rate λ to decrease the loss:

$$w_i \leftarrow w_i - \lambda \frac{\partial L}{\partial w_i}$$

After several epochs, the loss of the trained neural network would be small and can be used for making predictions. In addition, the optimiser will determine how the weights are modified to optimise the objective function: to minimise the loss.

3.3 Convolutional Neural Network (CNN)

A deep neural network is an artificial neural network, but it consists of multiple layers that successively transform the input into the outputs. These added layers are known as hidden layers due to their invisibility (Aggarwal, 2018). Convolutional Neural Network (CNN) is a deep neural network that is powerful for image classification problems (Indolia, et al., 2018).

In training a CNN, input data will be fed into multiple hidden layers: firstly, convolutional layers to extract the features of the input and reduce the spatial dimension of the input. Secondly, pooling layers enhance the features and further reduce the spatial dimension of data. Then the output will be fed into the dense layer before producing the final output, such as predicting the class for an image (Indolia, et al., 2018).

3.3.1 Input

In image classification tasks, images are the input data that will be fed into the subsequent hidden layers. To computers, an image is a 2-dimensional array of numbers represented in pixel values that are ranged from 0 to 255 (Aggarwal, 2018). A greyscale image has one channel. On the other hand, a colour image consists of 3 channels which are the red channel, green channel, and blue channel (Blanchard, 2020). Each of these colour channels is represented by a matrix of values that denote the red, green, and blue intensity, respectively.

3.3.2 Convolutional layer

In convolution operation, a filter that is a matrix slides across all the possible small regions of input so that certain features of the input are extracted. Dot products are performed between the filter and every small region of the input, bias is added, and activation function is used before producing a feature map. A spatial position (i, j) of input is passed for a convolution operation $*$ to produce a feature map a_{ij} :

$$a_{ij} = \sigma((\mathbf{w} * \mathbf{x})_{ij} + b)$$

where \mathbf{w} is the weight vector in the filter, \mathbf{x} is the input vector, b is the bias value, and σ is the activation function (Maggiori, et al., 2017), the values of the filter and bias are parameters (or weights) of the convolutional layer that are learned using backpropagation (Blanchard, 2020).

The convolution reduces the spatial dimension of inputs. Suppose in 2-dimensional CNN which is commonly used for image classification. In the q^{th} layer, a convolution operation is performed between an input (with length L_q and breadth B_q) and a square-shaped filter (with length F_q and breadth F_q). According to Aggarwal (2018), the length and breadth of the feature map, or in other words, the input of the $(q+1)^{\text{th}}$ layer is as follows:

$$L_{q+1} = L_q - F_q + 1$$

$$B_{q+1} = B_q - F_q + 1$$

Yet, the expression above is valid when convolution is applied at each position of the input. Strides can be used to control the movement of the filter across the input dimension including width and height (Blanchard, 2020). According to Aggarwal (2018), if the stride of S_q is used when performing convolution, the resulting feature maps have the following length and breadth:

$$L_{q+1} = (L_q - F_q)/S_q + 1$$

$$B_{q+1} = (B_q - F_q)/S_q + 1$$

Furthermore, multiple filters can be used to produce an output volume that contains multiple feature maps (Aggarwal, 2018). Hence, the number of filters used in the q^{th} layer will determine the number of feature maps, which is also the depth in the $(q+1)^{\text{th}}$ layer.

In CNN, even if the position of a pattern in an image is shifted, the pattern can be still recognised. This is known as translation equivariance. This is due to parameter sharing in CNN, which means that the same filters are applied across the input in convolution operation (Aggarwal, 2018). A shift in input pixel values

will cause a corresponding shift in the feature maps pixel values when convolution is applied. On top of that, CNN has the characteristic of local connectivity since each neuron is connected only to a small region of the input compared to ANN where each neuron is densely connected.

3.3.3 Layers

Similar to the convolution operation, the spatial dimension of feature maps is reduced by the pooling operation (Aggarwal, 2018). According to Blanchard (2020), pooling operation is performed by applying filters, however, there is no learnable parameter in the filters. Instead, the filters summarized the information in every small region of input, such as returning the maximum or average values in each small region (Aggarwal, 2018). These two types of pooling operations are known as max pooling and average pooling. Max pooling is more often used to enhance the feature of an input image by highlighting its most dominant feature (Kotu and Deshpande, 2019).

According to Aggarwal (2018), suppose in the q^{th} layer, pooling operation with size $P_q \times P_q$ is applied on the input with size $L_q \times B_q$, the length and breadth of the resulting feature map are as follow:

$$L_{q+1} = L_q - P_q + 1$$

$$B_{q+1} = B_q - P_q + 1$$

Similar to convolution, if the stride is applied when performing the pooling operation, the length and breadth of the resulting feature map are the following:

$$L_{q+1} = (L_q - P_q)/S_q + 1$$

$$B_{q+1} = (B_q - P_q)/S_q + 1$$

Besides, the pooling operation is performed independently at every feature map, unlike the convolution operation (Aggarwal, 2018). Hence, the number of feature maps before and after the pooling operation is the same.

Flatten layer is used to flatten the feature map into a 1-dimensional vector, before going through the dense layer (Chollet, 2018). In the dense layer, every neuron is fully connected with those in the previous layer (Blanchard, 2020). Whereas the output layer is a dense layer that is designed depending on the application (Aggarwal, 2018). In the \mathcal{L} -class problem, the number of nodes is the same as the number of classes.

3.4 Techniques to prevent overfitting

3.4.1 Data augmentation

Data augmentation refers to data augmentation that is performed during training to increase the model generalisation (Chollet, 2018). In computer vision tasks, random transformations such as horizontal shift, vertical shift, zoom in, zoom out, shear angle, and rotation can be applied to the existing training data such that the model will only see an image input once. Nevertheless, the fact that the augmented data come from a small set of images might not be sufficient to prevent the model from overfitting, highlighting the need of adding a dropout layer to the model, which is discussed in the next subsection.

3.4.2 Dropout layer

Dropout is a regularisation technique that is used to introduce noise to the learning process (Blanchard, 2020). This is to prevent the weights from co-adapting, thus reducing the overfitting of the model. During the training phase, some nodes are randomly dropped at a probability called as “dropout rate” (Kotu and Deshpande, 2019). Before running the next training epoch, the original model would be restored, and some other nodes would be randomly dropped.

CHAPTER 4

RESEARCH METHODOLOGY

4.1 Data exploration

Extended Malaysian Traffic Sign Dataset (EMTD), which consists of 1413 Malaysia traffic sign scenes, is used in this study (Sean Reilly, et al., 2018). Some of the traffic signs scenes are shown in Figure 1. By observation, scenes are collected from Google Street and captured using the camera. A total of 2540 traffic sign images are extracted from the scenes in EMTD. All these images are resized into 32 x 32 pixels as the input size for the Convolutional Neural Network (CNN) model is fixed in this study. Then, all traffic sign images are grouped into 71 classes such that each image is tagged with an associated label, i.e., class ID. We eliminated the classes with less than ten images to avoid model overfitting if these classes rely heavily on dataset expansion in section 4.2 later. There are 50 classes of traffic signs with a total of 2449 traffic sign images after elimination. Figure 2 (left) shows the distribution of traffic signs classes before elimination.

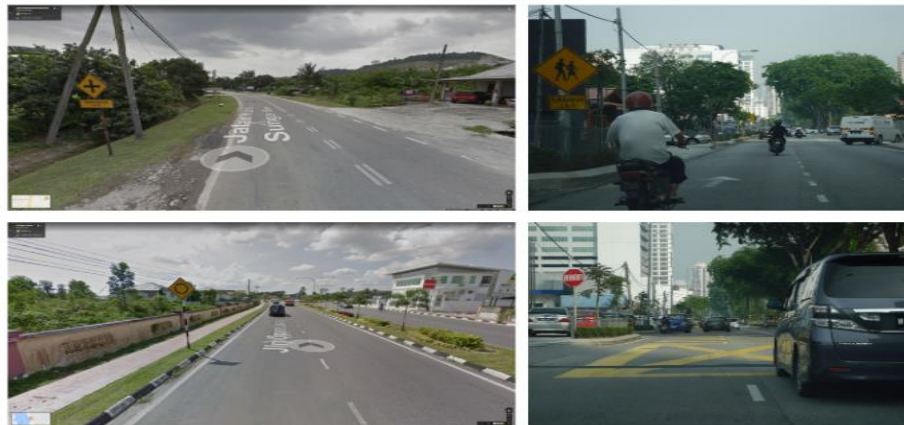


Figure 1: Examples of the traffic sign scenes in EMTD

Table 1: Traffic sign class ID and name in the dataset

Class ID	Traffic sign name	Class ID	Traffic sign name
0	jalan bercabang / berpisah arah	25	penanda pelepasan
1	halangan di hadapan	26	jalan sempit dari sebelah kiri
2	tanda arah selekoh ke kiri	27	persimpangan jalan (simpang empat)
3	tanda arah selekoh ke kanan	28	bulatan
4	laluan berbonggol	29	had laju 110km / j
5	beri laluan	30	lintasan pejalan kaki (kanan)
6	berhenti di simpang jalan	31	dilarang berpusing balik
7	lampu isyarat di hadapan	32	laluan motosikal sahaja
8	persimpangan jalan kecil di sebelah kiri	33	liku kiri
9	jalan bercabang / berpisah arah (kiri)	34	had laju 90km / j
10	simpang kiri	35	had laju 30km / j
11	jalan bercabang / berpisah arah (kanan)	36	dilarang membelok ke kiri
12	had laju 60km / j	37	had laju 40km / j
13	dilarang berhenti	38	tanda arah kiri dan kanan
14	dibenarkan berpusing balik	39	lintasan pejalan kaki (kiri)
15	had laju 80km / j	40	jalan menyimpang di sebelah kiri
16	laluan berbonggol di hadapan	41	dilarang membelok ke kanan
17	kawasan murid sekolah melintas	42	persimpangan jalan (simpang t)
18	dilarang meletak kenderaan	43	jambatan sempit
19	kenderaan dilarang masuk	44	zon kamera
20	zon tunda	45	kenderaan berat dilarang masuk
21	awas / beri tumpuan	46	had laju 70km / j
22	had laju 50km / j	47	dilarang memotong
23	kawasan pembinaan / kerja pembinaan sedang dijalankan	48	jalan bercabang / berpisah arah
24	simpang kanan	49	kenderaan melebihi 5T dilarang masuk

4.2 Data preparation

The algorithms are developed in Python language using Anaconda Spyder as the software. To reduce the class imbalance issue, the traffic sign images are randomly and minorly transformed including rotation, horizontal shift, vertical shift, and shear, using the Scikit-image library. In Figure 2 (right), it is shown that the distribution after the traffic sign images is expanded to a total of 6200

images by combining the original images with the randomly transformed images. Table 1 shows the list of the traffic sign classes and their respective names in the Malay language. Ten to fifteen short, slanted lines are added at random positions on all the traffic sign images using the OpenCV library, to create a dataset with Malaysia traffic signs with raindrop disturbance. Images in each class are assigned to the training set, validation set and testing set following the proportion of 60%, 20%, and 20%, respectively.

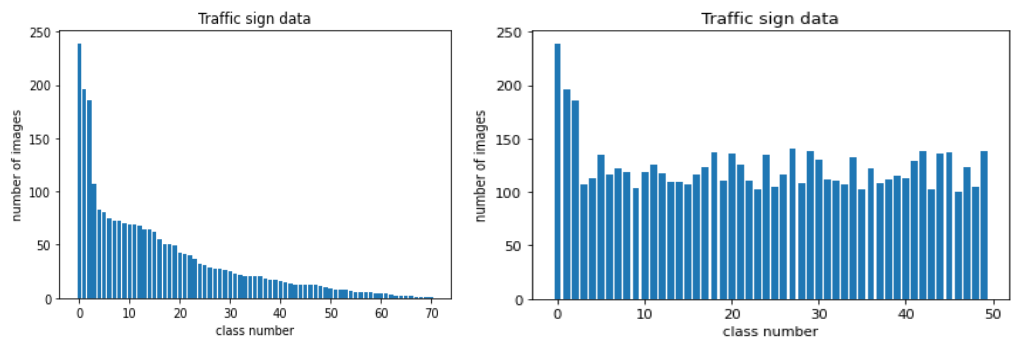


Figure 2: Distribution of traffic sign images extracted from EMTD before elimination (left) and after elimination with dataset expansion (right)

Table 2: Steps of data pre-processing

Training CNN with colour input	Training CNN with greyscale input
Step 1: equalisation	Step 1: greyscale conversion
Step 2: normalisation	Step 2: equalisation
	Step 3: normalisation

Table 2 shows the comparison of data pre-processing before training the CNN models with colour input and greyscale input, respectively in the first convolutional layer. A greyscale conversion on the image is implemented to investigate the colour effect on the classification accuracy. Before the training on the CNN models with colour inputs, two steps of data pre-processing are applied. First, equalisation is used to standardise the lighting of the images.

Second, normalisation is used to rescale the images' pixel values into smaller values which are between 0 to 255 into 0 to 1. While before the CNN models with greyscale inputs are trained, an additional step is performed to convert the coloured traffic sign images into greyscale before equalisation and normalisation are applied. Besides, one hot coding is applied as a technique to encode the categorical values of all the images. There are 50 binary variables created for each traffic sign class ID. For each observation, the value of one indicates the presence of the class and zero otherwise.

4.3 Convolutional Neural Network (CNN)

There are three characteristics of the Convolutional Neural Network (CNN) including local connectivity, parameter sharing and shift-invariant, that make it effective in image recognition. Table 3 shows the CNN architecture with 11 layers where the pre-processed and labelled traffic sign images in the training set are fed. These training data are fed into the model in every epoch so that the model is learned from the exposure to the data. The architecture is similar to the modified LeNet model proposed by Belghaouti, Handouzi, and Tabaa (2020), with modification by adding a dropout layer after the sixth layer, i.e., max pooling layer, as a regularisation technique to reduce overfitting issues. During the training process, the images in the training set are augmented on-the-fly using the Keras ImageDataGenerator class to closely mimic real-world road scenarios to increase the model generalisation. Note that this is different from the dataset generation mentioned in section 4.2 previously. Only the augmented batches of images are trained but the original batches of images are not used for training the CNN models (Rosebrock, 2019). Some of the augmented images are shown

in Figure 3. Right after completing each training epoch, images in the validation set are used to evaluate the model before running the next epoch.

Table 3: CNN architecture

Layer	Detail
1 st	Convolutional layer, filter size: 5x5, quantity of filter: 60, activation function: ReLU, stride: 1
2 nd	Convolutional layer, filter size: 5x5, quantity of filter: 60, activation function: ReLU, stride: 1
3 rd	Max pooling layer, pooling size: 2x2, stride: 2
4 th	Convolutional layer, filter size: 3x3, quantity of filter: 30, activation function: ReLU, stride: 1
5 th	Convolutional layer, filter size: 3x3, quantity of filter: 30, activation function: ReLU, stride: 1
6 th	Max pooling layer, pooling size: 2x2, stride: 2
7 th	Dropout layer
8 th	Flatten layer
9 th	Dense layer, quantity of nodes: 500, activation function: ReLU
10 th	Dropout layer
11 th	Dense layer, quantity of nodes: 50, activation function: softmax

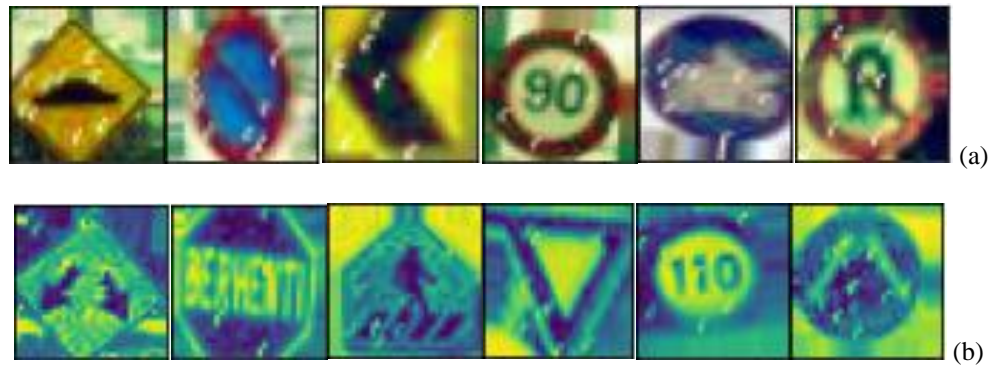


Figure 3: Examples of the augmented traffic sign images in training set for (a) greyscale version and (b) colour version

The model is trained using the Adam optimiser with a learning rate of 0.001, categorical cross-entropy loss function and batch size of 50. In each epoch, 200 batches of image data are trained. A total of 18 models with colour and greyscale input in the first convolutional layer, epochs of 10, 15, 20 and dropout rates of 0.1, 0.2, and 0.3 are constructed. To train the CNN models, the Keras library is

adopted with Tensorflow as the backend and the scripts are executed with the processor CPU Intel Core i5-8265U with 8GB of Random-access memory (RAM).

Since the model performance on the validation set is used as the feedback signal to tune the model parameters, the model might be overfitted to the validation set. The testing set is used to evaluate the model performance after the training process is completed. Since 18 models are being trained, their performances on the testing set are compared. The model with the best performance on the testing set is selected as our final model and interpretation is made based on the selected model.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Results

A total of 6200 images is divided into training, validation, and testing sets with a sample size of 3700, 1261 and 1239 images respectively. Referring to Table 4, the length and width of the feature maps are decreasing over layers. CNN models trained with colour input involve a total of 384,530 parameters, which contains 3000 more parameters compared to CNN trained with greyscale input. This is due to the difference in the input shape in the first layer of CNN. The colour image shape is (32, 32, 3) while the greyscale image shape is (32, 32, 1). The colour image that contains three channels, therefore, needs the filters with an additional depth of two in the first convolutional layer. As the first convolutional layer contains 60 filters with the size of 5 x 5, the training of CNN with colour inputs involves additional $(5 * 5 * 2 * 60) = 3000$ trainable parameters. Therefore, a longer time is needed to train the models with colour input, as shown in Table 5.

Table 4: Output shape and number of trainable parameters in each layer of the CNN model

Layer	Detail	Output shape	Number of parameters (Training CNN with colour input)	Number of parameters (Training CNN with greyscale input)
1 st	Convolutional layer	(28, 28, 60)	4,560	1,560
2 nd	Convolutional layer	(24, 24, 60)	90,060	

Table 4 continued: Output shape and number of trainable parameters in each layer of the CNN model

Layer	Detail	Output shape	Number of parameters (Training CNN with colour input)	Number of parameters (Training CNN with greyscale input)
3 rd	Max pooling layer	(12, 12, 60)	0	
4 th	Convolutional layer	(10, 10, 30)	16,230	
5 th	Convolutional layer	(8, 8, 30)	8,130	
6 th	Max pooling layer	(4, 4, 30)	0	
7 th	Dropout layer	(4, 4, 30)	0	
8 th	Flatten layer	(480)	0	
9 th	Dense layer	(500)	240,500	
10 th	Dropout layer	(500)	0	
11 th	Dense layer	(50)	25,050	
Total number of parameters			384,530	381,530

Table 5 shows the results of training Malaysia traffic signs images with raindrop disturbance using 18 different combinations of epochs and dropout rates with greyscale and colour image input in the first convolutional layer. Overall, a satisfactory average classification accuracy of 99.10% with a standard deviation of 0.23 is obtained. The nine CNN models trained with colour inputs achieved an average accuracy of 99.28% with a standard deviation of 0.13, while another nine CNN models trained with greyscale inputs achieved an average accuracy of 98.91% with a standard deviation of 0.14. The CNN models trained with greyscale input seem to have lower testing accuracies compared to those trained with colour input, but their difference in the standard deviation is relatively small.

From Table 5, it is observed that combinations 3 and 12 are overfitted as their accuracies on the testing set are lower compared to that on the training set. Furthermore, the higher the epoch value used, the longer the CNN models take to train. Despite that, the model trained with more epochs is not having higher accuracy. Out of the 18 combinations, combination 2 (CNN with colour input, 15 epochs and dropout rate of 0.1) is selected as it achieved the highest accuracy of 99.52% on the testing dataset.

Table 5: Results of training CNN with 18 different combinations of epochs and dropout rates with (a) colour input and (b) greyscale input

(a) Training with colour input

Combination	Epoch	Dropout rate	Accuracy (%)			Training execution time (in minutes)
			Training	Validation	Testing	
1	10	0.1	98.89%	99.29%	99.27%	81
2	15	0.1	99.12%	99.60%	99.52%	114
3	20	0.1	99.18%	99.37%	99.11%	165
4	10	0.2	98.34%	99.37%	99.27%	76
5	15	0.2	98.77%	99.37%	99.35%	113
6	20	0.2	98.91%	99.29%	99.19%	155
7	10	0.3	97.91%	99.21%	99.19%	80
8	15	0.3	98.39%	99.37%	99.44%	112
9	20	0.3	98.59%	99.05%	99.19%	154

(b) Training with greyscale input

Combination	Epoch	Dropout rate	Accuracy (%)			Training execution time (in minutes)
			Training	Validation	Testing	
10	10	0.1	98.67%	99.21%	98.79%	72
11	15	0.1	98.92%	98.81%	99.03%	106
12	20	0.1	99.16%	98.81%	99.03%	143
13	10	0.2	98.14%	99.44%	99.11%	71
14	15	0.2	98.59%	99.13%	98.87%	108
15	20	0.2	98.56%	98.89%	99.03%	143
16	10	0.3	97.31%	98.73%	98.79%	73
17	15	0.3	98.15%	98.49%	98.71%	105
18	20	0.3	98.12%	98.89%	98.87%	143

5.2 Discussion

Referring to Figure 4, the training accuracy at the first epoch was 82.35%, and it increased up to 95.72% at the second epoch. The highest training accuracy was obtained at the fifteenth epoch. The validation accuracy of the model was higher compared to training accuracy due to the application of the dropout layer. The training loss was the highest at the first epoch. It was observed that the model was in an efficient learning process as the training loss decreased by 77.54% after the second epoch and further decreased over epochs. However, the model might suffer from minor predictive errors as it has high testing accuracy (99.52%) and low loss (0.0487) after all the 15 epochs. Based on the result, a no parking road sign (class 13) is misclassified as a no stopping road sign (class 18) as both are blue circular signs with a red border. The only difference between these two road signs is that there is a red diagonal bar inside the no parking road sign, compared to the no stopping sign that contains a red cross. The misclassification due to the highly similar characteristics indicates that more training data are needed such that the algorithm provides a higher accurate classification rate. Out of 1239 testing images, 1233 images are correctly classified. Thus, the overall accuracy is 99.52%.

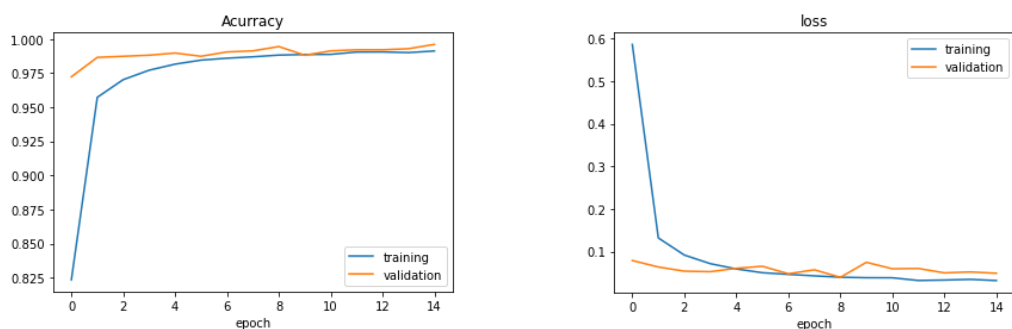


Figure 4: Accuracy (left) and loss (right) over the epoch of the selected model

Table 6: The macro and weighted precision, recall and F1 score

Method	Measure		
	Precision	Recall	F1 score
Macro average	99.53%	99.48%	99.50%
Weighted average	99.53%	99.52%	99.52%

As in Table 6, both the macro precision and weighted precision are the same, which is 99.53%. Traffic sign recognition is a multiclassification problem, hence F1 score should be focused on as a compromise between recall and precision. The weighted F1 score (99.52%) is slightly higher than the macro F1 score. This is due to the lower recall at those traffic sign classes with more testing images. So, the misclassification of traffic sign classes with more testing images accounts less when calculating the weighted average.

5.3 Model generalisation

To test the model generalisability, 95 Malaysia traffic sign images from ten classes are extracted from Islam and Raj (2017). The images are resized into 32 x 32 pixels, added with raindrops, pre-processed, and are fed into the selected model for prediction. Twenty-two images are being misclassified, resulting in an overall accuracy of 76.84% on this new set of testing data. Based on the results, there are four images with a pedestrian crossing road sign (class 30) being misclassified as school children crossing road sign (class 17). These two traffic sign classes also have some similarities, in which have a yellow background, black border and black pictogram at the centre. The pictogram for class 30 illustrates a walking pedestrian, whereas the pictogram for class 17 illustrates a walking parent with her children. Since the selected model performance on this new set of data is slightly dissatisfying, the need of training

the model with a larger dataset in the future is highlighted to improve the model's generalisability.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

Although previous research has shown the effectiveness of using the Convolutional Neural Network (CNN) in recognising traffic signs, these types of studies are lacking in Malaysia, especially the recognition of traffic signs in poor weather conditions. Since rainy weather is a common driving scenario, our study proposed a CNN as an image recognising system for Malaysia traffic signs with raindrop disturbance. The proposed algorithm is executed 18 times with different combinations of the epochs, dropout rates, and colour of the image input in the first convolutional layer (either colour or greyscale version). The best-selected model is trained with colour image inputs, 15 epochs and the dropout rate of 0.1, achieving an accuracy of 99.52% on the testing set. Hence, the objectives of this study have been successfully achieved.

6.2 Future research

To our best knowledge, this is the first study that develops a deep learning algorithm that can recognise Malaysia traffic signs with raindrop disturbance. However, the limitation that the CNN model in this study can only predict the class of the traffic signs should be noted. In future work, a detection algorithm can be added so that traffic signs with raindrop disturbance can be detected in the real-time environment before the recognition tasks. Besides, as discussed in

the results, more training images are needed to further improve the model accuracy on the testing set and the model generalisability because there are misclassifications due to the highly similar characteristics of the traffic signs. Also, if there are Malaysia traffic signs with real raindrops available in the future, the data should be used to train and test the model.

REFERENCES

- Aggarwal, C.C., 2018. *Neural Networks and Deep Learning: A Textbook*. Yorktown Heights, New York: Springer.
- Belghaouti, O., Handouzi, W., and Tabaa, M., 2020. *Improved Traffic Sign Recognition Using Deep ConvNet Architecture*. The 4th International Workshop on Connected and Intelligent Mobility (CIM 2020). Madeira, Portugal, 2-5 November 2020. Elsevier B.V.
- Blanchard, R., 2020. *Deep Learning for Computer Vision with SAS®: An Introduction*. Cary, North Carolina: SAS Institute Inc.
- Cao, J., Song, C., Peng, S., Xiao, F., and Song, S., 2019. Improved Traffic Sign Detection and Recognition Algorithm for Intelligent Vehicles. *Sensors (Basel)*, [e-journal] 19(18). <https://doi.org/10.3390/s19184021>
- Chollet, F., 2018. *Deep Learning with Python*. Shelter Island, New York: Manning Publications Co.
- Department of Statistics Malaysia, 2017. *Statistics on Causes of Death, Malaysia, 2017*. [press release] 31 October 2017. <https://www.dosm.gov.my/v1/index.php?r=column/pdfPrev&id=Y3psYUI2VjU0ZzRhZU1kcVFMMThGUT09>
- Department of Statistics Malaysia, 2018. *Statistics on Causes of Death, Malaysia, 2018*. [press release] 31 October 2018. <https://www.dosm.gov.my/v1/index.php?r=column/pdfPrev&id=aWg2VjJkZHHYcDdEM3JQSGloeTVIZz09>
- Department of Statistics Malaysia, 2019. *Statistics on Causes of Death, Malaysia, 2019*. [press release] 30 October 2019. <https://www.dosm.gov.my/v1/index.php?r=column/pdfPrev&id=RUxISDNkcNRVazJnakNCNVN2VGRdz09>

Department of Statistics Malaysia, 2020. *Statistics on Causes of Death, Malaysia, 2020*. [press release] 26 November 2020. <https://www.dosm.gov.my/v1/index.php?r=column/pdfPrev&id=QTU5T0dKQ1g4MHYxd3ZpMzhEMzdRdz09>

Department of Statistics Malaysia, 2021. *Statistics on Causes of Death, Malaysia, 2021*. [online] 16 November 2021. https://www.dosm.gov.my/v1/index.php?r=column/cthemByCat&cat=401&bul_id=R3VrRUhwSXZDN2k4SGN6akRhTStwQT09&menu_id=L0pheU43NWJwRWVSZklWdzQ4TlhUUT09

Fernando, K.R.M. and Tsokos, C.P., 2021. Dynamically Weighted Balanced Loss: Class Imbalanced Learning and Confidence Calibration of Deep Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, [e-journal]. <https://doi.org/10.1109/TNNLS.2020.3047335>

GTSRB – German Traffic Sign Recognition Benchmark, 2019. [Data set]. Kaggle. <https://www.kaggle.com/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign>

Indolia S., Goswami, A.K., Mishra, S.P. and Asopa, P., 2018. Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach. *Procedia Computer Science*, [e-journal] 132, pp. 679-688. <https://doi.org/10.1016/j.procs.2018.05.069>

Islam, K.T. and Raj, R.G., 2017. Real-Time (Vision-Based) Road Sign Recognition Using an Artificial Neural Network. *Sensors (Basel)*, [e-journal] 17(4), 853. <https://dx.doi.org/10.3390%2Fs17040853>

Janiesch, C., Zschech, P. and Heinrich, K. 2021. Machine learning and deep learning. *Electron Markets*, [e-journal] 31, p.p. 685–695. <https://doi.org/10.1007/s12525-021-00475-2>

- Kotu, V. and Deshpande, B., 2019. *Data Science: Concepts and Practice*. 2nd ed. Cambridge: Morgan Kaufmann Publishers.
- Lau, M.M., Lim, K.H. and Gopalai, A.A., 2015. *Malaysia traffic sign recognition with convolutional neural network*. 2015 IEEE International Conference on Digital Signal Processing (DSP). Singapore, 21-24 July 2015. IEEE.
- Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, [e-journal] 86(11), p.p. 2278-2324. <https://doi.org/10.1109/5.726791>
- Madani, A. and Yusof, R., 2016. Malaysian traffic sign dataset for traffic sign detection and recognition systems. *Journal of Telecommunication, Electronic and Computer Engineering*, [e-journal] 8(11), pp. 137-143. <https://jtec.utem.edu.my/jtec/article/view/1423>
- Maggiori, E., Tarabalka, Y., Charpiat, G. and Alliez, P., 2017. Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, [e-journal] 55(2), pp. 645-657. <https://ieeexplore.ieee.org/document/7592858/>
- Mostafa, B.M., El-Attar, N.E., Abd-Elhafeez, S.A. and Awad, W.A., 2020. Machine and Deep Learning Approaches in Genome: Review Article. *Alfarama Journal of Basic & Applied Sciences*, [e-journal] 2(1), pp. 105-113. <https://dx.doi.org/10.21608/ajbas.2020.34160.1023>
- Neoh, D.T.H., Sahari, K.S.M., Yew, C.H. and Basubeit, O.G. S., 2019. *Recognizing Malaysia Traffic Signs with Pre-Trained Deep Convolutional Neural Networks*. 2019 4th International Conference on Control, Robotics and Cybernetics (CRC). Tokyo, Japan, 27-30 September 2019. IEEE. <https://doi.org/10.1109/CRC.2019.00030>
- Quan, X.B. and Xiong, W.X., 2019. Real-Time Embedded Traffic Sign Recognition Using Efficient Convolutional Neural Network. *IEEE Access*, [e-journal] 7, pp. 53330-53346.

<https://doi.org/10.1109/ACCESS.2019.2912311>>

Rosebrock, A. 2019. *Keras ImageDataGenerator and Data Augmentation*. [online] PyImageSearch. Available at: <<https://www.pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/>> [Accessed 20 January 2022].

Sean Reilly, Ian Clancy, Stephen Foy and Michael Madden, 2018. Extended Malaysian Traffic Sign Dataset (EMTD) (Version 1) [Data set]. Zenodo. <https://doi.org/10.5281/zenodo.1217105>

Velamati, A. and Gopichand, G, 2021. Traffic Sign Classification Using Convolutional Neural Networks and Computer Vision. *Turkish Journal of Computer and Mathematics Education*, [e-journal] 12(3), pp. 4244-4250. <https://doi.org/10.17762/turcomat.v12i3.1715>

Wang, Y., Li, Y., Song, Y. and Rong, X., 2020. The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition. *Applied Sciences*, [e-journal] 10(5). <https://doi.org/10.3390/app10051897>


APPENDIX A

Script output for training the selected model (Combination 2)

```

runfile('D:/Documents/Y3S1/FYP/2-Traffic-Sign-Recognition/TSR_Train_colour.py',
wdir='D:/Documents/Y3S1/FYP/2-Traffic-Sign-Recognition')
Using TensorFlow backend.
Total classes detected in training set: 50
Importing classes from training set.....
0 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 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
44 45 46 47 48 49
Total classes detected in validation set: 50
Importing classes from validation set.....
0 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 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
44 45 46 47 48 49


```



```

Data Shapes
Train(3700, 32, 32, 3) (3700,)
Validation(1261, 32, 32, 3) (1261,)

```



```

2022-02-17 17:14:28.629531:
Model: "sequential_1"

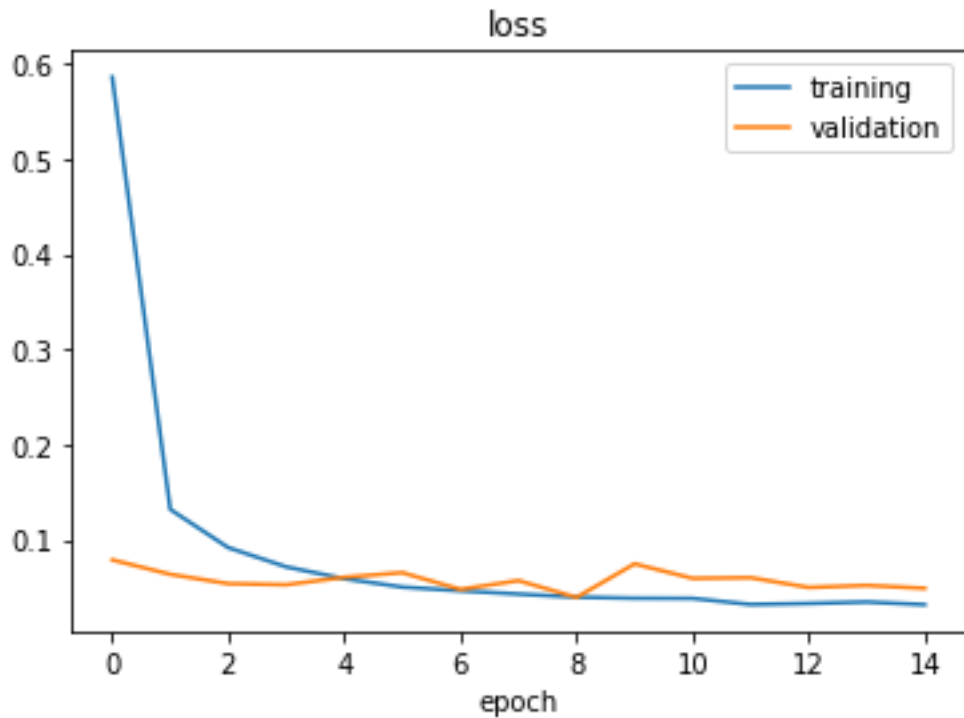
```

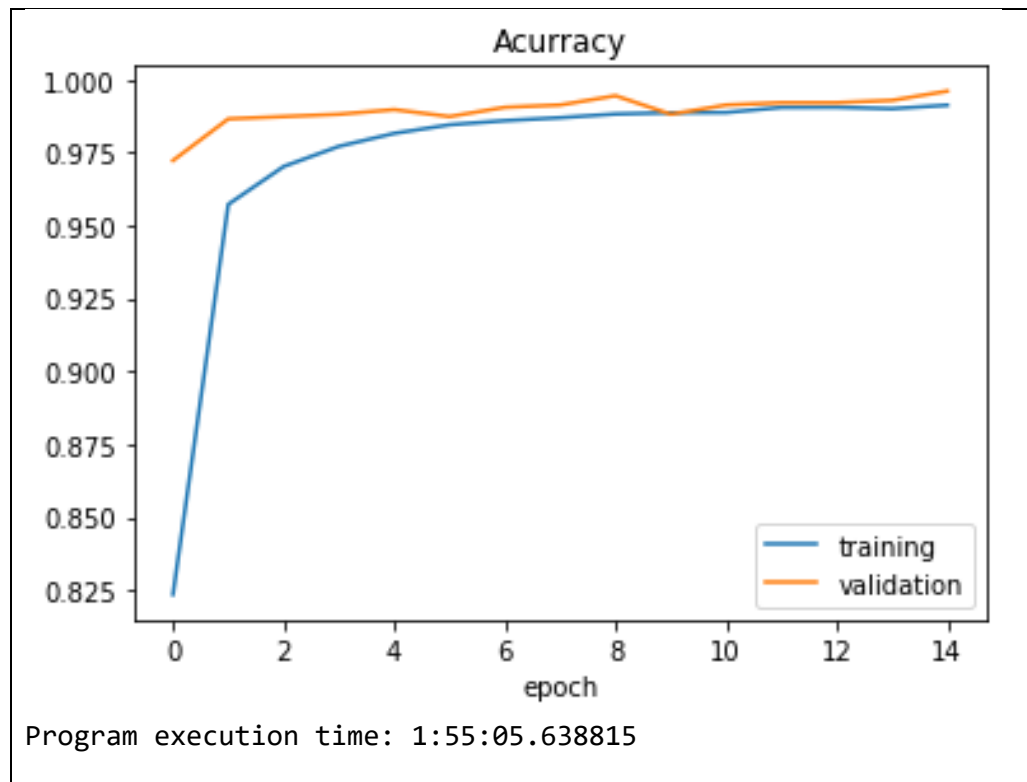
Layer (type) #	Output Shape	Param #
=====		
conv2d_1 (Conv2D)	(None, 28, 28, 60)	4560
conv2d_2 (Conv2D)	(None, 24, 24, 60)	90060
max_pooling2d_1 (MaxPooling2)	(None, 12, 12, 60)	0
conv2d_3 (Conv2D)	(None, 10, 10, 30)	16230
conv2d_4 (Conv2D)	(None, 8, 8, 30)	8130
max_pooling2d_2 (MaxPooling2)	(None, 4, 4, 30)	0
dropout_1 (Dropout)	(None, 4, 4, 30)	0

flatten_1 (Flatten)	(None, 480)	0
dense_1 (Dense)	(None, 500)	240500
dropout_2 (Dropout)	(None, 500)	0
dense_2 (Dense)	(None, 50)	25050
=====		
====		
Total params: 384,530		
Trainable params: 384,530		
Non-trainable params: 0		

None		
Epoch 1/15		
2000/2000 [=====] - 473s 237ms/step		
- loss: 0.5864 - accuracy: 0.8235 - val_loss: 0.0787 -		
val_accuracy: 0.9722		
Epoch 2/15		
2000/2000 [=====] - 470s 235ms/step		
- loss: 0.1317 - accuracy: 0.9572 - val_loss: 0.0635 -		
val_accuracy: 0.9865		
Epoch 3/15		
2000/2000 [=====] - 458s 229ms/step		
- loss: 0.0915 - accuracy: 0.9702 - val_loss: 0.0536 -		
val_accuracy: 0.9873		
Epoch 4/15		
2000/2000 [=====] - 456s 228ms/step		
- loss: 0.0713 - accuracy: 0.9771 - val_loss: 0.0525 -		
val_accuracy: 0.9881		
Epoch 5/15		
2000/2000 [=====] - 446s 223ms/step		
- loss: 0.0589 - accuracy: 0.9815 - val_loss: 0.0603 -		
val_accuracy: 0.9897		
Epoch 6/15		
2000/2000 [=====] - 447s 223ms/step		
- loss: 0.0500 - accuracy: 0.9845 - val_loss: 0.0652 -		
val_accuracy: 0.9873		
Epoch 7/15		
2000/2000 [=====] - 447s 223ms/step		
- loss: 0.0461 - accuracy: 0.9859 - val_loss: 0.0476 -		
val_accuracy: 0.9905		
Epoch 8/15		
2000/2000 [=====] - 449s 224ms/step		
- loss: 0.0427 - accuracy: 0.9869 - val_loss: 0.0569 -		
val_accuracy: 0.9913		
Epoch 9/15		

```
2000/2000 [=====] - 456s 228ms/step
- loss: 0.0394 - accuracy: 0.9882 - val_loss: 0.0391 -
val_accuracy: 0.9944
Epoch 10/15
2000/2000 [=====] - 484s 242ms/step
- loss: 0.0383 - accuracy: 0.9886 - val_loss: 0.0745 -
val_accuracy: 0.9881
Epoch 11/15
2000/2000 [=====] - 488s 244ms/step
- loss: 0.0381 - accuracy: 0.9887 - val_loss: 0.0592 -
val_accuracy: 0.9913
Epoch 12/15
2000/2000 [=====] - 454s 227ms/step
- loss: 0.0317 - accuracy: 0.9904 - val_loss: 0.0599 -
val_accuracy: 0.9921
Epoch 13/15
2000/2000 [=====] - 459s 230ms/step
- loss: 0.0328 - accuracy: 0.9905 - val_loss: 0.0497 -
val_accuracy: 0.9921
Epoch 14/15
2000/2000 [=====] - 448s 224ms/step
- loss: 0.0344 - accuracy: 0.9900 - val_loss: 0.0518 -
val_accuracy: 0.9929
Epoch 15/15
2000/2000 [=====] - 445s 222ms/step
- loss: 0.0315 - accuracy: 0.9912 - val_loss: 0.0487 -
val_accuracy: 0.9960
Training time: 1:54:41.183800
```

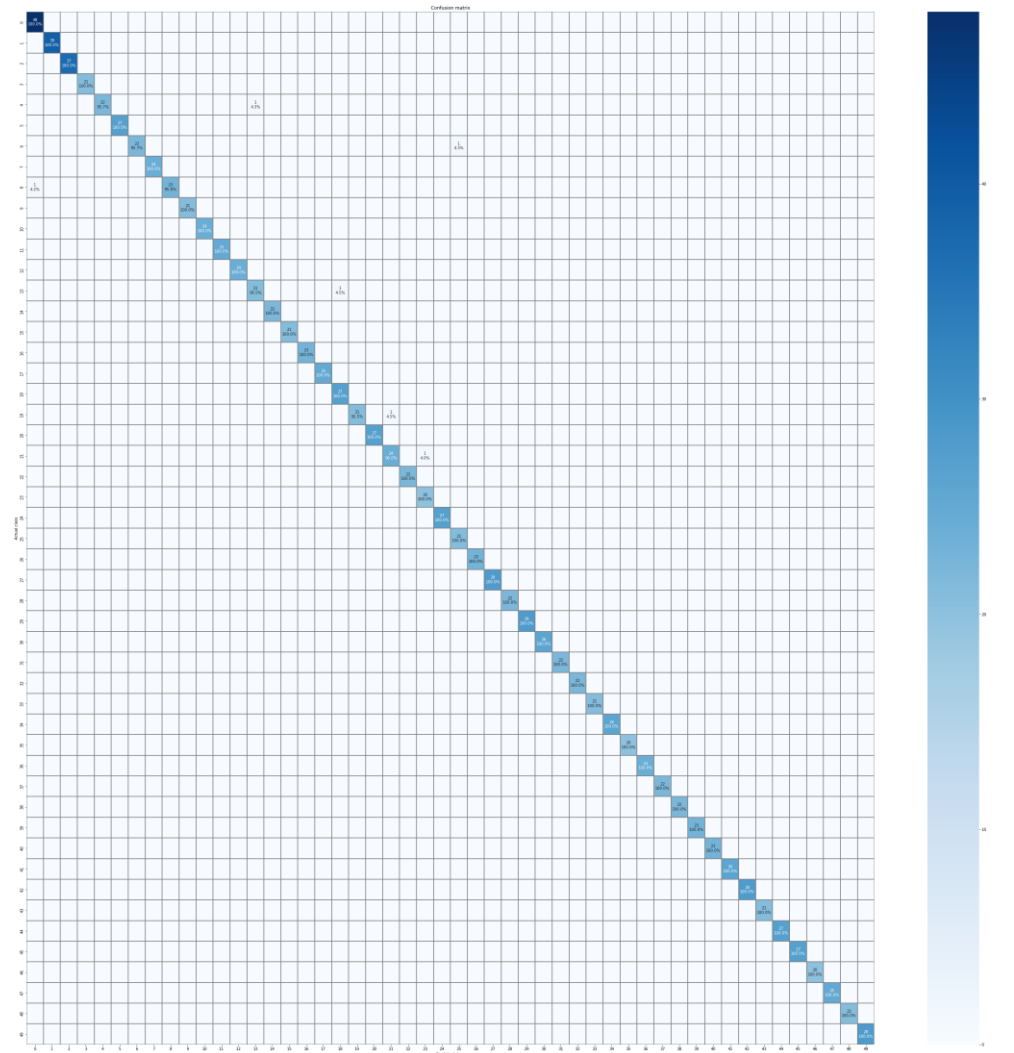




APPENDIX B

Script output for testing the selected model (Combination 2)

```
runfile('D:/Documents/Y3S1/FYP/2-Traffic-Sign-Recognition/TSR_Test_colour.py',  
wdir='D:/Documents/Y3S1/FYP/2-Traffic-Sign-Recognition')  
Total classes detected in testing set: 50  
Importing classes from testing set.....  
0 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 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43  
44 45 46 47 48 49  
Data Shapes  
Test(1239, 32, 32, 3) (1239,)  
Test Score: 0.04337273397751107  
Test Accuracy: 0.99435025  
Prediction time: 0:00:00.833247  
(1239,) (1239,)  
Found 1233 correct labels  
Found 6 incorrect labels  
Calculated test accuracy: 0.9951573849878934
```



	precision	recall	f1-score	support
Class 0	0.98	1.00	0.99	48
Class 1	1.00	1.00	1.00	39
Class 2	1.00	1.00	1.00	37
Class 3	1.00	1.00	1.00	21
Class 4	1.00	0.96	0.98	23
Class 5	1.00	1.00	1.00	27
Class 6	1.00	0.96	0.98	23
Class 7	1.00	1.00	1.00	24
Class 8	1.00	0.96	0.98	24
Class 9	1.00	1.00	1.00	21
Class 10	1.00	1.00	1.00	24
Class 11	1.00	1.00	1.00	25
Class 12	1.00	1.00	1.00	24
Class 13	0.95	0.95	0.95	22
Class 14	1.00	1.00	1.00	22
Class 15	1.00	1.00	1.00	21
Class 16	1.00	1.00	1.00	23
Class 17	1.00	1.00	1.00	25
Class 18	0.96	1.00	0.98	27
Class 19	1.00	0.95	0.98	22
Class 20	1.00	1.00	1.00	27
Class 21	0.96	0.96	0.96	25
Class 22	1.00	1.00	1.00	22
Class 23	0.95	1.00	0.98	20
Class 24	1.00	1.00	1.00	27
Class 25	0.95	1.00	0.98	21
Class 26	1.00	1.00	1.00	23
Class 27	1.00	1.00	1.00	28
Class 28	1.00	1.00	1.00	22
Class 29	1.00	1.00	1.00	28
Class 30	1.00	1.00	1.00	26
Class 31	1.00	1.00	1.00	22
Class 32	1.00	1.00	1.00	22
Class 33	1.00	1.00	1.00	21
Class 34	1.00	1.00	1.00	26
Class 35	1.00	1.00	1.00	20
Class 36	1.00	1.00	1.00	24
Class 37	1.00	1.00	1.00	22
Class 38	1.00	1.00	1.00	22
Class 39	1.00	1.00	1.00	23
Class 40	1.00	1.00	1.00	23
Class 41	1.00	1.00	1.00	26
Class 42	1.00	1.00	1.00	28
Class 43	1.00	1.00	1.00	21
Class 44	1.00	1.00	1.00	27
Class 45	1.00	1.00	1.00	27
Class 46	1.00	1.00	1.00	20
Class 47	1.00	1.00	1.00	25
Class 48	1.00	1.00	1.00	21
Class 49	1.00	1.00	1.00	28

accuracy			1.00	1239
macro avg	1.00	0.99	0.99	1239
weighted avg	1.00	1.00	1.00	1239
Program execution time: 0:00:04.050475				

APPENDIX C

Script output for testing the generalisability of the selected model (Combination 2)

```
Total classes detected in testing set: 50
Importing classes from testing set.....
0 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 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
44 45 46 47 48 49
Data Shapes
Test(95, 32, 32, 3) (95,)
Test Score: 1.5203397575177644
Test Accuracy: 0.76842105
Prediction time: 0:00:00.130595
(95,) (95,)
Found 73 correct labels
Found 22 incorrect labels
Calculated test accuracy: 0.7684210526315789
Program execution time: 0:00:00.904521

Predicted classes:
array([
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 4, 4, 4, 4, 4, 20, 4, 12, 0, 4, 6, 5,
5, 5, 5, 13, 49, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 7, 7, 7, 17, 7, 7, 26, 7, 0, 7, 6, 6, 19, 19,
19, 19, 6, 4, 4, 19, 20, 20, 20, 20, 20, 20, 20, 20, 20,
22, 15, 22, 22, 22, 16, 17, 17, 30, 21, 14, 30, 17, 30, 17],
dtype=int64)

Actual classes:
array([
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 19, 19, 19, 19,
19, 19, 19, 19, 19, 19, 20, 20, 20, 20, 20, 20, 20, 20,
20, 22, 22, 22, 22, 22, 22, 30, 30, 30, 30, 30, 30, 30, 30,
30])
```

APPENDIX D

E-Certificate of Presentation in SIRKM 2022

