Data Science and Engineering (DSE) Record, Volume 4, issue 1.

Development of Detection System for Motorcyclists without Helmet

Sukrit Akarametagul [1] and Paskorn Champrasert [2]

[1] Master's Degree Program in Data Science, Chiang Mai University, Chiang Mai, Thailand
[2] Department of Computer Engineering, Faculty of Engineering, Chiang Mai University, Chiang Mai, Thailand

sukrit_ak@cmu.ac.th [1] and paskorn.c@cmu.ac.th [2]

Abstract. This research is conducted to develop an artificial intelligence and machine learning system that can detect riders who do not wear helmets and to analyze and compare the capabilities of YOLO and RetinaNet algorithms in detecting these riders. The data from the CMU Smart Gate system's LPR (License Plate Recognition) camera, which detects the data of vehicles entering and exiting the gates of Chiang Mai University, was used for training and measuring the system performance. The results showed that both YOLO and RetinaNet algorithm could be used to develop a system to detect motorcyclists who do not wear helmets. However, the RetinaNet algorithm training model mean precision of 0.999 was higher than that of the YOLO algorithm which is 0.983. Precision specific to detecting motorcyclists without helmets both algorithms got the same result of 1.000. When the model was tested for processing time per image, the YOLO algorithm took less time to execute than the RetinaNet algorithm. At average value, the YOLO algorithm took 0.152 seconds. The RetinaNet algorithm took 1.659 seconds.

Keywords: Deep learning, Helmet use detection, Motorcycle, YOLO, RetinaNet.

1. Introduction

The behavior that poses a risk of road accidents includes speeding, driving under the influence, not wearing a helmet, not buckling up, not turning on motorcycle lights, and disregarding traffic signals, among others. Researchers are particularly interested in the behavior of not wearing a helmet because the statistics show that registered motorcycles in Thailand have a staggering 53.96% non-compliance rate. [1] The statistics also indicate that motorcycle accidents account for a high percentage of accidents in Thailand, reaching 39% [2], ranking first among all types of vehicles. Furthermore, only 45% of motorcycle users [3], including both riders and passengers, wear helmets in Thailand.

Despite the existence of laws and research confirming the benefits of wearing helmets, and even though the government campaigns for motorcycle riders and passengers to wear helmets at all times, the statistical data mentioned earlier still remains high. This is because the enforcement of these regulations requires personnel to physically check and ensure compliance, which is not feasible to do for motorcycle riders at all times.

In order to assist officers in monitoring helmet usage at all times, researchers are interested in utilizing artificial intelligence and machine learning techniques to detect motorcycle riders who do not wear helmets. They plan to use the CMU Smart Gate project, which can detect vehicle data entering and exiting the gates within Chiang Mai University. This includes capturing license plate numbers and registered card information. The system will employ cameras to monitor each gate of the university, making it suitable for detecting motorcycle riders who do not wear helmets. Two algorithms, namely YOLO and RetinaNet, will be tested to determine which one is most suitable for detecting motorcycle riders without helmets. The results of each algorithm will be analyzed and compared.

2. Literature Review

2.1. Previous Studies Using YOLO and RetinaNet algorithms

Wei Jia and colleagues [4] conducted a study on the development of a helmet detection system for motorcycle riders in China. The development process consisted of two main steps. In the first step, they developed a motorcycle detection system using closed-circuit cameras in China with image resolution of 1920 x 1080 pixels. In the second step, they used the output data from the first step as input for the second step to detect motorcycle riders without helmets. They utilized the YOLOv5 algorithm as the main approach and compared it with two-stage algorithms such as Faster R-CNN, Cascade R-CNN, and Libra R-CNN, as well as one-stage algorithms such as SSD, RetinaNet, YOLOv5, and FCOS. The overall results from both steps showed that the YOLOv5 algorithm achieved the highest mAP (mean Average Precision) of 97.7% and an F1-Score of 92.7%.

Hanhe Lin and colleagues [5] conducted a study on the development of a helmet detection system for motorcycle users using the CNN-based multi-task learning (MTL) method. They collected data from 91,000 frames of motorcycles, consisting of 10,006 motorcycles, from 12 different locations in Myanmar. The helmet detection system aimed to detect both riders and passengers sitting in various positions, including one front rider, one rear pillion, two rear pillions, and three rear pillions. The study compared two algorithms, YOLOv2 and RetinaNet. In this research, the helmet detection task involved detecting both riders and passengers, resulting in a total of 34 classes for training. The results of the study were presented in terms of F1-Score, where YOLOv2 achieved a score of 60.0% and RetinaNet achieved a score of 67.3%.

J. Sivaraj and colleagues [6] conducted a study on the development of a helmet detection system for road safety using Google Colab for development. The training data consisted of a total of 2,121 images collected from various open-source websites. These images were divided into 1,000 images of motorcycle riders and 1,121 images of helmets. Two models were trained: a motorcycle rider detection model and a helmet detection model. The study compared three algorithms: YOLOv3, SSD300, and Caffe Model. The results of the motorcycle rider detection models were as follows: YOLOv3 achieved an accuracy of 91.19%, SSD300 achieved 74.30%, and Caffe Model achieved 76.00%. The results of the helmet detection models were as follows: YOLOv3 achieved an accuracy of 90.13%, SSD300 achieved 85.00%, and Caffe Model achieved 81.00%.

3. Data preparation

3.1. Data

The researchers have collected data from the front gate camera of Chiang Mai University, which is recorded by the CMU Smart Gate system. The data was collected for a total of 3 days during the time periods of 08:00 - 09:00 and 15:00 - 16:00. The collected data is in the form of video footage captured by the LPR camera. The data is divided into 2 days for training the model and 1 day for testing the model.

3.2. Convert video data to image data

The data obtained from the CMU Smart Gate system was converted from video format to image data. The researchers used VLC software to convert the video data into images.

3.3. Bounding boxes in image data

The researchers used the LabelImg program to annotate the desired objects from the image data. They annotated the motorcycle riders wearing helmets, as shown in Image 1, and the motorcycle riders not wearing helmets, as shown in the example image in Fig. 1.

Fig. 1. Bounding box of a motorcycle rider wearing a helmet and rider not wearing a helmet from the LabelImg program.

From annotating all the image data, we obtained a set of images for training and testing the model, as shown in Table 1.

Table 1. Image Data for Training and Testing the Model

	Training data	Test data
The motorcyclist wearing a helmet	542	220
The motorcyclist not wearing a helmet	184	72
Total	726	292

3.4. YOLO Algorithm

The YOLO algorithm, or You Only Look Once, is a real-time object detection algorithm created by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi in 2015. It is notable for its speed, accuracy, and the ability to detect overlapping objects. The algorithm belongs to the One-stage Object Detection group and consists of three main components. The first component is the Backbone, which uses Darknet53, a Convolutional Neural Network (CNN), to extract image features. The second component is the Neck, which incorporates SSP (Spatial Pyramid Pooling) and PANet (Path Aggregation Network). The third component is the Head, responsible for predicting results using the YOLO algorithm [7], as shown in the example in Fig. 2.

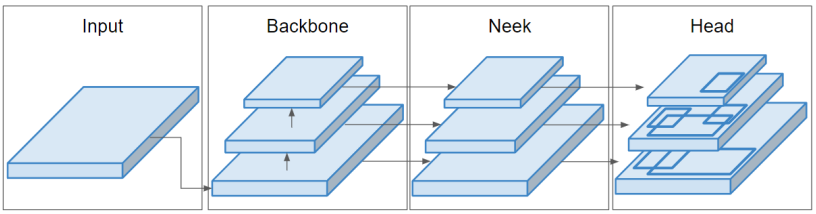

Fig. 2. YOLO Algorithm Architecture

3.5. RetinaNet Algorithm

The RetinaNet algorithm was developed by Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. It belongs to the One-stage Object Detection group, similar to the YOLO algorithm. Within the RetinaNet algorithm architecture, the Backbone utilizes ResNet (Residual Network) for extracting image features using Convolutional Neural Networks. The Neck employs FPN (Feature Pyramid Network) for detecting objects at multiple scales, and the Head utilizes the RetinaNet algorithm to make predictions based on the data from the Neck [8]. This is illustrated in Fig. 3.

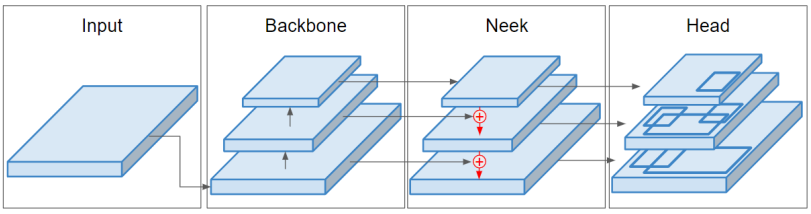

Fig. 3. RetinaNet Algorithm Architecture

4. Results

4.1. The method of performance evaluation

The measurement of accuracy is evaluated based on the values of Precision and Recall, as shown in Equations 1 and 2.

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

True Positive (TP): A test result that correctly indicates the presence of a condition or characteristic

True Negative (TN): A test result that correctly indicates the absence of a condition or characteristic

False Positive (FP): A test result which wrongly indicates that a particular condition or attribute is present

False Negative (FN): A test result which wrongly indicates that a particular condition or attribute is absent

In cases where it is not possible to choose between Precision or Recall, the F1-Score formula can be used. The F1-Score is the harmonic mean between Precision and Recall, calculated using Equation 3.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (3)$$

In cases where there is a need to evaluate overall performance in multi-class training, and the training involves more than one class, the mAP (mean Average Precision) formula can be used to assess the average accuracy of object detection. This can be calculated using Equation 4.

$$mAP = \frac{1}{k}\sum_{i}^{k} APi \quad (4)$$

"k" is the number of Epochs.

"APi" stands for the average precision and recall, which can be calculated using the formula shown in Equation 5.

$$AP = \sum_{n} (R_n - R_{n-1}) P_n \quad (5)$$

"R" is the value of recall.

"P" is the value of precision

4.2. The results of training the YOLO algorithm

The researchers trained the YOLO algorithm using a total of 726 images. They used 292 images for testing the trained model. The parameter "Epoch" was set to 100, 200, 300, and 400, while the parameter "Batch" was set to 1, 2, and 4.

Table 2. Results of YOLO Algorithm Training Experiment

Parameter	Epoch	100			200			300			400		
	Batch	1	2	4	1	2	4	1	2	4	1	2	4
Results	Epoch	100	100	100	179	142	156	237	**242**	166	218	115	161
	Precision A	0.889	0.938	0.933	0.938	0.964	0.960	0.969	0.972	0.945	0.924	0.938	0.953
	Recall A	0.968	0.982	0.995	1.000	1.000	0.994	0.994	0.964	1.000	1.000	0.960	0.982
	F1-Score A	0.927	0.959	0.963	0.968	0.982	0.977	0.981	0.968	0.972	0.960	0.949	0.967
	Precision B	0.859	0.940	0.984	0.925	**1.000**	0.969	0.997	0.974	**1.000**	0.985	0.947	**1.000**
	Recall B	0.847	0.872	0.854	0.917	**0.949**	0.880	0.931	0.917	0.876	0.892	0.819	0.873
	F1-Score A	0.853	0.905	0.914	0.921	**0.974**	0.922	0.963	0.945	0.934	0.936	0.878	0.932
	mAP	0.874	0.939	0.958	0.932	0.982	0.965	**0.983**	0.973	0.973	0.954	0.942	0.976

Precision A: The accuracy of detecting motorcycle riders wearing safety helmets.

Recall A: The correctness of detecting motorcycle riders wearing safety helmets.

F1-Score A: The average value between the precision and recall of detecting motorcycle riders wearing safety helmets.

Precision B: The accuracy of detecting motorcycle riders not wearing safety helmets.

Recall B: The correctness of detecting motorcycle riders not wearing safety helmets.

F1-Score B: The average value between the precision and recall of detecting motorcycle riders not wearing safety helmets.

mAP: Mean Average Precision, calculated as the average precision of detecting motorcycle riders wearing safety helmets and the precision of detecting motorcycle riders not wearing safety helmets.

4.3. The resource utilization during the training of the YOLO algorithm.

The researchers used Colab Pro as the computing resource for training the YOLO algorithm.

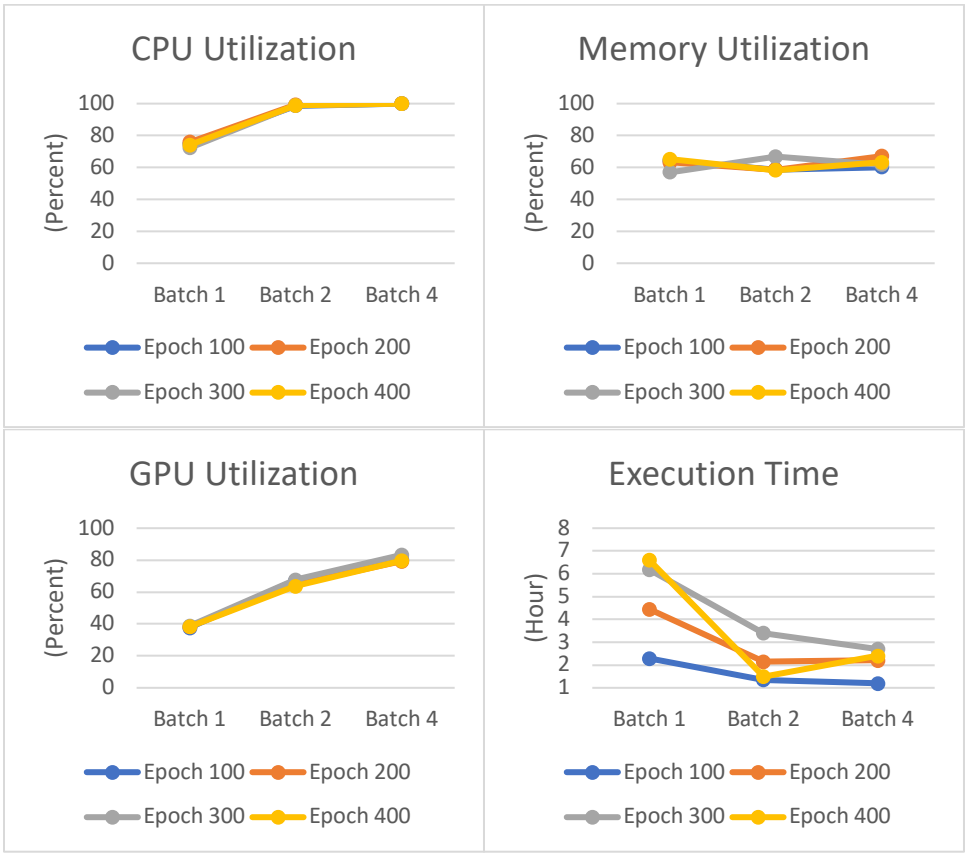

Fig. 4. Utilization of CPU, Memory, GPU, and Execution Time of the YOLO Algorithm.

The utilization of CPU, Memory, GPU, and Execution Time from the highest values, as shown in Fig. 4, reveals that as the Batch parameter increases, the CPU and GPU usage also increase, while the Execution Time decreases. This is because when the Batch parameter is set to 1, only one image is used for training, but when it is adjusted to 2, two images are used for training, resulting in higher CPU and GPU utilization. Conversely, as the training workload increases, the Execution Time decreases. In terms of Memory, it pertains to the utilization of the algorithm before processing through the GPU, which accounts for approximately 58% - 67% of the GPU's 14 GB resource. The Batch parameter does not affect Memory.

4.4. The results of training the RetinaNet algorithm

The results of training the RetinaNet model were as follows: The researcher used a total of 726 images for training the model and 292 images for testing. The parameter "Epoch" was set to 100, 200, 300, and 400, while the parameter "Batch" was set to 1, 2, and 4.

Table 3. Results of RetinaNet Training Experiment

Parameter	Epoch	100			200			300			400		
	Batch	1	2	4	1	2	4	1	2	4	1	2	4
	Steps	726	363	181	726	363	181	726	363	181	726	363	181
Results	Epoch	24	25	**40**	24	26	27	24	26	27	24	25	28
	Precision A	0.999	0.999	0.999	0.992	0.999	0.999	0.998	0.999	0.999	0.999	0.999	0.999
	Precision B	0.995	0.997	0.999	0.998	0.995	0.999	0.997	0.999	0.999	0.997	0.999	**1.000**
	mAP	0.996	0.998	**0.999**	0.995	0.997	**0.999**	0.998	**0.999**	**0.999**	0.998	**0.999**	**0.999**

Precision A is the accuracy of detecting motorcyclists wearing helmets.

Recall A is the correctness of detecting motorcyclists wearing helmets.

mAP (mean Average Precision) is calculated as the average accuracy of detecting motorcyclists wearing helmets and the accuracy of detecting motorcyclists not wearing helmets.

4.5. The resource utilization during the training of the RetinaNet algorithm.

The researchers used Colab Pro as the computing resource for training the RetinaNet algorithm.

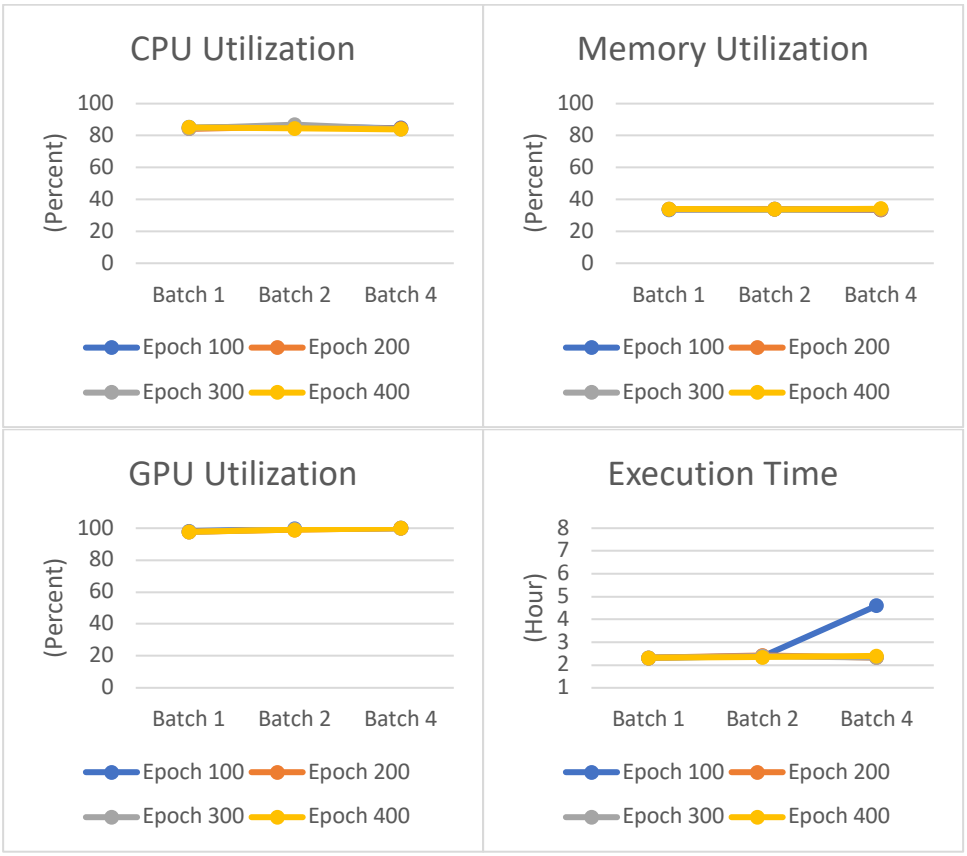

Fig. 5. CPU, Memory, GPU, and Execution Time Usage of RetinaNet Algorithm.

The utilization of CPU, Memory, GPU, and Execution Time from the highest values of the RetinaNet model, as shown in Fig. 5, reveals the following observations. CPU and Memory do not affect parameter adjustment since CPU utilizes approximately 83% - 87% of the 2.20GHz CPU resource, while Memory utilizes around 33.51% - 34.10% of the 14 GB Memory resource. As for GPU, it does not affect parameter adjustment but operates at 100% utilization from the 16 GB GPU resource, indicating that the RetinaNet algorithm maximizes GPU performance. Regarding Execution Time, it remains relatively consistent. The slight variation is due to the experimental results from different epochs, as depicted in Table 3, where the maximum value corresponds to the epoch at 40, resulting in slightly longer execution time.

4.6. The results of testing the processing time of the YOLO and RetinaNet algorithms per image.

The researcher conducted a performance test by processing a total of 292 test images to determine which algorithm has the fastest image processing time per image when using the trained models.

Table 4. Results of processing time testing for object detection algorithms YOLO and RetinaNet.

	MIN (s)	MAX (s)	AVG (s)	RANGE (s)
YOLO	**0.146**	**0.245**	**0.152**	**0.099**
RetinaNet	1.321	2.647	1.659	1.326

5. Discussion and Conclusion

From the results of the study on the YOLO and RetinaNet algorithms in Chapter 4, it was found that adjusting the parameters of Epoch and Batch for the YOLO algorithm did not have an impact on the number of training epochs and the average precision (mAP) value. When increasing the parameter values, the number of training epochs and mAP value sometimes increased, but in some cases, they decreased. As for the RetinaNet algorithm, adjusting the Epoch parameter did not affect the number of training epochs and the average precision. However, increasing the Batch parameter resulted in an increase in the number of training epochs and the average precision value.

The performance evaluation of the YOLO and RetinaNet algorithms on the output obtained after training the models shows that the YOLO algorithm has higher values for Epoch, Precision, Recall, F1-Score, and mAP compared to the RetinaNet algorithm, which only has values for Epoch, Precision, and mAP. This indicates that the YOLO algorithm provides better performance measurements. In terms of overall results from the experiments on training the models using images of motorcycle riders wearing and not wearing helmets, based on the average precision (mAP) values, the RetinaNet algorithm achieved the highest result at 0.999, while YOLO achieved 0.983. In terms of detecting motorcycle riders not wearing helmets, based on the precision values, both algorithms achieved the same result of 1.000. Comparing these precision values with relevant research, it was found that the precision values of the related research work in Chapter 2 were higher than 0.900, and the precision values of the researchers were also higher than 0.900. This indicates that the results of training the models using the YOLO and RetinaNet algorithms have precision values higher than 0.900, which are consistent with each other.

The utilization of computational resources during model training, the researchers only compared the Batch parameter because the Epoch parameter determines the maximum number of training rounds. However, the Batch parameter determines the number of iterations used to train the

model within one Epoch. For example, if there are 200 images and the Batch is set to 2, there will be 100 iterations, divided into 4 parts.

1. CPU Utilization: Increasing the Batch parameter of the YOLO algorithm results in higher CPU usage, whereas adjusting the Batch parameter of the RetinaNet algorithm does not significantly affect CPU usage. The YOLO algorithm consumes approximately 83% - 87% of the CPU resources at 2.20GHz.
2. Memory Utilization: Increasing the Batch parameter of both the YOLO and RetinaNet algorithms does not significantly impact memory usage. The YOLO algorithm consumes approximately 58% - 67% of the available memory, while the RetinaNet algorithm utilizes around 33% of the 14 GB memory resources.
3. GPU Utilization: Increasing the Batch parameter of the YOLO algorithm results in higher GPU usage, while adjusting the Batch parameter of the RetinaNet algorithm does not significantly impact GPU utilization. The YOLO algorithm utilizes approximately 100% of the available 16 GB GPU resources.
4. Execution Time: Increasing the Batch parameter of the YOLO algorithm results in reduced training time, while the RetinaNet algorithm maintains a similar execution time of approximately 2 hours and 20 minutes. This is because the experimental results for Epoch closely align with each other as shown in Table 3, except for one line that has the highest value. This discrepancy is due to the Epoch experiment being conducted for 40 rounds, resulting in increased execution time.

From the model testing results regarding the processing time of the YOLO and RetinaNet algorithms per image, the YOLO algorithm requires less processing time compared to the RetinaNet algorithm. Based on the AVG values, the YOLO algorithm takes 0.152 seconds, while the RetinaNet algorithm takes 1.659 seconds. The difference between them is 1.507 seconds in terms of the range (RANGE). Specifically, the YOLO algorithm has a difference of 0.099 seconds, whereas the RetinaNet algorithm has a difference of 1.326 seconds. When comparing these values, it is evident that the YOLO algorithm has a smaller difference than the RetinaNet algorithm.

The development of a system for detecting motorcycle riders without helmets using the YOLO and RetinaNet algorithms is possible. Both the YOLO and RetinaNet algorithms can be utilized to develop a system for detecting motorcycle riders who are not wearing helmets. When it comes to selecting and utilizing the algorithms, researchers have their own opinions. If the goal is to have a model that performs fast image processing and provides outputs such as Epoch, Precision, Recall, F1-Score, and mAP for model performance evaluation, YOLO algorithm can be chosen for training the model. On the other hand, if the objective is to have a highly accurate model without focusing on processing speed or having limited computing resources, the RetinaNet algorithm can be selected for training the model.

References

[1] Department of Land Transport, Transportation Statistics Report Year 2015 - 2019, 2019, pp. 3.

[2] Office of Transport and Traffic Policy and Planning, Analysis Report on Road Accident Situations by the Ministry of Transport, 2018, p. 7.

[3] ThaiRoads Foundation. The Helmet Usage Rate of Motorcycle Users in Thailand. [Online]. Available: http://trso.thairoads.org/statistic/helmet

[4] W. Jia, S. Xu, Z. Liang, Y. Zhao, H. Min, S. Li and Y. Yu, Real-time automatic helmet detection of motorcyclists in urban traffic using improved YOLOv5 detector, 2021, pp. 3623 - 3637.

[5] H. Lin , J.D. Deng , (Member, IEEE), Deike Albers and Felix Wilhelm Siebert, Helmet Use Detection of Tracked Motorcycles Using CNN-Based Multi-Task Learning, 2020, pp. 162073 - 162084.

[6] J. Sivaraj, R.S. Sudhan Adithya, Adhavan Alexander, M. Vishnudeep, S. Mohammed Farhanudin, P. Vishnu and T. Anusha, Helmet Violation Detection Application for Road Safety, 2021, pp. 448 - 454.

[7] OpenGenus IQ (2023). YOLO v5 model architecture. [Online]. Available: https://iq.opengenus.org/yolov5

[8] T.Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, Focal Loss for Dense Object Detection, 2017, pp. 2983 - 2984.