Indonesian Journal of Computer Science
ISSN 2549-7286 (*online*)
Jln. Khatib Sulaiman Dalam No. 1, Padang, Indonesia
Website: ijcs.stmikindonesia.ac.id | E-mail: ijcs@stmikindonesia.ac.id

Edge Computing-Based Automated Vehicle Classification System Using the MobileNet V2 Model

Rahardhita Widyatra Sudibyo[1], Haniah Mahmudah[2], Moch. Zen Samsono Hadi [3], Nihayatus Sa'adah[4]
[1]widi@pens.ac.id, [2]haniah@pens.ac.id, [3]zen@pens.ac.id, [4]nihayatus@pens.ac.id
Politeknik Elektronika Negeri Surabaya

Article Information

Submitted: 25 Oct 2022
Reviewed: 06 Nov 2022
Accepted : 20 Dec 2022

Keywords

Vehicle, Classification, Edge Computing, MobileNet V2

Abstract

The volume of traffic in one day is referred to as the average daily traffic volume. The Average Daily Traffic System (ADTS) is also used to detect road damage caused by excessive vehicle loads. In the ADTS system, vehicle data is still collected manually, with humans calculating the type and number of vehicles based on observations made. And then dividing into a time span. As a result, a system with a camera and deep learning data processing is required to automatically calculate the type and number of vehicles. The goal of this research is to develop edge computing systems by improving the system's performance in the calculation and classification of vehicles using the SSD MobileNet V2 model. The parameters of the MobileNet V2 have batch size 10, num classes 12, learning rate 0.005, number steps 200000 the results have the lowest loss value. The MobileNet V2 model can better classify vehicle types with a 65 FPS inference process.

A. Introduction

The average daily traffic volume is the volume of traffic in one day. Furthermore, the Average Daily Traffic System (ADTS) is used to detect road damage caused by excessive vehicle loads. This system is also frequently used to analyze the development of road transportation and the increase in the volume of vehicles each year to overcome traffic problems such as congestion [1]. This ADTS data is required by the Ministry of Transportation in order to determine road development priorities and to measure and evaluate demand on a road segment in order to determine the level of road service [2]. Vehicle data collection in the ADTS system is still done manually, with humans calculating the type and number of vehicles based on observations made and then divided into a time span. As a result, a system is required to automatically calculate the type and number of vehicles using a camera and data processing using deep learning. It requires an embedded ADTS hardware device that can quickly classify and count vehicles.

Many studies on images for detection and classification using artificial intelligence (AI) [3]-[15] have been conducted to improve system performance by combining various methods. The Convolutional Neural Network (CNN) method has been used in vehicle classification research. One of the image classification studies compares the architectures of 8 network layers and 9 network layers using the Convolutional Neural Network (CNN). The image data processing parameters are re-adjusted to 454x454, a batch size of 128, a learning rate of 0.02 and divided into 10 with 5.500.055 iterations each, with a maximum iteration limit of 450000. The data processing results demonstrate the level of accuracy in the nine-layer architecture. Using the Caffe deep learning framework, the network achieves more than 92.2% detection accuracy, which is higher than the experiment on eight network layers [3].

For the vehicle detection range, research is being conducted on the classification of vehicle types using the CNN method with the cascade method by adding LBP, Haar-like, and HOG feature extraction, PCA dimension reduction, and feature fusion processing. The detection range is then filtered by the SVM classifier using multifeatured fusion, and the filtered detection box is used as the CNN detection input, which is then cascaded. In a good driving environment, the accuracy rate is 98.69%. Furthermore, in a complex driving environment, the false detection rate and negligence rate of the vehicle detection method by the cascade method decrease by 1.37% and 0.7%, respectively [4].

The Faster-RCNN, YOLO, and SSD models in Real-Time Vehicle Type Recognition were compared to determine which model has the fastest process and highest accuracy for vehicle recognition. The models used in this study's implementation are YOLO v4, MobilNet v1 for SSD, and Inception v2 for Faster-RCNN. The algorithm was then tested on a GeForce RTX 2080ti device with 450 different images from the training. The F1-Score is used to evaluate the performance of each model, and the unit of measurement for processing speed is Frame Per Second (FPS).

Vehicles are classified into six categories in the study, with up to 3,188 images used as datasets. The YOLO v4 test results are the best. The Faster-RCNN model is the fastest among the RCNN models, but it lacks FPS because it employs CNN, and the SSD processes data quickly, but it employs mobile-v1 and the model is lightweight, resulting in lower accuracy [5]. In another study, we used TensorFlow as a framework to compare the performance of pre-trained models for object detection. Fast R-CNN,

Faster R-CNN, R-FCN, Mask R-CNN, YOLO, and SSD are among the models compared. The study's findings will be used to determine the matrix speed, accuracy, and number prediction of each existing method. COCO MS, PASCAL VOC 2007, and PASCAL VOC 2012 datasets were used. The best level of accuracy obtained is Faster R-FCN, while the best real-time processing rate is SSD [6][7].

The SSD method combined with the Pretrained MobileNet model can improve the SSD's real-time performance. MobileNet V2, which provides six feature maps with different dimensions to detect back-end networks for multi-scale object detection, is also being developed gradually. This method has a significant advantage when used as a model in the development and implementation of embedded systems. Because it not only supports fast data processing but also increased detection accuracy and devices with low-end hardware [8]-[10].

The goal of this research is to develop edge computing systems [8] by improving the system's performance in the calculation and classification of vehicles using the SSD MobileNet V2 model. The system is made up of several devices that work together to improve each other's performance. The system is composed of several devices that cooperate to enhance each other's performance. The camera sensor is used to capture images of vehicles passing by on the road, and the NVIDIA Jetson Nano is used to process videos captured with the SSD model pretrained method, namely MobileNet V2, to classify and count vehicles.

This study is organized in the following sequence: Part B proposes research methods, Part C results and discussion, and Part D contains the study's conclusions.

B. Research Method

The vehicle classification system is designed with edge computing in mind, with camera sensors used to detect vehicles and then classified using MobileNet V2 algorithms. The detection and classification algorithm are connected to the Jetson Nano, and the data is wirelessly transmitted to the server. The website application displays the vehicle classification data after processing data on the server using cloud computing, as shown in Figure 1.

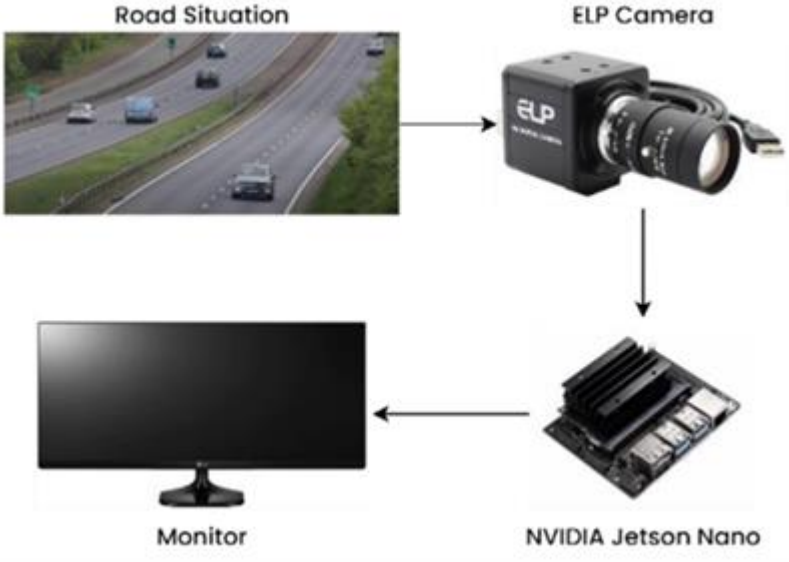


Figure1. ADTS Edge Edge Computing System

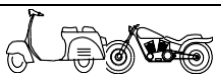
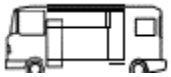
The system is comprised of several devices that collaborate to improve one another's performance. The camera sensor is used to capture images of vehicles passing by on the road, and the NVIDIA Jetson Nano is used to classify and count vehicles using videos captured with the SSD model pretrained method, namely MobileNet V2.

In this study, image data was collected after the hardware circuit was built, including dataset collection and pre-processing, such as quality adjustment and image annotation. Then, using MobileNet V2, create a pre-trained model, validate it, and display it on the website. The following is a detailed explanation of the work order:

1. Dataset Collection

The dataset collected consists of jpeg/jpg images of vehicles from groups 1, 2, 3, 4, 5a, 5b, 6a, 6b, 7a, 7b, and 7c. Each of these groups is searched for using keywords based on the type of vehicle, namely group 1, which includes motorcycles and three-wheeled vehicles, group 2, which includes sedans and jeeps, group 3, which includes medium passenger transportation such as elf cars, and group 4, which includes cars. Group 5a is a small bus (medium bus), group 5b is a large bus, group 6a is a 2-axle light truck, group 6b is a 2-axle medium truck, group 7a is a 3-axle truck, group 7b is a trailer truck, group 7c is a semi-trailer truck, and group 8 is non-motorized vehicles like bicycles and tricycles. According to the Circular Letter of PKRMS Manual No. 04/M/BM/2021, the following Table 1 shows the class of vehicles.

Table 1. Vehicle type group

Group	Vehicle Type Group	Transportation type
1	Motorcycles, 3-wheeled vehicles	
2	Sedan, jeep, station wagon	
3	Medium passenger transport	
4	Pickups trucks, micro trucks, and delivery cars	
5a	Small bus	
5b	Big bus	
6a	2 axle light truck	
6b	2 axle medium truck	
7a	T 3 axle truck	
7b	Trailer truck	

7c	Semi-trailer truck	
8	Non-motorized vehicles	

A vehicle type dataset in the form of images is required to create a classification system and automatic vehicle counting using the MobileNet V2 model (i.e. motorcycles, cars, buses, trucks, and non-motorized vehicles). The datasets were obtained through Google search and Kaggle.

2. Pre-Processing

The dataset consists of 19,516 raw images. To achieve the best results, images of vehicles with front, tilted (45 degrees), and side (90 degrees) angles were collected. The images are first processed so that they have the same dimensions of 640x640 pixels and the jpg extension type. The dataset is annotated or labeled in Pascal VOC format, which includes information about the image file name, label name, and bounding box area coordinates. 70% of the dataset will be used for training and 30% for model validation. Table 2 displays statistics on the distribution of each label or class in the dataset.

Table 2. Dataset label distribution [8]

Label	Type of Vehicle	Amount	Percentage
1	Motorcycle	1456	7.46%
2	Car	4079	20.90%
3	Car	1559	7.99%
4	Car	2807	14.38%
5a	Bus	1061	5.44%
5b	Bus	1922	9.85%
6a	Truck	1655	8.48%
6b	Truck	2031	10.41%
7a	Truck	1013	5.19%
7b	Truck	54	0.28%
7c	Truck	433	2.22%
8	Non-motorized vehicle	1446	7.41%
	Total	19516	100%

The inferencing process will be applied to the input in the form of road traffic video. If a vehicle is detected during the inference process, the MobileNet V2 model will classify it and create a bounding box for it. The tracker is then used on the bounding box to assign an ID to each bounding box. The number of vehicles counted only within the bounding box with an ID. The result of all previous stages is a video containing the classification results and the number of vehicles.

3. MobileNet V2 Model

In this research we develop MobileNet V2 model in edge computing. The training, validation, and export model processes are all part of the process of creating this MobileNet V2 model. The TensorFlow Object Detection API is used in the MobileNet

V2 model, where there is already a pre-trained MobileNet V2 model on the Model Zoo list belonging to the TensorFlow Object Detection API.

MobileNet V2 is a Convolutional Neural Network (CNN) architecture that can be used to overcome the need for large amounts of computing resources. Google researchers created a CNN architecture that can be used for mobile phones, as the name suggests. The primary distinction between MobileNet V2 and CNN architectures is that they both typically employ a convolution layer or a layer with a filter thickness equal to the thickness of the input image [10] as shown Table 3.

Table 3. Configuration MobileNet V2

Type/ Stride	Filter Shape	Input Size
Conv/ S2	3 x 3 x 3 x 32	224 x 224 x 3
Conv dw /S1	3 x 3 x 32 dw	112 x 112 x 32
Conv /S1	1 x 1 x 32 x 64	112 x 112 x 32
Conv dw /S2	3 x 3 x 64 dw	112 x 112 x 64
Conv /S1	1 x 1 x 64 x 128	56 x 56 x 64
Conv dw /S1	3 x 3 x 128 dw	56 x 56 x 128
Conv /S1	1 x 1 x 128 x 128	56 x 56 x 128
Conv dw /S2	3 x 3 x 128 dw	56 x 56 x 128
Conv /S1	1 x 1 x 128 x 256	28 x 28x 128
Conv dw /S1	3 x 3 x 256 dw	28 x 28 x 256
Conv /S1	1 x 1 x 256 x 256	28 x 28 x 256
Conv dw /S2	3 x 3 x 256 dw	28 x 28 x 256
Conv / s1	1 x 1 x 256 x 512	14 x 14 x 256
5 x Conv dw / s1 Cov / s1	3 x 3 x 512 dw	14 x 14 x 512
	1 x 1 x 512 x 512	14 x 14 x 512
Conv dw / s2	3 x 3x 512 dw	14 x 14 x 512
Conv / s1	1 x 1 x 512 x 1024	7 x 7 x 512
Conv dw / s2	3 x 3 x 1024 dw	7 x 7 x 1024
Conv / s1	1 x 1 x 1024 x 1024	7 x 7 x 1024
Avg Pool / s1	Pool 7 x 7	7 x 7 x 1024
FC / s1	1024 x 1000	1 x 1 x 1024
Softmax	Classifier	1 x 1 x 1000

The personal computer specifications for the training process are shown in Table 4.

Table 4. Computer Specification [8]

Operating System	CentOS Stream
CPU	AMD Ryzen 9 5950x
GPU	GeForce RTX 3080 10 GB
Memory	32 GB
TensorFlow	2.8.0
Python	3.8.10

The MobileNet V2 parameter training scenario changes are used to evaluate the ADTS system's performance in edge computing. The MobileNet V2 parameters are Batch size 10, Num Classes 12, Num steps 200000, and learning rate change. The learning rate is observed on changes in loss and model accuracy during the training process. 0.001 and 0.005 are the learning rates used. Table 5 shows the variables used in the training process.

Table 5. The MobileNet V2 parameter changes training scenario

Scenario	Batch Size	Num Classess	Learning Rate	Num Steps
1	10	12	0.001	200000
2	10	12	0.002	200000
3	10	12	0.003	200000
4	10	12	0.004	200000
5	10	12	0.005	200000

The training results are evaluated using several parameters, including loss, precision, and recall. Precision and recall are calculated using the following equations (1) and (2):

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

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

While TP stands for True Positive, FP stands for False Positive, and FN stands for False Negative.

Precision is a parameter used to determine the level of accuracy between the ground truth (real data) and the model's prediction results. Recall is used to determine how good a positive value is discovered. The total loss value is the sum of the losses accumulated during the training process. Average Precision (AP) is a metric used to assess the accuracy of object detectors. Average Precision computes the average precision for recall values ranging from 0 to 1.

Model MobileNet V2 calculation and accuracy are followed by the implementation of the ADTS system. A USB camera was used to record traffic conditions in this study. A 4K USB Camera was used in this video. The experiment setup video was shot on Manyar Kertoarjo in Surabaya, East Java, Indonesia. The system processes the results of the recording of road traffic conditions to perform operations to classify vehicle types and calculate the number of vehicles [8].

The ADTS system is tested using road traffic video as input. The video was shot during the day, during peak traffic hours. The video recording setup consists of the following components: camera, USB extension cable, and laptop. The camera is set up on a 425-centimeter-high pole. The camera pole is placed on a street lighting pole that can support the camera pole in the sidewalk area. The camera is angled towards the road at about 45 degrees. The video used in the testing process lasts 2 minutes and 16 seconds [8].

To avoid exhausting all GPU memory during the classification and counting process, the video used for testing is only about 2 minutes long. Although the video for the testing process is only about 2 minutes long, there are already vehicles in the video such as motorcycles, cars, buses, and trucks that can be used to see the results of the classification using the MobileNet V2 model. Figure 2 depicts the installation of a camera to record road traffic conditions [8].

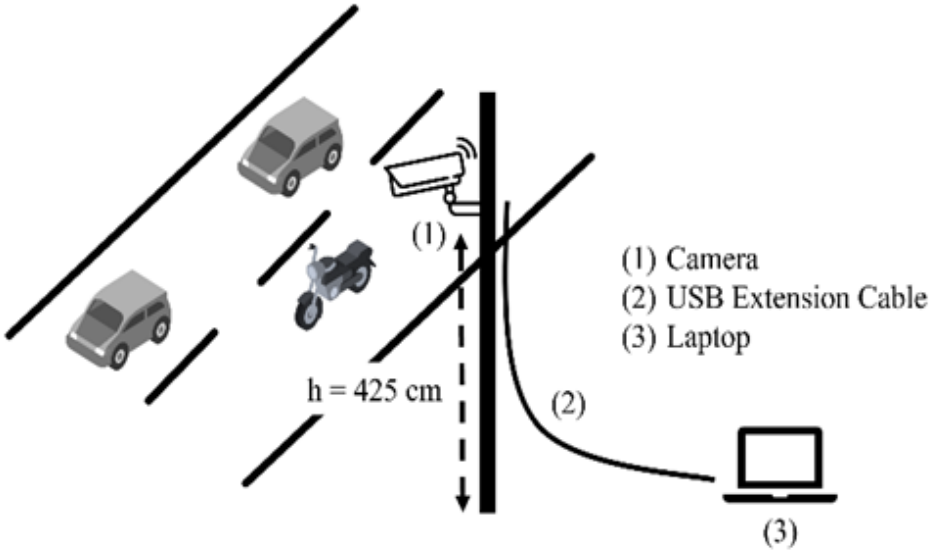


Figure 2. Setup video recording ADTS system [8]

C. Result and Discussion

In testing the MobileNet V2 model to see how much loss there is from the five existing scenarios. The learning rate parameter is transformed into a variable that varies depending on the scenario. Table 6 and Figure 3 show a comparison of the loss values for each learning rate.

Table 6. Loss comparison for each scenario

Step	Loss				
	Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**	**Scenario 5**
20000	0.407834	0.351481	0.298303	0.310144	0.302514
40000	0.333658	0.251744	0.230779	0.213337	0.222002
60000	0.343384	0.303895	0.226949	0.24842	0.224666
80000	0.355177	0.302547	0.268982	0.240288	0.213362
100000	0.374261	0.298286	0.240095	0.211002	0.196653
120000	0.335938	0.324044	0.25849	0.243371	0.263138
140000	0.320827	0.298181	0.242807	0.223319	0.23615
160000	0.40268	0.264515	0.242207	0.209296	0.20852
180000	0.393584	0.28265	0.246572	0.248834	0.219829
200000	0.362773	0.354858	0.240121	0.329267	0.287488

Different learning rates have different loss characteristics at each step, as shown in Table 7 and Figure 3. When compared to other learning rates, a learning rate of 0.001 has a very high loss value. In comparison to the learning rates of 0.002 and 0.004, which have a high spike at steps of 180,000 and 200,000. The learning rate 0.005 has a very low loss value in the first step, but it rises to 120,000 and then again to 180,000 and 200,000. And the learning rate of 0.003 has a relatively stable decrease in the loss value.

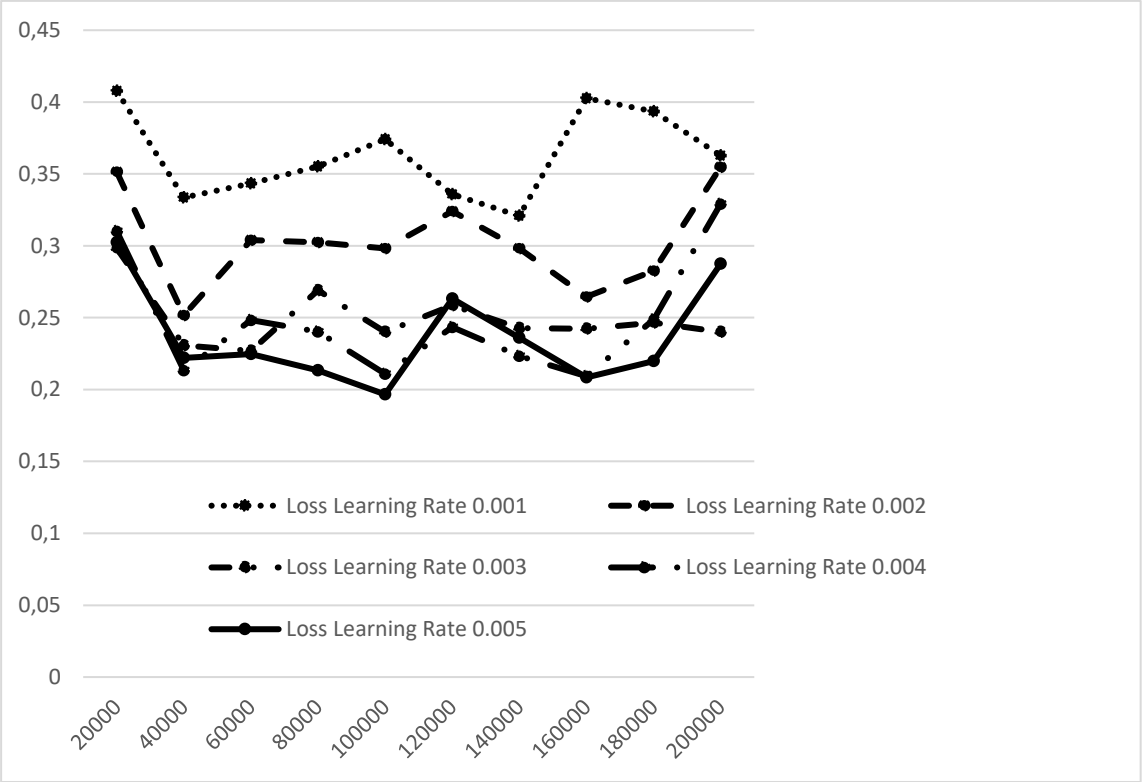


Figure 3. MobileNet V2 loss

Table 7. Comparison of Scenario Validation

Parameter	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
Average Precision	0.6875	0.761	0.7408	0.7637	0.7955
Recall	0.8455	0.8585	0.8664	0.8730	0.8787
Total Loss	0.3680	0.3309	0.3187	0.3086	0.3045

The training results, in the form of loss values, are used to validate the ADTS system's vehicle classification. Vehicle classification validation was performed 5 times based on the number of existing scenarios, and precision and recall values were obtained. Table 7 and Figure 4 show the validation data. The horizontal value shown scenarios 1-5 and the vertical value shown the probability performance system that are average precision, recall and total loss.

Different scenarios result in different parameters, as shown in Table 7 and Figure 4. When compared to other scenarios, scenario 1 produces the lowest average precision and recall and the highest loss. When compared to other scenarios, scenario 5 produces the highest average precision and recall as well as the lowest loss.

The validation test makes use of a video that was captured during the validation data collection stage. The video, which was originally one hour and one minute and sixteen seconds long, was later cut to four minutes and 32 seconds. The video fragment results will be used as validation. The server handles the compilation of the model on the video. The observation process is carried out by comparing classification results with threshold scores of 0.4, 0.45, and 0.5.

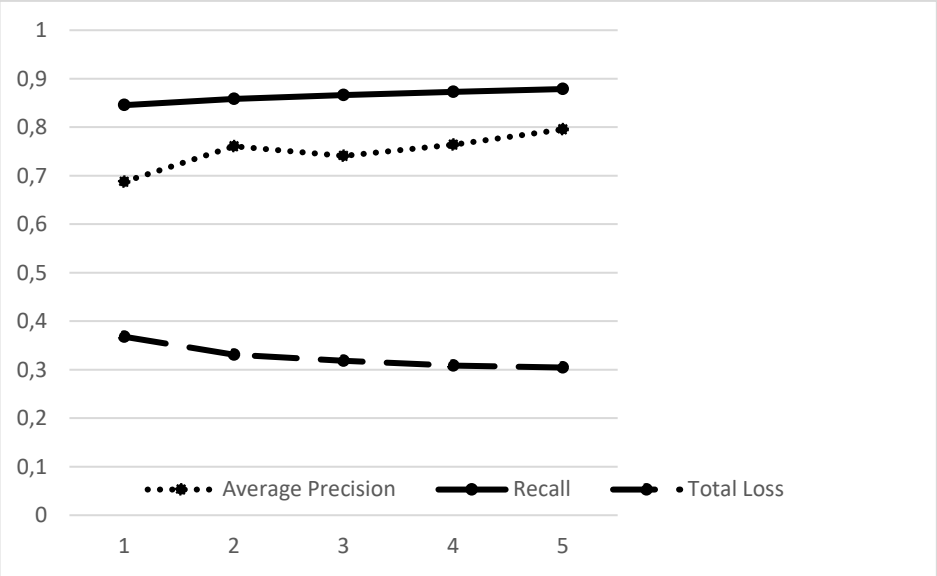


Figure 4. Performance of the system in all scenarios

The ADTS system uses threshold score value to determine vehicle classification predictions, where the system will not generate classification predictions below a predetermined threshold value, such as 0.5 or 50% of the prediction confidence score level. The ADTS system capable of detecting the same number of images at 65 frames per second (FPS) with a minimum score threshold of 40, 45, and 50. Table 8 shows that the MobileNet V2 model can classify and calculate the number 65 FPS in 52 seconds based on the type of vehicle.

Table 8. Comparison of Scenario Validation

Minimum Score Threshold		04		0.45		0.5	
Type of Vehicle	Actual Vehicle Count	FPS	Time(s)	FPS	Time(s)	FPS	Time (s)
Motorcycle	140	65	71	65	70	65	52
Car	69						
Bus	1						
Truck	2						
Non-motorized vehicle	0						

The Inception model's inference process receives only 5 FPS, which is slow. This is a trade-off from the Inception model, in which the model is slower to classify the object but produces good classification results[8]. The MobileNet V2 model can better classify vehicle types with a 65 FPS inference process. This MobileNet V2 final result outperforms the Inception model.

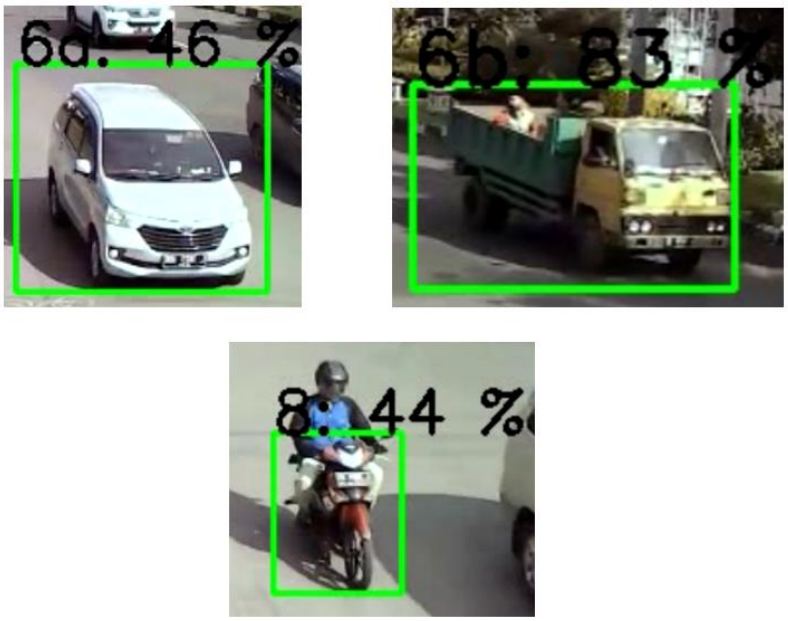


Figure 5. Video Classification Test Detection Results

A green bounding box in Figure 5 indicates the detected vehicle class, confidence score, and vehicle type class. The example of writing 6a indicates that the vehicle is detected as being from class 6a, and a percentage value of 46% indicates that the existing model detects the vehicle as being from class 6a and 46% of the time. The vehicle is detected as being from class 6b, and a percentage value of 53%. The MobileNet V2 model can better classify vehicle types.

D. Conclusion

Vehicle clasification with the MobileNet pre-training model was used in this study to detect cars, motorcycles, buses, trucks, and non-motorized vehicles. The model is used to optimize the model through repeated experiments on the minimum score threshold of hyperparameters with 5 scenarios and their comparisons. The results of the MobileNet model scenario 5 have the lowest loss value of the five scenarios. The MobileNet V2 model can better classify vehicle types with a 65 FPS inference process.

E. Acknowledgment

We are grateful to the Vocational Ministry of Education and Culture for funding research grants 093.SPK/D4/PPK/01.APTV/VI/2022.

F. References

[1] M. F. Barqireza, A. Setyawan, D. U. A. Putra and a. F. Helmi, "Analisis Volume Lalu Lintas Harian Rata-rata (VLHR)," *Marshall,* vol. 2, no. 1, pp. 16-29, January 2014.

[2] R. Mintorogo, S. AS, and S. N. Kadarini, “Evaluasi kinerja dan perbaikan kapasitas jalan Sungai Raya Dalam,” JeLAST J. PWK, Laut, Sipil, Tambang, vol. 2, no. 2, pp. 1–13, 2016.

[3] X. Luo, R. Shen, J. Hu, J. Deng, L. Hu and Q. Guan, "A Deep Convolution Neural Network Model for Vehicle Recognition and Face Recognition," International Congress of Information and Communication Technology (ICICT 2017), Sanya China, 1-2 January 2017.

[4] J. Hu, Y. Sun and S. Xiong, "Research on the Cascade Vehicle Detection Method Based on CNN," *Electronics,* vol. 10, no. 481, pp. 1-19, 2021.

[5] J. A. Kim, J. Y. Sung, and S. H. Park, "Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition," 2020 IEEE Int. Conf. Consum. Electron. - Asia, ICCE-Asia 2020, pp. 8–11, 2020.

[6] S. A. Sanchez, H. J. Romero, and A. D. Morales, "A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework," IOP Conf. Ser. Mater. Sci. Eng., vol. 844, no. 1, 2020

[7] Y. He, Z. Pan, L. Li, Y. Shan, D. Cao, and L. Chen, "Real-time vehicle detection from short-range aerial image with compressed MobileNet," Proc. - IEEE Int. Conf. Robot. Autom., vol. 2019-May, pp. 8339–8345, 2019.

[8] Imam Rifai, Arasy Dafa Sulistya Kurniawan, Haniah Mahmudah, Rahardhita Sudibyo, Mochammad Zen Samsono Hadi and Nihayatus Sa'adah, "Automatic Vehicle Classification and Counting System Using Inception Model", 2022 14th International Conference on Information Technology and Electrical Engineering (ICITEE), 2022

[9] Ilham Dwi Pratama, Hani'ah Mahmudah, Rahardhita Widyatra Sudibyo, "Design and Implementation of Real-time Pothole Detection using Convolutional Neural Network for IoT Smart Environment", 2021 International Electronics Symposium (IES), 2021.

[10] Z. S. Hernanda, H. Mahmudah and R. W. Sudibyo, "CNN-Based Hyperparameter Optimization Approach for Road Pothole and Crack Detection Systems," 2022 IEEE World AI IoT Congress (AIIoT), 2022, pp. 538-543, doi: 10.1109/AIIoT54504.2022.9817316

[11] Y. C. Chiu, C. Y. Tsai, M. Da Ruan, G. Y. Shen, and T. T. Lee, "Mobilenet-SSDv2: An Improved Object Detection Model for Embedded Systems," 2020 Int. Conf. Syst. Sci. Eng. ICSSE 2020, pp. 0–4, 2020.

[12] C. Szegedy, S. Ioffe, Vincent V., and Alexander A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning", in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), 2017

[13] Kaifeng Gan, Dingli Xu, Yimu Lin, Yandong Shen, Ting Zhang, Keqi Hu, Ke Zhou, Mingguang Bi, Lingxiao Pan, Wei Wu, and Yunpeng Liu, "Artificial intelligence detection of distal radius fractures: a comparison between the convolutional neural network and professional assessments", Acta Orthopaedica, 90:4, pp. 394-400, April 2019.

[14] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", arXiv:1506.01497v3, January 2016

[15] R. L. Galvez, A. A. Bandala, E. P. Dadios, R. R. P. Vicerra and J. M. Z. Maningo, "Object Detection Using Convolutional Neural Networks," in TENCON 2018 – 2018 IEEE Region 10 Conference, pp. 2023-2027, 28 October 2018.