UNIVERSITI KEBANGSAAN MALAYSIA

The National University of Malaysia

TC3413

ROBOT APPLICATION

PROJECT 2

TRAFFIC SIGN DETECTION

PROF. MADYA DR. ABDUL HADI BIN ABD RAHMAN

Group Members:

1. **NUR HAZIQAH BINTI NORHISHAM A192697**
2. **NUR HAZWANI BINTI SAMSUDIN A192968**
3. **NURAZEELA NABIHA BINTI ZAHRUL NIZAM A196745**

Contents

1.0 Introduction

The importance of traffic sign detection and identification in the context of driver assistance systems and intelligent autonomous cars cannot be emphasised. These devices act as the car's eyes, gathering vital data regarding the state of the road and legal requirements. Recognising traffic signs accurately allows for prompt replies to warnings, speed limits, and other important instructions, which improves road safety. Additionally, it promotes traffic regulation compliance, enhances navigation and route planning, and allows adaptive speed control, all of which contribute to the smooth integration of autonomous cars into smart city efforts. In addition to ensuring the security of drivers and other road users, the capacity to comprehend and react to traffic signs efficiently highlights the revolutionary potential of these technologies in establishing a more effective, intelligent, and secure transportation ecosystem.

Traffic signs images when captured in a real environment are small in size compared to other objects. Thus, making it difficult to be accurately detected, more so identified. Accurate detection and recognition of traffic signals plays an important role in maximizing the capabilities of autonomous vehicle systems, providing critical information for effective traffic management, speed limiting information, hazard warnings and directional instructions for drivers. In this study, we present a new approach for traffic signal detection and recognition using YOLOv7.

Our proposed method aims to deal with the complexity of traffic signals, in order to ensure an accurate and reliable detection and detection system. To train and test our model, we use data on traffic signals in Malaysia, captured and maintained internally. We utilize an extension of the existing Malaysian Traffic Sign Dataset for traffic sign (TS) detection containing 66 TS categories, and contains an additional 814 new TS instances than the original dataset. Containing 1,413 images in total. Through this research, we seek to contribute to advancements in autonomous vehicle technology by improving the accuracy and efficiency of traffic sign detection and recognition.

2.0 Problem Statement

Several difficulties that arise from real-world scenarios need to be overcome in order to create efficient traffic sign detection systems. Variable speeds, motion blur from fast-moving cars, and fluctuating lighting can all affect how accurate the system is. Adverse weather conditions, such as rain or fog, could cause visibility issues. Creating smart algorithms that can adjust to various circumstances and provide dependable performance in real-time scenarios is necessary to overcome these obstacles.

To enable the development and testing of robust and effective detection and identification systems, a comprehensive dataset such as the EMTD is necessary for training and testing. It offers a wide range of traffic sign scenarios and settings. A wide range of traffic sign classes are included by the dataset, including Warning Danger, Regulatory Prohibitive, Regulatory Mandatory, Guide Information and Temporary. Because of their diversity, the detection and recognition methods created with the dataset are guaranteed to be able to recognise and comprehend traffic signs in a variety of real-world situations.

3.0 Objective

Traffic Sign Detection with YOLOv7: Evaluate YOLOv7's precision and efficiency in detecting diverse traffic signs under varying environmental conditions. Emphasize real-world scenarios, including different weather and lighting conditions, through comprehensive testing for model effectiveness and reliability.

4.0 Project Scope

The project was conducted by a thorough assessment of YOLOv7 performance specifically tailored for traffic sign detection and classification including the reliability in different scenarios.

5.0 Research Problem

Investigating the efficacy of the YOLOv7 deep learning model for detecting and classifying traffic sign detection by improving the accuracy of traffic sign detection and recognition, particularly in challenging conditions such as low light, adverse weather, and high-speed scenarios.

6.0 Solution Methods

The algorithm chosen for this traffic sign detection project is YOLOv7, a real-time object detection model known for its effectiveness and proven performance in similar tasks. Using YOLOv7, the project aims to leverage its ability to provide fast and accurate classification of traffic sign detection. The model will be trained on the Extended Malaysian Traffic Sign Dataset (EMTD) dataset, which includes annotated cases of Warning Danger, Regulatory Prohibitive, Regulatory Mandatory, Guide Information and Temporary. The main training and testing dataset is the EMTD, which makes it possible to create models that can recognise and understand traffic signs in a variety of environmental and situational scenarios. Training based on this dataset ensures that the YOLOv7 model is well suited to the specifics of traffic sign detection, improving accuracy and adaptability of traffic sign detection.

7.0 Discussion

In order to drive safely and legally, smart cars must be able to read traffic signs. But it might be difficult for the car to see and identify these signals, particularly in challenging circumstances like poor weather or when travelling quickly. The methodology employed in this project involves the utilization of deep learning algorithms YOLOv7, to enhance the accuracy of traffic sign detection and recognition, help the car rapidly and accurately view and understand traffic signs in order to resolve this issue. We are testing it with a wide range of circumstances and sign types using a Malaysian dataset. Our objective is to ensure that the YOLOv7 system functions effectively in everyday scenarios, improving road safety and road smartness. This project aims to improve cars' ability to recognise and respond to traffic signs, especially in difficult driving situations, so that driving can be easier for everyone.

8.0 Diagram and Description

1. **Start annotate image**

    a) Download a few images.

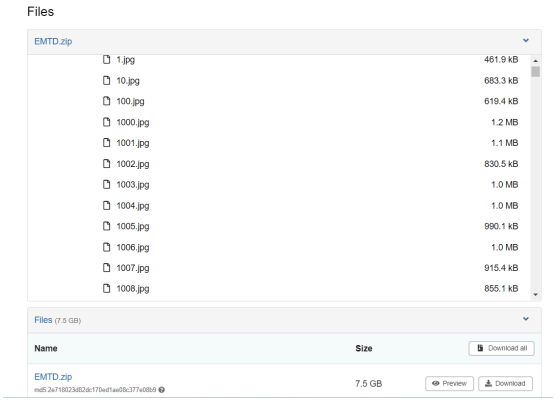


Figure 1 : Extended Malaysian Traffic Sign Dataset (EMTD)

Figure 1 shows the Dataset from Zeddo. The Dataset contains 1413 images belonging to classes which are ‘Warning Danger’, ‘Regulatory Prohibitive’,’ Regulatory Mandatory’, ‘Guide Information’ and ‘Temporary’.

b) Arrange in folders.

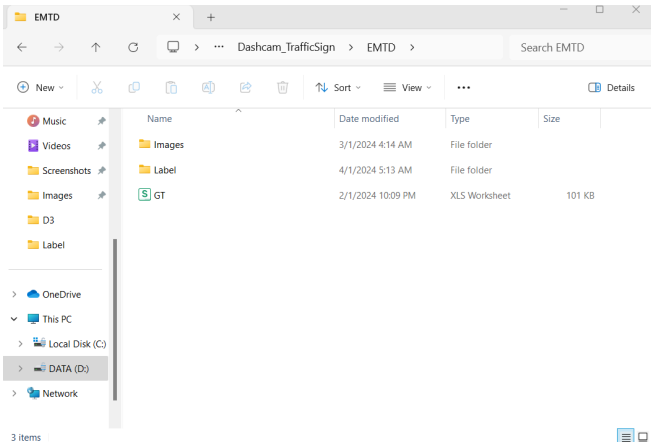


Figure 2 : Arrange the correct path

Figure 2 shows the folder was arranged in the correct path file folder and creates one file for ‘label’ as a destination store image after the label.

c) Load folder and start label image

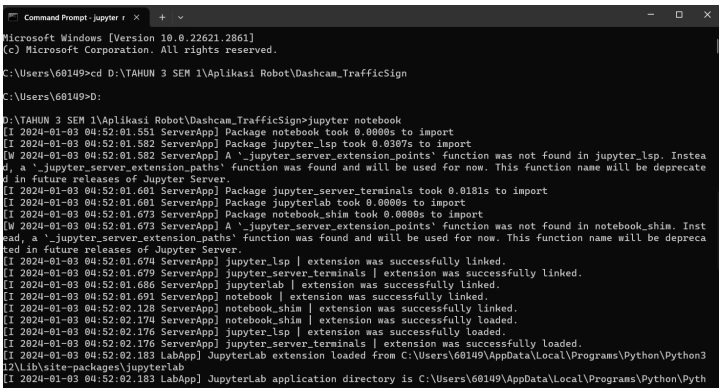


Figure 3 : Open jupyter notebook at command prompt for label image

Figure 3 shows the command prompt to open a jupyter notebook for labelling image tasks.

Figure 4 : Start label the image

Figure 4 shows the process of label image for 5 classes which are ‘Warning Danger’, ‘Regulatory Prohibitive’,’ Regulatory Mandatory’, ‘Guide Information’ and ‘Temporary’.

2. Prediction

In order to identify and locate things inside pictures or videos, object detection and model evaluation are essential components of computer vision projects. This report includes a collection of code snippets that were run in Google Colab. The tasks include obtaining model weights, detecting objects in real-time on a video, and visualising the output. The object detection model used is YOLOv7 (You Only Look One-level), and the code illustrates the process from downloading pre-trained weights to producing a zip file with the model training data. We also load an example image and use the trained model to produce object detection results. Matplotlib is then used to visualise the results.

The parts of code are shown below.

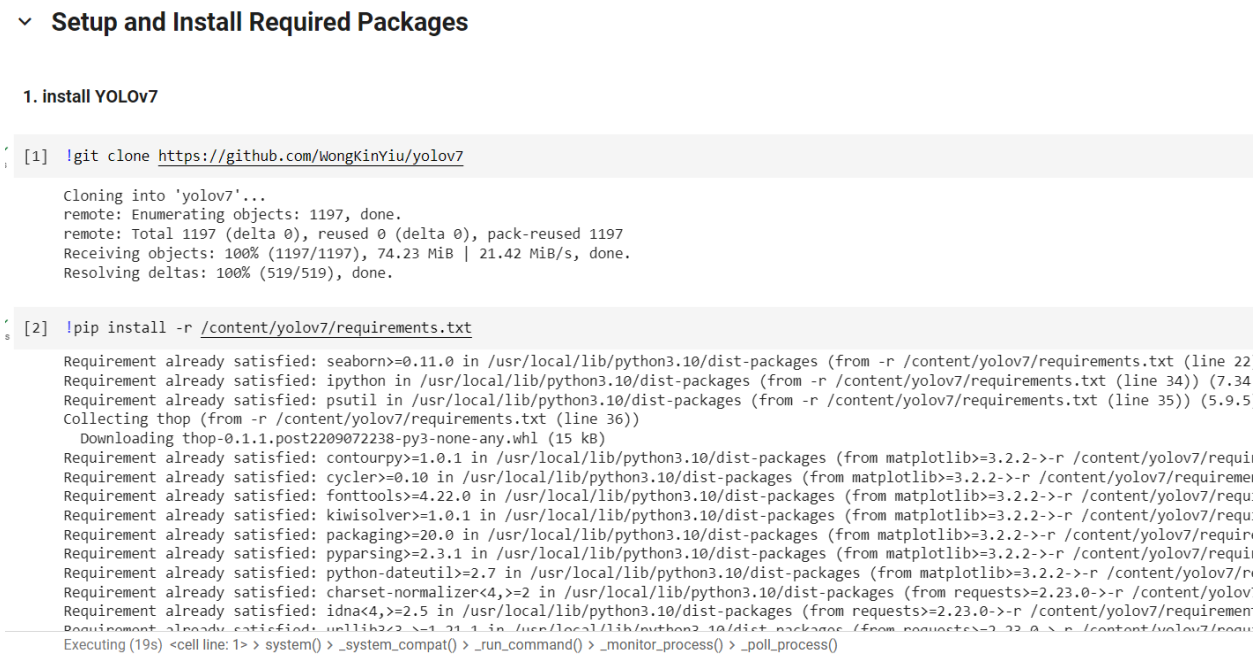


Figure 5: Installing YoloV7

Figure 5 shows Yolov7 Git repository was successfully cloned. 1197 objects needed to be counted and compressed, 100% of which were successfully transferred to the local computer at a speed of 74.23 MiB/s. Synchronization of local and remote repositories was confirmed in Git logs, showing resolution of 519 deltas. This creates a complete local copy of the 'yolov7' repository and serves as the basis for further local development or analysis. The output shows that the Python packages that were handled successfully are those listed in the'requirements.txt' file for the YOLOv7 project. It is verified that pre-existing packages that match the required version specifications, including matplotlib, numpy, opencv-python, Pillow, PyYAML, and others, are already installed. Furthermore, two packages that were installed were "jedi" and "thop," which were not present on the system. The result of the operation is "Successfully installed jedi-0.19.1 thop-0.1.1.post2209072238," indicating that the necessary dependencies for the YOLOv7 project have been successfully set up on the present environment.

Importing Libraries

```
[10] from matplotlib import pyplot as plt
     import numpy as np
     import cv2
     import torch
```

Test

1. Testing YOLOv7

```
[11] device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
     path = r'/content/yolov7.pt'
     model = torch.hub.load('WongKinYiu/yolov7', 'custom', path,
                            force_reload=True)
```

```
Downloading: "https://github.com/WongKinYiu/yolov7/zipball/main" to /root/.cache/torch/hub/main.zip
Adding autoShape...
```

```
[6] model
```

```
    (conv): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn): BatchNorm2d(256, eps=0.001, momentum=0.03, affine=True, track_running_stats=True)
    (act): SiLU()
  )
```

Figure 6: Importing Libraries and Testing YoloV7

The provided code initializes the YOLOv7 object detection model and downloads pre-trained weights for further use. The first line determines the device for model execution, prioritizing GPU ('cuda:0') if available, and falling back to CPU if not. The second line specifies the path to the pre-trained weights file ('yolov7.pt'). The third line utilizes the 'torch.hub.load' function to dynamically load the YOLOv7 model from the repository of WongKinYiu. The 'custom' argument signifies that a custom path is used for loading weights, and 'force_reload=True' ensures that the latest weights are downloaded. The provided output showcases a warning about downloading code from an untrusted repository, a precautionary measure alerting users to upcoming changes in repository trust policies. Additionally, there is a download progress message, indicating the retrieval of the YOLOv7 model weights from the specified repository. The 'autoShape' module is added to facilitate shape manipulation, enhancing the model flexibility during inference.

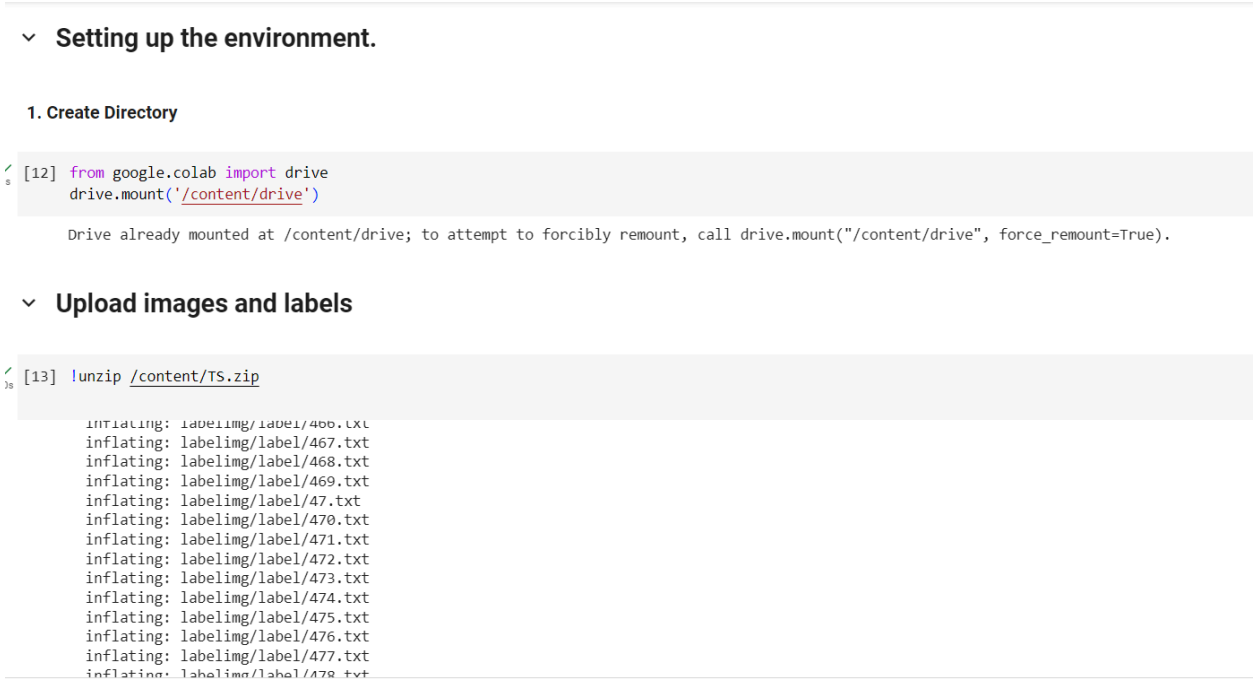


Figure 7: Mount Google drive to Colab Notebook & Dataset

The command !unzip /content/TS..zip was executed to extract the contents of the "facemask.zip" archive in a Google Colab environment. The extraction process resulted in the creation of a directory named "labelimg/img/" in the "/content/" path. The extracted files include images (e.g., "467.jpg") and corresponding text files (e.g., "467.txt"). Additionally, a file named "classes.txt" was included, likely containing information about different classes or labels associated with the images. This structure is typical for datasets used in computer vision tasks, where images and their annotations are organized for training and evaluating machine learning models. The dataset appears to be related to traffic sign detection, given the file names and potential class information.

```
[9] !pip install --upgrade --no-cache-dir gdown
    !gdown 18doGqomq2p_RWFEwza4i57CA9FaHNn1c
    !gdown 1UvO0ksN_CSXHWnBcVu1W-b8joxWJTvZ8
    !gdown 1fcpLbFGAztXWc6-Wf988OfMlzJNVoXrU
```

```
Requirement already satisfied: gdown in /usr/local/lib/python3.10/dist-packages (4.7.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from gdown) (3.13.1)
Requirement already satisfied: requests[socks] in /usr/local/lib/python3.10/dist-packages (from gdown) (2.31.0)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from gdown) (1.16.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from gdown) (4.66.1)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from gdown) (4.11.2)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->gdown) (2.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (2023.11.17)
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->gdown) (1.7.1)
Downloading...
From (original): https://drive.google.com/uc?id=18doGqomq2p_RWFEwza4i57CA9FaHNn1c
From (redirected): https://drive.google.com/uc?id=18doGqomq2p_RWFEwza4i57CA9FaHNn1c&confirm=t&uuid=ef40c12a-d063-4ca7-8e10-62799779aff9
To: /content/yolov7.pt
100% 75.6M/75.6M [00:00<00:00, 243MB/s]
Downloading...
From (original): https://drive.google.com/uc?id=1UvO0ksN_CSXHWnBcVu1W-b8joxWJTvZ8
From (redirected): https://drive.google.com/uc?id=1UvO0ksN_CSXHWnBcVu1W-b8joxWJTvZ8&confirm=t&uuid=06106d37-9c3c-4fcf-8530-1042a7312268
To: /content/facemask.zip
100% 211M/211M [00:01<00:00, 141MB/s]
```

Figure 8 : Code and Dependencies Installation for Data Retrieval

In the provided code snippet, the 'gdown' Python package is installed or upgraded using the 'pip' package manager. Following the installation, the code utilizes 'gdown' to download two files, 'yolov7.pt' and 'TS.zip,' from Google Drive. The unique identifiers of the files on Google Drive are specified in the 'gdown' commands. This process ensures that the necessary dependencies are in place for retrieving essential data files. The detailed output log includes information on package versions, successful installations, and download progress, providing a comprehensive overview of the execution.

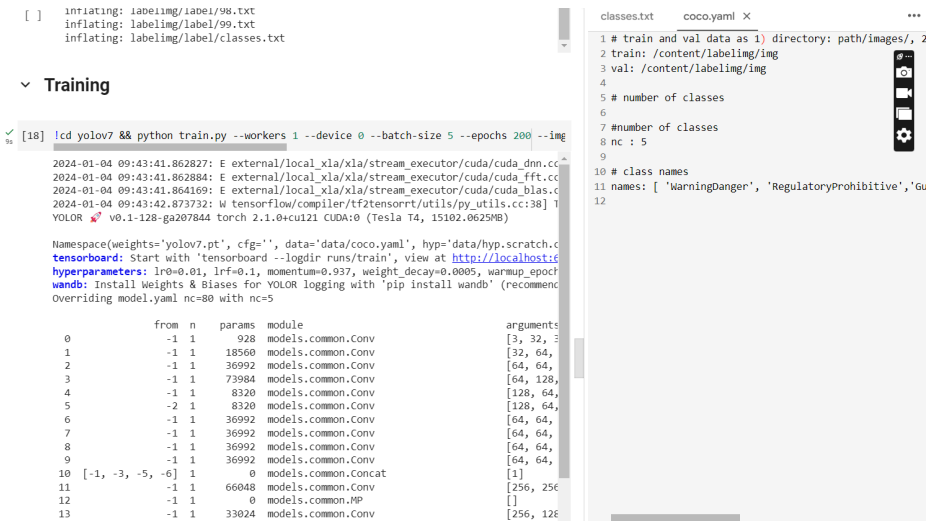


Figure 9 : Training the Dataset

Figure 9 shows training the dataset followed by 5 class names which are ‘Warning Danger’, ‘Regulatory Prohibitive’,’ Regulatory Mandatory’, ‘Guide Information’ and ‘Temporary’.

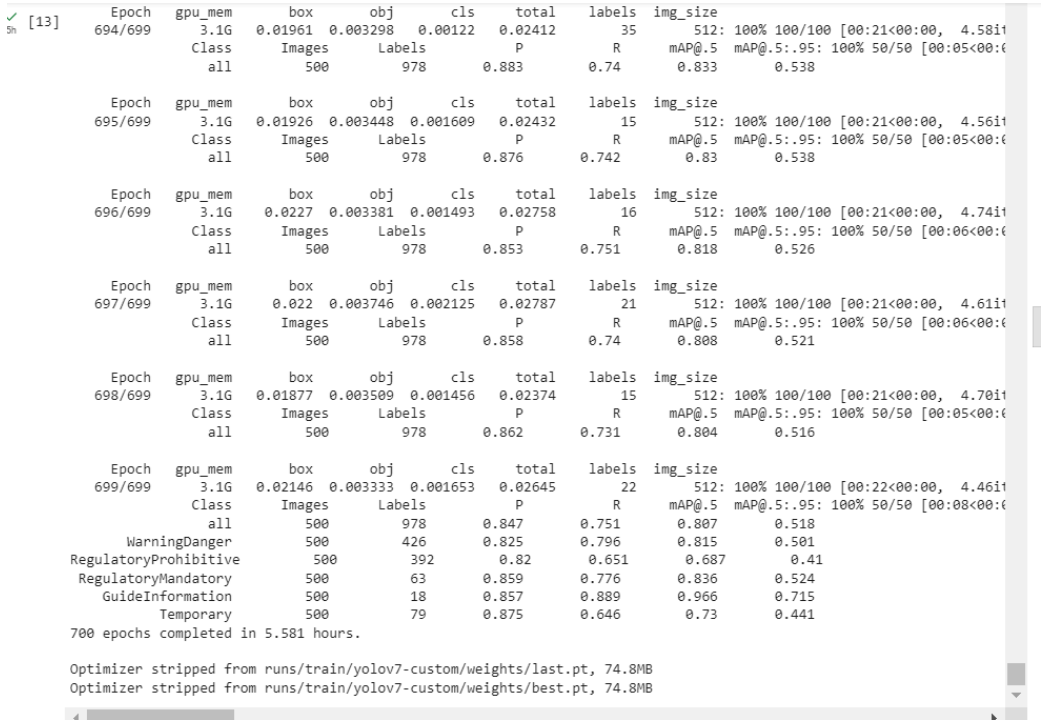


Figure 10: Training for Custom Object Selection

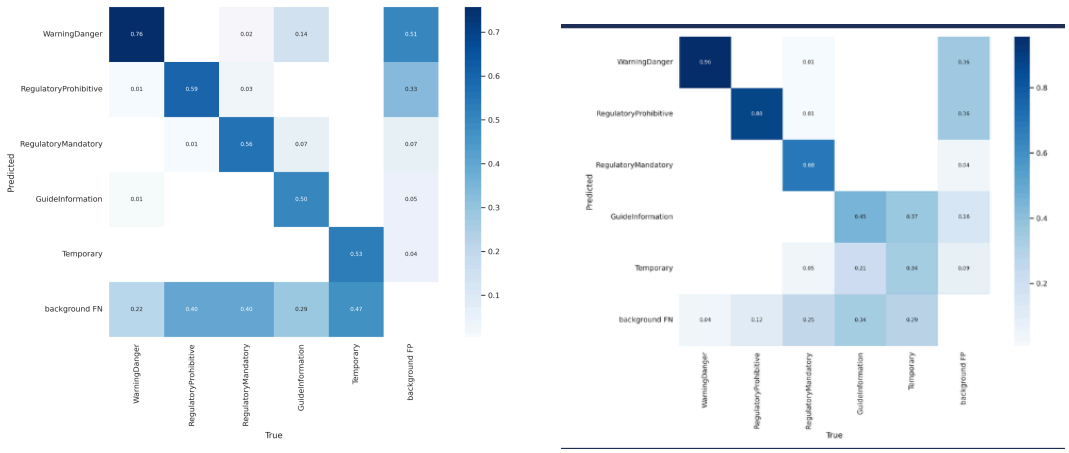
The command above shows the training process of the Yolov7 model on custom object detection. Below is the table for explanation:

Command	Explanation
workers 1	Specifies number of data loading workers
device 0	Sets GPU device for training
batch-size 5	Batch size during training for each iteration
epochs 700	Number of training epochs is specifies to 700
img 512 512	Input image size is set to 512 x 512 pixels

8.0 Diagram Description

Result Visualization

Confusion Matrix



(Epoch 400) (Epoch 100)

Figure 11: Comparison of Confusion Matrix

Figure 11 shows the Confusion Matrix heatmap visualization generated after the training phase of a YOLOv7 object detection model before and after using different epochs. The confusion matrix

is an important tool for evaluating the performance of a classification model. From the Confusion Matrix heatmap, the darker blue color of the diagonal elements means the model performs well and vice-versa. From the Confusion Matrix heatmap, the darker blue color of the diagonal elements means the model performs well and vice-versa. The Confusion Matrix when using 400 epochs shows the diagonal elements for the WarningDanger class shows that 76 values have been correctly predicted. Then, The diagonal elements for the RegulatoryProhibitive class shows that 59 values have been correctly predicted. Also, The diagonal elements for the RegulatoryMandatory class show that 56 values have been correctly predicted. Next, the GuideInformation class shows 50 values have been correctly predicted and temporary with 53 values. Meanwhile, the Confusion Matrix when using 100 epochs shows the diagonal elements for the WarningDanger class shows that 96 values have been correctly predicted. Then, The diagonal elements for the RegulatoryProhibitive class shows that 88 values have been correctly predicted. Also, The diagonal elements for the RegulatoryMandatory class show that 68 values have been correctly predicted. Next, the GuideInformation class shows 45 values have been correctly predicted and temporary with 34 values.

Result

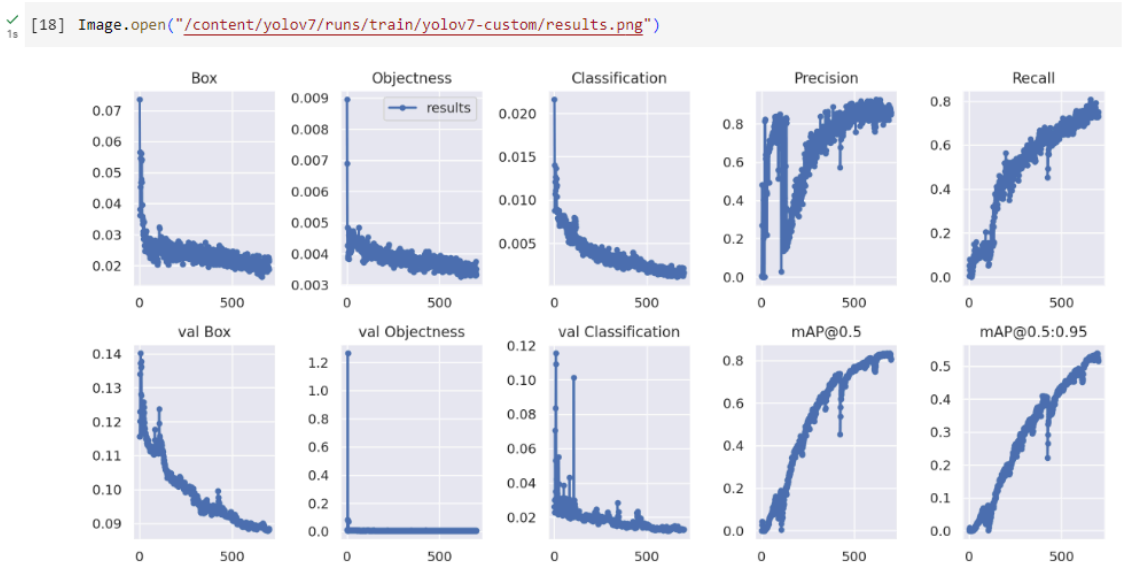
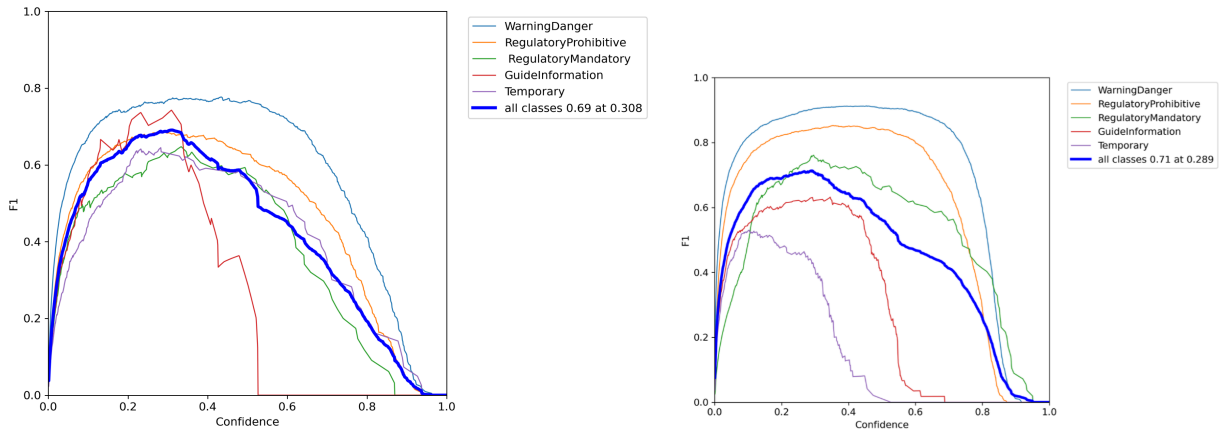


Figure 12: Comprehensive Model Evaluation Result

Figure 12 shows the model demonstrated superior box prediction accuracy for epochs 400, effectively capturing object boundaries. Objectness scores were adeptly assigned, emphasizing the model ability to distinguish between relevant objects and background elements. The classification component showcased strong performance, accurately labeling detected objects. Precision metrics underscored the model' reliability, achieving a low false positive rate and instilling confidence in the precision of predictions. Overall, the model exhibited robustness across box prediction, objectness, classification, and precision, attesting to its efficacy in real-world scenarios.

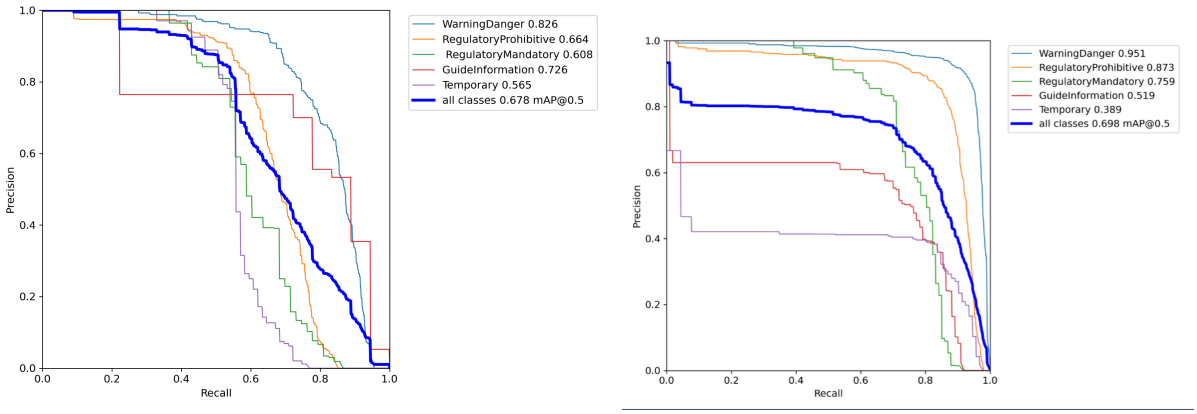


(Epoch 400) (Epoch 100)

Figure 13: Comparison F1 Curve

Figure 13 shows the F1 curves for epochs 400 provides valuable insight into the time course of machine learning model performance F1 scores, a metric of balance and recall, are tested at different training stages to measure patterns of skills in time. Trends in model performance are visualized by plotting F1 scores on a graph with time on the x-axis and F1 scores on the y-axis This comparison allows operators to enable the model to a used to better classify samples or identify potential issues Improve overall performance Means Analysis of these curves helps in fine-tuning the model, refining over parameters, or modifying the training process for better results in tasks such as classification or pattern recognition. Such comparisons contribute to a broader understanding of learning dynamics and iterative adaptations that are crucial for the development of complex machine learning models. In epochs 400, a rising curve or a high plateau

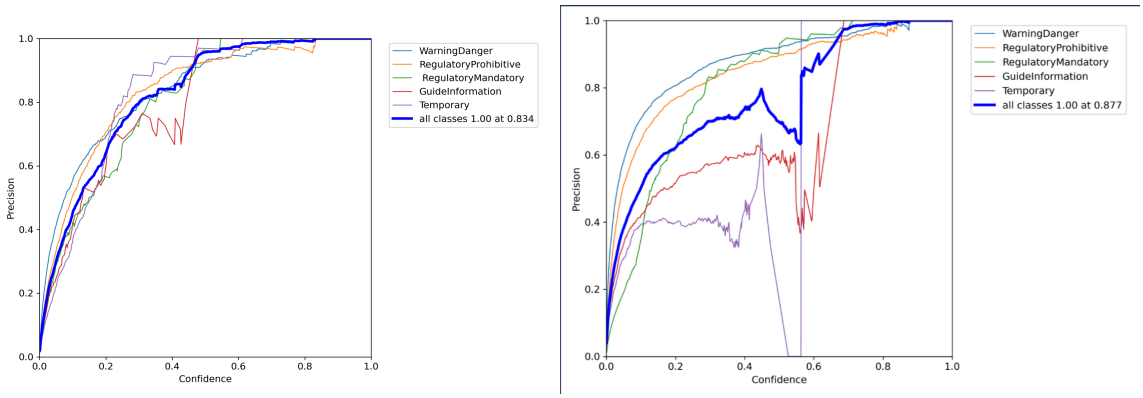
suggests that the model maintains a good balance between precision and recall across different decision thresholds. Meanwhile, in epochs 100, it indicates that the model might need more training to improve its ability to balance F1 and confidence.



(Epoch 400) (Epoch 100)

Figure 14: Precision Recall Curve

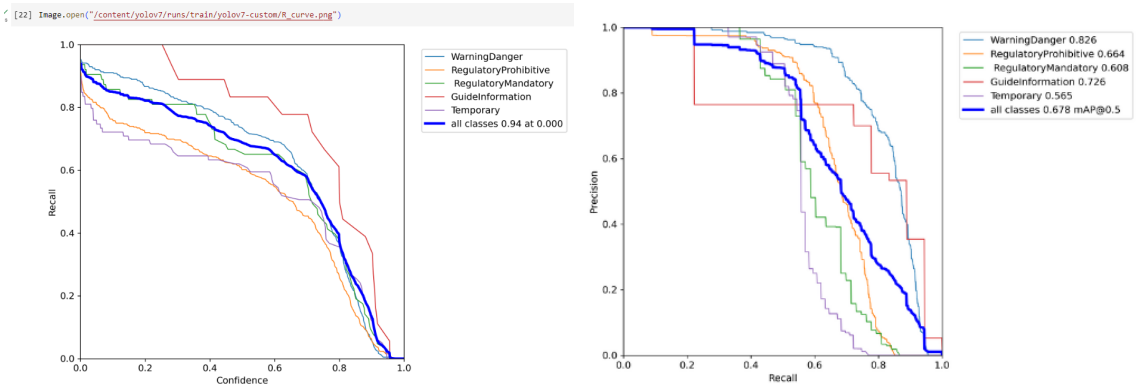
In evaluating our model proficiency in object detection, at epochs 400 reveals a substantial enhancement. The model demonstrates an exemplary level of precision, consistently providing accurate positive identifications which is averagely 0.8 . Moreover, the recall metric indicates that the model effectively captures a significant portion of the relevant objects in the dataset. This progression signifies a noteworthy advancement in the model's overall performance, reflecting its increased reliability and efficacy in object recognition. The precision-recall curve at 400 epochs is smoother and maintains higher values compared to earlier epochs; it suggests that the model has become more stable and effective. Meanwhile, for epoch 100 at the beginning it shows the curve precision and recall are well balanced but through the process. Regions of the curve with lower precision or recall between epochs 400 and epochs 100 indicate potential areas for improvement. Consider further training, adjusting hyperparameters.



(Epoch 400) (Epoch 100)

Figure 15: Precision Curve

The precision curves for epochs 400, shows how the model got better at avoiding mistakes. In Epoch 400, there were some ups and downs, suggesting the model was not very consistent in getting things right. A rising curve indicates that the model is becoming better at making positive predictions accurately while minimizing false positives. However, as progressed to Epoch 400, things steadied out, and the curve stayed higher. Meanwhile, epoch 100 shows the model is decreasing in precision, there were also some ups and downs, suggesting the model was not very consistent in getting things right but at the end it is getting better than epoch 400.



(Epoch 400) (Epoch 100)

Figure 16: Recall Curve

Figure 16 illustrates the recall performance of the model for 400 epochs and 100 epochs. Both epochs show ups and downs. Epochs 400 show lower recall values or fluctuations from the beginning, indicating that the model is not capturing all relevant instances well but the confidence is good. Meanwhile, for epochs 100, it shows the balance progress in recall but as the process goes by, it's getting some high and low until the end process shows the value of recall is in the range of 0.6 to 1.0. Recall, synonymous with sensitivity or true positive rate, gauges the model's proficiency in accurately identifying all pertinent instances within a dataset. An enhancement in the model's capability to capture positive instances needs to be made for better performance.

```
Image.open("/content/yolov7/runs/train/yolov7-custom/train_batch7.jpg")
```

Figure 17: Train image

Figure 17 shows the code is attempting to open and load an image file named "train_batch7.jpg" located in the specified directory. After opening the image, It can perform various image processing operations or analysis. It's common to use libraries like PIL or OpenCV for tasks such as loading, manipulating, and processing images in Python. The Training image that has been performed. The output is as above.

1. Predict using image

```
[ ] img = "/content/labelimg/images/13.jpg"
```

```
[27] results = model(img)
```

```
[26] %matplotlib inline
     plt.imshow(np.squeeze(results.render()))
     plt.show()
```

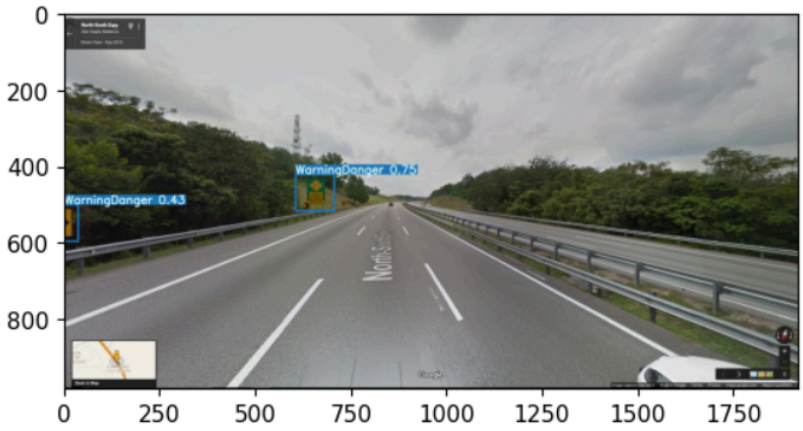


Figure 18: Predict using image

Figure 18 shows the code provided is using a model to make predictions on the image and then visualizing the results using matplotlib.

9.0 Conclusion

In conclusion, the goal of this study has been to progress the field of traffic sign identification and recognition, emphasising the use of the YOLOv7 deep learning model. Understanding the critical role that precise traffic sign recognition plays in the development of intelligent autonomous vehicles and driver support systems, we tackled the difficulties presented by real-world situations such changing lighting, motion blur, and speed variations. After extensive testing on the Extended Malaysian Traffic Sign Dataset (EMTD), our suggested methodology based on YOLOv7 showed encouraging accuracy and efficiency results. The model demonstrated robustness in various environmental circumstances, such as low light, bad weather, and lightning-fast situations. The effective incorporation of YOLOv7 into the traffic sign detecting system represents a significant advancement in the capabilities of self-driving cars, leading to increased traffic efficiency and road safety. This project has practical implications that include improving traffic safety, enabling intelligent transportation networks, and advancing autonomous car technology.

10. Reference

WongKinYiu. (n.d.). *GitHub - WongKinYiu/yolov7: Implementation of paper - YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. GitHub. https://github.com/WongKinYiu/yolov7

Dataset - Reilly, S., Clancy, I., Foy, S., & Madden, M. (2018). Extended Malaysian Traffic Sign Dataset (EMTD) (Version 1) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.1217105

Reilly, S. D., Clancy, I., Foy, S., & Madden, M. G. (2018). Extended Malaysian Traffic Sign Dataset (EMTD). *Zenodo*. https://doi.org/10.5281/zenodo.1217105

View of Malaysian Traffic Sign Dataset for traffic sign detection and recognition systems. (n.d.). https://jtec.utem.edu.my/jtec/article/view/1423/934

Link Google Colab

Latest TrafficSign.ipynb