CLASSIFICATION OF MOTORCYCLE RIDING PATTERN BASED ON COMPUTER VISION AND MACHINE LEARNING

VATTIYA JARUNAKARINT

MASTER OF SCIENCE
IN
INFORMATION TECHNOLOGY

SCHOOL OF INFORMATION TECHNOLOGY
MAE FAH LUANG UNIVERSITY
2021

CLASSIFICATION OF MOTORCYCLE RIDING PATTERN BASED ON COMPUTER VISION AND MACHINE LEARNING

VATTIYA JARUNAKARINT

THIS THESIS IS A PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
INFORMATION TECHNOLOGY

SCHOOL OF INFORMATION TECHNOLOGY
MAE FAH LUANG UNIVERSITY
2021

CLASSIFICATION OF MOTORCYCLE RIDING PATTERN BASED ON COMPUTER VISION AND MACHINE LEARNING

VATTIYA JARUNAKARINT

THIS THESIS HAS BEEN APPROVED
TO BE A PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE
IN
INFORMATION TECHNOLOGY
2021

EXAMINATION COMMITTEE

.....CHAIRPERSON
(Asst. Prof. Worasak Rueangsirarak, Ph. D.)

.....ADVISOR
(Surapong Uttama, Ph. D.)

.....EXTERNAL EXAMINER
(Asst. Prof. Teerawat Kamnardsiri, Ph. D.)

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to some of the individuals and organizations that have supported me through my research. First of all, I would like to express my sincere gratitude to Dr. Surapong Uttama for his enthusiasm, patience, insightful comments, useful information, practical advice, and tireless ideas and write this dissertation.

I feel very honored to receive this financial support of my thesis and research presentation from Mae Fah Luang University to complete my master's degree. I would also like to express our sincere gratitude for accepting admission to the graduate program at the Mae Fah Luang University.

Vattiya Jarunakarint

Thesis Title	Classification of Motorcycle Riding Pattern Based on Computer Vision and Machine Learning
Author	Vattiya Jarunakarint
Degree	Master of Science (Information Technology)
Advisor	Surapong Uttama, Ph. D.

ABSTRACT

Motorcycle accidents have become fatal road accidents in many areas, especially in developing countries. Many studies on vehicle detection have been published, but most of them focus on four-wheel vehicles. However, research on motorcycles is not sufficient, and no one has verified the detection efficiency of motorcycles. This research provided a comparison of CPU and GPU in the TensorFlow model and examined the performance of 34 different machine learning models in motorcycle detection under different conditions. The experiment included 13 video sets with a total of 672 frames. Three different views of the motorcycle (front, top, back) and different resolutions of the image were taken into account. Model efficiency was calculated by the quotient of detection accuracy and time. By lowering the resolution, efficiency has improved significantly on most models. The model that provides the most efficiency for front, and, and top is CenterNet HourGlass104 512x512 with efficiency above 300.

Another process is a new framework for using recorded video footage to detect abnormal riding in three dangerous cases: weaving, swerving, and drifting. The methodology consists of two main steps. First, we localized the motorcycle into a video frame using the best model from previous experiment. Next, obtaining centroid of the detected motorcycle and then using two linear regression models. Ordinary least

squares (OLS) and random sample consensus (RANSAC) have been tuned to find linearity. Riding patterns with high regression values tend to be normal riding. In the experiment, OLS worked well to distinguish between normal and abnormal driving. An OLS score or R-squared threshold of around 0.95. For classifying abnormal pattern, OLS can separate normal pattern and drifting pattern from the other two abnormal patterns by using R-squared score. Polynomial regression model was implemented to fit curve line such as swerving pattern and weaving pattern. OLS and Polynomial models could not classify sub-group of abnormal patterns such weaving (Big curve), and weaving (Small curve). Classification tree algorithm was applied to classify all the patterns. Therefore, 70/30 training and testing method, 10-fold cross validation were implemented. 70/30 training and testing method gave 70.5 % classification accuracy. 10-fold cross validation provided slightly higher classification accuracy than 70/30 method with 75.7% classification accuracy. Straight pattern, drifting (both big, and small curve) pattern, weaving (Big curve) was easily to be classified. On the contrary, swerving pattern was the most difficult to be classified.

Keywords: Abnormal Riding, Motorcycle Detection, Deep Learning, Machine Learning

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENTS	**(3)**
ABSTRACT	**(4)**
LIST OF TABLES	**(8)**
LIST OF FIGURES	**(10)**
CHAPTER	
1 INTRODUCTION	**1**
1.1 Background and Rational	1
1.2 Objective	2
1.3 Scope	2
1.4 Methodology	3
1.5 Project Plan	3
2 LITERATURE REVIEW	**4**
2.1 Motorcycle Detection	4
2.2 Detecting Abnormal Riding	6
2.3 Related Technology	8
3 METHODOLOGY	**9**
3.1 Comparing Motorcycle Detection	10
3.2 Measurement of Accuracy and Time	18
3.3 Abnormal Pattern Detection	19

TABLE OF CONTENTS (continued)

	Page
CHAPTER	
4 EXPERIMENTAL RESULT	**24**
4.1 Experimental Environment	24
4.2 Dataset	24
4.3 Experimental Result from TensorFlow 1 Models	24
4.4 Experimental Result from TensorFlow 2 Models	34
4.5 Detecting Abnormal Patterns of the Motorcycle	51
4.6 Classify Types of Abnormal Riding	57
4.7 Classification Tree	65
4.8 70/30 Training and Testing Method, and 10-Fold Cross Validation	67
5 DISCUSSION AND CONCLUSION	**73**
5.1 Discussion and Conclusion	73
5.2 Publication	75
REFERENCES	**76**
CURRICULUM VITAE	**83**

LIST OF TABLES

Table	Page
1.1 Project plan	3
4.1 Detection accuracy	26
4.2 Average detection time	27
4.3 Detection efficiency	29
4.4 Efficiency at two image resolutions	31
4.5 Average efficiency of the performances on CPU and GPU	32
4.6 Detection accuracy of TensorFlow 2 models	35
4.7 Average detection time	37
4.8 Average detection efficiency	38
4.9 Accuracy, time, and efficiency of front videos	40
4.10 Detection results on front-view images at the lower resolution	41
4.11 Average efficiency at two resolutions on front-view images	41
4.12 Average efficiency at two resolutions on back-view images	43
4.13 Detection accuracy of top-view videos	44
4.14 Detection efficiency of top-view videos (original size)	45
4.15 Average efficiency at three resolutions on the top-view image (T3)	46
4.16 Detection efficiency of TensorFlow 1 and TensorFlow 2	50
4.17 Detection accuracy of TensorFlow 1 models on back-view images	50
4.18 Example of detection result from simulated videos	52
4.19 R-Squared of ordinary least squared and RANSAC score	56
4.20 R-squared score from OLS model	58
4.21 R-squared score from OLS for straight pattern and drifting pattern	60
4.22 R-Squared of different exponents	62

LIST OF TABLES (continued)

Table **Page**

4.23 R-squared score from Polynomial regression model for swerving pattern and weaving pattern 63

4.24 Results from Polynomial regression (Sixth degree) 65

4.25 Classification accuracy from 70/30 method and 10-fold cross validation 69

4.26 Results from polynomial regression and OLS (Sixth degree, forth degree, two degree) 69

4.27 Classification accuracy from 70/30 method and 10-fold cross validation (Pre-possessing data included) 71

LIST OF FIGURES

Figure	Page
3.1 Overall scheme of methodology	9
3.2 Proposed methodology	10
3.3 Three selected clips for front, back and top views	12
3.4 Overall methodology of testing TensorFlow 2 models	13
3.5 Three videos represent front, top, and back	15
3.6 Three front videos	16
3.7 Three top videos	17
3.8 Low-light video	18
3.9 Abnormal riding	19
3.10 Methodology	20
3.11 Overall methodology	21
4.1 Example of detection result	25
4.2 Average accuracy versus time	30
4.3 Accuracy versus time of acceptable models	30
4.4 Example of motorcycle detection results on three different views	34
4.5 Accuracy versus time on back-view detection	42
4.6 Example of motorcycle detection with low-light condition	47
4.7 Comparison of average detection accuracy and COCO mAP	48
4.8 TensorFlow 1 and TensorFlow 2 (Accuracy versus Time)	49
4.9 OLS and RANSAC versus Polynomial regression	57
4.10 Samples of straight and drifting	59
4.11 Example of different value of exponents	61
4.12 Classification tree (Sixth degree)	66

LIST OF FIGURES (continued)

Figure	Page
4.13 Overall methodology (Classification algorithms)	67
4.14 Confusion matrix (Sixth degree)	68
4.15 Confusion matrix (70/30 method)	70
4.16 Confusion matrix (10-fold cross validation)	71
4.17 Classification tree (Pre-possessing data)	72

CHEPTER 1

INTRODUCTION

1.1 Background and Rational

It is common for individuals to get involved in motorcycle accidents, whether as a pedestrian or a car driver. In all types of accidents, motorcycle riders are usually the ones who suffer the most severe injuries. They are also the most frequent types of accidents that cause the most fatalities. In emerging country, motorcycle riding is the preferred type of transportation due to its low cost and ease of travel. In emerging country, most of the fatalities caused by traffic accidents are pedestrians and motorcycle riders. One of these is Thailand, which has a high number of fatal traffic accidents. According to a report released by the WHO [1], the number of road accidents in Thailand has not decreased since 2000. The number of fatal injuries caused by traffic accidents in Thailand was ranked 8th among all deaths in 2018. Also, over half of the fatal accidents that happened were involving pedestrians, cyclists, and motorcycle riders. In 2018, over 22,000 people died in road accidents in Thailand, and 74 percent of them were motorcycle riders. A report released by an accident data center of the country for June 15, 2022 [2], shows that there are over 398,000 injured individuals and 6,758 deaths caused by road accidents. Drunk driving, high speed, drowsiness, or bad drivers' behavior are causes of road accidents. These risky behaviors are related to abnormal driving which is a term explaining reckless behavior of drivers. Abnormal driving normally can be split into three types that are reckless driving, drowsy driving, and distracted driving. The abnormal driving is a very dangerous behavior for riders' life, for example, distracted driving is when the drivers are looking at something alongside the road or playing electronic devices such as a mobile phone while driving. This will take the drivers attention from the road and the accident is possible to occur.

Mostly this accident take place between cars and motorcycles because motorcycles are hard to be noticed in a specific area for drivers' view.

There are many studies on abnormal driving. One of these studies used a camera to monitor the eyes of drivers to determine if they were distracted while driving [3]. Another study used driving patterns to differentiate between normal and abnormal drying. According to the National Highway Traffic Safety Administration (1998) [4], these patterns can lead to accidents such as swerving and weaving. Although, detecting parts of drivers' body could be difficult because of a tint window or a helmet. Consequently, one of a suitable method is to apply software tools such as object detection models to cover this limitation. Due to the technological advancements that have occurred in the field of object detection in recent years, it has gained widespread acceptance. This discipline is studied and implemented in various academic and commercial organizations. For instance, in security systems, it can be used to identify people or objects that are hidden. In vehicle detection, it can be used to spot abnormal driving patterns.

1.2 Objective

1.2.1 To find the most efficient model for detecting motorcycle.

1.2.2 To classified abnormal riding patterns.

1.3 Scope

1.3.1 To focus on existing detection models which are available on public repositories.

1.3.2 To research on risky riding patterns of motorcycles.

1.3.3 To gather videos of motorcycles.

1.4 Methodology

Phase 1 –Defining the problems.

Phase 2 – Reviewing the literature reviews.

Phase 3 – Collecting data.

Phase 4 – Implementing detection models

Phase 5 – Applying regression models to detect abnormal pattern.

Phase 6 – Evaluation

1.5 Project Plan

This schedule consists of columns, which are the semester and rows, which are phases that was defined in methodology at above. The black highlight is the phase of work that are performed at that semester.

Table 1.1 Project plan

Phases/ Semester	**1st Semester**	**2nd Semester**	**3rd Semester**	**4th Semester**
Phase 1				
Phase 2				
Phase 3				
Phase 4				
Phase 5				
Phase 6				

CHEPTER 2

LITERATURE REVIEW

In this chapter, the researcher gives an overview of works related to motorcycle detection and detecting abnormal riding.

2.1 Motorcycle Detection

In vehicle detection, object detection plays an important role in reducing traffic accidents. For example, Jang and Ahn [3] gave the method to avoid road accidents from drowsiness by implementing machine learning and a CO2 sensor chip. The author used machine learning to predict drowsiness by detecting face recognition, and eye-blink. Hasanin and Hassan [5] developed a Hidden Markov Model (HMM), they propose a method for detecting drunk driving behavior. Data on driver behavior was collected via the Controller Area Network (CAN) connected to the on-board unit (OBU). They implemented a Recurrent Neural Network and achieved over 90% accuracy. In addition, Xinrong, Junwei, Jinghe and Yanchao [6] interested in the unusual driving behavior of buses aim to improve the safety and traffic safety of bus drivers. Data was collected using a smartphone accelerometer and direction sensor. A Bayesian classifier was implemented to detect different types of anomalous driving behavior, with the highest accuracy of 98.40 percent. The combination of software and hardware is widely used to detect vehicles with high accuracy. However, most hardware sensors are very expensive, and some detection techniques may require many sensors to be able to detect objects. Damaged hardware can lead to inaccuracies and errors during processing. As a result, improvements in detection software, algorithms, or models are gaining more and more attention from developers.

As a result, alternatives for detecting vehicles without specific hardware are attracting the attention of researchers. Various techniques were proposed to detect vehicles. Vision-based classifications have been found to recognize different views (side, front, back) of bicycles and motorcycles to distinguish between motorcycles and bicycles [7]. Vision-based systems have also been used to solve vehicle blockage problems, and one researcher has proposed a blockage segmentation method for detecting motorcycles [8]. Image processing is one of the well-known techniques for detecting vehicles. This study [9] used two main methods to detect vehicles from the front: edge detection and morphological processing. A key feature of edge detection is to enhance the object in the image. Noise is a major concern for inaccuracy in image processing, morphological processing is a process to reduce noise and detect vehicles precisely. Both methods were done in a popular scientific tool which is MATLAB. Another well-known technique is deep learning, which is a powerful tool for detecting objects in images. Authors [10] presented the idea of using a deep learning architecture to count the number of vehicles on the freeway. It uses a dataset of 11,129 images to improve vehicle detection and implements a model architecture called YOLOv3 (You Only Look Once) to detect the location and type of objects. By using ORB algorithm, the number of vehicles can be counted. Another demonstration [11] is a technique for distinguishing vehicle types in the traffic scene using a deep learning architecture. The author applied another well-known deep learning framework called Faster-RCNN, combining datasets from MIT, California Institute of Technology cars, and various car models on the Web. The purpose is to distinguish between the three types of vehicles commonly found on the road: cars, minibuses, and off-road vehicles. Another example of implementation of FASTER-RCNN and RCNN [12], This paper proposes a successfully trained model that provides effective vehicle detection.

Most vehicle detection studies often refer to four-wheeled vehicles such as cars and buses. Therefore, motorcycle detection is a little less researched than car detection. There is little work has been done on this subject. Nonetheless, several studies focused on motorcycle detection, for example, this paper have succeeded in detecting motorcycles and helmets with an accuracy of over 90% in a variety of weather conditions. Robust Object Segmentation technique was implemented to extract background from moving object and create a bounding box around object by using

connected component labeling [13]. The end result provides the speed and number of motorcycles. Use of a deep learning convolutional neural network model known as "EspiNet" and Markov decision process (MDP) for tracking motorcycles [14]. The author trained the model with 10,000 annotated images from a public website. This study [15] provided a trained, robust convolutional neural network (CNN) model with multiple layers and was able to detect a congested motorcycle in a single image. Moreover, some researches compared models or techniques. In addition, some researchers compared models and methods. According to [16], an analysis of the characteristics of the cognitive model and a description of the benchmark dataset. After the analysis process, the author gave an overview of the object detection method, which consists of one and two detectors. Finally, an improvement the object detection method and how to build an efficient model will be discussed. Furthermore, this study [17] A comparison of Faster-RCNN, MoG, and SVM systems shows that Faster-RCNN outperforms both MoG and SVM in terms of vehicle detection. Some researchers aimed to research the overall performance of item movement detection, automobile category and additionally monitoring algorithms [18]. Detection accuracy can be affected by hardware, camera placement, and lighting conditions. This paper [19] described the strengths and weaknesses of monocular vehicle detection, night vehicle detection, and detection for various applications. Authors [20] proposed a survey of Driver Assistance System (DAS) which is a warning system that could be very helpful when there is a situation where accidents likely to occur.

2.2 Detecting Abnormal Riding

Many descriptions of abnormal driving behavior have been proposed. Drivers who are drunk driving or have mental or physical appearance problems are usually quoted in driving behavior. NHTSA [4] showed some examples of problems in maintaining the correct lane position. Start by weaving the vehicle. When weaving, the vehicle moves to one side of the lane and then to the other side. This situation can occur during drunk driving. Another problem is vehicle drifting. The vehicle moves in a straight line and slides sideways into another lane. Problems can occur when there is

heavy rain that causes slippery roads. Swerving occurs when the vehicle leaves the lane. For example, if the driver finds the vehicle off track after grabbing the steering wheel and falling asleep, the driver will try to return to the track. There are many works that have contributed to solutions that detect abnormal driving using sensors. Kwangsoo, Bong, Jun and Dong [21] proposed a system that issues an alarm in abnormal driving conditions and warns the driver of imminent danger. Abnormal driving situations can be divided into three situations: distracted driving, drowsiness driving, and reckless driving. Researchers primarily used hardware sensors to detect dangerous situations and generate alarms. Jiadi et al. [22] proposed Driving behavior detection and identification system (D3). By Using smartphone sensors to monitor abnormal driving behavior. For example, use a smartphone to generate rough results and distinguish between normal and abnormal driving. It also uses a fine-grained monitor to detect anomalous driving and types of anomalous driving (swerving, weaving, fast U-turn side, slipping). 95.36% was a outcome from using D3. Jie, Xiaoqin and Steve [23] applied a deep learning approach to detect abnormal driving. Researchers have established a new deep learning-based model for detecting abnormal driving. Collect driver behavior data using instrumented vehicles. Applying an auto-coded model to learn the characteristics of anomaly driving, researchers were able to achieve 98.33 percent of anomaly detection accuracy. Xinrong et al. [6] focused on abnormal bus driving behavior with the aim of improving bus driver safety and traffic safety. They classified abnormal driving behavior into five types: lane change, sudden braking, fast turns, fast U-turns and long parking. In the early process, researchers used smartphone, accelerometers and directional sensors to collect bus driving data. Bayesian classifiers have been applied to identify different types of anomalous driving behavior and provided 98.40 percent accuracy. Sometimes social drinking is inevitable. It's about companionship. However, drinking too much can cause mental damage and affect your driving ability. There are studies aimed at detecting the behavior of drunk drivers. Hasanin and Hassan [5] studied drunk driving, they have developed a hidden Markov model (HMM) from the previous work. Data on driver behavior was collected via the Controller Area Network (CAN) connected to the on-board unit (OBU). They expanded their work by applying a recurrent neural network and were able to achieve an accuracy of 90% or higher. Sandeep, Ponnam and Sura [24] used Raspberry Pi3 and multiple

sensors such as heart rate sensor, GPS device, drunk driver detection alarm device, etc. Facial recognition is used to know if a driver is drunk. Saif, Ali and Hussein [25] combined intelligent component of a transportation system called Vehicular Ad-hoc Networks (VANET) with sensors that detect driving abnormalities very effectively. They used VANET's context recognition system to get information about the driver's behavior and environment. In addition, they used different types of sensors to collect contextual information. Lastly, they proposed an algorithm that takes information as input and creates a system that can detect abnormal driving and alert other vehicles to avoid accidents.

The topic of abnormal driving detection has been widely proposed by many researchers in several ways. Most methods typically used the sensor as the main core for collecting data from the driver. Therefore, the proposed technique mainly uses a camera to capture video, and therefore most sensors are very expensive, reducing the cost of the equipment. Detection techniques may require many sensors to archive the objective. If any of them are corrupted, problems will occur during the detection process. The three proposed methods can also capture video from CCTV, which is usually placed in the position required for the detection system. Also, as far as I know, research on abnormal driving detection focuses mainly on four-wheeled vehicles such as automobiles and buses. However, our job was to detect unusual driving behavior of motorcycles. Our work only requires video from cameras or CCTV and open-source software to detect abnormal driving. This means that resource costs are minimized and no sensors are needed.

2.3 Related Technology

2.3.1 A free and open-source software for machine learning called TensorFlow. TensorFlow is flexible ecosystem of tools, easy to build the model, and libraries resources for researchers. It has wide range of usage and it focuses on training and deep learning networks.

2.3.2 Scikit-learn is open source tool for predictive data analysis [26]

CHEPTER 3

METHODOLOGY

There are mainly six stages for the research. Firstly, collecting the motorcycle videos. Then, comparing all chosen detection models with two resolutions which are 1920 x 1080 pixels and 480 x 270 pixels. Then, pick the most suitable model to capture the motorcycles riding abnormally. Next, obtain centroid of each motorcycle in every image frame and plot them. Applying these results in regression models to classify them into abnormal riding and normal riding. Finally, evaluation will be covered. Figure 3.1 depicts an overall scheme of our proposed methodology.

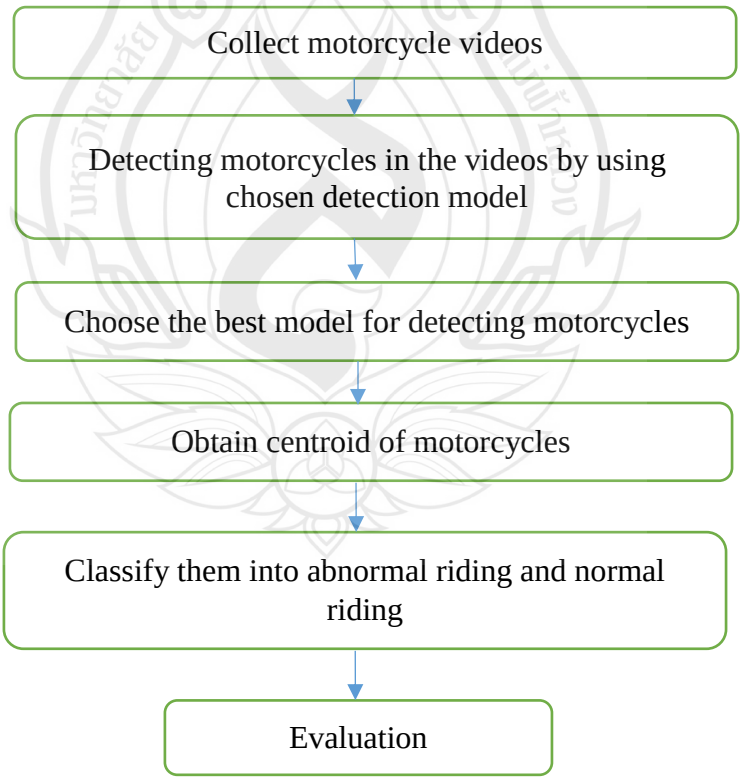


Figure 3.1 Overall scheme of methodology

3.1 Comparing Motorcycle Detection

This section, we propose an experimental investigation of motorcycle detection via both classical machine learning technique and deep learning approaches. We applied totally 34 different models to three short videos containing motorcycles and then measured detection accuracy and time. The ratios between these two measurements are compared to yield the detection efficiency. Then the top models were chosen to be evaluated again against the same videos but at a lower resolution. Proposed methodology is shown in figure 3.2.

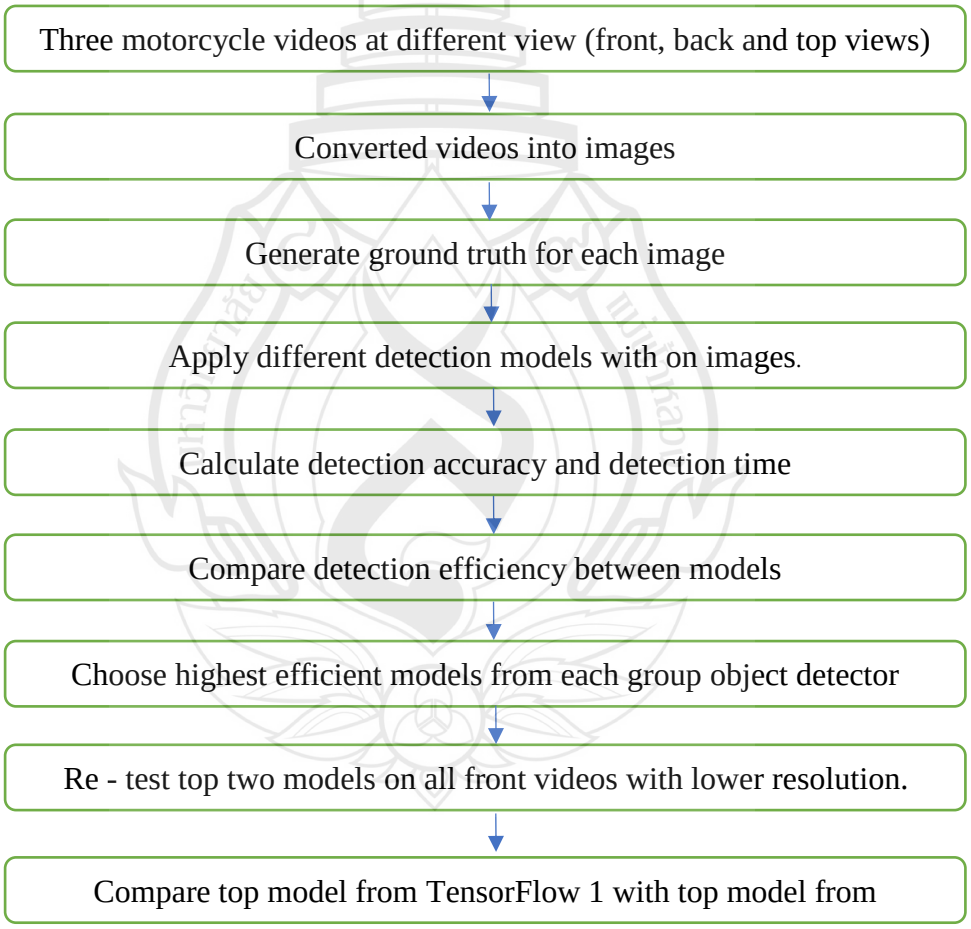


Figure 3.2 Proposed methodology

Most of the detection models are selected from TensorFlow which are 31 models in total. There are two versions of TensorFlow models that are TensorFlow 1 and TensorFlow 2 [27]. Thus, we divided our work for testing efficiency of the models into two parts. The first part is to have experimental investigation on TensorFlow 1 and TensorFlow 2 models. The second part is to find the best model by comparing all models from TensorFlow 1 and TensorFlow 2 with certain dataset. In addition, evaluate the CPU and GPU performance running on the TensorFlow1 model. Retesting all TensorFlow1 models on the GPU. The comparison of existing measurements with our measurements is tested.

3.1.1 TensorFlow 1 Models

3.1.1.1 Data preparation

First, three short video clips with motorcycles in front, back and top views as shown in Figure 3.3 were collected from our public repositories and web recordings. Each video has a resolution of 1920 x 1080 pixels for horizontal videos and 1080*1920 pixels for vertical videos. We then take video frames in which motorcycles are presented, that's a total of 49 images i.20, 16 and 13 frames from the front, back and top. We then labeled motorcycles in each frame to create ground truth. These steps were also repeated for images rescaled to 480x270 pixels.

3.1.1.2 Selection of motorcycle detection algorithm

We examined all accessible machine learning algorithms available in public repositories and found two main sources. One is a popular object detection algorithm based on Convolution Neural Network (CNN) implemented in TensorFlow. Then we acquired 17 working models trained on COCO dataset [27] and 2 models trained on YOLO [28] dataset which includes motorcycle. Another is a classical object discovery model reckoned on Haar Cascade classifier [29]. Therefore completely 20 models were chosen for comparison in this part.

Figure 3.3 Three selected clips for front, back and top views

3.1.2 TensorFlow 2 Models

In this section, by working on both machine learning and deep learning techniques, we are investigating the performance of motorcycle detection. TensorFlow2 Models Implemented 14 different models from the zoo. The TensorFlow 2 Model Zoo consists of more than 10 different models with different types of object detectors such as Centernet, SSD (Single Shot Detector), EfficientDet, FasterRCNN and more.

We implemented all models to a few short films wherein motorcycles are presented. Then, detection accuracy and detection time have been measured. We used consequences from those measurements to compute the performance of the models. Next, we pick out top models from each group of object detectors which can be Centernet, SSD (Single Shot Detector), EfficientDet, and Faster-RCNN and implemented those top models to a few new videos. Next, the two best models with close results were selected. The two models were tested at both the original and lower resolutions. Finally, we compared the top model of TensorFlow 1 with the top model of TensorFlow 2. TensorFlow 2 is an updated version of TensorFlow 1 with a new object detector added to TensorFlow 2. An overall process is shown in Figure 3.4

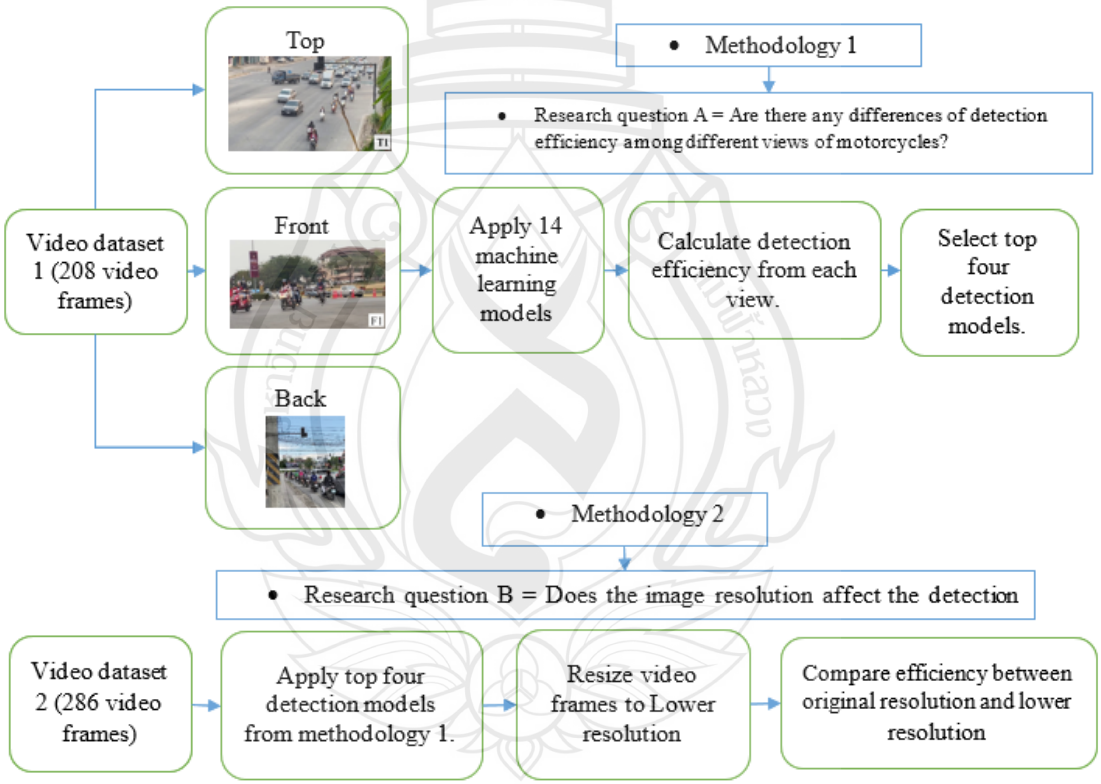


Figure 3.4 Overall methodology of testing TensorFlow 2 models

3.1.2.1 Data preparation

There are ten motorcycle videos along with 4 front-view movies, one back-view video, four top-view videos, and one video with low-light condition completely of 623 video frames. In the early stage, a group of three short videos which includes front, top and back perspectives have been collected from our camera as proven in Figure 3.5. The original length of the videos is 1980x1080 pixels for horizontal videos and 1080x1980 for vertical videos. Next, we amassed video frames wherein the motorcycle has existed. There are 56, 105, 47 video frames (208 video frames in total) for the front, pinnacle, and returned respectively. Next, I labeled each motorcycle displayed on the frame to create a ground truth. Also, most traffic cameras are installed to record the vehicle from the front. Therefore, we collected more videos on the front view to observe performance means of recording three more movies with specific angles of the camera, distances, and traffic congestion scene as proven in Figure 3.6. There are 123 video frames for three new movies i.e., 56, 46, 21 from the first, second, and 1/3 movies. For a clean under-standing, we call the front movies as F1, F2, F3, and F4 for a discussion in a later section. In a real traffic scene, the camera is usually at the same height as the pole of the traffic light. The position of the camera is similar to the position of the CCTV. This makes the top view of the motorcycle the most important of the three different views for recognizing the motorcycle and applying it to the actual scenario. To find the best model for detecting Top-view motorcycles, we have further collected three videos, including 163 video frames for Top-view motorcycles at different angles and distances. We call top-view images as T1, T2, T3, and T4. The three top-view images constitute in Figure 3.7. Moreover, low-light condition has a huge effect on motorcycle detection. We collected one top-view video containing 129 video frames with low-light to check the performance of all models as proven in Figure 3.8

3.1.2.2 Selection of Motorcycle detection models

We investigated a new TensorFlow2 detection model trained on the COCO dataset [27] available in public repositories. Next, we selected 14 models with different architectures than all available models. There are four main object detectors used in this study: FasterRCNN, SSD, CenterNet, and EfficientDet. FasterRCNN is a common object detection architecture that uses a convolutional neural network (CNN) built from three parts: a convolutional network, a region proposal network, and a fully connected neural network. FasterRCNN also has a two-tier network that provides high precision

and large scale for multiscale resolution [30]. Another object detector is the SSD, which has a CNN architecture and a VGG16 base network. It provides a fast detection time [31]. Considering CenterNet, this is a point-based object detection architecture. It can resolve the multi-label issues and has a stability among detection time and accuracy [32]. EfficientDet has CNN structure and is capable of gain excessive accuracy with few training epochs. It is told to beat YOLO with minimal computation power [33].

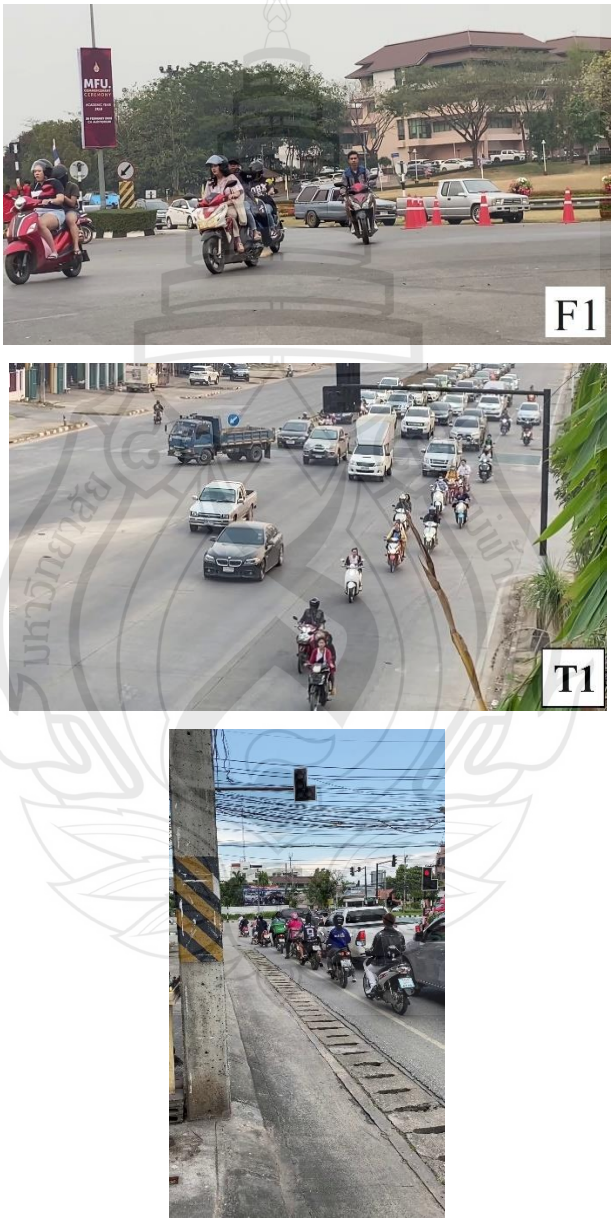


Figure 3.5 Three videos represent front, top, and back

Figure 3.6 Three front videos

Figure 3.7 Three top videos

Figure 3.8 Low-light video

3.2 Measurement of Accuracy and Time

There is a measurement provided by TensorFlow that was originally used to measure the speed and timing of detection models. The measurement is defined with COCO mAP (average accuracy) and speed (ms) [34]. However, the rating given appears to measure all classes of objects and may not be appropriate to measure a specific class, such as a motorcycle class or a car class. Therefore, we proposed a simpler and more common technique to measure accuracy based on the known confusion matrix [35].

Detection accuracy is based on confusion matrix and is computed using (1). Moreover, if the detection accuracy is 70%, it will not mean the models can detect 70 motorcycles out of 100 motorcycles. Detection accuracy is described as the ratio of correctly predicted examples to the total number of examples.

$$\text{Accuracy} = (TP/TN) / \text{Total samples} \quad (1)$$

TP represents True positive and TN represents True negative.

Regarding the detection time, we start a timer when the detection starts and stop the counter when the objects detection ends. We then divide the total detection time by a number of frames and use these numbers to calculate the efficiency.

Efficiency = Accuracy / Time per frames (2)

Efficient models usually need to provide high detection accuracy in a short amount of time.

3.3 Abnormal Pattern Detection

This section presents an experimental study to identify abnormal driving patterns using motorcycle detection and liner models. The proposed methodology consists of three parts: data preparation, motorcycle detection model selection, and anomaly pattern detection. Classify certain types of abnormal driving patterns, such as weaving, bending, and drifting, as shown in Figure 3.9.

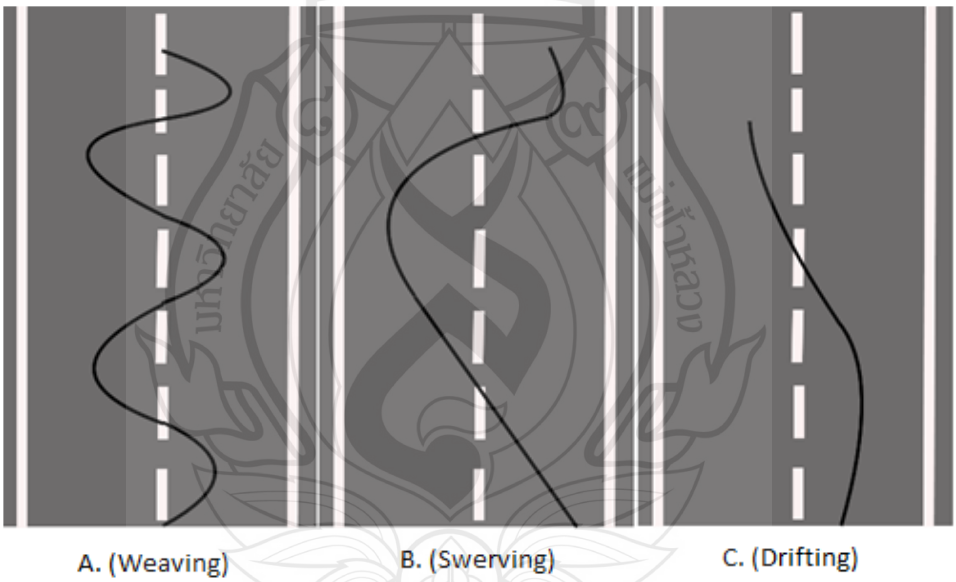


Figure 3.9 Abnormal riding

The first process is to collect videos of both normal and abnormal driving patterns. It is very difficult to find certain types of abnormal driving on the road, as traffic jams and other vehicles can disturb the scene. Therefore, I created a scene where an abnormal driving pattern actually occurs. There are 70 videos in total (10 videos per riding pattern). In order to complete a following process, we need to find the most

efficient model first. Next process is to feed the detected motorcycle video by implementing a champion model from 34 models to gather X and Y coordinate of motorcycle from every video`s frame. The model chosen to detect the motorcycle to provide the highest efficiency in terms of time and accuracy. When you feed the video to the detection model, the model recognizes the motorcycle and creates a bounding box for each frame. For each frame, calculate the x coordinate (the left corner of the bounding box and the right corner of the bounding box divided by 2) and the y coordinate (the value divided by 2 at the top of the bounding box and the bottom of the bounding box). The next process is to use the centroid as input to the normal least squares and RANSAC algorithms. Finally, we observed the results of both algorithms to distinguish between normal and abnormal driving. The overall scheme of the proposed methodology is shown in Figure 3.10.

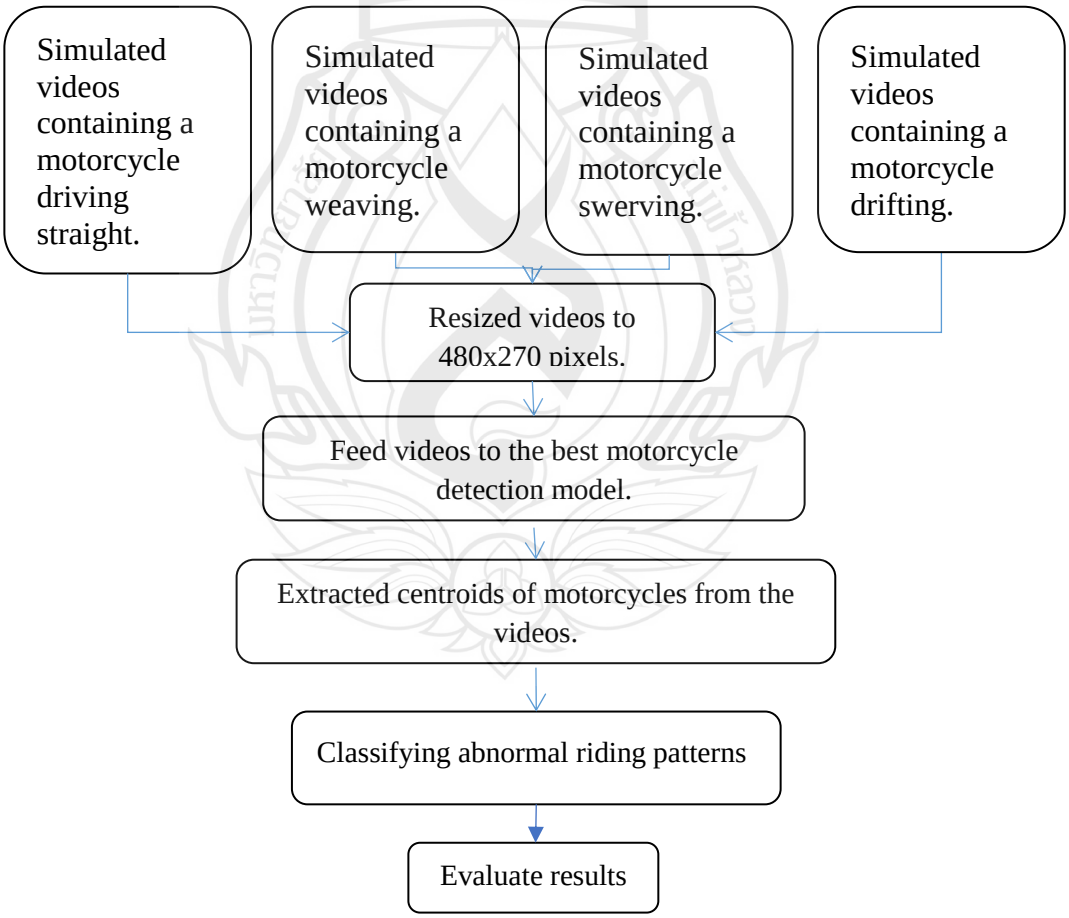


Figure 3.10 Methodology

For the classification part, straight pattern and drifting pattern seem to be linear. Therefore, Ordinary least square model could be a suitable tool to classify these two patterns. On the other hand, swerving pattern and weaving pattern provide quite curve line. Thus, polynomial regression model is more suitable than OLS. An overall method is shown in Figure 3.11.

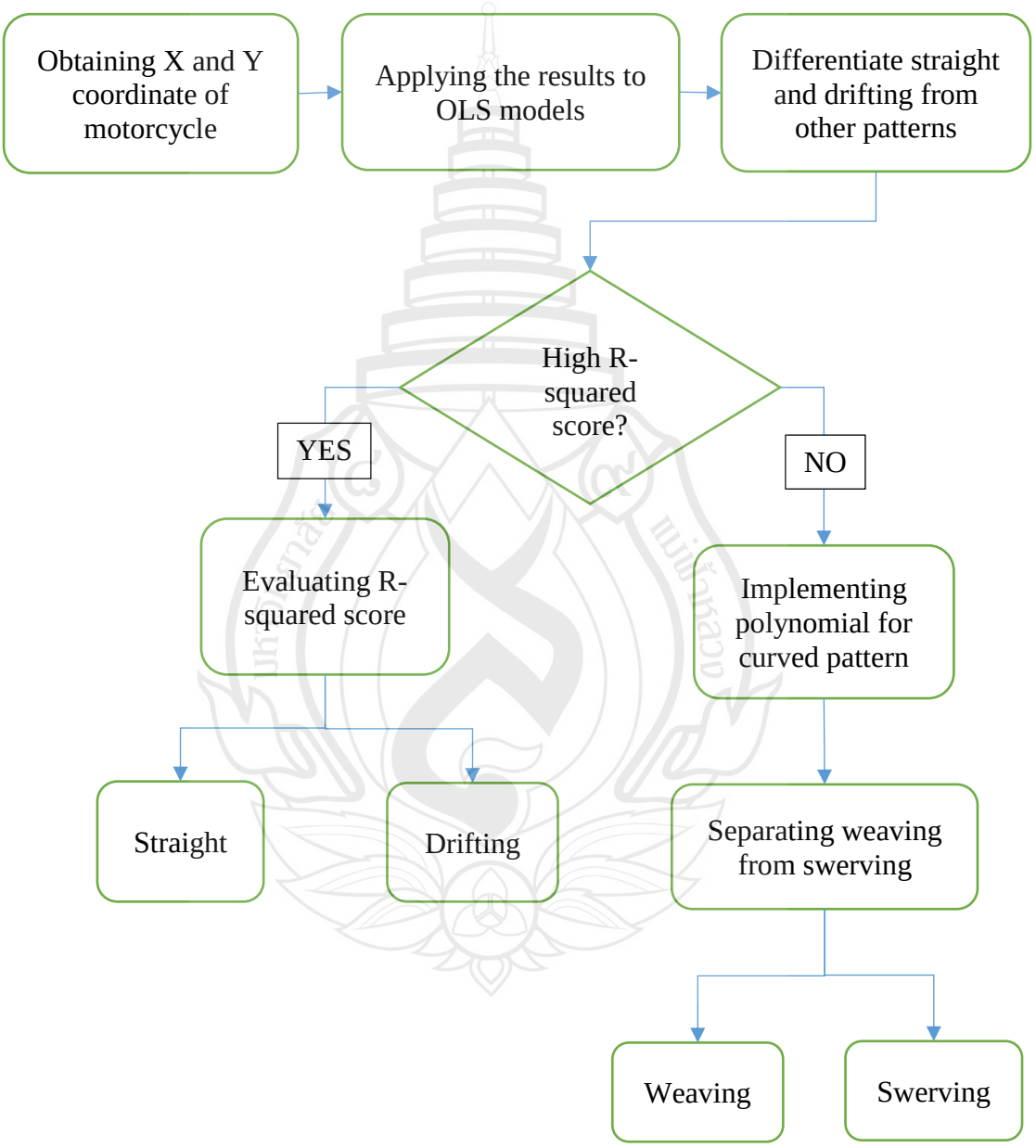


Figure 3.11 Overall methodology

Selection of Algorithms for Detecting Abnormal Riding Patterns

After obtaining centroids of the motorcycle, we aimed to distinguish normal riding from abnormal riding. At the primary stage, normal riding can be defined, while the motorcyclist rides on instantly. Thus, R Squared (1) can be an appropriate approach to become aware of normal riding sample through the usage of ordinary least square (2) due to the fact that straight line will normally provide high value of R Squared. R Squared is measurement for a regression model which specify the proportion of variance for a dependent variable that is explained by an independent variable. In addition, R squared demonstrates the goodness of fit of data for regression model. a variability of the data will not be explained if the R-square value is 0 percentage. On the opposite hand, 100% value of R squared explains all variability of the data. Riding weaving, drifting, or swerving may have effect on low value of R squared, then we might differentiate normal pattern from abnormal pattern with the aid of using R squared which R squared is computed the use of 1).

$$R^2 = 1 - \frac{Unexplained\ Variation}{Total\ Variation} \qquad (1)$$

Where Unexplained Variation is error component of regression equation and Total variation means the sum of explained and unexplained variation.

An equation of Ordinary least squares is

$$Y_i = B_0 + B_1 X_i + \varepsilon_i \qquad (2)$$

Whera Y is independent variable, $B_0 + B_1 X_i$ is linear component and ε_i is random error component.

RANSAC is provided by [26], that is linear, model that getting used on this paper to take a look at an end result offering with the aid of using some other linear model aside from OLS.

After using OLS to distinguish normal pattern from abnormal pattern, we aim to use polynomial regression to classify each type of abnormal pattern. Polynomial Regression is a shape of linear regression wherein the connection among independent variable x and independent variable y is modeled polynomial. Polynomial regression suits a nonlinearity among the value of x and the corresponding conditional suggest of

y, denoted E (y |x). Implementing polynomial regression for curve line because most of abnormal pattern will be presented at curve line. In opposition, normal pattern will mostly present straight line.

70/30 Training and testing set [36, p. 30] will be apply to classify abnormal patterns. In most cases, the percentage chosen is 70% for the training set and 30% for the test. The more training data you have, the better, but the more test data you have, the more accurate your error estimation will be, as the classification model will improve. Another classification algorithm is 10-fold cross validation [37]. In the 10-fold cross-validation, the fitting procedure is performed a total of 10 times. Each fitting is performed on a training set consisting of 90% of the total randomly selected training set, the remaining 10% being used as a retention set. Finally, classification tree [38] will be test on 70 patterns by using coefficients from polynomial regression model. The decision tree is a simple representation for classifying examples. This is supervised machine learning that continuously divides data according to specific parameters.

CHAPTER 4

EXPERIMENTAL RESULT

4.1 Experimental Environment

The tested PC environment is Intel Core i7-8750H CPU @ 2.20 GHz, 16 GB RAM and SSD storage, and NVIDIA GeForce RTX 2070. The software platforms are Python version 3.8.1, TensorFlow version 2.2.1, imageAI version 2.1.6, and Scikit-learn version 0.22 on Windows 10.

4.2 Dataset

There are 13 videos in total. The videos are consisting of 3 different views (front, top, and back) of motorcycles for testing on TensorFlow 1 models. In addition, 4 front-view videos, 4 top-view videos, 1 back-view video, and 1 low-light videos are collected for the experiment on TensorFlow 2 models. The total image frames of 13 videos are 672 frames.

4.3 Experimental Result from TensorFlow 1 Models

4.3.1 Detection Accuracy

38 frames of 1920 x 1080 pixels were tested on the chosen machine learning model. The detection is labeled as shown in Figure 4.1. Please note that we are only focusing on motorcycle detection results.

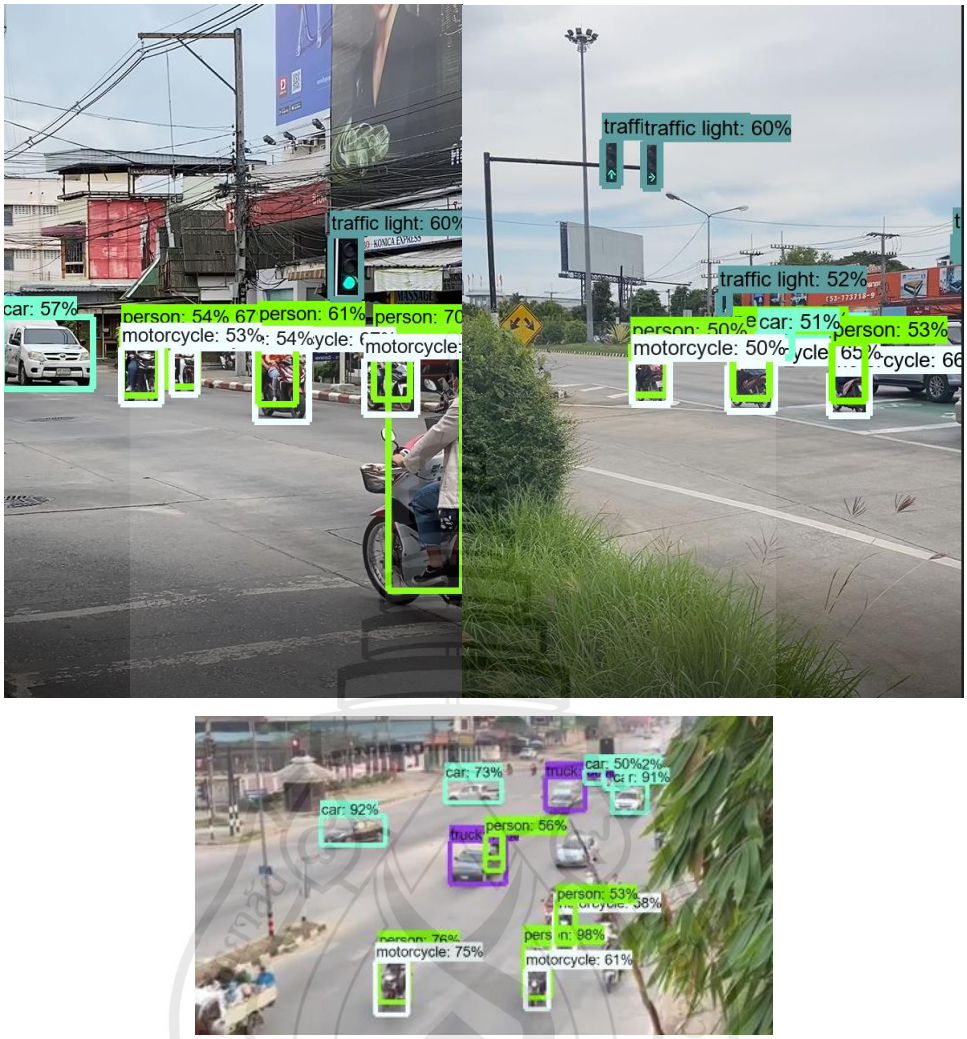


Figure 4.1 Example of detection result

Table 4.1 Detection accuracy

Model No.	Model	Accuracy (%)			
		Front	**Top**	**Rear**	**Average**
1	ssd_mobilenet_v1_coco	70.76	0.00	5.15	25.3
2	ssd_mobilenet_v1_0.75_depthcoco	58.03	8.97	23.25	30.08
3	ssd_mobilenet_v1_ppn_coco	62.22	10.00	41.2	37.8
4	ssd_mobilenet_v1_fpn_coco	84.25	7.69	25.32	39.08
5	ssd_resnet_50_fpn_coco	92	19.40	49.85	53.75
6	ssd_mobilenet_v2_coco	65.09	4.00	15.84	28.31
7	ssdlite_mobilenet_v2_coco	59.29	8.86	23.63	30.59
8	ssd_inception_v2_coco	71.8	6.30	42.23	40.11
9	faster_rcnn_inception_v2_coco	87.1	35.59	44.13	55.6
10	faster_rcnn_resnet50_coco	89.6	27.27	57.74	58.2
11	faster_rcnn_resnet50_lowproposal_coco	86.56	16.67	41.38	48.2
12	rfcn_resnet101_coco	83.86	37.17	63.98	61.67
13	faster_rcnn_resnet101_coco	96.19	35.90	68.16	66.75
14	faster_rcnn_resnet101_lowproposals_coco	83.03	32.05	61.15	58.74
15	faster_rcnn_inception_resnet_v2_atrous_coco	98.98	41.77	62.68	67.81
16	faster_rcnn_nas_lowproposal	78.28	16.67	57.4	50.78
17	mask_rcnn_inception_resnet_v2_atrous_coco	88.59	44.87	84.42	72.62
18	Haar Cascade classifier	72.75	42.90	21.24	45.63
19	YoloV3	93	63.52	83.3	79.94
20	TinyYoloV3	75.28	35.06	15.19	41.84
	Average	**79.98**	**24.73**	**44.36**	

The average accuracy (%) is computed as proven in Table 4.1

First, most models are best detected at 79.98% in front view with average accuracy, but lowest at 24.73%. This is consistent with the fact that the COCO and YOLO datasets used for mannequins mainly contain images of motorcycles in front-view. However, it may not be practical for a typical traffic camera that is usually placed above vehicle.

A model namely “YoloV3” provides the best average detection accuracy which is 79.94%. Also, this model reached the satisfied accuracy of 93 % for the front view. While the Haar Cascade classifier which does not use the deep learning technique, it gave most effective 45.63% of common accuracy and 72.75% for the front images

4.3.2 Detection Time

We calculated total detection time in seconds from the first to the last frame of each video. After which calculate the total detection time or detection time per frame. The final results are provided in Table 2. Remark that the model numbers in Table 4.2 are as same as those in Table 4.1

Table 4.2 shows that most models require different detection times for the front, back, and top images. On average, the model ssd_mobilenet_v1_0.75_ Depth_coco consumes the shortest detection time of 0.93 seconds per frame. In contrast, the mask_rcnn_inception_resnet_v2_atrous_coco model offers a fairly high detection accuracy of 72.62%, but with the longest detection time of 60.95 seconds per frame.

In addition, the Haar cascade classifier requires 4.12 seconds per frame

Table 4.2 Average detection time

Model No.	Average Detection Time (seconds/frame)			
	Front	**Top**	**Rear**	**Average**
1	0.91	1.14	0.91	0.98
2	0.94	1.04	0.81	0.93
3	1.13	0.83	0.97	0.97
4	1.69	1.60	1.5	1.61
5	2.37	2.26	2.15	2.26
6	1.44	1.11	1.02	1.19
7	1.06	0.96	0.87	0.96
8	1.46	1.20	1.28	1.31
9	2.76	1.70	1.92	2.12
10	4.67	2.55	3.33	3.51

Table 4.2 (continued)

Model No.	Average Detection Time (seconds/frame)			
	Front	**Top**	**Rear**	**Average**
11	3.27	2.13	2.48	2.62
12	6.19	3.40	4.41	4.67
13	6.73	3.51	4.76	5
14	6.17	2.98	3.85	4.33
15	64.96	41.39	46.65	51
16	7.69	6.78	7.40	7.29
17	81.71	43.15	57.98	60.95
18	4.2	3.79	4.38	4.12
19	2.86	2.0	3.05	2.63
20	2.65	1.53	2.12	2.10

For real-time detection, commonly a smooth detection frame rate needs to be at the least 24 frames or 0.04 seconds per frame. All models and environments examined on this paper do now no longer fulfill this criterion. Hence withinside the subsequent section, we can speak greater experiments on smaller resolution which tried to deal with this limitation.

4.3.3 Detection Efficiency

Commonly, accuracy is inversely proportional to time. That is, a model that provides high detection accuracy usually consumes more time. Therefore, we proposed to evaluate detection efficiency by finding the relationship between accuracy and time, as shown in Table 4.3.

Table 4.3 Detection efficiency

Model No.	Detection Efficiency			
	Front	**Top**	**Rear**	**Average**
1	77.75	0.00	5.65	27.80586
2	61.73	8.60	28.70	33.02092
3	55.06	12.04	42.47	36.52812
4	49.85	4.81	16.88	23.84611
5	38.81	8.59	23.18	23.52
6	45.20	3.61	15.52	21.44
7	55.93	9.21	27.16	30.77
8	49.17	5.25	32.99	29.14
9	31.55	20.97	22.98	25.15
10	19.18	10.71	17.33	15.73
11	26.47	7.82	16.68	16.99
12	13.54	10.92	14.50	12.99
13	14.29	10.23	14.31	12.94
14	13.45	10.74	15.8	13.36
15	1.52	1.01	1.34	1.29
16	10.17	2.46	7.75	6.79
17	1.08	1.04	1.45	1.19
18	17.32	11.32	4.84	11.16
19	32.51	31.41	27.31	30.52
20	28.4	22.86	7.16	19.49

The end result shows that the model no. 3 or “ssd_mobilenet_v1_ppn_coco” affords the overall efficiency at 36.52. However, if we cognizance on handiest the front view, the version no. 1 or “ssd_mobilenet_v1_coco” yields the very best performance at 77.75.

The high efficiency model in Table 4.3 may seem like the best choice for motorcycle detection, but it may not be accurate enough. In other words, the high efficiency here can be due to medium to low accuracy and very short detection times. To illustrate these concerns, Figure 4.2 shows a graph between accuracy and time.

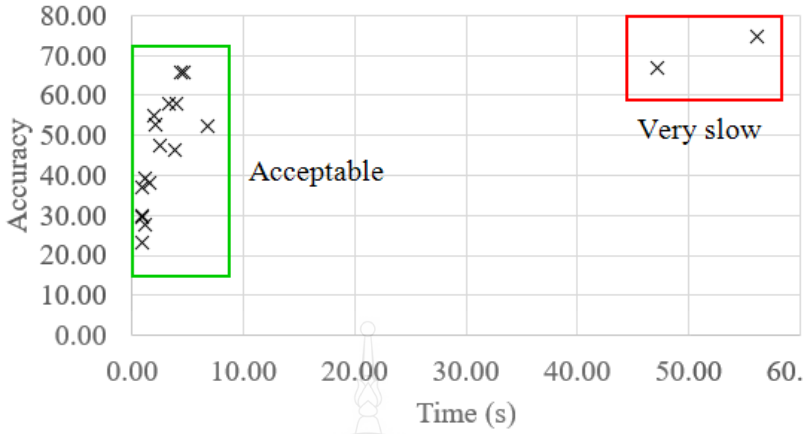


Figure 4.2 Average accuracy versus time

The plot of average accuracy over time in Figure 4.2 shows that the detection model can be divided into two main groups. Acceptable medium to high precision model and models that provided very slow detection time which are not preferable. Therefore, ignore the two models in the upper right and redraw the graph as shown in Figure 4.3.

From Figure 4.3, it's clear that the model no. 3 was the most efficient model whose slope is steepest. This end result is much like the one cited in Table 4.3. Nevertheless, this version renders pretty small accuracy (41.2%) which can be now no longer reliable.

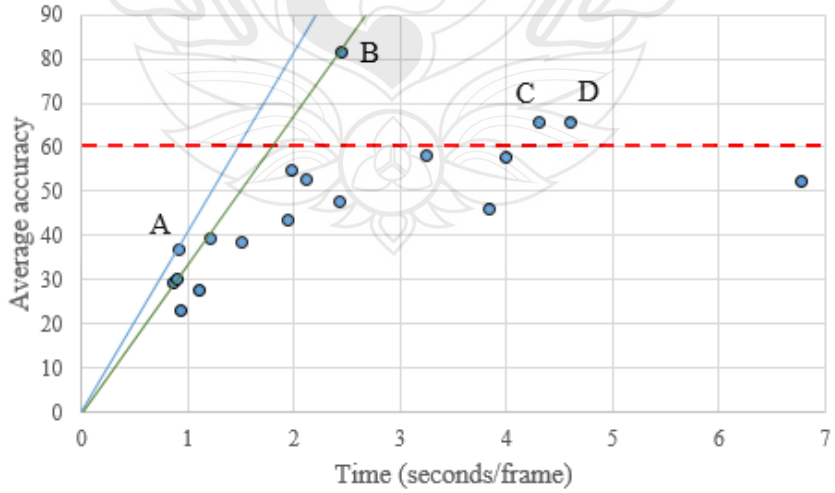


Figure 4.3 Accuracy versus time of acceptable models

Thus, we suggest to filter the models by using a threshold i.e. 60% and only three models (point B, C, and D) are qualified. Among them the model at point B or model no. 19 (YoloV3) whose accuracy in line with time (slope) is the highest will become the satisfactory choice. In addition, whilst we appearance returned at Table 4.1, this model additionally gives a very high accuracy at 79.94% and 93% for front-view detection accuracy

4.3.4 Detection Efficiency on Lower Resolution Images

Model no. 19 offers the highest efficiency with an accuracy of over 60%, but the detection time per frame (2.63 seconds) far exceeds the real-time detection (0.04 seconds). An obvious factor is the input image resolution (1920 x 1080), which is much larger and much larger than the COCO and YOLO training image datasets (600 x 600). As a result, the input image is reduced from 1/16 to 480x270 pixels and the accuracy and time of the top three models (No. 19, 12, 13) are retested. The results are shown in Table 4.4.

Table 4.4 Efficiency at two image resolutions

ModelNo.	**Accuracy (%)**		**Time (seconds/frame)**		**Efficiency**	
Resolution	1920	480	1920	480	1920	480
12	63.98	62.55	4.41	2.31	12.99	27.06
13	68.16	62.07	4.76	2.81	12.94	22.01
19	83.3	72.94	3.05	7.98	30.52	9.13

Surprisingly, Table 4.4 shows that reducing the image size from 1920x1080 to 480x270 pixels does not significantly change the detection accuracy of models No. 12 and 13. It also cuts the detection time by about half. This doubles the detection efficiency, but the model no.12 is still better than the no 13.

Another unexpected result is the model no. 19. In this case, detecting a small image will reduce the accuracy, but on the contrary, it will take much more time. This will significantly reduce efficiency.

Though lowering image resolution renders quicker detection substantially for a few models, the best detection time per frame is at 2.31 seconds, that is manifestly too lengthy for real-time processing (0.04 seconds). Further lowering image resolution to shorten detection time isn't always advocated because of a trouble for observers to observe accuracy.

4.3.5 CPU vs GPU

Comparing the performance of the TensorFlow model on the CPU and GPU. Detection time has a significant impact on the efficiency of the model, as efficiency is calculated as the ratio of accuracy to detection time per frame. Therefore, running a model using the GPU may produce different results than what is running on the CPU. The first experiment is to get the result from Table 4.3. TensorFlow1 running on the CPU. We then retested all TensorFlow 1 models on the GPU using the same dataset (1980x1080 pixel images) and observed the average efficiency of all three-view images. Table 4.5 shows the results of average detection efficiency based on CPU and GPU.

Table 4.5 Average efficiency of the performances on CPU and GPU

Model No.	Model	Efficiency	
		CPU	GPU
1	ssd_mobilenet_v1_coco	27.80	32.18
2	ssd_mobilenet_v1_0.75_depth_coco	33.02	39.24
3	ssd_mobilenet_v1_ppn_coco	36.52	47.41
4	ssd_mobilenet_v1_fpn_coco	23.84	53.19
5	ssd_resnet_50_fpn_coco	23.52	65.34
6	ssd_mobilenet_v2_coco	21.44	38.86
7	ssdlite_mobilenet_v2_coco	30.77	38.94

Table 4.5 (continued)

Model No.	Model	Efficiency	
		CPU	GPU
8	ssd_inception_v2_coco	29.14	49.59
9	faster_rcnn_inception_v2_coco	25.15	60.62
10	faster_rcnn_resnet50_coco	15.73	63.14
11	faster_rcnn_resnet50_lowproposal_coco	16.99	56.66
12	rfcn_resnet101_coco	12.99	69.76
13	faster_rcnn_resnet101_coco	12.94	63.92
14	faster_rcnn_resnet101_lowproposals_coco	13.36	66.25
15	faster_rcnn_inception_resnet_v2_atrous_coco	1.29	45.08
16	faster_rcnn_nas_lowproposal	6.79	43.47
17	mask_rcnn_inception_resnet_v2_atrous_coco	1.19	25.74

It is first determined that GPU handiest has a huge impact on a few SSD detectors including ssd_mobilenet_v1 and ssd_resnet_50 whose performance is double while strolling on GPU. Another factor to be observed is that every model of Faster-RCNN have about 4 instances better performance on GPU than performance on CPU. In addition, there are three types of model numbers. 15, # 16, and # 17 are significantly more efficient when run on the GPU. Therefore, GPUs are a great tool for increasing efficiency and achieving real-time detection. As a result, we continued to test all TensorFlow 2 models using the GPU.

4.4 Experimental Result from TensorFlow 2 Models

4.4.1 Detection Accuracy

An evaluation of 14 TensorFlow models became examined on three exclusive perspectives of images containing front, top, and back at 56, 105, 46 photo frames. Each object is labeled with the aid of using bounding box and the name of the item is supplied at the top of the bounding box as proven in Figure 4.4. The object detection model typically provides the class number of each object detected. For example, a car is assigned class number 1 and a motorcycle is assigned class number 2. This allows you to focus on a particular object and avoid other objects by paying attention to the object's class number and name.

Please note that we are only interested in motorcycle detection. Table 4.6 shows the accuracy of all views for each model.

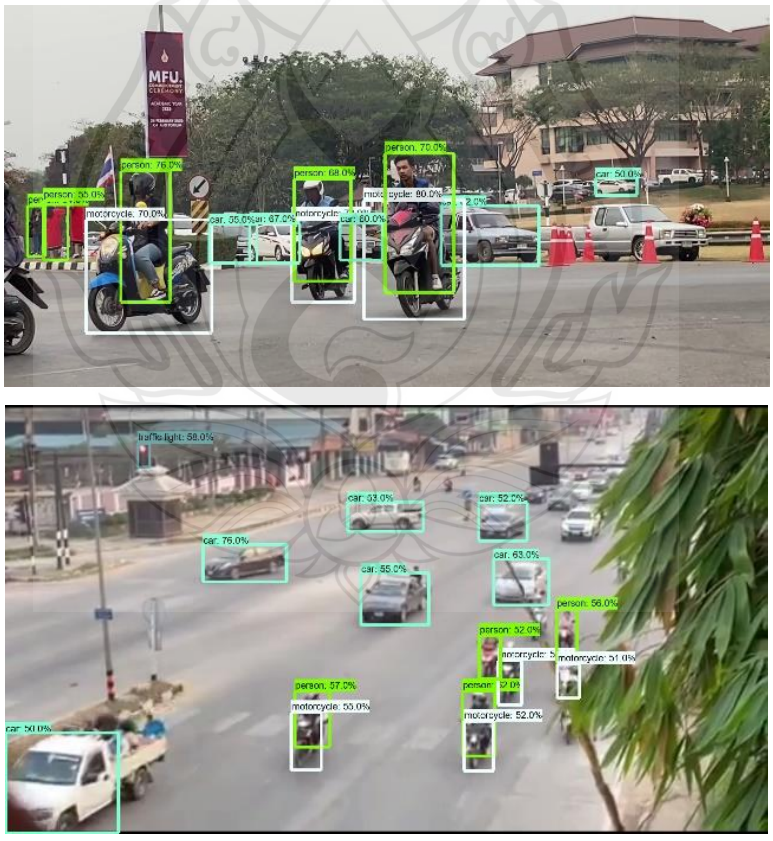


Figure 4.4 Example of motorcycle detection results on three different views

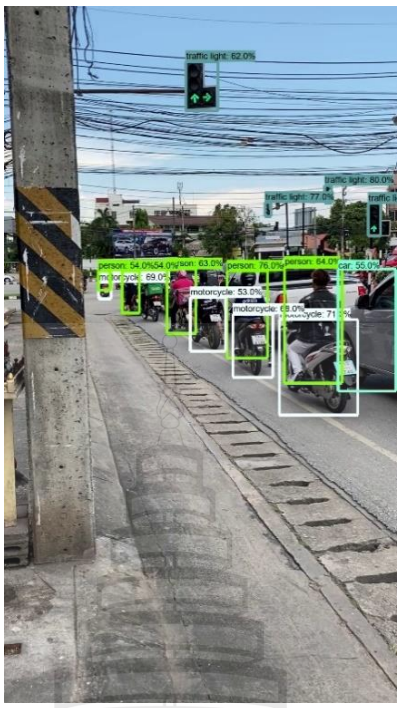


Figure 4.4 (continued)

Table 4.6 Detection accuracy of TensorFlow 2 models

Model No.	Model	Accuracy (%)			
		Front	Top	Back	Average
1	CenterNet HourGlass104 512x512 (CH104)	80.45	67.21	71.83	73.16
2	CenterNet Resnet50 V1 FPN 512x512 (C_R50_V1)	60.9	5.19	50	38.69
3	CenterNet Resnet101 V1 512x512 (C_R101_V1)	66.16	4.12	42.25	37.51
4	CenterNet Resnet50 V2 512x512 (C_R50_V2)	36.84	29.87	38.02	34.91
5	EfficientDet D0 512x512 (ED_D0)	49.25	7.79	21.83	26.29
6	SSD MobileNet v2 320x320 (SSD_MN_V2)	57.24	13.92	26.05	32.40

Table 4.6 (continued)

Model No.	Model	Accuracy (%)			
		Front	**Top**	**Back**	**Average**
7	SSD MobileNet V2 FPNLite 320x320 (SSD_MN_V2FPNL)	33.83	0	20.07	17.96
8	SSD ResNet50 V1 FPN 640x640 (SSD_R50_V1)	69.92	6.49	45.05	40.48
9	SSD ResNet101 V1 FPN 640x640 (SSD_R101_V1)	65.41	16.88	40.84	41.04
10	SSD ResNet152 V1 FPN 640x640 (SSD_R152_V1)	69.17	7.79	39.08	38.68
11	Faster R-CNN ResNet50 V1 640x640 (FRCNN_R50_V1)	88.97	38.46	41.52	56.31
12	Faster R-CNN ResNet101 V1 640x640 (FRCNN_R101_V1)	87.23	36.36	41.97	55.18
13	Faster R-CNN ResNet152 V1 640x640 (FRCNN_R152_V1)	87.59	38.46	45.07	57.04
14	Faster R-CNN Inception ResNet V2 640x640 (FRCNN_IR_V2)	80	6.49	73.59	53.36
	Average	**66.64**	**19.93**	**42.655**	

Looking at the accuracy of all models, most models show the best performance in the front view, averaging 66.64 percent. Of all the results from the three different views provided by FRCNN_R50_V1, the highest accuracy in the front view is 88.97 percent. Most models, on the other hand, have an average accuracy of 19.93%, which is poor performance in top view. In the rear view, there are only two models with more than 70%, CH104 and FRCNN_IR_V2. The performance of the rest of the models in the rear view is very low because it couldn't reach 50% accuracy.

We conclude that COCO models have been in general trained on front-view motorbike images. Moreover, we additionally determined the detection accuracy from TensorFlow 1 and the outcomes confirmed that maximum of TensorFlow 1 models preformed excellent at front-view images as well.

4.4.2 Detection Time

The detection time was calculated in seconds from the first frame to last frame of each video. Next, we divide detection time by the number of frame to obtain detection time per frame. A show of detection time for all models is represented in Table 4.7.

Note that the model numbers in Table 4.7 are shared with the model numbers in Table 4.6 and all models are running on the GPU

Table 4.7 Average detection time

Model No.	Average Detection Time (seconds/frame)			
	Front	**Top**	**Back**	**Average**
1	0.86	0.97	0.82	0.89
2	0.77	0.76	0.69	0.74
3	0.79	0.75	0.69	0.74
4	0.74	0.74	0.64	0.70
5	0.86	0.91	0.73	0.83
6	0.90	0.92	0.79	0.87
7	0.70	0.66	0.62	0.66
8	0.86	1.10	0.80	0.92
9	0.94	1.15	0.79	0.96
10	0.91	0.96	0.82	0.90
11	0.88	0.96	0.77	0.87
12	0.95	0.99	0.81	0.91
13	0.93	0.99	0.80	0.91
14	1.37	1.48	1.22	1.36

Table 4.7 shows that SSD_MN_V2FPNL has the shortest detection time of 0.66 seconds per frame and FRCNN_IR_V2 has the longest detection time of 1.36 seconds per frame compared to all models. In general, the higher the detection accuracy, the longer the detection time. In contrast, poor detection accuracy seems to require less

detection time. This means, SSD SSD_MN_V2FPNL offers pretty a low detection accuracy at 17.96 % of average accuracy and it is only model that is not able to detect on any motorcycle at the top view.

24 frames per second is a common rate for achieving a smooth frame rate, and none of the TensorFlow 2 models can run on a particular GPU. We also decided to resize the image because the size of the image can be an important factor for a long-time consuming. This topic will be discussed in a later section.

4.4.3 Detection Efficiency

The measurement of detection efficiency is the ratio of detection accuracy to detection time. Due to the high detection accuracy and long detection time, the efficiency value can be quite low. Efficient models, on the other hand, seem to provide high accuracy with short detection times. Table 4.8 shows the detection efficiency.

Table 4.8 Average detection efficiency

Model No.	Detection Efficiency			
	Front	**Top**	**Back**	**Average**
1	92.90	49.78	86.72	76.47
2	78.09	6.77	72.22	52.36
3	82.95	5.44	60.99	49.80
4	49.67	40.32	59.08	49.69
5	57.20	8.54	29.54	31.76
6	63.22	15.01	32.56	36.93
7	48.01	0	31.99	26.66
8	80.84	5.85	56.28	47.66
9	69.57	14.58	51.52	45.22
10	75.55	8.03	47.15	43.58
11	100.0	39.99	53.81	64.60
12	91.51	36.61	51.61	59.91
13	93.84	38.54	55.79	62.72
14	58.34	4.36	59.94	40.88

Table 4.8 shows that model # 1 or CH104 achieves the highest overall efficiency at 76.47. This model also offers the highest top view efficiency at 49.78 and the highest rear-view efficiency at 86.72. In the front view, model number 11 or FRCNN_R50_V1 is number one in terms of efficiency. By spotting at each detection accuracy in Table 4.6 and Detection efficiency, Front-view images are the great option amongst all 3 exclusive perspectives for detecting motorcycles. Hence, we determined to have a in addition test on front-view images. Record the three frontal videos shown in Figure 4.5 at a different camera angle, further distant, and then select a location where traffic congestion is occured to observe the performance of the selected model. Discussions on this topic will be presented in the next section.

4.4.4 Front-view Images Detection

From all 14 models, choose a model from each group of the most efficient object detectors when tested on front-view. There are four models selected: model no 1 (CH104), model no 5 (ED_D0), model no 8 (SSD_R50_V1), and model no 11 (FRCNN_R50_V1). Next, we tested the selected model with a 123-frame front image. The image size is 1980x1080 pixels in the horizontal view and 1080x1980 pixels in the vertical view. Notice that the front videos are named F2, F3, and F4, respectively, as shown in Figure 3.6. F1 refers to the first front video in Figure 3.5. A demonstration of an labeled image is shown in Figure 4.4.

Table 4.9 shows the accuracy, time, and efficiency of the selected model. Table 4.9 shows the model numbers. 5 and model # 8 provide fairly low accuracy for all three videos, but require the same long detection times as the other two models with high accuracy for F2 and F3 videos. For F4, we recorded the video in which there had been a large number of of vehicles riding around. As a result, model no.11 performs poorly on the F4 video. Moreover, model no.11 detects bikes as good as model no.1 for F4 video. However, a reason that the accuracy of model no.11 is only 55.2 % due to the fact the model detected an abundance of False Positives of motorcycle while motorcycle riding very near each other. This will lessen the accuracy of the model while it's far computed with the aid of using a confusion matrix.

Table 4.9 Accuracy, time, and efficiency of front videos

No.	Accuracy (%)			Time (seconds/frame)			Efficiency		
	F2	F3	F4	F2	F3	F4	F2	F3	F4
1	80.6	82.0	70	0.98	0.87	0.88	81.9	93.6	78.6
5	25	37.3	21.2	0.88	0.84	0.82	28.3	43.9	25.6
8	56.7	61.9	46.9	0.95	0.91	0.90	59.2	67.8	51.9
11	78.8	81.6	55.2	0.99	0.89	0.93	79.2	91.3	59.3

In summary, Model no.1 (CH104) appears to be the great option for motorbike detection amongst all 4 models. The model gives the very best performance for all 3 videos i.e., 81.9, 93.6, and 78.6 from F2, F3, and F4 videos. However, looking back at Tables 4.6 and 4.8, Model no 11 offers the highest accuracy and efficiency in the front image (F1) with 88.97% accuracy and 100% efficiency. With the exception of F4, model no 11 also gives very similar results to model no 1. Thus, model no.11 may be some other efficient model to detect front-view images. The model will offer the best result while the motorcycles are simply visible at the images and now no longer stacking.

4.4.5 Detection Efficiency of Front-View Images on Lower Resolution

For further experimentation, none of the models can achieve real-time detection (0.04 seconds per frame), so downscale all front video to 1 / 16 480x270 pixels. Therefore, lowering the resolution of the image may be a possible solution to this problem. In this experiment, we select the top two models that achieve the highest efficiency from all four models, as shown in Table 4.9. Table 4.10 shows the results at low resolution.

Table 4.10 Detection results on front-view images at the lower resolution

Model No.	**1**				**11**			
Name	**F1**	**F2**	**F3**	**F4**	**F1**	**F2**	**F3**	**F4**
Accuracy	79.69	76.48	76.11	62.5	88.72	74.21	74.81	52.5
Time (seconds/frame)	0.19	0.22	0.19	0.20	0.18	0.216	0.17	0.18
Efficiency	399.5	342.0	384.7	308.8	476.3	342.8	417.2	287.1

Calculation of average accuracy, time, and efficiency for selected models of all front videos in both high and low resolutions. This process was performed to compare the performance of the selected model at two resolutions. The results are shown in Table 4.11.

Table 4.11 Average efficiency at two resolutions on front-view images

Model No.	**Accuracy (%)**		**Time (seconds/frame)**		**Efficiency**	
Resolution	1920	480	1920	480	1920	480
1	78.26	73.69	0.89	0.20	87.39	358.78
11	76.14	72.56	0.92	0.19	82.93	380.86

It is remarkable from Table 4.11 that from resizing images from 1980x1080 to 480x270 pixels, the detection time is substantially decreased even as detection accuracy is barely reduced for each model. This way detection performance is about quadruples in 480x270 pixels resolution. Because of its low resolution, model no 11 slightly outperforms model no 1 in terms of average efficiency. Model no 11 is less accurate than model no 1 for F2, F3, and F4 video, but has a shorter detection time and is about 10% more accurate than model no 1 for F1 video.

4.4.6 Back-view Image Detection

Obtain the detection accuracy from Table 4.6 and the detection time data from Table 4.7. Next, plotting the relationship between accuracy and time in the rear view in Figure 4.5 shows that No. 1 and Model 14 are just two models with an accuracy of over 70%. Provides less than 50% accuracy for the remaining models.

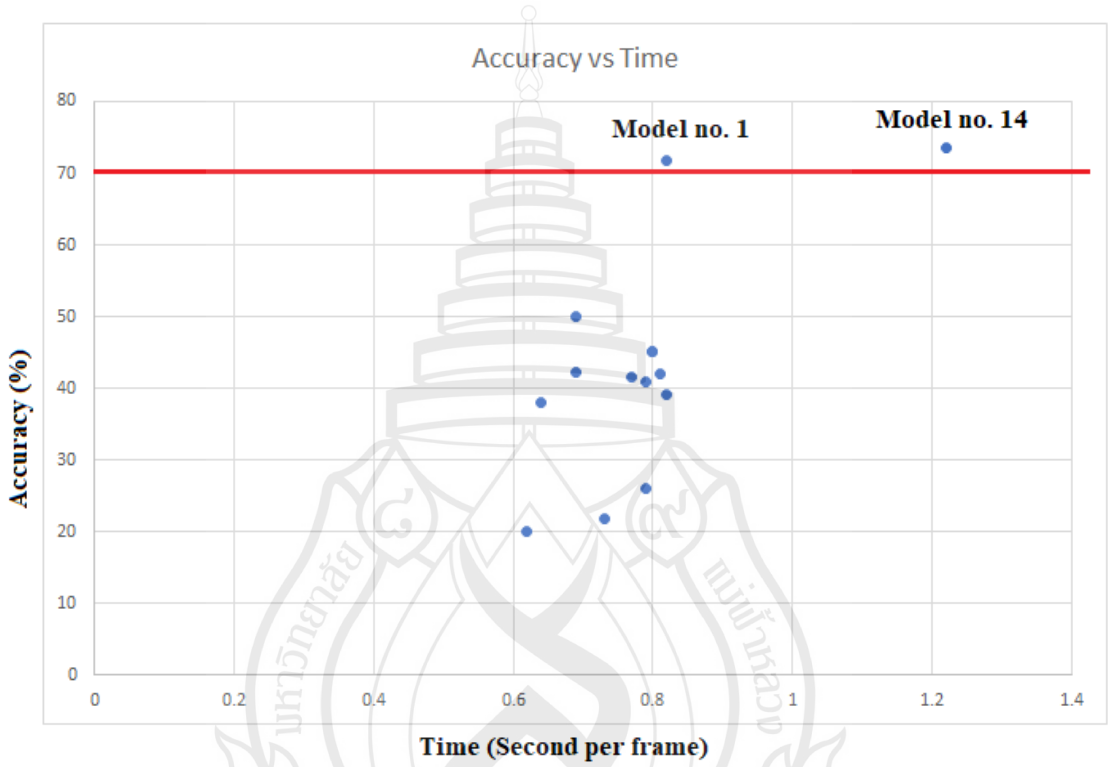


Figure 4.5 Accuracy versus time on back-view detection

As an end result, we propose to apply a threshold i.e., 70 chances accuracy to filter the model that offer accuracy beneath the threshold. There are simplest model that pass the condition which are CH104 and FRCNN_IR_V2 as proven in Figure 4.5. Then, we determine to copy a step that we've got accomplished on front-vie. Downscaling back-view image with the aid of using 1/16 from 1080x1980 pixels to 270x480 pixels and apply those images on model no.1 and model no.14 to observe a change of the end result. The end result is represented in Table 4.12.

Table 4.12 Average efficiency at two resolutions on back-view images

Model No.	Accuracy (%)		Time (seconds/frame)		Efficiency	
Resolution	1920	480	1920	480	1920	480
1	71.83	69.36	0.82	0.22	86.72	303.81
14	73.59	72.13	1.22	0.59	59.97	120.60

The outcome from Table 4.12 appears to be same direction from the outcome from Table 4.11. Downscaling decision has a massive effect on detection time that is decreased notably even as detection accuracy infrequently changes. Model no 1 is superior to model no 14 with an efficiency of 303.81. Therefore, the CH104 is the best rear view motorcycle detection model.

4.4.7 Top-view Image Detection

According to Table 4.6, most models provided insufficient accuracy in top view detection. The main reason for the low accuracy may be that the video (T1) is evaluated in an occluded scenario. The lowest accuracy provided by SSD_MN_V2FPNL is 0 and the highest accuracy provided by CH104 is 67.21. However, using CCTVs or cameras to record vehicles in real-world traffic conditions is often done in high places such as Traffic light pole. Thus, I decided to record three more videos of the Top view motorcycle to observe the performance of all the models. Table 4.13 shows the detection accuracy of all three videos.

Table 4.13 Detection accuracy of top-view videos

Model No.	Detection Accuracy		
	T2	**T3**	**T4**
1	74.34	76.4	59.85
2	22.3	23.03	51.4
3	20.81	13.48	35.91
4	42	42.1	21.12
5	7.8	2.24	3.5
6	11.89	24.71	10.56
7	0.37	0	0
8	10.03	11.79	8.4
9	18.21	21.91	10.56
10	11.89	12.35	11.97
11	41.63	45.5	38.73
12	48.69	38.2	43.21
13	37.91	46.62	38.02
14	1.48	7.3	0.71

From Table 4.13, The end result clearly reveals that other than model no.1 (CH104), none of different model should yield above 55 percentages of detection accuracy. Moreover, we once more pick the top models from every object detector to be examined in lower resolution. By noticing average detection performance in Table 4.14, T3 video offers the best performance amongst all top-videos. Therefore, we choose T3 video as a pattern video for use in low resolution experiment.

Table 4.14 Detection efficiency of top-view videos (original size)

Model No.	Detection efficiency		
	T2	**T3**	**T4**
1	76.47	82.94	73.97
2	26.90	25.99	69.98
3	25.20	16.58	46.62
4	51.94	53.59	29.39
5	9.12	2.60	4.43
6	12.77	26.12	11.83
7	0.50	0	0
8	10.12	11.50	8.92
9	17.83	21.35	10.16
10	10.64	10.17	10.39
11	42.18	49.60	38.01
12	48.41	41.21	39.34
13	38.53	48.74	39.34
14	1.03	5.36	0.45
Average	**26.55**	**28.27**	**28.09**

We decrease the original resolution (1920x1080) two and four time which can be 960x540 and 480x270 via way of means of retaining 16x9 aspect ratio. The detection accuracy, time, and performance of all three resolutions are offered in Table 4.15.

Table 4.15 Average efficiency at three resolutions on the top-view image (T3)

Model No.	**Accuracy (%)**			**Time (seconds/frame)**			**Efficiency**		
Resolution	1920x 1080	960x 540	480x 270	1920x 1080	960x 540	480x 270	1920x 1080	960x 540	480x 270
1	76.4	74.41	71.34	0.92	0.34	0.19	82.94	216.33	367.44
5	2.24	2.24	1.2	0.86	0.34	0.18	2.60	6.48	6.48
8	24.71	23.03	19.5	0.94	0.30	0.23	26.12	75.25	83.68
13	45.5	44.97	38.56	0.91	0.37	0.20	49.60	120.19	188.18

From Table 11, we will summarize that model no.1 (CH104) still gives the best detection efficiency. The end result of this test is similar to the low-resolution front-view image. By decreasing the image resolution, detection accuracy barely decreases and detection time substantially will increase for all of the models. Model no 1 improves detection efficiency by about 160% and improves detection efficiency from 1920x1080 pixels to 480x270 pixels by about 343% by reducing the image resolution from 1920x1080 pixels to 960x540 pixels. Therefore, lowering the resolution may be an essential way to increase the efficiency of the model.

By lowering image resolution significantly shorten detection time, the exceptional detection time per frame is 0.18 which nevertheless cannot fulfill the real-time detection (0.04 according per frame). For reaching real-time detection, suggestions are reducing number of classes in object detection, search for higher GPU, or regulate the models` architecture, or fine-tune the models` parameters.

4.4.8 Motorcycle Detection with Low-Light Condition

Select the best model for Top-view detection (CH104) to run your experiments in the dark. As shown in Figure 4.6, finding a motorcycle is difficult. Low light conditions have a significant effect on detection accuracy. In summary, all TensorFlow models have a low ability to detect motorcycles in the dark.

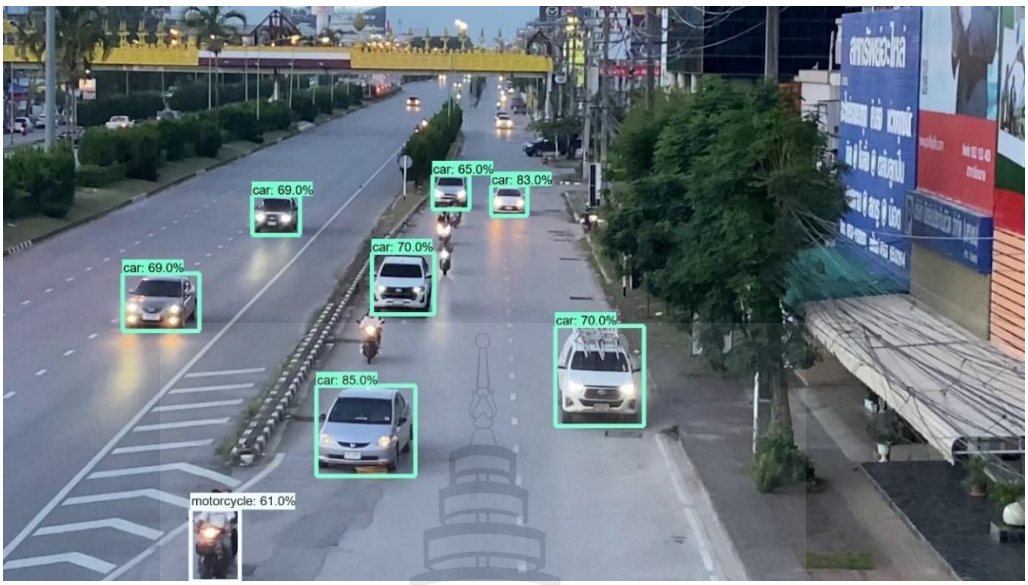


Figure 4.6 Example of motorcycle detection with low-light condition

4.4.9 Detection Accuracy vs COCO mAP

TensorFlow model zoo provides a standard measurement for detection accuracy called COCO mAP [34]. Nevertheless, a question is “Is this measurement also suitable for measuring accuracy of motorcycle detection? “. To answer the question based on particular observation that we will compare the average accuracy from Table 4.6 with COCO mAP for all models as shown in Figure 4.7.

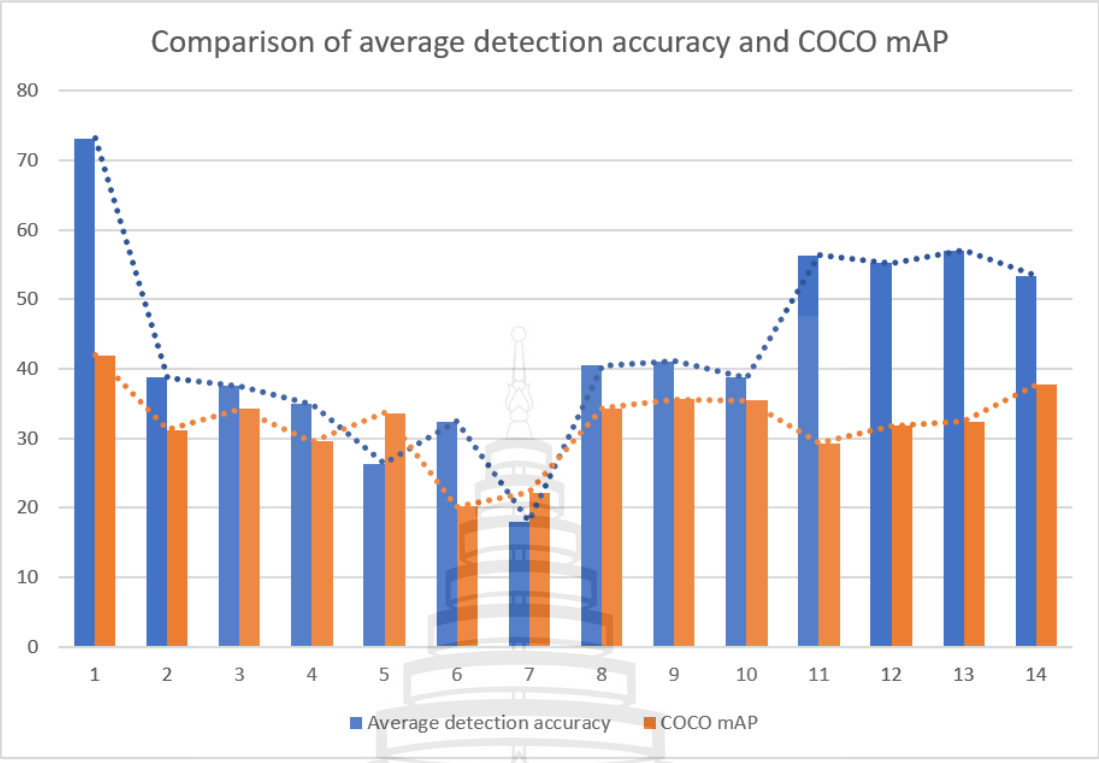


Figure 4.7 Comparison of average detection accuracy and COCO mAP

Figure 4.7 shows that Ch104 (Model no 1) is the best model with the highest scores in both detection accuracy and COCO mAP. Some models may be better than others calculated by COCO mAP, but with a focus on motorcycle detection, they are less accurate. For example, model no 14 ranks second in COCO mAP, but in experiment it ranks fifth, with lower detection accuracy than models no 11, 12, and 13. There are several models with different ratings when comparing motorcycle detection and COCO mAP. Another thing to be noticed is that the model with the lowest COCO mAP is model no 6. However, in our experiments, model no 7 is the model that gives the lowest detection accuracy. Therefore, COCO mAP is not in accordance with our measurements.

4.4.10 TensorFlow 1 vs TensorFlow 2

Our final experiment is to compare models from TensorFlow 1 and models from TensorFlow 2. Now, we can conclude that both TensorFlow models perform best at front-view motorcycles and at 480x270 pixels. From Table 4.9, we have the top models from each object detector from TensorFlow 2. Then, we choose top TensorFlow 1 models from each group of object detectors. The selected models that provide best efficiency are ssd_resnet_50_fpn_coco, faster_rcnn_resnet101_lowproposals_coco, rfcn_resnet101_coco, and mask_rcnn_inception _resnet_v2_atrous _coco. We test selected models from TensorFlow 1 on all four front-view videos at 480x270 pixels and compute average accuracy and time. Plotting the results in Figure 4.8.

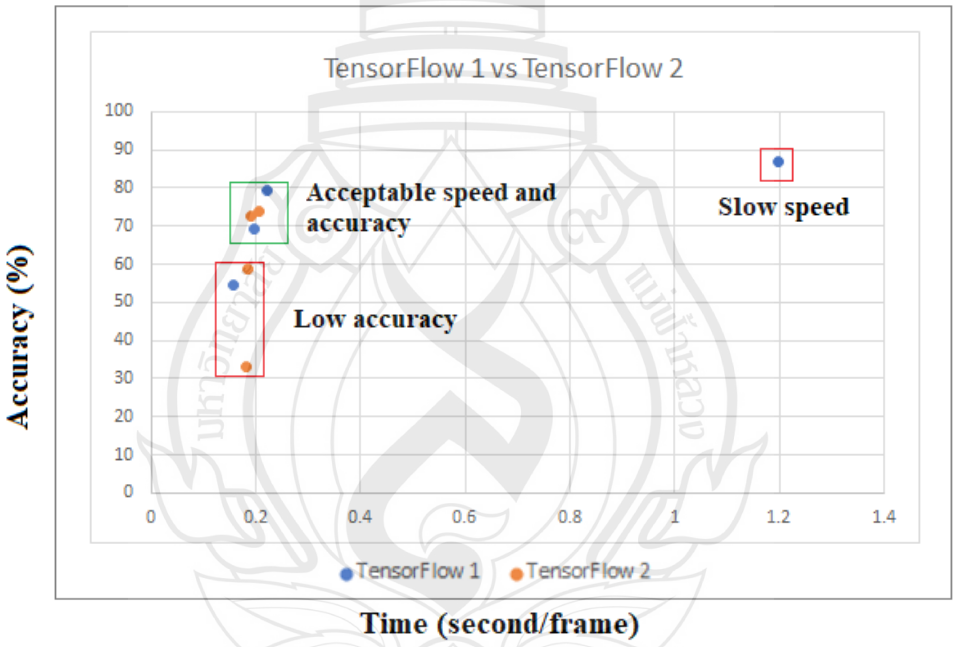


Figure 4.8 TensorFlow 1 and TensorFlow 2 (Accuracy versus Time)

From Figure 4.8, it is observed that two models from TensorFlow 1 are not satisfied with the results which one of them provides unreliable accuracy and another one takes quite a long to compute the detection. Thus, there are four models left are CenterNet HourGlass104 512x512 and Faster R-CNN ResNet50 V1 640x640 from TensorFlow 2. Another two models are faster_rcnn_resnet101_lowproposals_coco, and

rfcn_resnet101_coco from TensorFlow 1. rfcn_resnet101_coco seems to provide the highest accuracy among the four models and it is also a champion model from TensorFlow 1. To be clear, we calculate the efficiency of all four models as shown in Table 4.16.

Table 4.16 Detection efficiency of TensorFlow 1 and TensorFlow 2

Model	Efficiency				
	F1	**F2**	**F3**	**F4**	**Average**
CenterNet HourGlass104 512x512	399.5	342.0	384.7	308.8	358.78
Faster R-CNN ResNet50 V1 640x640	476.3	342.8	417.2	287.1	80.86
rfcn_resnet101_coco	396.01	323.30	385.95	358.98	366.06
faster_rcnn_resnet101_ lowproposals_coco	425.51	312.53	342.52	361.04	360.40

By observing Table 4.16, all four model provide a very close result to each other and the model that yields the highest efficiency is Faster R-CNN ResNet50 V1 640x640 from TensorFlow 2 at 380.86.

For back-view images, we test top models from TensorFlow 1 with as same dataset as we tested on TensorFlow 2. However, none of TensorFlow 1 models can surpass 70 percentages of accuracy. The outcome is shown in Table 4.17.

Table 4.17 Detection accuracy of TensorFlow 1 models on back-view images

Model name	Accuracy
ssd_resnet_50_fpn_coco	40.14
faster_rcnn_resnet101_lowproposals_coco	37.67
rfcn_resnet101_coco	55.98
mask_rcnn_inception_resnet_v2_atrous _coco	58.45

As a result, CenterNet HourGlass104 512x512 is still the best option for detecting back-view motorcycles.

For top-view images, none of TensorFlow 1 models can yield above 60% of detection accuracy. Only YOLOv3 can meet the condition. However, YOLOv3 has long time-consuming even at lower-resolution. In summary, CenterNet HourGlass104 512x512 is the best option for detecting all view of the motorcycles.

4.5 Detecting Abnormal Patterns of the Motorcycle

Top-view is the view that might be carried out in an actual scene due to its camera angle perspective as comparable as CCTV. Therefore, we simulate the scene where motorcycle riding abnormally on top-view. From the previous experiment, we have the chosen model to detect motorcycle and obtain X and Y coordinate from bounding boxes. The model is CenterNet HourGlass104 512x512 (CH104). This model offers the highest motorcycle detection efficiency in terms of accuracy per detection speed for top-view. Table 4.18 shows an example of the detection result.

Table 4.18 Example of detection result from simulated videos

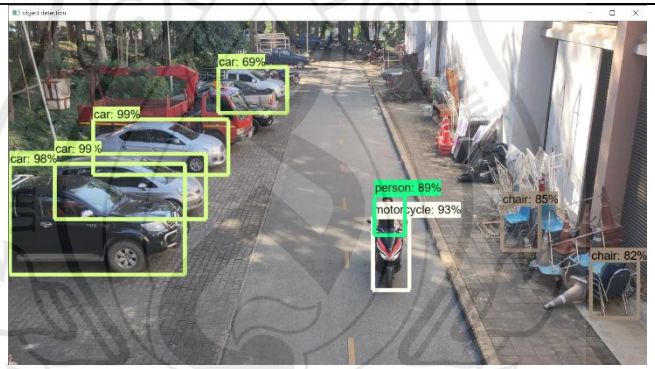
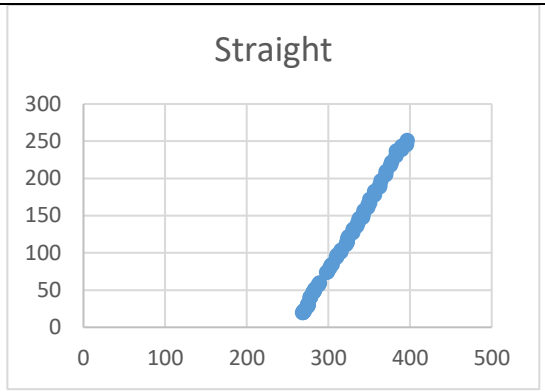
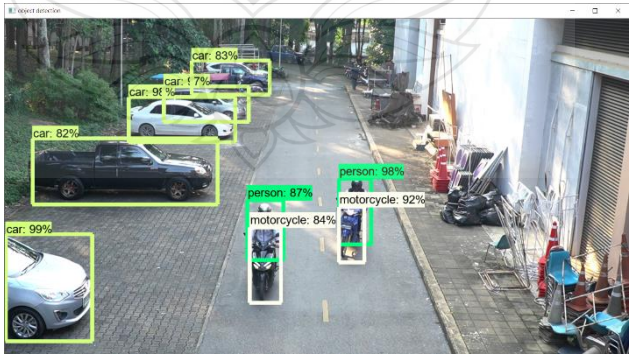
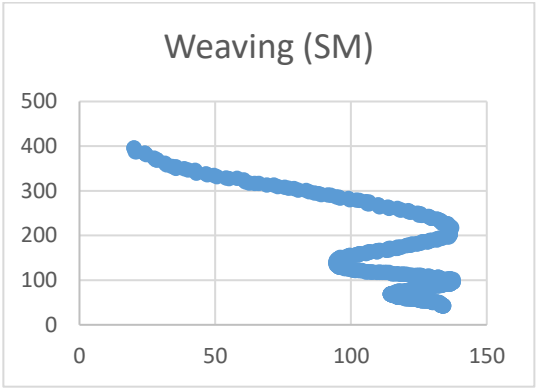
Name	Riding Patterns	
Straight		
Weaving (Small curve)		

Table 4.18 (continued)

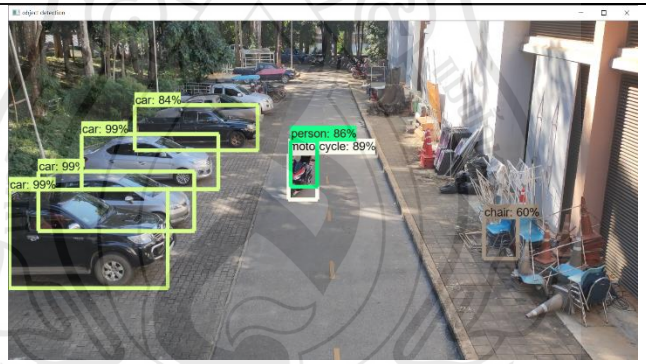
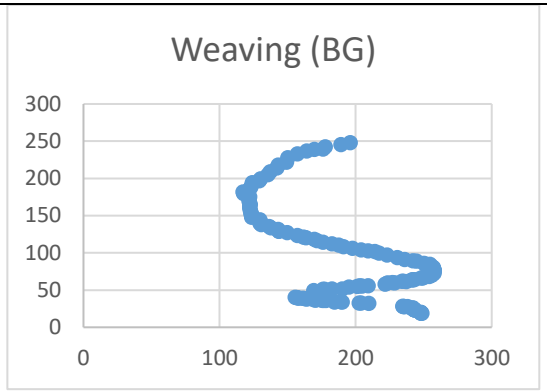
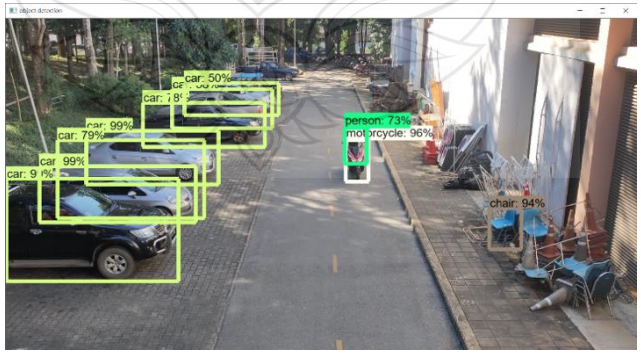
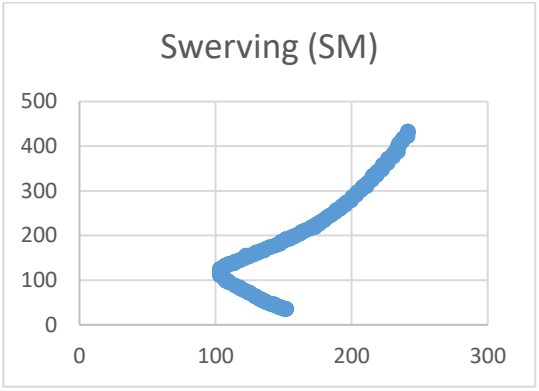
Name	Riding Patterns	
Weaving (Big curve)		
Swerving (Small curve)		

Table 4.18 (continued)

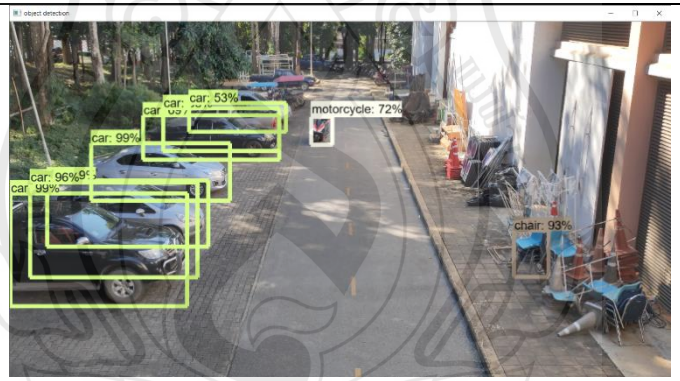
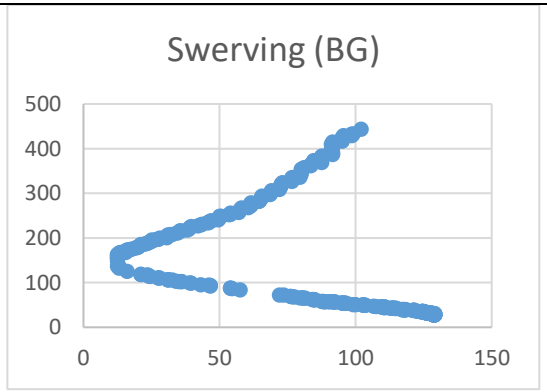
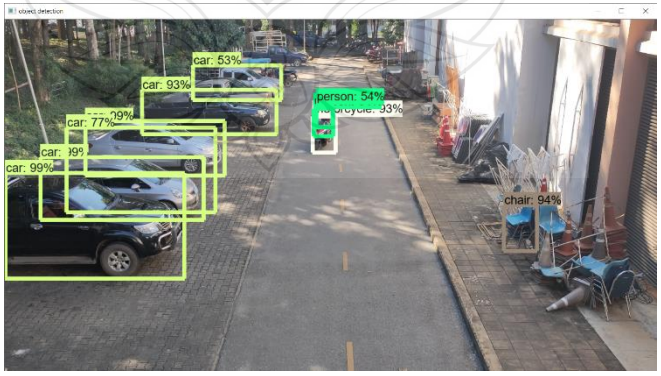
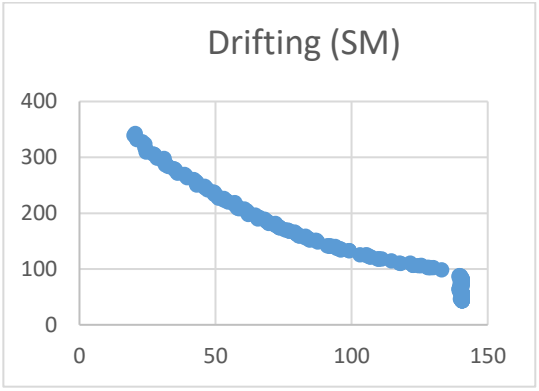
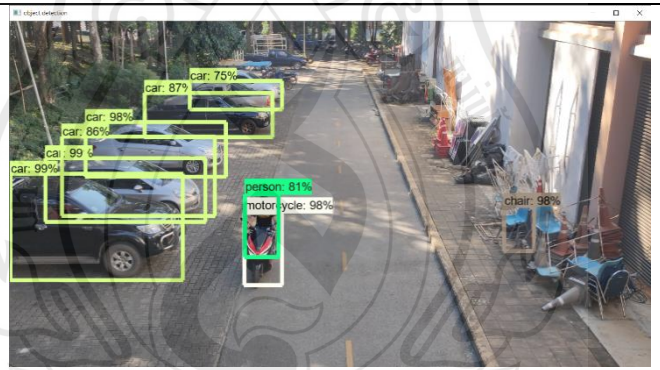
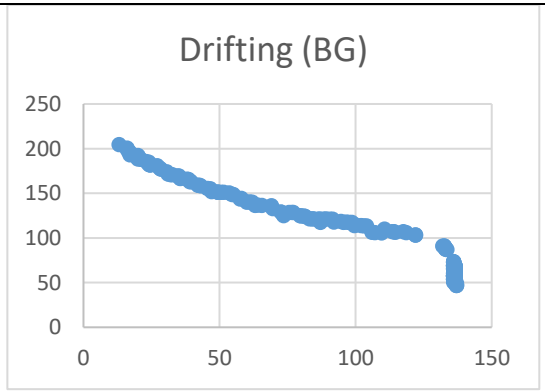
Name	Riding Patterns	
Swerving (Big curve)		
Drifting (Small curve)		

Table 4.18 (continued)

Name	Riding Patterns	
Drifting (Big curve)		

Eighteen videos of abnormal driving and three videos of normal driving were tested on an ordinary least squares model. The results of the three R-square are obtained in normal riding which are 0.9596, 0.9938 and 0.4015 as shown in Table 4.19.

Table 4.19 R-Squared of ordinary least squared and RANSAC score

Patterns	R-Squared	RANSAC score
Straight	0.9936	0.9936
Straight	0.9956	0.9956
Straight	0.4015	0.9754
Weaving (Small curve)	0.3258	0.0878
Weaving (Small curve)	0.8881	0.8572
Weaving (Small curve)	0.6790	0.8537
Weaving (Big curve)	0.0121	-0.7239
Weaving (Big curve)	0.0085	-0.7524
Weaving (Big curve)	0.1158	-0.8548
Swerving (Small curve)	0.9361	0.9352
Swerving (Small curve)	0.9275	0.9256
Swerving (Small curve)	0.7047	0.6050
Swerving (Big curve)	0.5850	0.5101
Swerving (Big curve)	0.6793	0.6542
Swerving (Big curve)	0.8225	0.8004
Drifting (Small curve)	0.7889	0.7087
Drifting (Small curve)	0.7438	0.7124
Drifting (Small curve)	0.7505	0.7689
Drifting (Big curve)	0.9474	0.9124
Drifting (Big curve)	0.9231	0.9268
Drifting (Big curve)	0.9425	0.9336

On the other hand, big curved have lower R-square for all three videos. In addition, the fact that the R-square results are less than 0.95 can be classified as abnormal riding. The results of OLS and RANSAC are very similar in some riding cases such as straight, swerving (small curve), swerving (large curve), drifting (small

curve), and drifting (large curve). According to the table 4.17. The results of the R-square for normal riding are 0.9956, 0.9938, and 0.4015. Keep in mind that the main problem is unwanted motorcycle noise for normal riding, the R-squared is very low at 0.4015. While recording the simulated video, some motorcycles were parked in the scene and the detection model was able to find them. As a result, the OLS result was low. To solve this problem, RANSAC was applied to remove the noise and the data was perfectly fitted to the model as it scores 0.9754 as shown in Table 4.19.

4.6 Classify Types of Abnormal Riding

By using OLS, we distinguish normal riding from abnormal riding. However, OLS are not suitable for classifying each type of abnormal riding. Therefore, we have a further experiment on polynomial regression. Because OLS will not be an effective tool for fitting a curve line such as lines from weaving or swerving pattern as shown in Figure 4.9. Polynomial regression could be suitable solution for this problem. Thus, we implement a polynomial regression model from python [39].

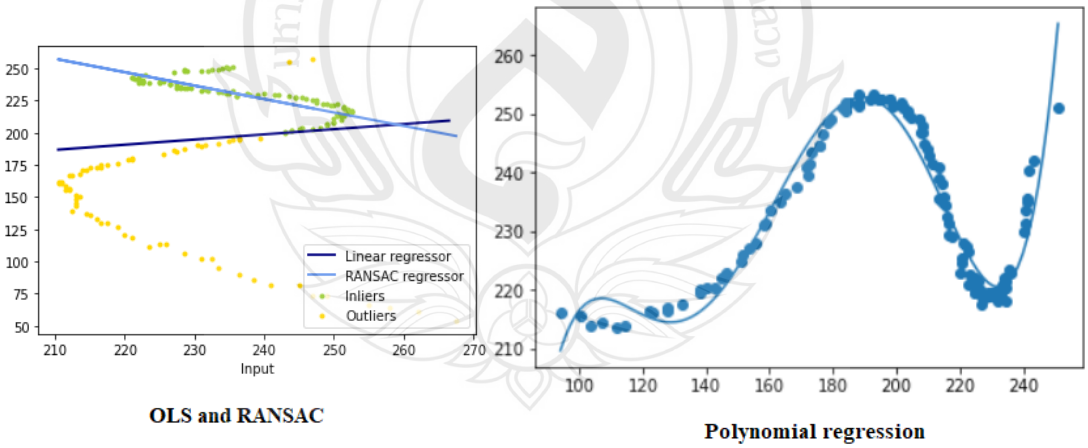


Figure 4.9 OLS and RANSAC versus Polynomial regression

We have run all the patterns which are 70 videos in total (10 videos for each pattern) using OLS. Then, we pick three sample results for each pattern as shown Table 4.20.

Table 4.20 R-squared score from OLS model

Riding patterns	R-squared
Straight	0.995551822
Straight	0.998739129
Straight	0.996711652
Weaving (BG)	0.37909021
Weaving (BG)	0.123874077
Weaving (BG)	0.049482301
Weaving (SM)	0.398814596
Weaving (SM)	0.067392272
Weaving (SM)	0.300402987
Swerving (BG)	0.378351275
Swerving (BG)	0.190217431
Swerving (BG)	0.126398282
Swerving (SM)	0.262534267
Swerving (SM)	0.646246984
Swerving (SM)	0.082460483
Drifting (BG)	0.945523609
Drifting (BG)	0.940613668
Drifting (BG)	0.938828048
Drifting (SM)	0.969063958
Drifting (SM)	0.944036037
Drifting (SM)	0.961707252

By observing Table 4.20, we can clearly notice that straight and drifting provide very high R-squared which is not surprised for straight patterns. In addition, drifting has a high R-squared as well as straight because the pattern of drifting is when the motorcyclist changing lane. There is no big curve occur during riding as shown in Figure 4.10. As a result of this, we can distinguish straight and drifting from weaving and swerving patterns. For now, we have two big groups which are first group consisting of straight and drifting patterns, and second group consisting of weaving and swerving patterns. Before sparating sub-group (small and big curve), we catagorize each pattern first.

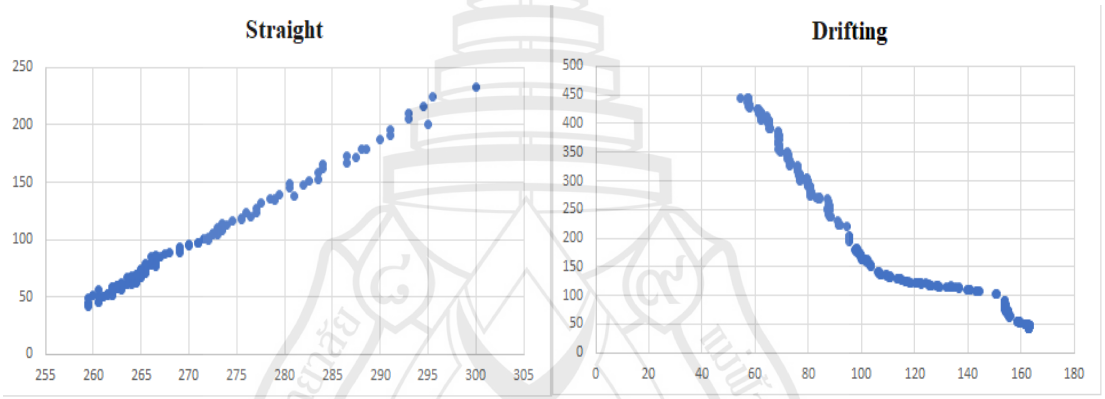


Figure 4.10 Samples of straight and drifting

For the first group, we can easily separate these two patterns from each other by again oberserve R-Squared results from OLS. Table 4.21 shows all the results for straight and drifting.

Table 4.21 R-squared score from OLS for straight pattern and drifting pattern

Riding patterns	R-squared
Straight	0.995551822
Straight	0.998739129
Straight	0.996711652
Straight	0.998636025
Straight	0.986800694
Straight	0.996115001
Straight	0.997636306
Straight	0.983578142
Straight	0.994625498
Straight	0.994330954
Drifting (BG)	0.945523609
Drifting (BG)	0.940613668
Drifting (BG)	0.938828048
Drifting (BG)	0.948541612
Drifting (BG)	0.936146537
Drifting (BG)	0.949102715
Drifting (BG)	0.933399985
Drifting (BG)	0.952635899
Drifting (BG)	0.912928462
Drifting (BG)	0.940355399
Drifting (SM)	0.969063958
Drifting (SM)	0.944036037
Drifting (SM)	0.961707252
Drifting (SM)	0.95796721
Drifting (SM)	0.960666576
Drifting (SM)	0.955569794
Drifting (SM)	0.950758277
Drifting (SM)	0.902526271
Drifting (SM)	0.906867909
Drifting (SM)	0.915513737

We notice that most of straight patterns provide R-squared above 0.99 with the highest of 0.998 and the lowest of 0.983. For drifting, most of them provide more than 0.90 R-squared score but not above 0.97. The highest score is 0.969 and the lowest score is 0.902 (including both big and small curve). Therefore, if we propose the threshold of R-squared score i.e 0.97, we could split Striagh patterns from drifting.

For second group, Both OLS and RANSAC could not fit a curve line which is a result from weaving pattern. Nonetheless, polynomial regression could perfectly fit the curve line. The polynomial regression will require three main input to generate the results which are X and Y coordinate and an exponent. With high number of exponents, the model will be better at fitting the line as shown in Figure 4.11.

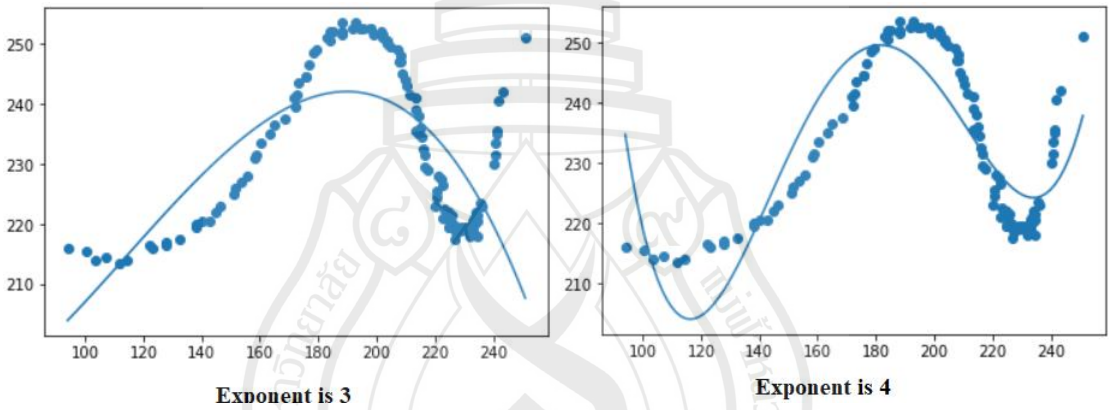


Figure 4.11 Example of different value of exponents

Normally, R-square defines how good the model can fit the line. With the high value of exponent, R-square will always be high even the line is very curved. Thus, R-square value from polynomial regression model will not be a good tool for classifying types of abnormal riding, if we apply only one number of the exponent. However, if we compare the R-square value of different exponent, we will observe the difference between riding pattern. Examples of R-square for each different number of exponents is shown in Table 4.22.

Table 4.22 R-Squared of different exponents

Patterns/Exponent	6	5	4	3
Straight	0.9963	0.9962	0.996	0.9957
Swerving (BG)	0.9834	0.9816	0.9741	0.8823
Swerving (SM)	0.9854	0.9688	0.9593	0.9531
Drifting (BG)	0.9976	0.995	0.9912	0.9867
Drifting (SM)	0.987	0.9755	0.9621	0.9597
Weaving (BG)	0.9348	0.9319	0.746	0.4657
Weaving (SM)	0.9383	0.9382	0.9231	0.9188

By observing Table 4.22, we clearly notice that straight pattern will always provide the high score of R-square even low value of exponents. weaving pattern (Big curve) have high R-square score when exponent is 6 or 5. In addition, when exponent is decreased to 4 or 3, weaving pattern (Big curve)'s R-square score is significantly decreased.

Our main objective is to separate swerving from weaving pattern. Thus, we propose to compare R-squared with different value of exponent. We obtain results from the polynomial regression model and set exponent to 6 and 3 as shown in Table 4.23.

Table 4.23 R-squared score from Polynomial regression model for swerving pattern and weaving pattern

Riding pattern	Exponent (6)	Exponent (3)	Riding pattern	Exponent (6)	Exponent (3)
Swerving (BG)	0.999189779	0.993821	Weaving (BG)	0.908848199	0.609166
Swerving (BG)	0.998571206	0.983242	Weaving (BG)	0.82568726	0.738115
Swerving (BG)	0.999260752	0.987265	Weaving (BG)	0.870008427	0.74765
Swerving (BG)	0.997496827	0.955439	Weaving (BG)	0.741774863	0.669071
Swerving (BG)	0.998828209	0.982314	Weaving (BG)	0.854887981	0.788064
Swerving (BG)	0.998758738	0.980795	Weaving (BG)	0.882085518	0.801468
Swerving (BG)	0.998293881	0.993224	Weaving (BG)	0.931997961	0.867103
Swerving (BG)	0.960994632	0.944225	Weaving (BG)	0.896017455	0.639034
Swerving (BG)	0.98341902	0.97415	Weaving (BG)	0.35991039	0.334433
Swerving (BG)	0.985912555	0.957013	Weaving (BG)	0.921875455	0.785876
Swerving (SM)	0.992623678	0.916914	Weaving (SM)	0.855518817	0.771969
Swerving (SM)	0.996218723	0.990015	Weaving (SM)	0.86615789	0.766197
Swerving (SM)	0.996182753	0.981993	Weaving (SM)	0.86210164	0.641134
Swerving (SM)	0.994389588	0.970362	Weaving (SM)	0.827340689	0.701926

Table 4.23 (continued)

Riding pattern	Exponent (6)	Exponent (3)	Riding pattern	Exponent (6)	Exponent (3)
Swerving (SM)	0.993419198	0.981273	Weaving (SM)	0.884666061	0.828679
Swerving (SM)	0.996472471	0.975882	Weaving (SM)	0.826836215	0.714603
Swerving (SM)	0.997817622	0.974926	Weaving (SM)	0.763017983	0.637585
Swerving (SM)	0.997688925	0.976301	Weaving (SM)	0.879640843	0.785865
Swerving (SM)	0.994913203	0.985653	Weaving (SM)	0.938397121	0.923174
Swerving (SM)	0.995039664	0.980468	Weaving (SM)	0.952242189	0.867045

By comparing results from different exponents, most of weaving patterns especially big curve have greatly increased in R-squared score from exponent (3) to exponent (6). On the other hand, swerving patterns rarely increased in R-squared by increasing exponent value. We can separate weaving from swerving by observing a change in R-squared when exponent is increased.

In summary, OLS seems to be suitable tool for fitting straight and drifting pattern. Polynomial regression model will fit swerving pattern with setting exponent to 4. Additionally, weaving requires quite high value of exponent to be fit.

R-squared score seem to be a good choice for classifying types of riding patterns. However, R-squared score is not a satisfied tool for classifying sub-category of each abnormal patterns i.e., weaving (big curve), swerving (small curve), drifting (bug curve) etc.

4.7 Classification Tree

Therefore, we have further experiment on classifying abnormal pattern. Obtaining the coefficients from polynomial regression model to be an input for classification tree algorithm. If the data mining task involves classification or prediction of results and a goal is to generate rules that can be easily explained and translated into SQL or a natural query language, the classification tree method is recommended. Firstly, sixth degree of polynomial model would be suitable because at sixth degree, the model can fit all the patterns. The example of the data (sixth degree) is presented in Table 4.24.

Table 4.24 Results from Polynomial regression (Sixth degree)

Patterns	x^6	x^5	x^4	x^3	x^2	x^1	**Constant**
Straight	2.656e-11 x	-2.052e-08 x	6.139e-06 x	0.0008913 x	0.06394 x	-1.554 x	274.1
Weaving (BG)	3.165e-10 x	- 2.24e-07 x	5.785e-05 x	-0.006598 x	0.3271 x	-6.922 x	315.4
Weaving (SM)	-3.947e-12 x	6.114e-09 x	-3.57e-06 x	0.0009838 x	- 0.1317 x	7.847 x	-35.76
Swerving (BG)	-8.018e-13 x	1.438e-09 x	-9.882e-07 x	0.0003189 x	-0.04544 x	1.614 x	130.5
Swerving (SM)	-2.457e-12 x	3.389e-09 x	-1.754e-06 x	0.0004092 x	-0.03801 x	0.6133 x	161.5
Drifting (BG)	3e-10 x	-2.435e-07 x	7.889e-05 x	-0.01288x	1.097 x	- 46.22 x	892.7
Drifting (SM)	6.56e-12 x	-4.918e-09 x	1.116e-06 x	-1.209e-05 x	-0.0242 x	1.92 x	87.21

Then, we apply these results from Table 4.24 to generate classification tree. The classification tree is provided in Figure 4.12.

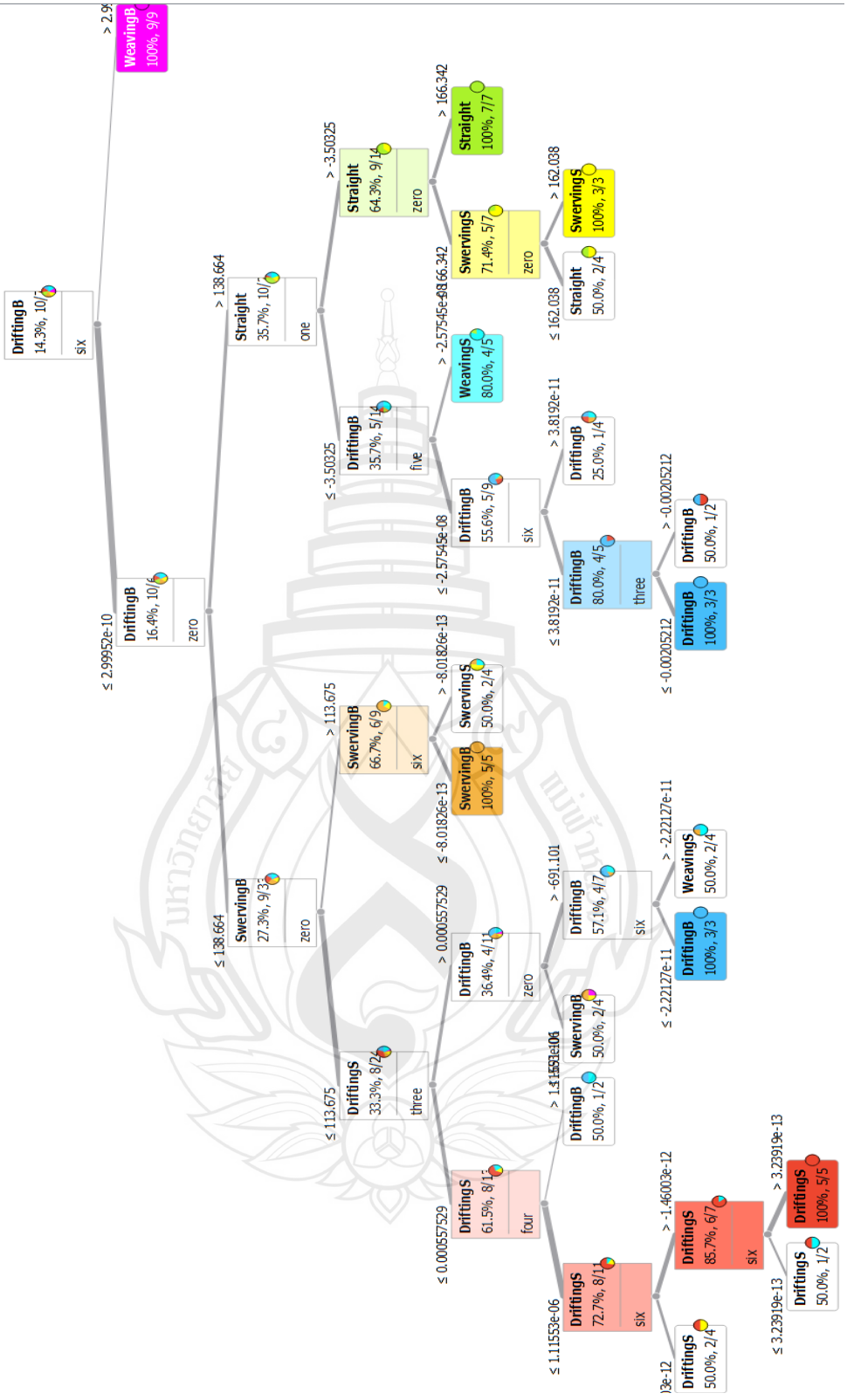


Figure 4.12 Classification tree (Sixth degree)

By using parameter from sixth degree, the algorithm, can easily classify weaving (Big curve). Straight pattern also has high classification approximately 70% with a use of parameter from constant value. Other patterns seem to have quite low accuracy. From Figure 4.12, we can conclude that some patterns need certain degree to be classified. For example, weaving (Big curve) require sixth degree and straight pattern require constant value. However, overall accuracy for all pattern is low. In order to test that we apply 70/30 training and testing method, and 10-fold cross validation.

4.8 70/30 Training and Testing Method, and 10-Fold Cross Validation

Using all data for classification model could result in overfitting. Therefore, we applied 70% training 30% testing split method in machine learning, 10-fold cross validation, and classification tree. Overall scheme is shown in Figure 4.13.

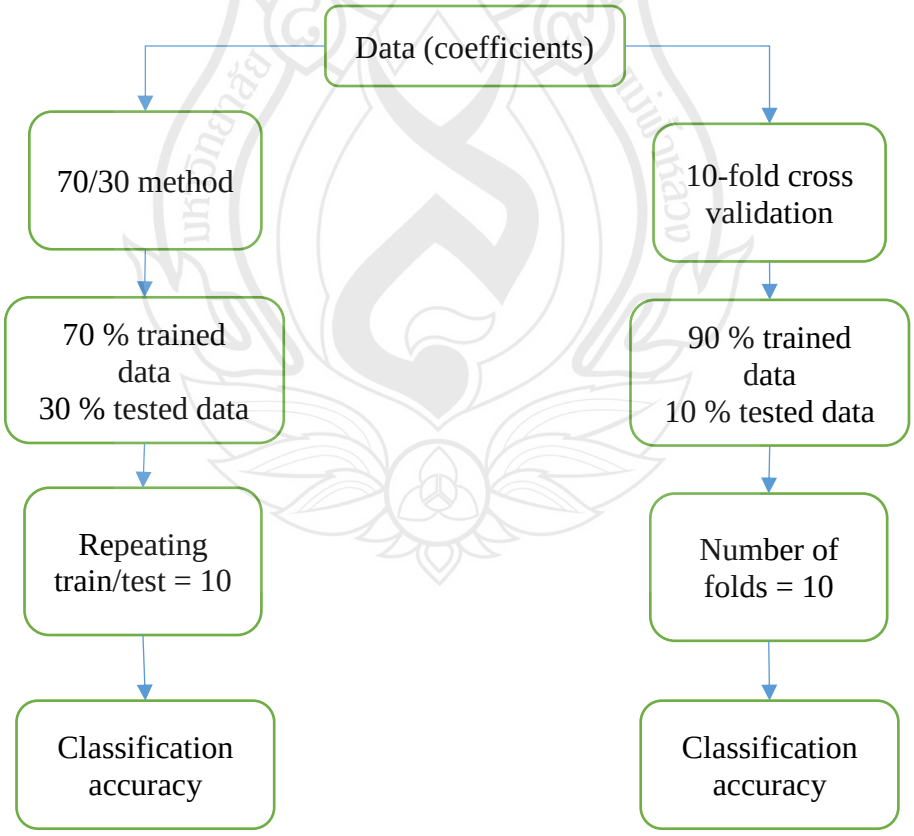


Figure 4.13 Overall methodology (Classification algorithms)

Firstly, we acquire coefficients from polynomial regression model for every patterns. We have three main experiments for 70/30 method, 10-fold, and classification. The experiments are investigating on three different value of exponents which are four exponent, six exponent, and customized data. Swerving pattern seem to be fit very at four exponents, so we tested all patterns with four exponents on classification models. Then, we also test all patterns with sixth degree because sixth degree is a choice for fitting weaving patterns that has a biggest curve among all patterns. Classification accuracy is computed by using Confusion matrix ($Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$). Example of confusion matrix is shown in Figure 4.14.

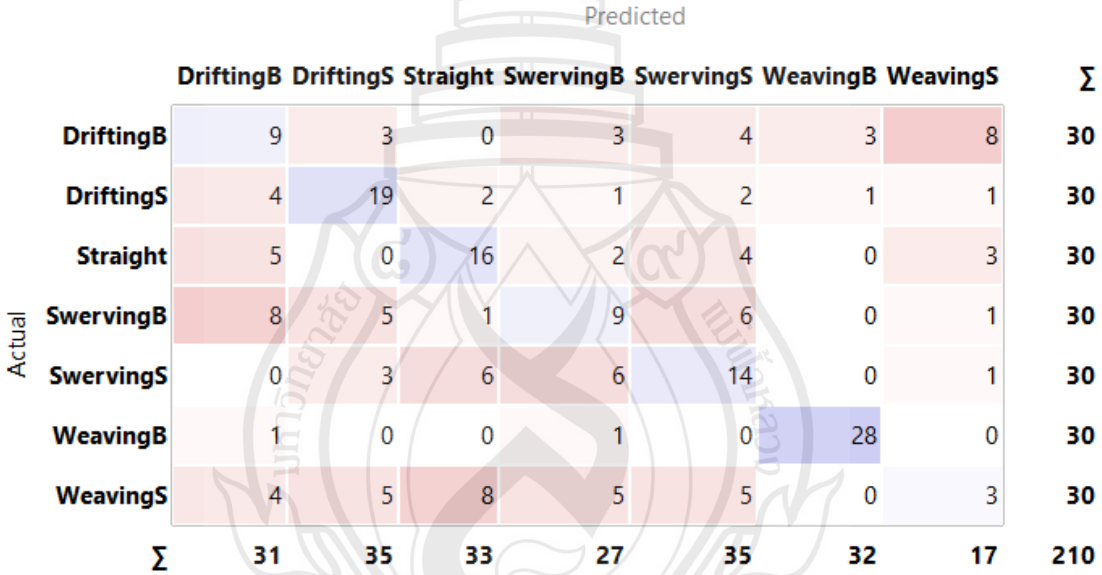


Figure 4.14 Confusion matrix (Sixth degree)

Classification accuracy of 70/30 method, 10-fold cross validation is presented in Table 4.25.

Table 4.25 Classification accuracy from 70/30 method and 10-fold cross validation

Exponent	70/30 method (Accuracy)	10-fold cross validation (Accuracy)
6	55.2%	58.6%
4	46.6%	42.9%

We can notice that both values of exponent provided quite low classification accuracy. Thus, we customized data according to previous experiment. straight and drifting are fit best at OLS model and swerving is fit best at polynomial models with four exponents. Lastly, weaving is fit best at six exponents. As a result of this, we obtain the coefficients from each model that the patterns fit best. For example, straight and drifting will have 2 values which are coefficient and intercept. In opposition, swerving will have 5 values such as $S = \pi x^5 + nx^4 + nx^3 + nx^2 + nx^1 + constant$, and weaving will have 6 coefficients + constant. The example of data is shown in Table 4.26

Table 4.26 Results from polynomial regression and OLS (Sixth degree, forth degree, two degree)

Patterns	x^6	x^5	x^4	x^3	x^2	x^1	Constant
Straight	0	0	0	0	0	2.14E+00 x	-5.44E+02
Drifting (BG)	0	0	0	0	0	-1.42E+00 x	2.83E+02
Drifting (SM)	0	0	0	0	0	-1.98E+00 x	3.37E+02
Swerving (BG)	0	0	7.56E-07	-4.79E-04	1.06E-01	-9.94E+00	6.06E+02
Swerving (SM)	0	0	-1.32E-08	8.18E-07	6.29E-03	-2.07E+00	3.23E+02
Weaving (BG)	3e-10 x	-2.435e-07 x	7.889e-05 x	-0.01288 x	1.097 x	- 46.22 x	892.7
Weaving (SM)	6.56e-12 x	-4.918e-09 x	1.116e-06 x	-1.209e-05 x	-0.0242 x	1.92 x	87.21

With splitting 70 percentages of data for training and 30 percentages of data for testing. Then, we retrain, and retest for ten rounds. The classification accuracy seems is 70.5% which is satisfied outcome. A confusion matrix is presented in Figure 4.15.

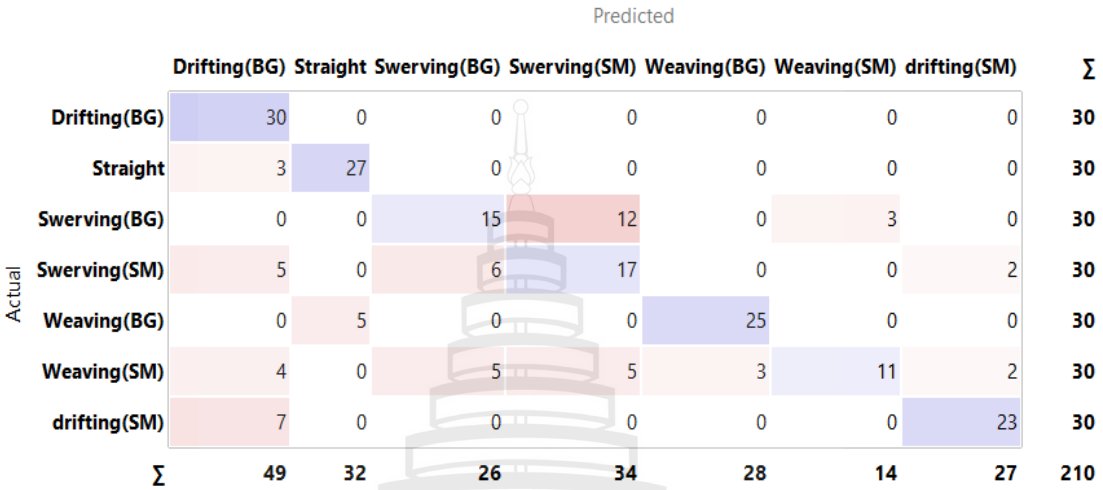


Figure 4.15 Confusion matrix (70/30 method)

From Figure 4.15, we can see that a lot of false classification came from swerving (Big curve) and swerving (Small curve). We can conclude that swerving (Big curve) and swerving (Small curve) have similar riding pattern.

Moving to another experiment that is 10-fold cross validation. The data spilt into 9:1 ratio. 90 percentages of data are used for training and 10 percentages of data are used for testing. The number of folds is ten. The confusion matrix is shown in Figure 4.16. For 10-fold method, the Classification accuracy is slightly increased which is 75.7% as shown in Table 4.27.

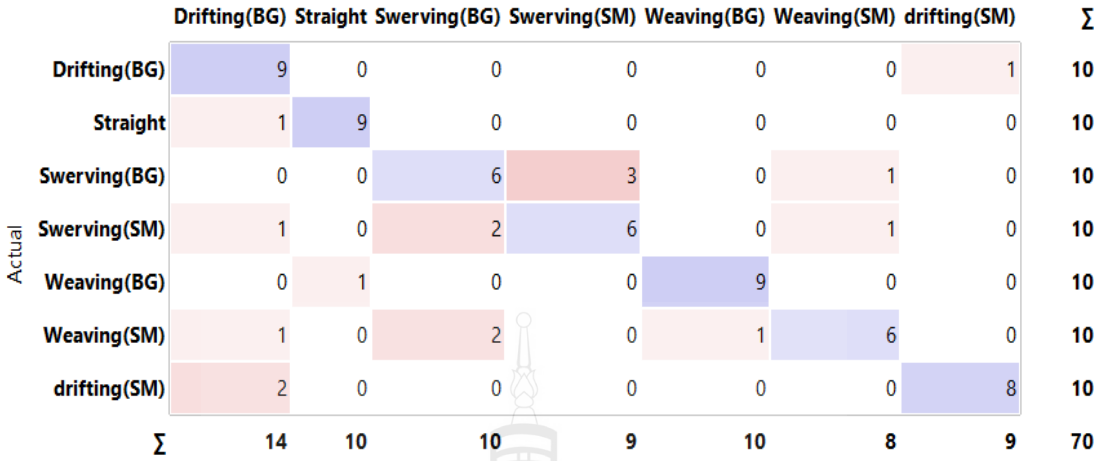


Figure 4.16 Confusion matrix (10-fold cross validation)

Table 4.27 Classification accuracy from 70/30 method and 10-fold cross validation (Pre-possessing data included)

Exponent	70/30 method (Accuracy)	10-fold cross validation (Accuracy)
6	55.2%	58.6%
4	46.6%	42.9%
Pre-possessing data	70.5%	75.7%

The result is similar to 70/30 method, the model provides most of false positives which are from both swerving (Big curve) and swerving (Small curve).

We also apply customized data to classification tree. From Figure 4.17, straight and weaving pattern are recognized at early stage. The algorithm predicts 90% accuracy for both weaving (Big curve) and straight pattern. Using more parameters to classify other patterns, drifting (Small curve) and drifting (Big curve) still get high classification accuracy. From all the patterns, swerving seems to be the hardest pattern to be classified.

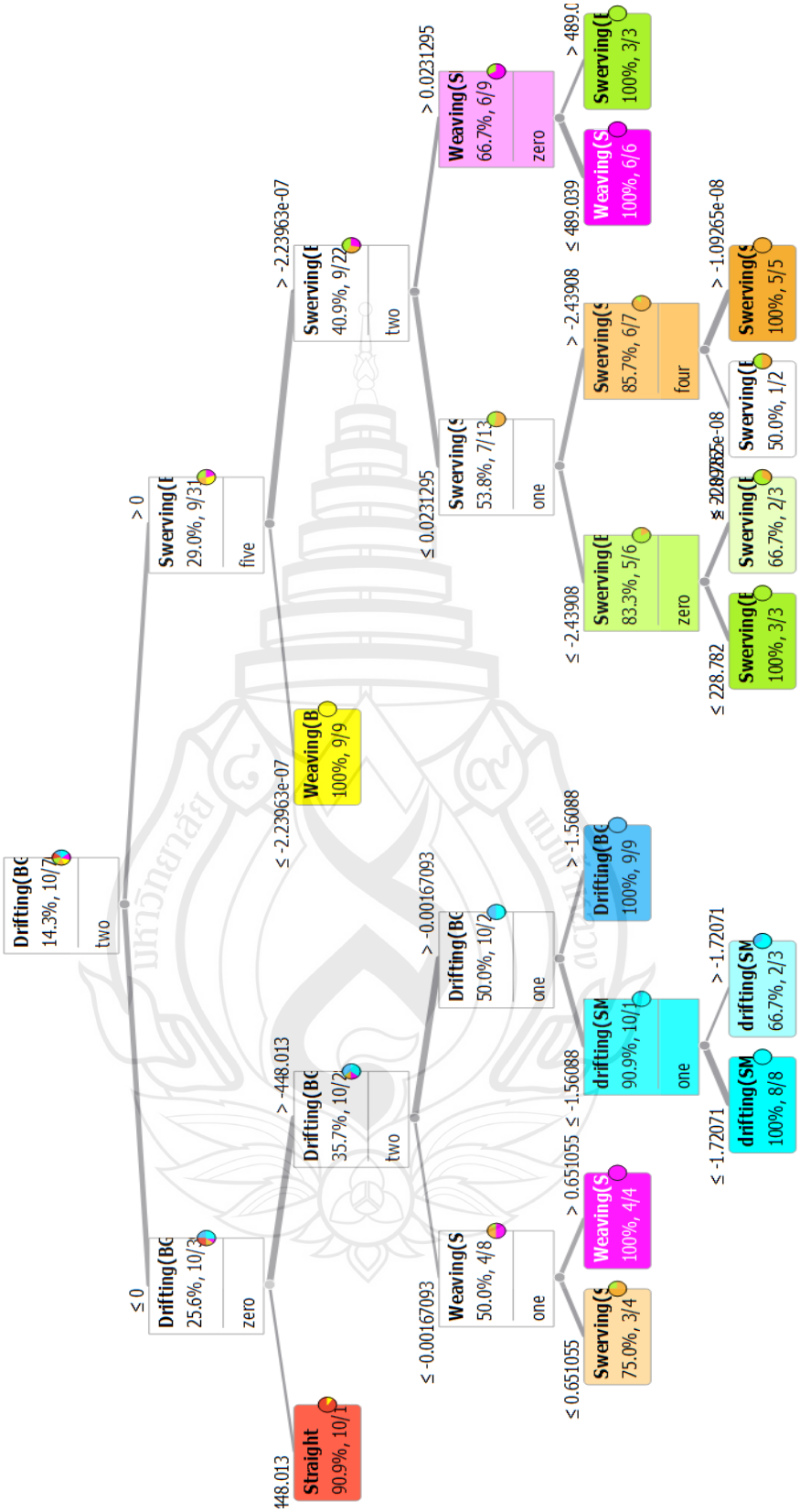


Figure 4.17 Classification tree (Pre-possessing data)

CHAPTER 5

DISCUSSIONS AND CONCLUSION

5.1 Discussion and Conclusion

From the experimental results, the best candidates from TensorFlow 1 model were further investigated with low resolution images of the same video and could double the efficiency on some models. Finally, the most efficient models are Yolo V3 and rfcn_resnet101_coco for large (1920x1080) and small (480x270) video frames. For TensorFlow 2, it seems that most models were best detected in the front view image, but not in the top view image. Only two models gave satisfactory results to the back view image with an accuracy of 70% or higher. The best candidates from each group of models were tested with three additional videos focused on the front view. Two models provided efficient results: CH104 and FRCNN_R50_V1. Therefore, these models were further tested at low resolution on all frontal images. The two models provide very similar results, with FRCNN_R50_V1 slightly above CH104, with an average efficiency of 380.86 and an average accuracy of 72.56. For back view images, the CH104 is the perfect model for both high resolution (1080 x 1980) and low resolution (270 x 480). The detection accuracy from the top view can be very low compared to other views. Nevertheless, Top-view is a view that can be applied to real traffic scenes similar to CCTV, depending on the camera angle. However, results from TensorFlow 1 models are very poor at top-view which none models could yield above 50 % detection accuracy. YoloV3 yielded 63.52 % but it requires long time consuming even with low resolution. Thus, the further experiment is tested on TensorFlow 2 models only.

All models were tested in the other three videos around the top view. With the exception of CH104, which achieved an accuracy of over 70%, no model could achieve an accuracy of over 55%. In addition, the top model for each object detector is selected and further evaluated at the low resolution of the three frontal videos. The CH104 achieved the highest efficiency and was able to improve efficiency by about 160% by reducing the image resolution from 1920x1080 pixels to 960x540 pixels. By reducing the resolution from 1920x1080 pixels to 480x270 pixels, the efficiency of CH104 has improved significantly by about 360%. CH104 was still the best candidate for top view detection, but the other candidates were inefficient. Finally, FRCNN_R50_V1 is the model that offers the highest efficiency of all front view detection models. However, if a goal is to detect all three views of your motorcycle, then CH104 is recommended.

For abnormal riding detection, the results of RANSAC and OLS are similar when it comes to certain riding cases such as swerving, and weaving. Table 3 shows the R-squared of normal riding, which is around 0.9956, 0.9938, and 0.4015. It is noticeable that the low R-squared of normal riding results in noise is caused by the unwanted noises generated by the motorcycle. While filming the simulated video, the detection model could detect parked motorcycle in the scene. Thereby, R-square score from OLS was low.

By implementing OLS, we successfully differentiate straight pattern and drifting pattern from the other two patterns. Polynomial regression model was applied to fit curve line such as swerving pattern and weaving pattern. R-squared score from polynomial model can differentiate weaving pattern from swerving pattern. However, R-squared score is not able to classify sub-group of each abnormal riding i.e., weaving (Small curve), weaving (big curve) etc. Therefore, we further have experimental investigation on classification methods which are 70/30 split train and test method based on machine learning, and 10-fold cross validation. Obtaining the coefficients from polynomial regression model and OLS. We tested three different ways. Firstly, we have fixed value of exponent and run it all patterns. With four exponent and six exponent, classification accuracy is quite low which are below 60% for both 70/30 method and 10-fold cross validation. Thereby, we customized the data by setting coefficients for each pattern. From experiment, we concluded that straight and drifting have satisfied result from OLS model and swerving and weaving pattern have been well

fit by using polynomial regression model with 4 exponents and 6 exponents respectively. By customizing data, the classification accuracy significantly increases for both 70/30 method and 10-fold method. The classification accuracy is 70.5% for 70/30 method and 75.7 for 10-fold method. In conclusion, 10-fold cross validation provides slightly higher classification than 70/30 method. Classification tree algorithm was also implemented for classifying types of the patterns. The algorithm provides good classification results for weaving (Big curve), straight, and drifting (both small and big curve). swerving seems to be the most difficult pattern to be classified for all method that we have tested.

Nonetheless, none of the tested models can perform real-time motorcycle detection due to their detection times are above the usual video play-time which the fastest model provides 0.18 per frame. There is still a room for further investigation including how we could improve the detection efficiency by either increasing the detection accuracy and/or decreasing the detection time. These improvements could be in different directions such as calibrating the detection models, limiting the detection class to merely motorcycle, using a better detection environment e.g., better GPU or cloud-based settings, fine-tuning the detection models' parameters, and even modify the internal structure of the models. For classification, more algorithms or methods should be explored. Additionally, the amount of data is limited in our work, so more data should be collected. Tracking motorcycle is very important for real traffic scene. Tracking method should be implemented.

5.2 Publication

Jarunakarint, V., Uttama, S., & Rueangsirarak, W. (2020). Survey and experimental comparison of machine learning models for motorcycle detection. In *2020 - 5th International Conference on Information Technology (InCIT)* (pp. 320–325). https://doi.org/10.1109/InCIT50588.2020.9310954 [40]

Jarunakarint, V., & Uttama, S. (2021). Detection of risky riding patterns of motorcyclists based on deep learning and linear regression. *NU Int. J. Sci., 18*(1), 136-151. [41]

REFERENCES

REFERENCES

[1] World Health Organization (WHO). (2018). *Launch of the global status report on road safety 2018 in Thailand*. https://www.who.int/thailand/news/detail/19-12-2018-launch-of-the-global-status-report-on-road-safety-2018-in-thailand

[2] Thai RSC. (n.d.). *Accident data center of Thailand*. https://www.thairsc.com/

[3] Jang, S. W., & Ahn, B. (2020). Implementation of detection system for drowsy driving prevention using image recognition and IoT. *Sustainability, 12*(7), 3037. https://doi.org/10.3390/su12073037

[4] National Highway Traffic Safety Administration. (1998). *The visual detection of DWI motorists*. U.S. Department of Transportation, National Highway Traffic Safety Administration.

[5] Harkous, H., & Artail, H. (2019). A two-stage machine learning method for highly-accurate drunk driving detection. In *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, (pp. 1–6). IEEE. https://doi.org/10.1109/WiMOB.2019.8923366

[6] Wu, X., Zhou, J., An, J., & Yang, Y. (2018). Abnormal driving behavior detection for bus based on the Bayesian classifier. In *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)* (pp. 266–272). IEEE. https://doi.org/10.1109/ICACI.2018.8377618

[7] Messelodi, S., Modena, C., & Cattoni, G. (2007). Vision-based bicycle/ motorcycle classification. *Pattern Recognit. Lett., 28*(13), 1719–1726. https://doi.org/10.1016/j.patrec.2007.04.014

[8] Chiu, C.- C., Ku, M.- Y., & Chen, H. -T. (2007). Motorcycle detection and tracking system with occlusion segmentation. In *Eighth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS '07)* (pp. 32–32). IEEE. https://doi.org/10.1109/WIAMIS.2007.60

[9] Kalyan, S. S., Pratyusha, V., Nishitha, N., & Ramesh, T. K. (2020). Vehicle detection using image processing. In *2020 IEEE International Conference for Innovation in Technology (INOCON)* (pp. 1–5). IEEE. https://doi.org/10.1109/INOCON50539.2020.9298188

[10] Song, H., Liang, H., Li, H., Dai, Z., & Yun, X. (2019). Vision-based vehicle detection and counting system using deep learning in highway scenes. *Eur. Transp. Res. Rev., 11*(1), Article number: 51 (2019). https://doi.org/10.1186/s12544-019-0390-4

[11] Suhao, L., Jinzhao, L., Guoquan, L., Tong, B., Huiqian, W., & Yu, P. (2018). Vehicle type detection based on deep learning in traffic scene. *Procedia Comput. Sci., 131,* 564–572. https://doi.org/v10.1016/j.procs.2018.04.281

[12] Yılmaz, A., Guzel, M., Askerzade, I., & Bostanci, G. E. (2018). A vehicle detection approach using deep learning methodologies. *arXiv:1804.00429.* https://doi.org/10.48550/arXiv.1804.00429

[13] Ku, M.- Y., Chiu, C. -C., Chen, H. -T., & Hong, S. -H. (2008). Visual motorcycle detection and tracking algorithms. *WSEAS Transactions on Electronics, 5*(4), 121-131.

[14] Espinosa, J. E., Velastin, S. A., & Branch, J. W. (2019). Detection and tracking of motorcycles in congested urban environments using deep learning and markov decision processes. In J. Carrasco-Ochoa, J. Martínez-Trinidad, J. Olvera-López & J. Salas (Eds.), *Pattern recognition. MCPR 2019. Lecture Notes in Computer Science* (vol. 11524, pp. 139–148). Springer, Cham. https://doi.org/10.1007/978-3-030-21077-9_13

[15] Huynh, K., Le, T. -S., & Hamamoto, K. (2016). Convolutional neural network for motorbike detection in dense traffic. In *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)* (pp. 369-374). IEEE. https://doi.org/10.1109/CCE.2016.7562664.

[16] Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., & Qu, R. (2019). A survey of deep learning-based object detection. *IEEE Access, 7*, 28837–128868. https://doi.org/10.1109/ACCESS.2019.2939201

[17] Arinaldi, A., Pradana, J., & Gurusinga, A. (2018). Detection and classification of vehicles for traffic video analytics. *Procedia Comput. Sci., 144*, 259–268. https://doi.org/10.1016/j.procs.2018.10.52

[18] Sri Jamiya, S., & Esther Rani, P. (2019). A survey on vehicle detection and tracking algorithms in real time video surveillance. *International Journal of Scientific & Technology Research, 8*(10), 2266-2276. https://doi.org/10.13140/RG.2.2.14975.84648

[19] David, A., Kumar, K., & Kumar, S. (2016). Vision-based vehicle detection survey. *Int. J. Recent Contrib. Eng. Sci. IT IJES, 4*(1), 31-35. https://doi.org/10.3991/ijes.v4i1.5590.

[20] Mukhtar, A., Xia, L., Tang, T. B., & Abu Kassim, K. A. (2013). On-road approaching motorcycle detection and tracking techniques: A survey. In *2013 IEEE International Conference on Control System, Computing and Engineering* (pp. 63–68). IEEE. https://doi.org/10.1109/ICCSCE.2013.6719933

[21] Kim, K., Kim, B. W., Lee, J. W., & Park, D. -H. (2018). Driver reaction acceptance and evaluation to abnormal driving situations. In *2018 International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 1377–1379). https://doi.org/10.1109/ICTC.2018.8539705

[22] Yu, J., Chen, Z., Zhu, Y., Chen, Y., Kong, L., & Li, M. (2017). Fine-grained abnormal driving behaviors detection and identification with smartphones. *IEEE Trans. Mob. Comput.*, *16*(8), 2198–2212. https://doi.org/10.1109/TMC.2016.2618873

[23] Hu, J., Zhang, X., & Maybank, S. (2020). Abnormal driving detection with normalized driving behavior data: A deep learning approach. *IEEE Trans. Veh. Technol.*, *69*(7), 6943–6951. https://doi.org/10.1109/TVT.2020.299324

[24] Sandeep, K., Ravikumar, P., & Ranjith, S. (2017). Novel drunken driving detection and prevention models using internet of things. In *2017 International Conference on Recent Trends in Electrical, Electronics and Computing Technologies (ICRTEECT)* (pp. 145–149). IEEE. https://doi.org/10.1109/ICRTEECT.2017.38

[25] Al-Sultan, S., Al-Bayatti, A. H., & Zedan, H. (2013). Context-aware driver behavior detection system in intelligent transportation systems. *IEEE Trans. Veh. Technol.*, *62*(9), 4264–4275. https://doi.org/10.1109/TVT.2013.2263400

[26] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, É. (2012). Scikit-learn: Machine Learning in python. *J. Mach. Learn. Res.*, *12*(85), 2825−2830.

[27] Yu, H., Chen, C., Du, X., Li, Y., Rashwan, A., Hou, L., . . . Li, J. (2020). *TensorFlow model garden*. https://github.com/tensorflow/models

[28] Olafenwa, M. (2021). *OlafenwaMoses/ImageAI*. Retrieved June 18, 2021, from https://github.com/OlafenwaMoses/ImageAI

[29] Pai, A. Y. (2021). *AdityaPai2398/vehicle-and-pedestrian-detection-using-haar-cascades*. https://github.com/AdityaPai2398/ Vehicle-And-Pedestrian-Detection-Using-Haar-Cascades

[30] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell., 39*(60), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

[31] Jia, S., Diao, C., Zhang, G., Dun, A., Sun, Y., Li, X., . . . Zhang X. (2019). Object detection based on the improved single shot MultiBox detector. *J. Phys. Conf. Ser., 1187*(042041). https://doi.org/10.1088/1742-6596/1187/4/042041

[32] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (2019). CenterNet: Keypoint triplets for object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 6568–6577). IEEE. https://doi.org/10.1109/ICCV.2019.00667

[33] Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 10778–10787). IEEE. https://doi.org/10.1109/CVPR42600.2020.01079

[34]*COCO - Common Objects in Context*. (n.d.). https://cocodataset.org/#detection-eval

[35] Kulkarni, A., Chong, D., & Batarseh, F. A. (2020). 5 - Foundations of data imbalance and solutions for a data democracy. In F. A. Batarseh & R. Yang (Eds.), *Data democracy* (pp. 83–106). Academic Press. https://doi.org/10.1016/B978-0-12-818366-3.00005-8

[36] Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). *Why 70/30 or 80/20 relation between training and testing sets: A pedagogical explanation*. https://scholarworks.utep.edu/cs_techrep/1209

[37] Berrar, D. (2018). Cross-Validation. *Encyclopedia of Bioinformatics and Computational Biology, 1*, 542-545. https://doi.org/10.1016/B978-0-12-809633-8.20349-X

[38] Jijo, B., & Mohsin Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. *J. Appl. Sci. Technol. Trends, 2*, 20–28.

[39] *Machine Learning Polynomial Regression*. (n.d.). Retrieved June 18, 2022, from https://www.w3schools.com/python/python_ml_polynomial_regression.asp

[40] Jarunakarint, V., Uttama, S., & Rueangsirarak, W. (2020). Survey and experimental comparison of machine learning models for motorcycle detection. In *2020 - 5th International Conference on Information Technology (InCIT)* (pp. 320–325). https://doi.org/10.1109/InCIT50588.2020.9310954

[41] Jarunakarint, V., & Uttama, S. (2021). Detection of risky riding patterns of motorcyclists based on deep learning and linear regression. *NU Int. J. Sci.*, *18*(1), 136-151.