Bigint Computing Journal
Volume 4, Issue 1, 2026, pp. 9-14
ISSN: 3032-5374
DOI: https://doi.org/10.55537/bigint.v4i1.1527
https://journal.aira.or.id/bigint/index

Performance Evaluation of YOLOv8 for Vehicle License Plate Detection Using Standard Object Detection Metrics

Kardandi Alfarizi Siregar [a,*], Bhagaskara Cahyadi [a], Legiman Samosir [a], Supiyandi [b]

[a] Department of Computer Science, Universitas Islam Negeri Sumatera Utara, Medan, Indonesia
[b] Department of Computer System, Universitas Pembangunan Panca Budi, Medan, Indonesia

A B S T R A C T

Vehicle license plate detection is a crucial computer vision task for traffic monitoring, automated parking, and vehicle identification. This study evaluates the performance of a YOLO-based license plate detection system implemented in Python and executed on Google Colab to ensure reproducibility. A public dataset of vehicle images with variations in lighting conditions and viewing angles is used for testing. Performance is assessed using precision, recall, F1-score, mAP@0.5, and mAP@0.5:0.95. The results show a precision of 0.7653 and a recall of 0.6809, yielding an F1-score of 0.7206. The mAP@0.5 reaches 0.7776, while the mAP@0.5:0.95 drops to 0.3572. As a contribution, this work provides a simple and replicable baseline evaluation workflow for YOLO-based license plate detection using standard object-detection metrics. The large gap between mAP@0.5 and mAP@0.5:0.95 indicates that the model often detects the presence of license plates but struggles to localize them precisely under stricter IoU thresholds, highlighting localization sensitivity for small objects under real-world variations. These findings can guide future improvements through dataset diversification, augmentation, and higher-resolution training to enhance bounding box accuracy.

KEYWORD: Google Colab, License Plate Detection, Object Detection, Performance Evaluation, YOLOv8.

* Corresponding author:
Kardandi Alfarizi Siregar ,
Department of Computer Science, Universitas Islam Negeri Sumatera Utara, Medan, Indonesia
Email: kardandi83@gmail.com
Article History: Received: 2026-01-14; Accepted: 2026-01-22

1. INTRODUCTION

Advances in computer vision technology have significantly contributed to automating object identification processes across various sectors, including transportation and security. One prominent application is vehicle license plate detection and recognition, commonly referred to as Automatic License Plate Recognition (ALPR), which reduces reliance on manual identification procedures [1]. Within the context of intelligent transportation systems, ALPR is an essential component for intelligent tolling, access control, security surveillance, and data-driven traffic management [2].

As the number of vehicles continues to increase, the demand for fast and accurate automated detection systems has grown accordingly [3]. Deep learning-based methods have been widely developed because they can adaptively extract features from images and perform robustly under diverse environmental conditions. A popular approach for real-time object detection is You Only Look Once (YOLO), introduced by Redmon et al. [4], which has evolved through multiple variants emphasizing the speed–accuracy trade-off (e.g., YOLOv4 and YOLOv7). In practical deployments, modern YOLO families (e.g., YOLOv8) are widely available through the Ultralytics framework and can be readily integrated with Python for experimentation and reproducibility [5] [6].

Although YOLO has been extensively adopted, license plate detection presents specific challenges due to the relatively small object size, variations in camera viewpoint, illumination changes, motion blur, and potential vehicle occlusion [7]. Large-scale license plate dataset studies indicate that real-world variations such as distance differences, tilt, and weather disturbances significantly affect localization quality and recognition accuracy [8].

Therefore, evaluation should employ metrics that not only assess detection success at a single IoU threshold but also measure localization quality more strictly, such as mAP@[0.5:0.95] [9].

Nevertheless, an explicit research gap remains. Many ALPR studies tend to focus on implementation outcomes or complete pipelines (detection + OCR), and performance reporting is often limited to a single IoU threshold (e.g., mAP@0.5). Such reporting can obscure a key issue for small objects like license plates: bounding-box localization precision under stricter IoU thresholds. Moreover, there are still relatively few evaluation-oriented studies that provide a simple, well-documented, and easily reproducible workflow for YOLO-based license plate detection on public image datasets, while explicitly reporting stricter metrics such as mAP@0.5:0.95 to distinguish detection reliability ("finding the object") from localization accuracy.

Based on this context, this study evaluates the performance of a YOLO-based vehicle license plate detection system implemented in Python and deployed on Google Colab. The research questions include: (1) how the license plate detection system is implemented using YOLO in Python on Google Colab; (2) how well the developed detection system performs; (3) how performance is reflected through precision, recall, F1-score, and mAP; and (4) how effectively the system detects license plates under variations in illumination and viewing angles. The objectives are to implement the system, evaluate its performance using relevant metrics, and analyze detection capability under diverse image conditions. Practically, this work is expected to serve as a concise reference for developing efficient and easily reproducible license plate detection systems.

To maintain the scope of this study, several limitations are applied: the work is restricted to license plate detection only (without character recognition/OCR), uses YOLO without comparison to other methods, relies on static images (not video), and does not discuss detailed model parameter optimization. The evaluation focuses on detection metrics (precision, recall, F1-score, and mAP); therefore, computational time measurement is not examined in depth. Accordingly, the main contribution of this study is to provide a simple and reproducible baseline evaluation for YOLO-based license plate detection on Google Colab, while highlighting the performance gap between mAP@0.5 and mAP@0.5:0.95 as an indicator of localization sensitivity for small objects under varying conditions.

2. METHODOLOGY

This research uses an experimental approach by implementing and testing a vehicle license plate detection system using YOLO based on Python. This experiment focuses on evaluating the detection performance on a vehicle image dataset.

The dataset used in this research consists of 359 images for the training data (train), 45 images for the testing data (test), and 45 images for the validation data (val). These images were obtained from a public dataset source that includes variations in lighting conditions, angles of view, and object distances, which present challenges for the model in detecting license plates under various real-world conditions.

The dataset consists of vehicle images with license plate objects that have been manually labeled. The training data is used to train the YOLO model, while the testing and validation data are used to evaluate the model's performance. Before inference, the dataset undergoes preprocessing, which includes resizing and normalizing the images to match the input requirements of the YOLOv8 model.

The system is implemented using Google Colab as a cloud-based computing environment, with support from the OpenCV, NumPy, and Ultralytics YOLO libraries for training and inference of the YOLOv8 model. The choice of YOLOv8 is based on the availability of a mature implementation, extensive deployment support, and competitive detection performance across various object scenarios [10].

In this study, the YOLOv8 model is fine-tuned using a specific dataset for vehicle license plate detection[11]. In this experiment, the model does not use an existing pretrained model but is trained from scratch with the prepared dataset. This approach was chosen to obtain a more optimal model for detecting vehicle license plates, considering the unique and varied characteristics of the dataset.

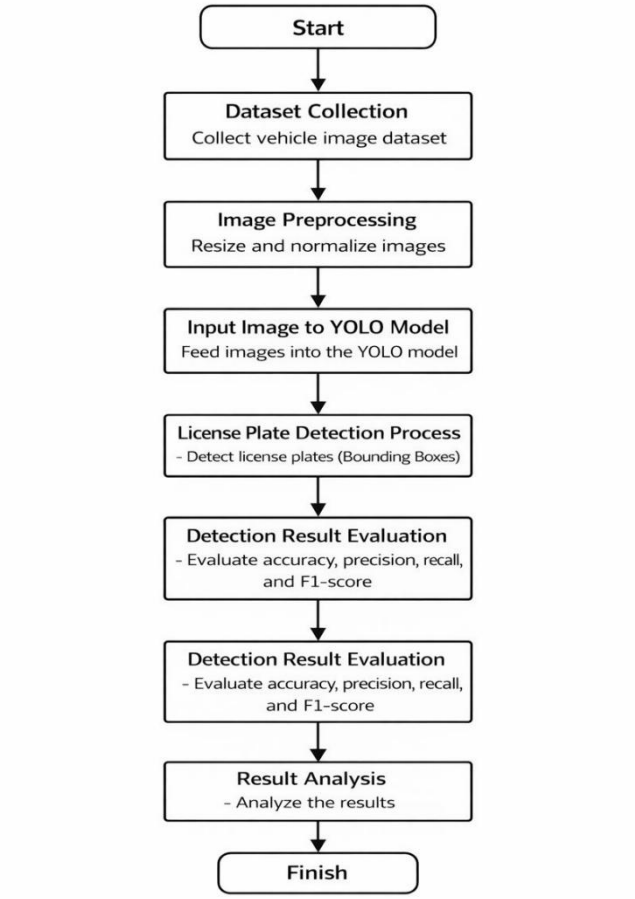

Figure 1. Research Flowchart

The research flow consists of five main steps:

1. Dataset Collection: Data is obtained from a public dataset source focusing on vehicle images with clearly visible license plates.
2. Image Preprocessing: The images are processed with resizing and normalization to match the input requirements of the YOLOv8 model.
3. Inference using YOLO Model: The YOLO model is used to generate bounding boxes that mark the location of license plates in vehicle images.
4. Detection Result Evaluation: The detection results are compared with the ground truth to measure detection accuracy using predefined evaluation metrics.
5. Result Analysis: The detection results are analyzed to identify the system's strengths and limitations.

Performance evaluation is conducted using precision, recall, and mean Average Precision (mAP) metrics. Precision and recall are calculated based on the concepts of True Positive (TP), False Positive (FP), and False Negative (FN). Meanwhile, mAP is calculated following modern object detection evaluation practices, which measure the average area of the precision-recall curve across different Intersection over Union (IoU) thresholds. On the COCO benchmark, AP is calculated at various IoU thresholds and is often reported as mAP@[0.5:0.95], which is the average AP at IoU ranging from 0.50 to 0.95 with a 0.05 interval [12] [13].

3. RESULTS AND DISCUSSION

The discussion below addresses the hardware limitations that impacted the license plate detection system's performance. In this study, the use of Google Colab as the computational environment posed limitations in terms of computational capacity, particularly in processing large and complex datasets. Although the evaluation results were quite good, these limitations may have affected detection accuracy and inference speed, especially for images with extreme conditions.

Testing was conducted on the YOLOv8 model, which was used to detect a single class of objects: vehicle license plates. Performance evaluation was carried out using key metrics: precision, recall, mAP@0.5, and mAP@0.5:0.95, as commonly used in object detection evaluations. Based on the evaluation results, the model shows good performance but still faces challenges under more complex conditions, especially when the IoU threshold is tightened.

Table 1. Metrics and Values

Metrik	Nilai
Precision	0,7653
Recall	0,6809
F1-score (2PR/(P+R))	0,7206
mAP@0.5	0,7776
mAP@0.5:0.95	0,3572

The precision value of 0.7653 indicates that most of the bounding box predictions made by the model are correct. However, the recall value of 0.6809 suggests that a significant number of license plates in the test images were not detected. Based on the precision and recall values, the F1-score of 0.7206 reflects a good balance in performance. Nevertheless, there is still room for improvement, particularly in the model's ability to capture all license plate objects.

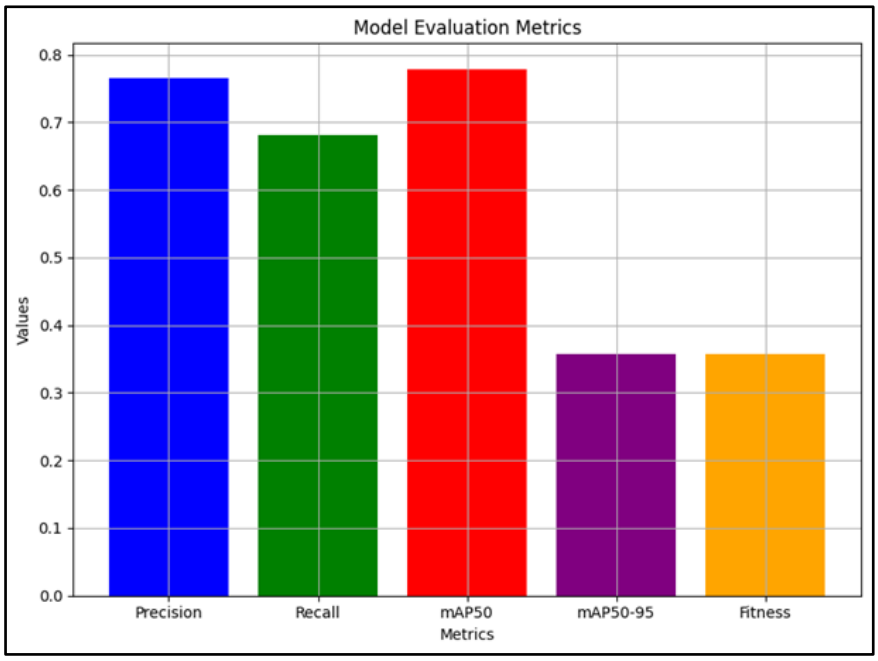

Figure 2. Model evaluation matriks

The mAP@0.5 value of 0.7776 shows good detection accuracy at an IoU threshold of 0.5. However, the drop in mAP@0.5:0.95 to 0.3572 suggests that when evaluation is conducted at stricter IoU thresholds, the precision of bounding box positioning and size remains unstable [14]. This can be attributed to factors such as:

1. Small license plate sizes, which make it harder for the model to make accurate detections.
2. Light reflections that can obscure the visibility of the plates in images.
3. Partial occlusion or cases where parts of the license plate are blocked by other objects.
4. Extreme angles, which make the license plates difficult to read.

In general, mAP@[0.5:0.95] is stricter than mAP@0.5, as it measures localization accuracy across various IoU thresholds [15]. Therefore, a large difference between mAP@0.5 and mAP@[0.5:0.95] often indicates that while the model is consistent in finding objects, its bounding box precision still needs improvement, particularly for small objects like vehicle license plates [16].

Figure 3. Successful License Plate Detection

Based on the results obtained, performance improvements can be directed towards several areas:

1. Expanding and diversifying the dataset, to include various conditions such as daytime and nighttime, rain, long-distance shots, and extreme angles to increase the model's robustness in real-world scenarios.
2. Relevant data augmentation, such as adjusting brightness/contrast, adding blur, and changing the perspective of images, to make the model more adaptive to different conditions.
3. Re-training with higher input resolution, to improve localization accuracy for small objects like license plates, which often require more detailed image information for accurate detection.
4. Hyperparameter optimization and more robust training strategies, to improve the model's resilience against a wider range of real-world conditions and enable better performance with more complex images.

4. CONCLUSION

The vehicle license plate detection system using YOLO based on Python was successfully implemented in the Google Colab environment and demonstrated the ability to automatically generate bounding boxes around license plates in vehicle images.

The evaluation results indicate that the model performs relatively well, with a precision of 0.7653, recall of 0.6809, F1-score of 0.7206, mAP@0.5 of 0.7776, and mAP@0.5:0.95 of 0.3572. The mAP@0.5 value is relatively high, indicating good detection accuracy at the IoU threshold of 0.5. However, the mAP@0.5:0.95 value shows a noticeable decline, suggesting that localization accuracy at stricter IoU thresholds needs improvement.

These findings demonstrate that while YOLO is capable of detecting license plates reliably, especially in less complex conditions, there is still room for improvement in terms of detection precision, particularly under challenging conditions, such as small object detection, occlusion, and extreme angles.

This study provides a baseline evaluation of YOLO performance for simple and easily replicable license plate detection systems. However, the study's limitations, such as reliance on Google Colab's computational resources and challenges in detecting small objects, should be addressed in future research.

Future research should focus on optimizing model parameters, increasing dataset diversity, and expanding testing to include real-time video data or more challenging scenarios to bring license plate detection systems closer to real-world implementation needs. Additionally, improving input resolution, utilizing data augmentation techniques, and experimenting with multi-scale detection strategies can further enhance the model’s performance in detecting small objects like license plates under various environmental conditions.

REFERENCES

[1] L. Satya, M. R. D. Septian, M. W. Sarjono, M. Cahyanti, and E. R. Swedia, "Sistem pendeteksi plat nomor polisi kendaraan dengan arsitektur YOLOv8," *Sebatik*, vol. 27, no. 2, pp. 753–761, Dec. 2023. https://doi.org/10.46984/sebatik.v27i2.2374

[2] L. Alashrafi, R. Badawood, H. Almagrabi, M. Alrige, F. Alharbi, and O. Almatrafi, "Benchmarking lightweight YOLO object detectors for real-time hygiene compliance monitoring," *Sensors*, vol. 25, no. 19, Oct. 2025. https://doi.org/10.3390/s25196140

[3] N. H. Harani, C. Prianto, and M. Hasanah, "Deteksi objek dan pengenalan karakter plat nomor kendaraan Indonesia menggunakan metode convolutional neural network (CNN) berbasis Python," *Jurnal Teknik Informatika*, vol. 11, no. 3, pp. 47–53, Aug. 2019. https://doi.org/10.23960/jitet.v11i1.2897

[4] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of YOLO algorithm developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022. https://doi.org/10.1016/j.procs.2022.01.135

[5] M. R. Joyonegoro and E. Setyati, "Deteksi dan pengenalan dua variasi plat nomor kendaraan bermotor di Indonesia dengan variasi waktu dan pencahayaan memanfaatkan YOLOv8 dan CNN," *KONVERGENSI*, vol. 20, no. 1, pp. 11–17, Jan. 2024. https://doi.org/10.24912/jiksi.v10i1.17852

[6] R. Sapkota and M. Karkee, "Ultralytics YOLO evolution: An overview of YOLO26, YOLO11, YOLOv8, and YOLOv5 object detectors for computer vision and pattern recognition," *arXiv preprint*, Oct. 2025. https://doi.org/10.48550/arXiv.2510.09653

[7] N. A. Kurniawan and C. A. Sari, "Automatic license plate detection system with YOLOv11 algorithm," *Journal of Applied Informatics and Computing (JAIC)*, vol. 9, no. 6, p. 3097, Dec. 2025. https://doi.org/10.30871/jaic.v9i6.11484

[8] H. Moussaoui *et al*., "Enhancing automated vehicle identification by integrating YOLOv8 and OCR techniques for high-precision license plate detection and recognition," *Scientific Reports*, vol. 14, no. 1, Dec. 2024. https://doi.org/10.1038/s41598-024-65272-1

[9] R. Laroca *et al*., "A robust real-time automatic license plate recognition based on the YOLO detector," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, Apr. 2018. https://doi.org/10.1109/IJCNN.2018.8489629

[10] F. Martadinata, A. Firdaus, and M. R. T. Amal, "Detection and identification of vehicle license plates in Indonesia transportation system based on deep learning using YOLOv11 and EasyOCR," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 14, no. 4, pp. 573–578, Oct. 2025. https://doi.org/10.32736/sisfokom.v14i4.2524

[11] N. B. S. Makkar, "VajraV1—The most accurate real-time object detector of the YOLO family," *arXiv preprint*, Dec. 2025. https://doi.org/10.48550/arXiv.2512.13834

[12] M. S. Zandi and R. Rajabi, "Deep learning based framework for Iranian license plate detection and recognition," *Multimedia Tools and Applications*, Jan. 2022. https://doi.org/10.1007/s11042-022-12023-x

[13] R. Sapkota, Z. Meng, M. Churuvija, X. Du, Z. Ma, and M. Karkee, "Comprehensive performance evaluation of YOLOv12, YOLO11, YOLOv10, YOLOv9, and YOLOv8 on detecting and counting fruitlet in complex orchard environments," *arXiv preprint*, Feb. 2025. https://doi.org/10.48550/arXiv.2407.12040

[14] S. Gholinavaz, N. Saeedi, and S. S. Gharehveran, "Robustness analysis of YOLO and Faster R-CNN for object detection in realistic weather scenarios with noise augmentation," *Scientific Reports*, vol. 15, no. 1, Dec. 2025. https://doi.org/10.1038/s41598-025-28737-5

[15] N. Jegham, C. Y. Koh, M. Abdelatti, and A. Hendawi, "YOLO evolution: A comprehensive benchmark and architectural review of YOLOv12, YOLO11, and their previous versions," *arXiv preprint*, Mar. 2025. https://doi.org/10.48550/arXiv.2411.00201

[16] V. N. Ribeiro and N. S. T. Hirata, "Efficient video-based ALPR system using YOLO and visual rhythm," *arXiv preprint*, Jan. 2025. https://doi.org/10.48550/arXiv.2501.02270