

# YOLO-Based Vehicle Detection and Classification for Multi-Lane Free Flow Systems

Nihayatus Sa'adah\*, Faridatun Nadziroh, Rahardhita Widyatra Sudibyo, Haniah Mahmudah,  
Fitri Fitriah, Hari Wahjuningrat Suparno, Muhammad Hendra Syahputra  
*Politeknik Elektronika Negeri Surabaya, Indonesia*

\*Corresponding author: Nihayatus Sa'adah; email: nihayatus@pens.ac.id

Manuscript received 3 May 2026, revised 20 June 2026, accepted 3 July 2026

DOI: <https://doi.org/10.68129/jeis.v1i2.6>

**Abstract** - Multi-Lane Free Flow (MLFF) systems have emerged as a promising approach to reducing congestion at toll gates by enabling uninterrupted toll transactions without requiring vehicles to stop. To support this implementation, this study proposes a YOLOv9-based vehicle detection and classification system optimized using grid search and the SGD optimizer. The proposed system is designed for edge computing deployment and evaluated in a real multi-lane toll environment using an NVIDIA Jetson AGX Orin device. Experimental results show that the optimized model achieves a mean Average Precision at IoU threshold 0.50 (mAP@50) of 92.6%, an F1-score of 92.0%, and an inference time of 70.8 ms, corresponding to 14.1 FPS. These results indicate that the proposed system provides a balanced trade-off between detection accuracy and computational efficiency for edge-based MLFF applications. Unlike previous studies that primarily evaluated vehicle classification models using offline datasets, this study includes implementation and testing in a real multi-lane toll scenario. The findings demonstrate the potential of YOLO-based vehicle detection and classification to support practical MLFF deployment, although further evaluation under more diverse traffic, lighting, and environmental conditions is still required.

**Keywords:** You only look once version 9 (YOLOv9), Vehicle classification, Multi lane free flow (MLFF)

## I. Introduction

High mobility in Indonesia, especially in densely populated areas, supports the Indonesia government in building toll roads. In accordance with KEPMEN PU/PR No. 370/KPTS/M/2007, toll roads are public roads where users are required to pay tolls [1]. Although the construction of toll roads has a positive impact on the speed of mobility, congestion can still occur, especially at toll gates during peak hours and national holidays. Toll roads in Indonesia use the GTO (Automatic Toll Gate) system where users pay via e-toll. However, the existence of toll roads still causes due to problems during payment such as expired e-toll cards, unreadable, and others [2]. In practice, GTO (Automatic Toll Gate) technology still causes payment queues at toll gates. The queue begins when vehicles arrive and stop to tap the GTO sensor. Users must wait for the e-toll balance to be deducted and for the gate bar to open. On the current system, the waiting time averages 2.29 minutes. This condition worsens during peak hours, especially after 5:00 p.m. local time, causing an increase in queues and leading to congestion [3].

To overcome this problem, instructed the Minister of Public Works and Public Housing in 2016 to use an automated system connected to a bank account to reduce congestion. This system is called Multi-Lane Free-Flow (MLFF). Multi Lane Free Flow (MLFF) is the latest payment system that facilitates payment access without having to stop [4]. According to

research in Serbia, the MLFF system can reduce CO<sub>2</sub> levels by 25-45% and NO<sub>x</sub> levels by 32-98%[5]. It has the capacity to eliminate congestion more effectively than current technologies, such as Automatic Toll Gate (GTO)[6].

In order to support the implementation of the MLFF (Multi-Lane Free Flow) system, an automatic and efficient vehicle detection technology is required. Artificial intelligence and deep learning offer a potential approach for detecting and determining vehicle classes in real time. The resulting detection and classification information can subsequently be used to support rapid decision-making within the MLFF system. Bochkovski et al. developed YOLOv4 as an efficient and powerful object detection model that provides a balance between detection accuracy and processing speed. YOLOv4 achieved 43.5% AP and 65.7% AP<sub>50</sub> on the MS COCO dataset while operating at approximately 65 FPS on a Tesla V100 GPU, indicating its suitability for real-time object detection application [7]. Object detection identifies objects in an image using previous visual data, which is then processed and classified into certain categories. The technology in MLFF uses wireless communication with RFID devices that allow unique vehicle identification between readers and tags also for payment [4][8].

To ensure proper classification, vehicle detection systems must have a high degree of accuracy. Currently, various pre-

trained deep learning models can be used for this purpose. However, when selecting such models, it is important to consider the detection performance that best suits the system requirements [6]. Object detection using deep learning with the You Look Only Once (YOLO) method allows this system to be supported by combining the detection and classification processes of objects. With real-time detection capability and lightweight model, YOLO suitable for MLFF concept. YOLO (You Only Look Once) is a real-time object detection method that combines detection and classification processes [9]. There are many versions of YOLO; the latest is YOLOv9, which outperforms other versions. Utilizing the YOLOv9 architecture allows for the efficient and effective detection and classification of vehicles and lanes, supporting the MLFF system in Indonesia.

To maximize vehicle detection performance, the best hyperparameter combination of the model must be determined using a grid search method [10]. Additionally, selecting an optimizer during the grid search process is important for identifying the most suitable optimization method for training the model. These optimization methods play an important role in the model's learning process and pattern recognition [11]. This research uses several optimizers, including Adam, SGD, and RMSprop. Adam (Adaptive Moment Estimation) combines the advantages of momentum and RMSprop when adjusting the learning rate. SGD (Stochastic Gradient Descent) is a classical method that updates weights based on random samples. In contrast, RMSprop adaptively adjusts the learning rate for each parameter by utilizing gradient smoothing [12].

Although vehicle detection and classification for toll payment systems has been studied extensively, many of these studies are limited to single-lane scenarios or require sophisticated computational resources. There is still a substantial gap in the development of a system that can detect multiple classes of vehicles in real time across multiple lanes and be small enough to be installed on edge devices. This research employs the grid search method to optimize a YOLOv9-based solution via hyperparameter tuning. The system's ability to function effectively in Multi-Lane Free Flow (MLFF) conditions enables large-scale, practical implementation in Indonesia's toll road infrastructure, offering precise classification with minimal latency, high FPS, and low inference time.

## II. Related Works

To complete this research, it is necessary to review the relevant literature and previous studies. This step establishes a solid foundation for understanding theories and methods related to deep learning. It also supports the development of reliable vehicle detection and classification system by providing insight into existing approaches and technologies.

Febriansyah, et al [1] proposed a cost-efficient vehicle classification system using YOLO to detect axles (tires) and ResNet to classify vehicle types based on the detection results. Using a dataset of 1,196 vehicle images across five classes, their system achieved 97% classification accuracy, offering a practical solution to reduce operational costs while improving the accuracy of toll classification systems.

Nurul Yudho Fauzan, et al [6] conducted a study on object detection algorithms for vehicle classification at toll gates in Indonesia to support a barrierless payment system. The researchers compared YOLOv4, Faster R-CNN, and SSD. YOLOv4 achieved the best performance, with a weighted average F1-score of 0.880, indicating its strong classification ability across all vehicle classes.

Deepaloke Chattopadhyay, et al [13] conducted a study on real-time vehicle classification on an electronic toll system by comparing two models, namely YOLOv3 and YOLOv3 Tiny. The results showed that the YOLOv3 had 45 FPS and YOLOv3 Tiny had 220 FPS in real-time, this shows that the framing speed of YOLOv3 Tiny is better than YOLOv3. YOLOv3 Tiny is better than YOLOv3 Tiny also has a better mAP than YOLOv3 with 33,34% accuracy. The comparison between studies that have been done before and the system we propose can be described as follows:

TABLE I  
COMPARISON WITH PREVIOUS RESEARCH

| Previous Research | Method                           | Implementation                |
|-------------------|----------------------------------|-------------------------------|
| [1]               | YOLOv4 + FastRCNN                | No<br>Only prototype          |
| [6]               | YOLOv9 + Resnet 50               | No<br>Only image test         |
| 13                | YOLOv3 and YOLOv3 Tiny           | No<br>Only image test         |
| Proposed System   | YOLOv9 + Grid Search + Optimizer | Yes<br>NVIDIA Jetson AGX Orin |

## III. Proposed Methodology

The overall system block diagram is shown in Fig 1. This research focuses on designing Artificial Intelligence with the pre-trained YOLOv9 model to custom model with custom dataset. Custom models in this research will work on Jetson Orin with integration on several components shown in the blue block in the figure. Through Jetson Orin, this custom model will detect 5 vehicle groups using a camera. The camera will be placed on a pole with a maximum height of 5,1 m toll gate according to BPJT design standards. The model will do real-time video framing when detecting and classifying vehicles, the results of this detection and classification will be the basis of the payment system on MLFF.

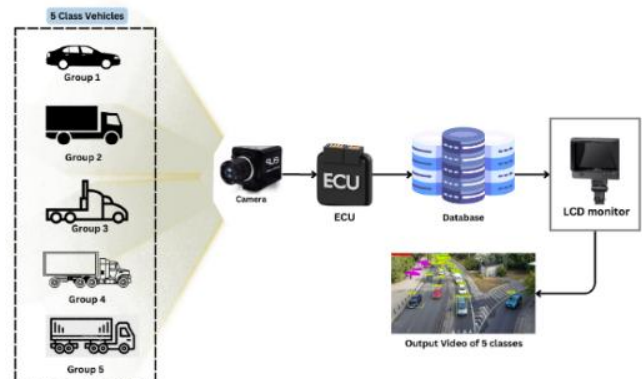


Fig. 1. System design

In the proposed methodology, there are several stages as shown Fig 2. This stage has similarities with several research, but this study uses the YOLO model with a relatively new

version, namely version 9. This research also enables vehicle detection and classification on low-spec devices to measure the performance quality of custom models.

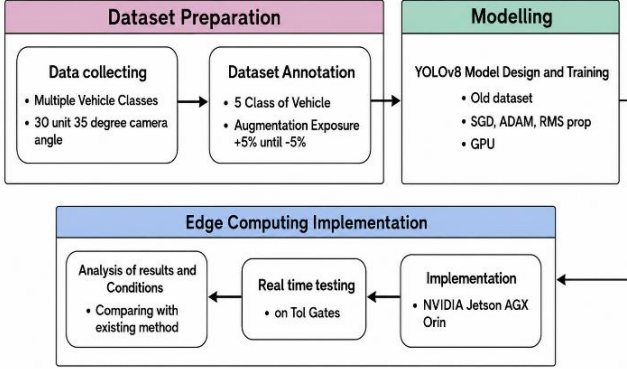


Fig. 2. Research methodology

### A. Data Preparation

Custom datasets for used to train the model were obtained from several sources, including taking videos directly at toll gate then framing them every 5 seconds and capturing data through platforms such as Youtube. The Custom datasets consist annotation of five classes according to BPJT [14], these classes are shown in Fig. 6.

TABLE II  
TOLL VEHICLES CLASS

| Class   | Vehicle Types                     |
|---------|-----------------------------------|
| Group 1 | Sedan, SUV, pickup truck, and bus |
| Group 2 | Two axle truck                    |
| Group 3 | Three axle truck                  |
| Group 4 | Four axle truck                   |
| Group 5 | Five axle truck                   |

This dataset considers the angle of video capture from the side, ranging from an estimated angle of 30 to 35 degrees to ensure visibility of the axles and exposure augmentation about -5% to 5%. Example of annotated frame are shown in Fig 3. The total dataset is 10000 images before augmentation, with a division of 87% train set, 9% valid set, and 4% test set. The details of the dataset can be seen in Table III.



Fig. 3. Annotation examples more than one classes

TABLE III  
DATASET LABELLING DETAILS

| Label | Vehicle Type                             | Count | Percentage |
|-------|------------------------------------------|-------|------------|
| G1    | Sedan, jeep, pickup/small truck, and bus | 2000  | 20%        |
| G2    | 2-axle truck                             | 2000  | 20%        |
| G3    | 3-axle truck                             | 2000  | 20%        |
| G4    | 4-axle truck                             | 2000  | 20%        |
| G5    | 5-axle truck                             | 2000  | 20%        |

### B. Modelling

The pre-processed data will then be processed and triangulated using the pre-trained YOLOv9 model. Before training with a high number of epochs, the grid search method is used to determine the optimal combination of hyperparameter for time efficiency. The parameters to be tuned are optimizer and batch size that shown in Table IV.

TABLE IV  
GRID SEARCH SCENARIO

| Epoch | Optimizer | Batch Size | Learning Rate               |
|-------|-----------|------------|-----------------------------|
| 10    | RMSprop   | 8          | 16                          |
|       | SGD       | 8          | 16                          |
|       | Adam      | 8          | 16                          |
|       |           |            | lr0 = 0.001<br>lrf = 0.0001 |

After that, the optimal hyperparameter is trained using the various learning rate at high epoch. The final optimal model is evaluated using the following metrics and losses:

- Box loss : Measures how well the bounding box prediction approaches the ground truth.
- Cls loss : Measures the error in class prediction.
- Dfl loss : Optimizes the prediction of bounding box coordinates with greater precision.
- mAP : Measures the average detection performance across all vehicle classes by computing the mean of Average Precision (AP) values derived from the precision-recall curve at a specific Intersection over Union (IoU) threshold. In this study, mAP@50 is used, where a prediction is considered correct if its IoU with the ground-truth bounding box is at least 0.50.

$$mAP@50 = \frac{1}{C} \sum_{c=1}^C AP_c @ IoU 0.50 \quad (1)$$

- Precision : Measures how many predictions are correct
- $$Precision = \frac{TP}{TP + FP} \quad (2)$$

- Recall : Measures how many objects actually exist
- $$Recall = \frac{TP}{TP + FN} \quad (3)$$

- F1 Score : Comparison between recall and precision.
- $$F_1 \text{ Score} = \frac{2(Precision \times Recall)}{Precision + Recall} \quad (4)$$

### C. Edge Computing Implementation

Edge computing enables local data processing, reducing latency and improving system response. Edge computing will be implemented on an NVIDIA Jetson AGX Orin device with the specifications listed in the Table V [15]. The system will be tested in real time at the toll gate using Logitech MX Brio camera. The test will evaluate the system's accuracy and speed in live vehicle detection.

Fig 4 shows the installation location of the camera and monitoring pole on the left side of the road, its above 29 meters before the toll gate, This setup is used to record passing vehicles at a 30-35 degrees angle. This location was chosen to obtain the optimal viewing angle for the vehicle detection and classification process. While Fig 5 illustrates the implementation process until the detection and classification of each frame is complete. The test result are videos that will be analyzed and compared with the existing method to assess the system's overall superiority [16].

TABLE V  
JETSON AGX ORIN SPECIFICATION

|                 |                                         |
|-----------------|-----------------------------------------|
| <b>GPU</b>      | 2048-core NVIDIA with 64 Tensor Cores   |
| <b>CPU</b>      | 12-core Arm® Cortex®-A78AE 64-bit ARMv8 |
| <b>RAM</b>      | 32GB 256-bit                            |
| <b>Storage</b>  | 64 GB eMMC 5.1                          |
| <b>Software</b> | Linux                                   |
| <b>O/S</b>      | Ubuntu 20.04                            |



Fig. 4. Real-time testing location

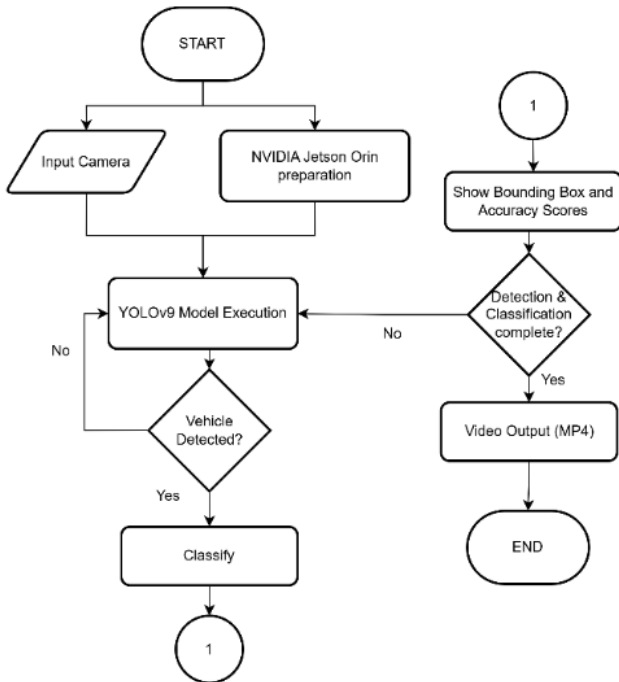


Fig. 5. Implementation Flowchart

#### IV. Experimental Result

##### A. Training Result

The grid search results show that SGD with a batch size of 8 provides the best performance, with an mAP @50% of 0.911, a precision of 0.833, and a recall of 0.839. This value is relatively high. Additionally, this combination produces the lowest total loss compared to tuning with other parameters. Adam optimizer tuning shows quite good performance, while RMSprop produces the lowest results. Therefore, SGD with a batch size of 8 is chosen as the optimal hyperparameter combination. The presence of Grid Search results helps the subsequent training process on a dataset without requiring more time to estimate good parameters. This combination will undergo further training at higher epochs.

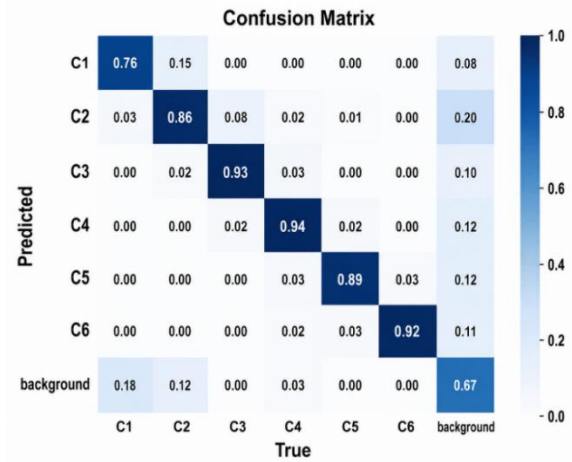


Fig 6. Confusion matrix result

These hyperparameter combination then will trained with epoch 100 that shown in Table VI, the results of the model training are quite optimal compared to grid search prediction. The SGD optimizer with a batch size of 8 is optimized by varying the initial and final learning rates. According to Table VI, the best model has an initial learning rate of 0.017 and a final learning rate of 0.0025. Based on metric evaluation, this model achieves an mAP @50% of 0.926, a precision of 0.914, a recall of 0.867, and an F1 score of 0.920. These values are high compared to those of other models. Meanwhile, when judged by the loss function, this model ranks second from the bottom. However, considering the metric evaluation, this model has the optimal balance compared to other models. Therefore, the model with  $lr_0 = 0.017$  and  $lrf = 0.0025$  is selected for implementation.

The confusion matrix results in Fig. 6 demonstrate the model's predictive ability for the five classes (G1–G5). In general, the model correctly identified most of the classes. Classes G2, G3, G4, and G5 showed high accuracy, with values ranging from 0.86 to 0.95. This model is optimal for correctly detecting these class groups. However, the value for correctly detecting G1 is slightly low at 0.79, and there are incorrect classifications as G1 (0.17) and G2 (0.04). This indicates that the model occasionally misclassifies this class, though the misclassification rate is quite low. Overall, the model's accuracy exceeds 70%.

TABLE VI.  
TRAINING RESULT

| Optimizer | Lr0    | Lrf    | Batch Size | mAP   | Precision | Recall | F1score | Box Loss | Class Loss | Dfl Loss |
|-----------|--------|--------|------------|-------|-----------|--------|---------|----------|------------|----------|
| SGD       | 0,001  | 0,0001 | 8          | 0,922 | 0,888     | 0,856  | 0,9047  | 0,2546   | 0,1299     | 0,8438   |
|           | 0,0001 | 0,001  | 8          | 0,923 | 0,89      | 0,836  | 0,9062  | 0,4369   | 0,3504     | 0,9718   |
|           | 0,017  | 0,0025 | 8          | 0,926 | 0,914     | 0,867  | 0,9200  | 0,2742   | 0,1405     | 0,8561   |
|           | 0,018  | 0,003  | 8          | 0,925 | 0,897     | 0,884  | 0,9108  | 0,2764   | 0,1461     | 0,8542   |

### B. Edge Computing Implementation Result

The training model has a format in .pt. To test whether the model works well according to the metric evaluation and loss results, implementation is required on the Jetson Orin device. This deployment is an inference process that shows whether the model works well on low-spec devices. Deployment of this model can be done directly using a camera and indirectly using video. The camera used is a Logitech MX BRIO camera.

The following are the results of the best custom model deployment showed at Table VII. The Fps result of 14,1 and inference time of 70,8 ms shows the performance of the real-time system. This FPS value is quite good for real-time monitoring. An Inference time below 100 ms also shows that the prediction process is running quite fast in a multi lane environment and can detect more than one vehicle optimally. The following are some detection results using the custom model created.

Table VII provides a comparison of the performance of the existing system and some other research. Study [1] used the YOLOv4 model and achieved fairly good accuracy with mAP of 82.97% and precision of 89.1%. However, the system was not implemented in real-time or on edge devices, so it was not that deployable in real life. Study [6] used a combination of YOLOv9 with ResNet50 and had the best mAP of 95.83% and precision of 97.14% but with a very slow inference time of 1.137 seconds per image, which will not be feasible for real-time applications. Moreover, the model was not even tested on a real multi-lane environment. On the other hand,

research [13] contrasted YOLOv3 and YOLOv3 Tiny, in which YOLOv3 Tiny demonstrated stunning FPS (220), which is ideal for real-time applications. However, accuracy was extremely low at 33.34% mAP with no real-world use case in multi-lane environments.



Fig 7. Result detection and classification in multi-lane and vehicle

The presented model, however, employs YOLOv9 in combination with Grid Search hyperparameter tuning and the SGD optimizer to yield an optimized trade-off between accuracy and efficiency. It delivers a mAP of 92.6%, precision of 91.4%, F1-score of 92.0%, and inference time of 70,8 ms (approximately 16.2 FPS). Unlike previous studies, this model was put to the test immediately on a multi-lane toll booth in real-time and showed to be effective under actual conditions. Its best performance and support for edge devices make it an ideal choice for actual implementation on MLFF-based tolling systems.

TABLE VII.  
COMPARISON FINAL RESULT BETWEEN RELATED WORKS

| Study    | Method                               | FPS        | Inference Time (ms)    | mAP (%)                 | Precision | Recall | F1-Score | Multi Lane Test | Implementation                           | Notes                                                            |
|----------|--------------------------------------|------------|------------------------|-------------------------|-----------|--------|----------|-----------------|------------------------------------------|------------------------------------------------------------------|
| [1]      | YOLOv4                               | 35         | -                      | 82,973%                 | 89,1%     | 89,15  | 87,7%    | X               | No, only testing with image from dataset | High accuracy, not real-time or edge-based.                      |
| [6]      | YOLOv9 + Resnet 50                   |            | 1,1370 sec/image       | 95.83%                  | 97.14%    | 95,83% | 96,44%   | X               | No, only testing with image from dataset | Accurate but slow and complex, no real-time test.                |
| [13]     | YOLOv3 and YOLOv3 Tiny               | 220 (Tiny) | 104.41<br>27.34 (Tiny) | 33.34% (tiny)<br>55.56% | 98.5      | 100    | -        | X               | No, only testing with image from dataset | Very fast, low mAP, dataset-only test.                           |
| Proposed | YOLOv9 + Grid Search + SGD optimizer | 14,1       | 70,8                   | 92,6%                   | 91,4%     | 86,7%  | 92%      | √               | Yes, real-time testing at toll gates.    | Balanced accuracy, real-time edge deployment & detect multi-lane |

## V. Conclusion

In this study, the use of deep learning as a supporter of the Multi Lane Free Flow system in detecting and classifying vehicles in real-time on low-spec devices has proven to be efficient. This is evidenced by the good model in the metric evaluation and loss function after training and validation. The best model is the model with the most optimal hyperparameters assessed through the metric evaluation and loss function. Before training with a larger iteration scale, Grid Search is performed to make it easier to find the best Hyperparameter (tuning). Furthermore, based on the Grid Search reference, the model is trained with a new dataset. The result is the best model with a combination of SGD Optimizer parameters, Batch Size 8, Lr0 = 0.001, Lrf = 0.0001, and Epoch 100 with an accuracy of 92,6%. This is evidenced by the good model when deployed on low-spec devices. This research can still be developed for research related to the Multi Lane Free Flow system. It is hoped that this research will be useful for other researchers and toll road users.

## Conflict of Interest

The authors declare that they have no conflict of interest regarding the publication of this paper.

## References

- [1] A. A. Febriyansah and G. Kosala, "Classification of Vehicle Classes at Toll Gates Using YOLOv9 and Resnet50," in *Proceedings of the 2025 International Conference on Advancement in Data Science, E-learning and Information System (ICADEIS)*, Bandung, Indonesia, 2025, pp. 1-6, doi: 10.1109/ICADEIS65852.2025.10933473.
- [2] A. M. S. Sufanir, D. R. Shaumi, and V. R. A. Siregar, "Pemeriksaan Standar Pelayanan Gerbang Tol Pasteur 1 Ditinjau Dari Kecepatan Transaksi," *Jurnal Rekayasa Sipil (JRS-Unand)*, vol. 18, no. 3, p. 194, Dec. 2022, doi: 10.25077/jrs.18.3.194-201.2022.
- [3] N. Mujahidah, H. R. Azmi, A. N. Zaman, and F. Lestari, "Analysis of Automatic Toll Gate Payment Queue System," in *Proceedings of the Second Asia Pacific International Conference on Industrial Engineering and Operations Management*, Surakarta, Indonesia, September 14-16, 2021.
- [4] Harnanda, A. Y., Priyanto, S., & Irawan, M. Z. (2022). "Determining Factors Of Interest In The Use Of Technology Readness Based Multi Lane Free Flow (Mlff)," *International Journal of Economics, Business and Accounting Research (IJEBA)*, vol. 6, no. 4, Dec. 2022. <https://doi.org/10.29040/ijebar.v6i4.7565>.
- [5] Y. A. Saputra and Y. Latief, "Analytical hierarchy process on the implementation of multi lane free flow toll road transaction system in Indonesia," *Edelweiss Applied Science and Technology*, vol. 9, no. 3, pp. 217–227, 2025, doi: 10.55214/25768484.v9i3.5181.
- [6] O. F. Nurul Yudho Fauzan, R. S. Perdana, and T. Adiono, "Development of Vehicle Type Detection System on Multi Lane Free Flow Toll E-Collection," in *Proceedings of 2022 International Conference on Data and Software Engineering, ICoDSE 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 42–46.
- [7] A. Bochkovski, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv preprint arXiv:2004.10934*, 2020.
- [8] A. H. Ranguti, A. Calvin, W. C. Kosasih, R. R. Febriano, D. R. Hermanus, and A. I. Arrahmah, "Enhanced Vehicle Classification via Axle-Based Deep Learning Models," in *Proceedings of 2024 7th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, Yogyakarta, Indonesia, 2024, pp. 137–142, doi: 10.1109/ISRITI64779.2024.10963458.
- [9] J. Vinod, D. T. M. and R. Rajesh, "Interactive YOLO-based Object Detection using a Polygonal Region of Interest for Airborne Surveillance Applications," in *Proceedings of 2024 IEEE Space, Aerospace and Defence Conference (SPACE)*, Bangalore, India, 2024, pp. 901–905, doi: 10.1109/SPACE63117.2024.10668012.
- [10] A. D. Sulistya Kurniawan, Y. Imanda Purnama, A. B. Wicaksono and H. Mahmudah, "Comparative Analysis and Optimization of Deep Learning Models for Object Detection Using Grid Search Hyperparameter Tuning," in *Proceedings of 2024 International Electronics Symposium (IES)*, Denpasar, Indonesia, 2024, pp. 587–592, doi: 10.1109/IES63037.2024.10665797.
- [11] S. Sveleba, I. Katerynychuk, I. Kunyo, V. Brygilevych, V. Franiv and O. Krychevets, "Convolutional Neural Network with Optimized Learning Parameters," in *Proceedings of 2024 14th International Conference on Advanced Computer Information Technologies (ACIT)*, Ceske Budejovice, Czech Republic, 2024, pp. 170–173, doi: 10.1109/ACIT62333.2024.10712630.
- [12] S. Makamo, "Optimization Algorithms in Deep Learning Models for Improving the Forecasting Accuracy in Sequential Datasets with Application in the South African Stock Market Index: A Review," *International Journal of Artificial Intelligence and Machine Learning*, vol. 4, no. 2, pp. 1–8, Jul. 2024, doi: 10.51483/ijaiml.4.2.2024.1-8.
- [13] D. Chattopadhyay, S. Rasheed, L. Yan, A. A. Lopez, J. Farmer and D. E. Brown, "Machine Learning for Real-Time Vehicle Detection in All-Electronic Tolling System," in *Proceedings of 2020 Systems and Information Engineering Design Symposium (SIEDS)*, Charlottesville, VA, USA, 2020, pp. 1–6, doi: 10.1109/SIEDS49339.2020.9106682.
- [14] P. Utomo, L. Fatmawati, and D. Nor Amadi, "Assessment of Toll User Satisfaction in East Java Using Simple Additive Weighting (SAW)," *Journal of Information Systems and Informatics*, vol. 4, no. 2, 2022. doi: <https://doi.org/10.51519/journalisi.v4i2.259>.
- [15] H. Mahmudah, A. S. Aisjah, S. Arifin and C. Arif Prastyantyo, "iYOLOv7-TPE-SS: Leveraging Improved YOLO Model With Multilevel Hyperparameter Optimization for Road Damage Detection on Edge Devices," *IEEE Access*, vol. 13, pp. 79247–79263, 2025, doi: 10.1109/ACCESS.2025.3562612.
- [16] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information," in *Proceedings of European conference on computer vision*, Cham: Springer Nature Switzerland, Feb. 2024, pp. 1–21. doi: <https://doi.org/10.48550/arXiv.2402.13616>.

## Author Biography



artificial intelligence.

**Nihayatus Sa'adah** received the bachelor's degree in Telecommunication Engineering from Politeknik Elektronika Negeri Surabaya, Indonesia. She received the master's degree in Electrical Engineering from Politeknik Elektronika Negeri Surabaya, Indonesia. She is currently affiliated with Politeknik Elektronika Negeri Surabaya, Indonesia. Her research interests include telecommunication systems, wireless communication, the Internet of Things, and applied



artificial intelligence.

**Faridatun Nadziroh** received the bachelor's degree in Telecommunication Engineering from Politeknik Elektronika Negeri Surabaya, Indonesia. She received the master's degree in Electrical Engineering from Institut Teknologi Sepuluh Nopember, Indonesia. She is currently affiliated with Politeknik Elektronika Negeri Surabaya, Indonesia. Her research interests include telecommunication systems, wireless communication, the Internet of Things, and applied



telecommunication systems, and applied machine learning.

**Rahardhita Widyatra Sudibyo** received the bachelor's degree in Telecommunication Engineering from Politeknik Elektronika Negeri Surabaya, Indonesia. He received the master's degree in Electrical Engineering from Institut Teknologi Sepuluh Nopember, Indonesia. He also received the doctoral degree from Okayama University, Japan. He is currently affiliated with Politeknik Elektronika Negeri Surabaya, Indonesia. His research interests include wireless communication, computer networks, telecommunication systems, and applied machine learning.



**Haniah Mahmudah** received the bachelor's degree from the Engineering Physics Department, Institut Teknologi Sepuluh Nopember, Indonesia. She received the master's degree from the Electrical Engineering Department, Institut Teknologi Sepuluh Nopember, Indonesia. She also received the doctoral degree from the Engineering Physics Study Program, Institut Teknologi Sepuluh Nopember, Indonesia. She is currently a Lecturer with the Electrical Engineering Department, Politeknik Elektronika Negeri Surabaya, Indonesia. Her research interests include signal processing, wired and wireless communication, the Internet of Things, and the application of machine learning. She is a member of ASIOTI and MASTEL.



**Fitri Fitriah** received the bachelor's degree from Politeknik Elektronika Negeri Surabaya, Indonesia. Her contribution in this manuscript is associated with vehicle classification data processing, experimental validation, and performance analysis for MLFF-oriented intelligent transportation systems.



**Hari Wahjuningrat Suparno** received the bachelor's degree in Electrical Engineering from Institut Teknologi Sepuluh Nopember, Indonesia. He received the master's degree in Electrical Engineering from Institut Teknologi Sepuluh Nopember, Indonesia. He is currently affiliated with Politeknik Elektronika Negeri Surabaya, Indonesia. His research interests include electrical engineering and telecommunication system.



**Muhammad Hendra Syahputra** is affiliated with Politeknik Elektronika Negeri Surabaya, Indonesia. His work in this manuscript relates to investigation, software implementation, data handling, and visualization of YOLO-based vehicle detection and classification results in a toll-gate environment.