

Automatic License Plate Detection System with YOLOv11 Algorithm

Nicholas Alfandhy Kurniawan¹, Christy Atika Sari^{2*}

Informatics Engineering, Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia
111202214004@mhs.dinus.ac.id¹, christy.atika.sari@dsn.dinus.ac.id²

Article Info

Article history:

Received 2025-10-11

Revised 2025-10-27

Accepted 2025-11-08

Keyword:

Character Recognition,
License Plate Detection,
Plate License,
YOLOv11.

ABSTRACT

The increasing number of motor vehicles in Indonesia demands technological solutions to enhance efficiency and security, particularly in automatic license plate recognition systems. This study aims to develop an automatic license plate detection system using the YOLOv11 algorithm to detect license plates and their characters in real-time. The research methodology includes collecting datasets from Kaggle, RoboFlow, and manual acquisition, followed by annotation, data augmentation, model training, and interface development using Tkinter and OpenCV. The dataset comprises 4000 license plate images and 3000 characters images, divided for training, validation, and testing. Evaluation results demonstrate strong model performance, with precision of 0.891, recall of 0.911, mAP50 of 0.906, and mAP50-95 of 0.631 for license plate detection, and precision of 0.889, recall of 0.912, mAP50 of 0.907, and mAP50-95 of 0.629 for character detection. Real-time testing showed that 12 out of 12 license plates were successfully recognized, influenced by lighting conditions, distance, and plate orientation. This study produced an efficient system for parking security, with potential for further development.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.

1.1. INTRODUCTION

The number of vehicles is increasing as every family certainly owns a vehicle, and it could be more than one to facilitate activities or serve as a tool for work [1], [2]. Each vehicle, such as motorcycles and cars, certainly has different license plates; however, there are also plates that share similarities because in Indonesia, license plates can be ordered according to the desires of the owner. The license plate itself is a form of identity for motor vehicles registered with the police, which is rectangular and consists of letters and numbers containing information about the province and region where the vehicle is registered. In Indonesia, license plates come in several colors as the background of the plate, indicating the different functions of each vehicle.

Automatic license plate recognition has become an attractive solution to improve efficiency and security in various sectors. Automatic license plate recognition is an important concept that allows identification and reading of characters (a-z, 0-9) on license plates without involving human observation, making it more efficient and practical.

This system can be applied to the parking security system of an agency [3], where the system automatically detect every license plate seen by the camera and only read license plates that have been registered in the database so that it can tighten and also limit so that not all vehicles can park in the agency's parking area which make the security of the area maintained and reduce the risk of motor vehicle crime. Therefore, computer vision is one of the studies that have a significant role in the development of automatic license plate detection systems. The technology help to detect and read the letters and numbers of the license plate automatically so as to speed up the operational process both from the detection system side and the vehicle side.

YOLO or "You Only Look Once" is an algorithm that has been designed to detect an object in real-time [4], where the detection process carried out quickly by combining the prediction of the bounding box so that it allows the system to immediately analyze objects that have been given a bounding box and produce object detection in a short time [5], [6], [7]. In this study, we chose to adopt the YOLOv11 algorithm, a newer version of YOLO that provides better

improvements in accuracy and speed in real-time license plate detection.

The goal of this research is to improve the existing license plate recognition system by increasing its accuracy and efficiency. The research uses artificial intelligence technology and the YOLO (You Only Look Once) object detection algorithm to identify the license plate and recognize the characters (A-Z, 0-9) on it. The process by which the license plate recognition system works with YOLO can be explained briefly as follows. First, YOLO detects the license plate area and marks it with a bounding box on the identified region using RoboFlow. RoboFlow is a computer vision development platform that supports data collection, preprocessing, and model training. Through RoboFlow, users can share and upload their own datasets as well as annotate objects to be detected with bounding boxes and apply data preprocessing and augmentation. YOLO then focuses on detecting characters within the marked license plate area while ignoring characters or elements outside the bounding box. The researcher used the Tkinter GUI, a Python model. The Tkinter module is the most widely used because it is easy for beginners to implement as a basis for learning GUI operations [8].

This research is made on the literature conducted by the researcher on research that has been done by previous researchers. In previous research, Andik Yulianto has conducted research related to license plate recognition systems with YOLO v8 in 2024. From the results of this study, researchers obtained good results for a license plate recognition system, 0.82 for the mAP50-95 value, and 99% for the accuracy of the detection model and 81% for recognition [9]. In contrast to this research, Iwan Virgiawan conducted research related to object detection using YOLO v3 by getting maximum results, namely 100% successful detection of objects with a range of up to 100cm from the camera and objects in the brightest light with a detection time of less than 3 seconds [10]. Fitri Utaminigrum [11], has conducted research on the detection of residential access license plates using YOLO v5 and obtained maximum results, 100% for successfully detecting license plates and 95.83% for detecting characters (a-z, 0-9) from license plates, but researchers do license plate detection at close range. Priyanto Hidayatullah conducted research on the comparison of YOLO versions where this study revealed that each version of YOLO (v8 to v11) showed improvements in architecture and feature extraction, but some blocks remained unchanged, such as SPPF. YOLO v10 introduced efficiency through NMS-free and SCDown training, while YOLO v11 improved detection capabilities with C3k2 and C2PSA. In addition, YOLO v11 also has the ability to detect small objects. However, the lack of academic publications and official architecture diagrams for YOLO v11 makes it difficult to understand and develop further [12].

The selection of YOLO v11 as the main algorithm in this study is based on its superiority in real-time object detection,

especially for small-sized objects such as characters on license plates, which are often difficult to detect accurately by previous versions of YOLO. YOLO v11 was chosen for its ability to combine computational efficiency with high accuracy, which is crucial for real-time applications such as parking security systems. Unlike the YOLO v3 used by Iwan Virgiawan (achieving 100% detection at a distance of 100 cm under brightly lit conditions), YOLO v11 offers more consistent performance across a wide range of environmental conditions, including low lighting and non-ideal viewing angles. In addition, compared to YOLO v5 used by Fitri Utaminigrum (100% license plate detection and 95.83% characters detection at close range), YOLO v11 has a more advanced architecture with the addition of modules such as C3k2 and C2PSA, which improve feature extraction and detection capabilities at small scales. Research by Priyanto Hidayatullah shows that YOLO v11 introduces significant improvements over YOLO v10 through NMS-free training and architecture optimization, which reduces computational complexity while maintaining high accuracy. In addition, YOLO v11 supports training with pre-trained models such as "yolo11s.pt", which enables training time efficiency and performance improvement through transfer learning, as applied in this study with 35 epochs. Therefore, YOLO v11 was chosen for its ability to overcome the limitations of previous versions in terms of accuracy, speed, and adaptability to environmental variations.

YOLOv11 introduces the C3k2 (Cross Stage Partial with Kernel Size 2) and C2PSA (Cross Stage Partial with Spatial Attention) modules, which significantly improve feature extraction capabilities for small-sized objects, such as characters on license plates. The C3k2 module uses a smaller convolution kernel to preserve visual details, reducing information loss during the downsampling process. Meanwhile, C2PSA integrates spatial attention mechanisms to focus the model on relevant areas in the image, such as license plates, despite complex background noise. According to research by Zhang [13], in Scientific Reports, these modules improve multiscale detection accuracy by up to 8% over YOLOv8, especially on small objects with heterogeneous backgrounds.

YOLO v11 adopts an NMS-free (Non-Maximum Suppression-free) training approach, which eliminates the need for traditional NMS post-processing. This approach reduces computational latency by simplifying the inference pipeline, making it suitable for real-time applications such as license plate detection in parking security systems. Research by Zhang [13], showed that the NMS-free approach in YOLO v11 reduced the inference time by 12% compared to YOLO v10, while maintaining a high mAP50 value (0.8920 in tests on remote sensing datasets). This allows YOLO v11 to process images at high speed without compromising accuracy.

Previous detection methods such as SSD and Faster R-CNN have been able to produce high accuracy, but are still constrained by inference speed and their inability to consistently recognize small objects. Meanwhile, YOLOv5 and YOLOv8 improve detection efficiency, but their performance tends to decline in low-light conditions, oblique angles, and when detecting small characters on license plates. In addition, the use of the Non-Maximum Suppression (NMS) mechanism in previous versions causes detection redundancy and increases processing time.

As the latest development, YOLOv11 offers architectural improvements through the C3k2 and C2PSA modules that strengthen small-scale feature extraction, as well as an NMS-free training approach that speeds up inference without reducing accuracy. With these capabilities, YOLOv11 addresses the limitations of previous methods in terms of speed, accuracy, and adaptability to complex environmental conditions, making it the most suitable solution for real-time vehicle license plate detection systems.

II. METHOD

This research had been used two types of datasets that have been collected by researchers, namely datasets for license plates and datasets for characters such as letters and numbers contained in the license plate. The license plate dataset in this study was taken from several sources, namely Kaggle, RoboFlow, and datasets taken using the camera from the researcher. The overall dataset consists of 4000 images with a division of 1000 images coming from the Kaggle dataset, 2500 images taken from datasets originating from RoboFlow, and 500 images taken through the researcher's cell phone manually in the area around where the researcher lives. All images in this dataset are colour license plate images with varying resolutions with various lighting conditions and viewing angles. In Table 1, there are details of the data division that researchers do, this dataset is divided into three data processing, namely 70.25% for training data, 19.825% for validation data, and 9.925% for testing data. The numbers from the division of the three data processing above produce decimal numbers because researchers use the division of the RoboFlow system. This dataset division is done to prevent the dataset from overfitting and to evaluate the model accurately [14]. Researchers carried out the dataset augmentation method during dataset processing which later the percentage of the three data changed again because when augmentation is carried out, the training data has been multiplied by three by the RoboFlow system so that the training results can be maximized so that it can detect objects (license plates) better. The character dataset (a-z, 0-9) of this license plate uses a license plate where each character has a bounding box on each character. The total dataset of this character is 3000 license plate images taken from various sources, namely 1500 images from RoboFlow, 1000 images

from Kaggle, and 500 images taken by researchers using a cell phone camera around where the researchers live.

The images were manually captured by researchers using mobile phone cameras in varying environmental conditions to represent real-world scenarios commonly encountered in everyday vehicle use. Images were captured both during the day and at night to ensure diversity in light intensity and color temperature. Daytime photos were taken under natural sunlight, resulting in high contrast and sharp reflections on the metal surfaces of the vehicles, while nighttime photos utilized street lighting and the phone's built-in flash to ensure the legibility of license plate characters. The shooting angles varied between front, oblique (approximately 30°–60°), and slightly elevated angles to simulate camera placements commonly found in parking or surveillance systems. Additionally, the backgrounds of the recorded vehicles varied, including asphalt roads, building walls, and vegetation, to enrich the dataset and test the robustness of the YOLOv11 model against background noise and lighting variations.

TABLE 1
DIVISION OF PLATE LICENSE DATA BEFORE AUGMENTATION

Type of data	Amount of data
Train	2810
Valid	793
Test	397

TABLE 2
DIVISION OF PLATE LICENSE CHARACTERS DATA BEFORE AUGMENTATION

Type of data	Amount of data
Train	2100
Valid	595
Test	305

Based on Table 2, there are details of the data division that researchers do, this dataset is divided into three data processing, namely 70% for training data, 19.83% for validation data, and 10.17% for testing data. The numbers from the division of the three data processing above are the same as the license plate dataset producing decimal numbers because researchers use the division of the RoboFlow system. This dataset division is done to prevent the dataset from overfitting and to evaluate the model accurately [14]. Researchers carried out the dataset augmentation method during dataset processing which later the percentage of the three data changed again because when augmentation is carried out, the training data multiplied by three by the RoboFlow system so that the training results can be maximized so that it can detect license plate characters better.

The research method is a scientific method that aims to obtain data in accordance with the objectives of the research so as to obtain the results as expected. For this research, the

YOLO v11 algorithm is used to find an automatic license plate recognition system. YOLO v11 is used by researchers because it simply has the ability to detect small objects, in this case aiming to detect characters (letters and numbers) on license plates. The stages in this research are dataset collection, bounding box, data augmentation, training, testing, Tkinter GUI creation, real-time testing of license plate detection [15]. The following is an image of the flow of this research in flowchart.

In Figure 1, you can see the research flow made by researchers, where researchers collect license plate datasets with two data collection methods, namely public datasets and private datasets. Public datasets are obtained by researchers through Kaggle and RoboFlow, while private datasets are obtained by researchers through researchers' cell phone cameras in locations around where researchers live. Furthermore, researchers input the dataset into the RoboFlow application which aims to provide a bounding box in the license plate section and also the characters of each license plate. Researchers conducted bounding boxes on license plates and characters (a-z, 0-9) from license plates using two different datasets in order to get maximum results. In the initial stage, the annotation process was carried out on the vehicle license plate image dataset. In this dataset, annotation is done using only one class, the "license plate" class, which represents the entire license plate area in the image. After that, annotation is performed on the character dataset contained in the license plate. In the character dataset, annotation is performed on 36 classes, consisting of 10 number classes (0-9) and 26 alphabet letter classes (a-z).

Since this dataset used for training the object detection model using the YOLO algorithm, the annotation method applied is to provide a bounding box on each target object, both the license plate as a whole and each individual character to ensure that the model can recognize and detect the object in question correctly during the training process [16].

Then, preprocessing is carried out on the two datasets by resizing the image resolution to 640 x 640 through the RoboFlow system [17]. After preprocessing, there is also an augmentation process carried out on both datasets, the license plate dataset is carried out an image rotation augmentation process between -25° and $+25^\circ$, and shear 20° horizontal, 25° vertical. For the character dataset on the license plate, the augmentation process is carried out, namely image rotation between -25° and $+25^\circ$, and shear 30° horizontal, 30° vertical [15], [18]. After the augmentation process, the new data division obtained as in Table 3 and Table 4.

TABLE 3
DIVISION OF PLATE LICENSE DATA AFTER AUGMENTATION

Type of data	Amount of data
Train	8430
Valid	793
Test	397

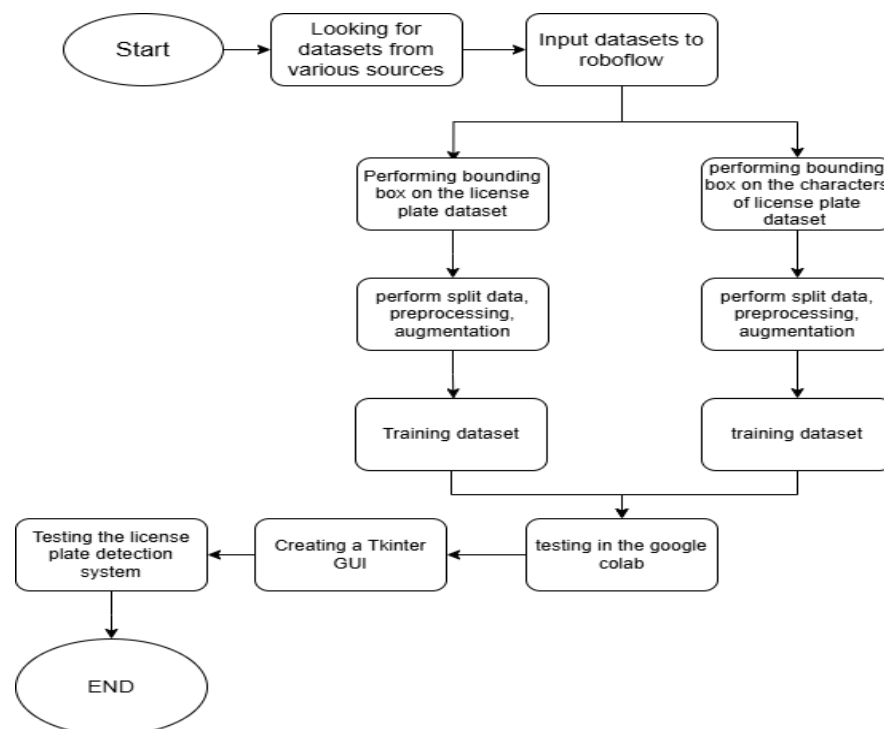


TABLE 4
DIVISION OF PLATE LICENSE CHARACTERS DATA BEFORE
AUGMENTATION

Type of data	Amount of data
Train	6300
Valid	595
Test	305

Based on Table 3 and Table 4, it can be seen there is a change in the amount of data that occurs, especially in the training data which changes from 2810 to 8430 for the license plate dataset (Table 3) and 2100 to 6300 for the license plate character dataset (Table 4). It can be seen there is an increase of three times from the initial number. Data augmentation helps prevent overfitting, which is a condition where the model overfits itself to the training data and thus loses the ability to generalize to new data. This augmentation process occurs because the researcher wants to increase the number and variety of the dataset so that the model to be detected can be learned in advance to recognize objects in diverse conditions. Increasing the amount and variety of data ensures that the YOLO v11 model learns the general features of license plates and characters, rather than just memorizing specific patterns. In addition, increasing the number of model variations allows the model to recognize a wider range of license plates and also results in higher detection accuracy [19].

After the augmentation process, 35 epochs were used to train on two different datasets [20]. In addition, the default pre-training model of YOLO v11, “yolo11s.pt”, was implemented in the dataset training process. The utilization of this model aims to improve efficiency and save training time, as there is no need to train from scratch. By using a pre-trained model as a base, the training of a new model can be done more optimally as the model already has prior knowledge of common visual features. After the dataset training process using the YOLO algorithm is completed, the researcher obtained the best training model which is automatically saved in the “runs” directory. This model is saved in a file named “best.pt”, which represents the model weights with the most optimal performance during training. The “best.pt” file be saved and used as the main model in the deployment process of the developed vehicle license plate recognition system as shown in Figure 2.

The training process was carried out using the YOLOv11 architecture, with initial parameters referring to the base model that had been previously developed by the open-

Figure 1 The Research Flow

source community. The model uses transfer learning, which utilizes pretrained weights from initial training on a general dataset as a starting point for the learning process. Thus, the model does not learn from scratch, but rather continues learning by adjusting the parameter weights to the characteristics of the vehicle license plate dataset used in this study.

The result of this training process is a file containing the best model weights, which is automatically saved in the “best.pt” format. These weights represent the model parameters with the most optimal performance based on the evaluation of precision, recall, and mean Average Precision (mAP) metrics on the validation data. These weights are then used as pretrained weights in the implementation stage of the real-time vehicle license plate detection system, as well as the basis for further development for testing the model in different environmental conditions.

Training and model testing were conducted on an Acer Nitro 5 AN515-57-921P laptop equipped with an Intel® Core™ i9-11900H processor, 16 GB DDR4 RAM, and an NVIDIA GeForce RTX 3060 GPU (6 GB GDDR6). The programming environment used Python, with supporting libraries such as OpenCV, NumPy, and Tkinter. The process was run on Windows 11 64-bit.

A balance between training speed and real-time inference efficiency. The RTX 3060 GPU supports high-speed parallel processing, while the memory capacity and multithreaded processor ensure system stability when handling large datasets and sequential images. With this configuration, the model is able to achieve optimal performance in detecting license plates in real-time without experiencing processing bottlenecks.

At the deployment stage, a user interface was built using Tkinter, a Python built-in GUI library that is lightweight and easy to use. Tkinter was chosen because it is able to provide basic components such as buttons, frames, and labels, which are sufficient for the needs of the system without having to build a web-based application. In its implementation, frames are used as video display containers, while labels are used to display text and video footage from cameras that are processed in real-time. To support image and video processing, OpenCV is used, a computer vision library that is capable of handling various visual processing operations efficiently [21]. The use of OpenCV greatly supports the development of object detection systems such as vehicle license plate recognition [22].

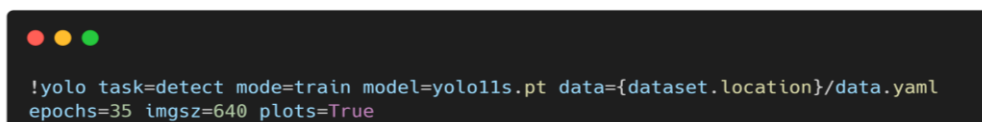


Figure 2 Epochs Used



Figure 3 GUI Tkinter

III. RESULTS AND DISCUSSION

This section presents and discusses the experimental findings derived from the proposed model. The results are analysed to assess the model's performance and to provide insights into its strengths and limitations. The following

Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100% 19/19
all	595	4080	0.889	0.912	0.907	0.629
0	160	172	0.93	0.929	0.921	0.661
1	179	211	0.913	0.896	0.912	0.534
2	260	295	0.907	0.956	0.903	0.628
3	224	269	0.932	0.974	0.946	0.665
4	226	264	0.905	0.938	0.915	0.615
5	217	258	0.947	0.973	0.955	0.685
6	246	299	0.934	0.95	0.941	0.668
7	150	173	0.952	0.926	0.926	0.654
8	136	142	0.964	0.944	0.964	0.707
9	167	183	0.93	0.866	0.956	0.661
A	124	141	0.918	0.959	0.946	0.666
B	133	142	0.907	0.963	0.948	0.666
C	66	67	0.788	0.97	0.883	0.6
D	62	74	0.943	0.889	0.911	0.674
E	140	142	0.895	0.965	0.931	0.674
F	49	51	0.912	0.902	0.923	0.637
G	49	50	0.886	0.931	0.902	0.604
H	316	332	0.949	0.962	0.945	0.636
I	43	47	0.831	0.936	0.856	0.578
J	28	28	0.869	1	0.928	0.649
K	138	150	0.924	0.947	0.931	0.653
L	51	51	0.92	0.901	0.929	0.646
M	28	30	0.818	0.967	0.907	0.658
N	36	37	0.846	0.838	0.889	0.623
O	18	18	0.476	0.722	0.681	0.465
P	62	65	0.921	0.892	0.869	0.628
Q	52	52	0.768	0.699	0.806	0.577
R	38	39	0.941	0.923	0.927	0.623
S	54	63	0.872	0.868	0.829	0.534
T	57	60	0.928	0.863	0.934	0.622
U	29	30	0.857	0.9	0.868	0.551
V	36	36	0.82	0.833	0.896	0.584
W	38	44	0.969	0.909	0.955	0.74
X	13	13	0.801	0.846	0.774	0.631
Y	23	23	0.963	0.957	0.993	0.636
Z	27	29	0.961	0.931	0.951	0.605

Figure 4 Results of Training Dataset for Number and Letter of License Plates

subsection begins with the evaluation of key performance metrics.

A. Evaluation

Here, we evaluate the performance of the YOLO v11 model for license plate and character detection (letters and numbers) using precision, recall, mean Average Precision (mAP50), and mAP50-95 metrics. Based on the training results for 35 epochs, the model for license plate detection produces a precision value of 0.891, recall of 0.911, mAP50 of 0.906, and mAP50-95 of 0.631. For character detection, the model achieved a precision of 0.889, recall of 0.912, mAP50 of 0.907, and mAP50-95 of 0.629 (Figure 4). These values indicate a strong performance in detecting license plates and their characters, with high precision and recall indicating that the model is able to identify most of the target objects with a good level of accuracy. The mAP50 value close to 0.9 indicates the model's ability to detect objects with high confidence at an Intersection over Union (IoU) threshold of 0.5, while the lower mAP50-95 (around 0.63) indicates that the model still faces challenges in achieving high accuracy at stricter IoU thresholds.

Specifically, the precision value of 0.891 for license plates indicates that 89.1% of the positive predictions generated by the model were correct, while the recall of 0.911 indicates that 91.1% of the license plates that should have been detected were successfully recognized. The mAP50 value of 0.906 reflects the model's ability to detect license plates with high accuracy at an IoU of 0.5, which is relevant for real-time applications such as parking security systems. However, the lower mAP50-95 value (0.631) indicates that the model is less consistent in detecting license plates at higher IoU thresholds, which is often necessary to ensure highly accurate detection of small objects such as characters. For character detection, similar results are seen, with performance variations between classes (letters and numbers) caused by the uneven distribution of data in the dataset, as shown in Figure 4.

The relatively low values of mAP50-95 (0.631 for license plates and 0.629 for characters) compared to mAP50 (0.906 and 0.907) indicate that the model faces difficulties in achieving high accuracy at tighter IoU thresholds. This could be due to the challenge of detecting small objects, such as characters on license plates, which require very high bounding box precision. In this study, data augmentations such as rotation (-25° to $+25^{\circ}$) and shear (20° - 30°) helped to improve detection, but did not fully overcome the challenges at high IoU, especially for characters with small size or extreme distortion. The evaluation results for character detection show performance variations between classes (Figure 4), which is likely due to the uneven distribution of data across letters and numbers in the dataset. Some characters, such as infrequently occurring letters (e.g., 'Q' or 'Z'), may have fewer samples than common characters such as 'A' or 'I', which affects the detection accuracy per class. In this study, although the character dataset consists of 3000 images augmented to 6300, the uneven distribution remains a challenge, which can be addressed in the future with techniques such as oversampling of minority classes or specific augmentation for certain classes.

B. Modelling

Based on the graph visualization results of the YOLOv11 model training process in Figure 5, it can be observed that the model experienced significant performance development during the training process. The loss values in the training data consisting of box_loss, cls_loss, and dfl_loss show a consistent downward trend as the number of epochs increases. This indicates that the model is able to learn effectively in adjusting the bounding box prediction, class classification, and label distribution to the training data used. A similar pattern is also seen in the validation data, where all three types of loss also experience a steady decline. This indicates that the model not only performs well on training data, but also shows good generalization ability to new data.

In addition, the evaluation of the model's performance metrics showed satisfactory results. The precision value increases sharply at the beginning of training and tends to stabilize at around 0.89 after the 15th epoch, indicating that most of the positive predictions generated by the model are correct. Similarly, the recall value increased significantly and stayed at around 0.91, which means that most of the objects that should have been detected were successfully recognized by the model. Furthermore, the mean Average Precision or mAP50 metric shows a high value of around 0.90, while mAP50-95 which is a higher precision metric shows a value of around 0.63. These two mAP values strengthen the indication that the trained YOLO v11 model has a good detection accuracy of the objects in the dataset.

Overall, the training results graph shows that the model managed to gradually and steadily improve its performance, both in terms of loss minimization and detection accuracy improvement. These results show that the YOLO v11 model can be reliably used in the developed vehicle license plate recognition system.

Figure 6 displays the training results of the YOLOv11 object detection model consisting of various metrics and loss functions during the training process. At the top of the graph, it can be seen that the three loss components in the training data, namely box_loss, cls_loss, and dfl_loss, decrease consistently until the end of the epoch. This indicates that the model has successfully optimized the prediction of the bounding box position, object classification, and label distribution effectively. This steady decline is an indication that the learning process on the training data is going well. On the other hand, the bottom graph shows the loss on the validation data. Both box_loss and cls_loss in the validation data also show a downward trend, which indicates the ability of the model to generalize to data that has never been seen before. However, in val/dfl_loss, there is a significant fluctuation throughout the epoch. Nonetheless, the variation is still within a relatively stable range and does not indicate extreme overfitting.

Meanwhile, the model evaluation metrics also showed positive performance. The precision and recall values show a sharp increase in the early training phase, and then stabilize above 0.9, indicating that the model is able to recognize most objects with high accuracy and completeness. The mean Average Precision (mAP) metric also reinforces these findings, with mAP50 reaching values close to 0.90, and mAP50-95 hovering around 0.60. Overall, these training result graphs show that the YOLO v11 model has been effectively trained and is able to achieve good performance on both training and validation data. The evaluation results support that the model is feasible for further use in the development of object recognition systems, such as vehicle license plate identification systems.

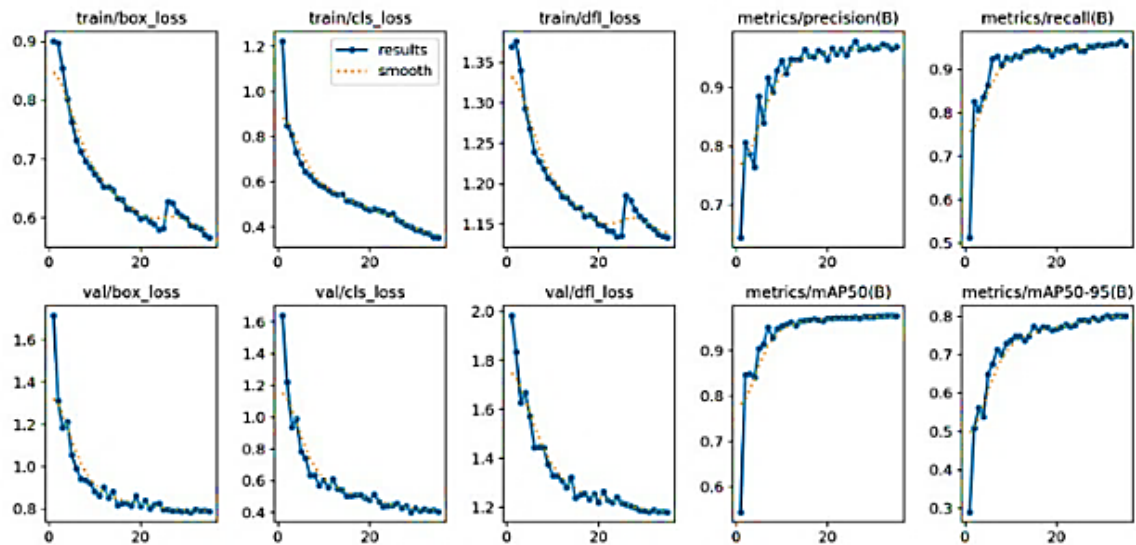


Figure 5 Results of the License Plate Training Process

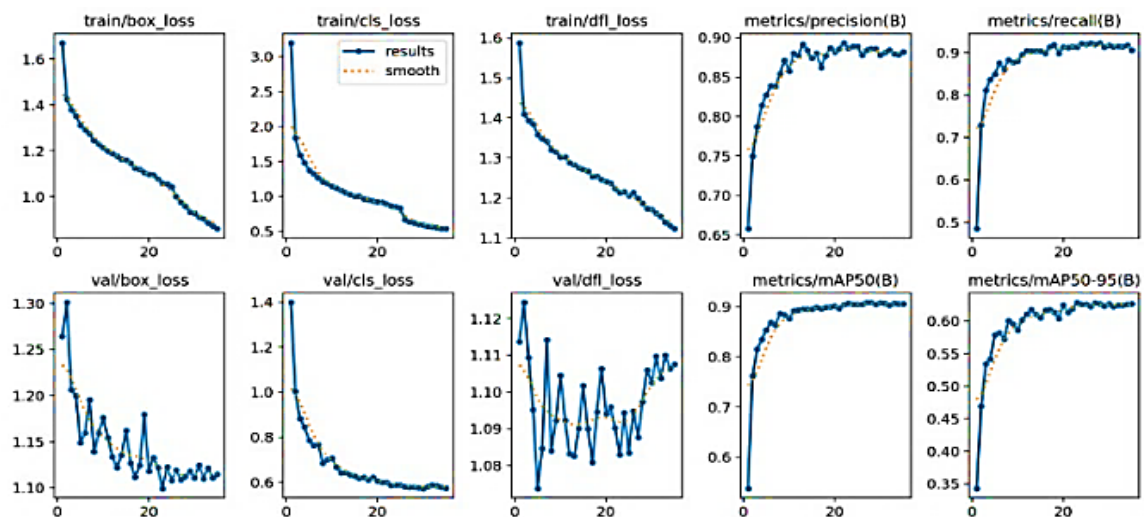


Figure 6 Results of the Character Training Process on License Plates


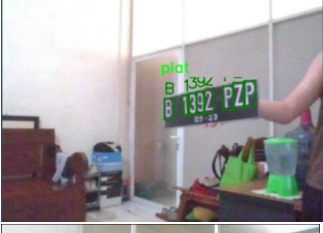




C. Result of the Detection

Table 5 shows the results of license plate detection tests using the YOLO v11 model with three different license plates, tested in 12 experiments with varying distances (50 cm, 75 cm, and 150 cm) and license plate positions (straight, tilted to the side, tilted backward, and tilted upward). Out of 12 trials, all license plates were detected correctly (100%), and all license plate characters were recognized accurately. The successful detection of license plates in all experiments

shows that the YOLO v11 model has a robust ability to identify license plate areas, even under varying distance and position conditions. At distances of 50 cm, 75 cm, and 150 cm with a straight position, the model successfully detected license plates with accurate bounding boxes, as shown in the detection results image (Table 5). This is supported by data augmentation such as rotation (-25° to $+25^{\circ}$) and shear (20° - 25° for license plates, 30° for characters), which allows the model to recognize license plates from various angles.

TABLE 5
THE RESULT OF LICENSE PLATE DETECTION

License Plate Figure	Result of the Detection	Description
	Detected Plat: B1392PZP	The position of the plate is 75 cm from the camera.
	Detected Plat: B1392PZP	The position of the plate is 150 cm from the camera.
	Detected Plat: B1083TFS	The position of the plate is 75 cm from the camera.
	Detected Plat: B1362UZX	The position of the plate is 75 cm from the camera.
	Detected Plat: B1362UZX	The position of the plate is 150 cm from the camera.
	Detected Plat: B1083TFS	The plate is positioned 50 cm from the camera at an angle to the side.

	Detected Plat: B1392PZP	The plate is positioned 75 cm from the camera.
	Detected Plat: B1392PZP	The plate is positioned 75 cm from the camera at an angle leaning backward.
	Detected Plat: B1362UZX	The plate is positioned 50 cm from the camera at an angle leaning backward.
	Detected Plat: B1083TFS	The position of the plate is 75 cm from the camera.
	Detected Plat: B1083TFS	The plate is positioned 75 cm from the camera at an angle to the side.
	Detected Plat: B1083TFS	The plate is positioned 75 cm from the camera at an angle to the side.

Although all license plates are detected, detection errors can occur due to several factors. Factors that may cause this failure include the tilted position of the license plate, which can cause significant visual distortion, making it difficult for the model to recognize small characters, and suboptimal lighting conditions, such as shadows or glare, which may have affected character recognition because the characters on the license plate are small objects that are difficult to detect with high accuracy.

This research can be considered significant because it demonstrates the potential of YOLO v11 in everyday applications, such as parking security systems, where fast and accurate license plate detection can restrict access to unregistered vehicles, thereby enhancing security. This system also supports operational efficiency by reducing reliance on human observation. Nevertheless, the failure of character recognition under extreme conditions highlights the need for further development to ensure consistency in various

environmental conditions, such as low lighting or less than ideal viewing angles.

D. Comparison of Researches

In this research, researchers obtained YOLOv11 model performance with precision 0.891, recall 0.911, mAP50 0.906, and mAP50-95 0.631 for license plate detection, and precision 0.889, recall 0.912, mAP50 0.907, and mAP50-95 0.629 for character detection. These values demonstrate the model's strong ability to detect license plates and their characters with high accuracy at an IoU (Intersection over Union) of 0.5, although performance degrades at tighter IoUs (mAP50-95 around 0.63). This degradation is due to the challenges in detecting small objects such as characters on license plates, especially under varying environmental conditions.

Compared to the research by Hanae Moussaoui et al. (Table 6) [23], which used YOLOv8 and OCR, they achieved 99% license plate detection accuracy and 98% character recognition accuracy with a smaller dataset (270 images). The accuracy of that study is lower than Nicholas' study, which may be due to the much smaller dataset used, so that model generalization may be limited compared to the dataset in this study, which includes 4000 license plate images and 3000 characters images. In addition, the approach used was incremental OCR, which required pre-processing such as k-means clustering and thresholding, whereas this study relied directly on YOLOv11, which is more efficient for real-time

applications. The study by Ming-An Chung et al. [24] with YOLO-SLD (based on YOLOv7) reported a license plate detection accuracy of 98.91% on the CCPD dataset, with a significant improvement in extreme lighting conditions (from 93.5% to 96.7%). Although there is no significant difference in accuracy, YOLOv11 in this study shows superiority in detecting small objects (characters) thanks to the C3k2 and C2PSA modules, which improve feature extraction. In addition, the NMS-free approach in YOLOv11 reduces computational latency, providing an efficiency advantage over YOLO-SLD, which still uses traditional NMS. Reda Al-batat et al. [25] used YOLOv2 for vehicle detection, YOLOv4 for license plate and character detection, and ResNet50 for vehicle type classification, achieving an average character recognition accuracy of 90.3%. While this accuracy is slightly higher for character recognition than Nicholas' study (88.9% precision), Reda Al-batat's approach is more complex as it involves three processing stages, which may increase latency compared to the more integrated single approach of YOLOv11.

This study shows that the YOLOv11 model successfully detected all license plates in 12 experiments with varying distances (50 cm, 75 cm, 150 cm) and plate positions (straight, sideways, backward, upward), as shown in Table 5. However, factors such as visual distortion and suboptimal lighting conditions can be one of the causes of character recognition failure.

TABLE 6
RELATED RESEARCH

Title	Author's Name	Discussion and Research Result
Enhancing Automated Vehicle Identification by Integrating YOLO v8 and OCR Techniques for High-Precision License Plate Detection and Recognition	Hanae Moussaoui, Nabil El Akkad, Mohamed Benslimane, Walid El-Shafai, Abdullah Baihan, Chaminda Hewage, Rajkumar S. Rathore	This research proposes a vehicle identification system based on YOLO v8 for license plate detection as well as OCR (Optical Character Recognition) for character recognition. The dataset consists of 270 images annotated using CVAT and tested under various lighting conditions. For image quality improvement, we applied k-means clustering, thresholding, and morphological operations. The experimental results show an accuracy rate of 99% for license plate detection and 98% for character recognition. In addition, the system generates a text file that extracts the country of origin of the vehicle automatically. Evaluation was conducted using metrics such as precision, recall, F1-score, and CLA, and showed superior performance over previous methods in the literature [23].
YOLO-SLD: An Attention Mechanism-Improved YOLO for License Plate Detection	Ming-An Chung, Yu-Jou Lin, Chia-Wei Lin	This study introduces YOLO-SLD, a modified version of YOLOv7 with the integration of SimAM (parameter-free attention module). The main focus of the research is to improve the accuracy and efficiency of license plate detection under complex lighting conditions and image capture angles. Experiments were conducted on the CCPD dataset, where YOLO-SLD improved the accuracy from 98.44% to 98.91% compared to YOLOv7. In the subset of images with dark and bright conditions, the accuracy increased from 93.5% to 96.7%. In addition, the number of model parameters was reduced by 1.2 million parameters, making the model more lightweight and efficient. These results show that the addition of SimAM strengthens the feature extraction capability and improves performance without increasing the computational burden [24].
An End-to-End Automated License Plate Recognition System Using YOLO Based Vehicle and License Plate Detection with Vehicle Classification	Reda Al-batat, Anastassia Angelopoulou, Smera Premkumar, Jude Hemanth, Epameinondas Kapetanios	This research develops an end-to-end ALPR system consisting of three main stages: vehicle detection using YOLOv2, license plate detection and character recognition using YOLOv4, and vehicle type classification (truck, emergency vehicle, etc.) using ResNet50. Unlike previous approaches, this system does not rely on fixed rules or country-specific license plate layout-based preprocessing. The evaluation was conducted on five public datasets from various countries (AOLP, UFPR-ALPR, Caltech Cars, etc.), covering a variety of lighting conditions, shooting angles, and plate formats. Results show an average character recognition accuracy of 90.3% and a processing speed that can still be used on mid-range GPUs (decent FPS for real-time applications). As such, the system is considered robust and flexible enough to handle real-life scenarios [25].

Compared to the work of Moussaoui et al. [23] which achieved 98% character recognition accuracy under various lighting conditions, the performance of YOLOv11 in this study was slightly higher (100% character recognition success in real-time testing). However, the Moussaoui study used a much smaller dataset and did not mention license plate position variations, which may limit the generalization of their model compared to the larger and more diverse dataset in this study. The study by Chung et al. [24] showed lower accuracy in extreme lighting conditions (96.7%). However, YOLOv11 has an advantage in detecting license plates at longer distances (150 cm), which was not explicitly tested in Chung's study. In addition, the C3k2 and C2PSA modules in YOLOv11 provide better character detection capabilities at small scales than YOLOv7, although character recognition results are still affected by extreme plate positions. Reda Albatat et al. [25] reported an average character recognition accuracy of 90.3% on datasets from various countries, which is lower than the character recognition success rate (100%) in real-time testing in this study. However, their approach using YOLOv4 for plate and character detection requires a mid-range GPU device to achieve real-time speed, while YOLOv11 with a non-NMS approach is lighter and more efficient, suitable for applications with limited resources.

II. CONCLUSION

This study successfully developed an automatic license plate detection system using the YOLO v11 algorithm, which demonstrated accurate and efficient performance in detecting license plates and characters in real time. With a dataset of 4,000 license plate images and 3,000 characters images, the trained model produced satisfactory precision, recall, and mAP values, demonstrating good generalization capabilities. Tests showed that the system was able to detect license plates well, even though character detection was affected by factors such as lighting, distance, and plate position. The evaluation results showed strong performance with a precision of 0.891, recall of 0.911, mAP50 of 0.906, and mAP50-95 of 0.631 for license plate detection, as well as precision of 0.889, recall of 0.912, mAP50 of 0.907, and mAP50-95 of 0.629 for character detection. Real-time testing successfully detected all license plates (100%) at a distance of 50-150 cm, and also successfully detected all characters (100%). Data augmentation (-25° to +25° rotation, 20°-30° shear) improved model generalization, although challenges such as class imbalance and high IoU thresholds remain. This system has the potential to improve parking security in institutions through automated vehicle identification. In this study, the developed system focuses exclusively on the detection process, where the YOLOv11 algorithm identifies and localizes vehicle license plates along with their characters. The recognition phase, which involves reading and converting the detected characters into text, has not yet been integrated. For future development, the system can be combined with an Optical Character Recognition (OCR)

module, such as Tesseract or EasyOCR, to facilitate automatic reading and extraction of vehicle license plate data.

REFERENCES

- [1] E. Ektrada, L. Hakim, and S. P. Kristanto, "Sistem Tracking dan Counting Kendaraan Berbasis YOLO untuk Pemetaan Slot Parkir Kendaraan," *Softw. Dev. Digit. Bus. Intell. Comput. Eng.*, vol. 1, no. 02, pp. 55–60, Mar. 2023, doi: 10.57203/session.v1i02.2023.55-60.
- [2] Y. Galahartlambang, T. Khotiah, Z. Fanani, and M. B. D. Rahmat, "Pengenalan Plat Nomor Kendaraan Real-Time pada Kondisi Gelap dan Hujan Menggunakan Deep Learning," *J. Inform. Rekayasa Elektron.*, vol. 8, no. 1, pp. 188–194, 2025.
- [3] A. Malfaresa, M. Masril, O. E. Putra, H. Awal, and B. Hendrik, "Inovasi Sistem Kontrol Akses Gerbang Kantor Pemerintahan Berbasis Teknologi Multisensor dan Deteksi Plat Nomor Kendaraan," *J. QUANCOM QUANTUM Comput. J.*, vol. 2, no. 2, pp. 1–8, Dec. 2024, doi: 10.62375/jqc.v2i2.433.
- [4] A. Meirza and N. R. Puteri, "Implementasi Metode YOLOV5 dan Tesseract OCR untuk Deteksi Plat Nomor Kendaraan," *J. Ilmu Komput. Dan Desain Komun. Vis.*, vol. 9, no. 1, pp. 424–435, 2024.
- [5] O. Alfiano and S. Rahayu, "Implementasi Algoritma Deep Learning YOLO (You Only Look Once) untuk Deteksi Kualitas Kentang Segar dan Busuk Secara Real Time," *J. Res. Publ. Innov.*, vol. 2, no. 3, pp. 2470–2478, 2024.
- [6] F. Agustina and M. Sukron, "Deteksi Kematangan Buah Pepaya Menggunakan Algoritma YOLO Berbasis Android," *J. INFOKAM*, vol. XVIII, no. 2, pp. 71–78, 2022.
- [7] N. J. Hayati, D. Singasatia, and M. R. Muttaqin, "Object Tracking Menggunakan Algoritma You Only Look Once (YOLO)v8 untuk Menghitung Kendaraan," *KOMPUTA J. Ilm. Komput. Dan Inform.*, vol. 12, no. 2, pp. 91–99, 2023.
- [8] S. Nova, N. Khotimah, and M. Y. A. Wahyuningrum, "Pemanfaatan Chatbot Menggunakan Natural Language Processing untuk Pembelajaran Dasar-Dasar GUI Tkinter pada Bahasa Pemrograman Python," *J. Ilm. Tek.*, vol. 3, no. 1, pp. 58–65, 2024.
- [9] Anthony, Herman, and A. Yulianto, "Pengembangan Sistem Pengenalan Plat Nomor Indonesia Menggunakan YOLOv8 dan EasyOCR," *J. Ilm. KOMPUTASI*, vol. 23, no. 4, pp. 571–578, 2024.
- [10] I. Virgiawan, F. Maulana, M. A. Putra, D. D. Kurnia, and E. Sinduningrum, "Deteksi dan Tracking Objek Secara Real-Time Berbasis Computer Vision Menggunakan Metode YOLO V3," *J. Ris. Penelit. Univers.*, vol. 05, no. 2, pp. 42–52, 2024.
- [11] M. R. Rais, F. Utaminirum, and H. Fitriyah, "Sistem Pengenalan Plat Nomor Kendaraan untuk Akses Perumahan menggunakan YOLOv5 dan Pytesseract berbasis Jetson Nano," *J. Pengemb. Teknol. Inf. Dan Ilmu Komput.*, vol. 7, no. 2, pp. 681–685, 2023.
- [12] P. Hidayatullah, N. Syakrani, M. R. Sholahuddin, T. Gelar, and R. Tubagus, "YOLOv8 to YOLO11: A Comprehensive Architecture In-depth Comparative Review A PREPRINT," 2025.
- [13] L. He, Y. Zhou, L. Liu, W. Cao, and J. Ma, "Research on object detection and recognition in remote sensing images based on YOLOv11," *Sci. Rep.*, vol. 15, no. 1, p. 14032, Apr. 2025, doi: 10.1038/s41598-025-96314-x.
- [14] R. G. Wijnarko, A. I. Pradana, and D. Hartanti, "Implementasi Deteksi Drone Menggunakan YOLO (You Only Look Once)," *J. FASILKOM*, vol. 14, no. 2, pp. 437–442, 2024.
- [15] M. IKBAL and R. A. Saputra, "Pengenalan Rambu Lalu Lintas Menggunakan Metode YOLOv8," *JIKA J. Inform.*, vol. 8, no. 2, pp. 204–212, Apr. 2024, doi: 10.31000/jika.v8i2.10609.
- [16] A. A. Rasjid, B. Rahmat, and A. N. Sihananto, "Implementasi YOLOv8 Pada Robot Deteksi Objek," *J. Technol. Syst. Inf.*, vol. 1, no. 3, pp. 1–9, Jul. 2024, doi: 10.47134/jtsi.v1i3.2969.
- [17] M. R. Joyonegoro and E. Setyati, "Deteksi dan Pengenalan Dua Variasi Plat Nomor Kendaraan Bermotor di Indonesia dengan Variasi Waktu dan Pencahayaan Memanfaatkan YOLO V8 dan CNN," *KONVERGENSI*, vol. 20, no. 1, pp. 11–17, 2024.
- [18] M. H. Rais, A. Musnansyah, and H. Fakhrrurroja, "Penerapan Algoritma YOLO V8 untuk Pengenalan Pengendara Sepeda Motor

- Tanpa Helm dalam Sistem Pemantauan Pelanggaran Lalu Lintas,” *E-Proceeding Eng.*, vol. 12, no. 1, pp. 1531–1538, 2025.
- [19] R. Hesananda, I. A. Noviani, and M. Zulfariansyah, “Implementasi YOLOv5 untuk Deteksi Objek Mesin EDC: Evaluasi dan Analisis,” *BIOS J. Teknol. Inf. Dan Rekayasa Komput.*, vol. 5, no. 2, pp. 104–110, Aug. 2024, doi: 10.37148/bios.v5i2.127.
- [20] B. D. Prasetya, E. P. Amalia, P. D. Juliana, and R. K. Niswatin, “Implementasi Object Detection Berbasis Web untuk Klasifikasi Jenis Sampah Menggunakan YOLOv8,” in *Prosiding Seminar Nasional Teknologi dan Sains*, 2025, pp. 60–65.
- [21] W. P. N. Putra, A. I. Pradana, and Nurchim, “Implementasi Sistem Penghitungan Volume Kendaraan Menggunakan YoloV8,” *J. FASILKOM*, vol. 14, no. 2, pp. 443–450, 2024.
- [22] M. Zakiyamani, T. I. Cahyani, D. Riana, and S. Hardianti, “Deteksi dan Pengenalan Plat Karakter Nomor Kendaraan Menggunakan OpenCV dan Deep Learning Berbasis Python,” *J. Inf. Technol. Comput. Sci. INTECOMS*, vol. 5, no. 1, pp. 56–64, 2022.
- [23] H. Moussaoui and others, “Enhancing automated vehicle identification by integrating YOLO v8 and OCR techniques for high-precision license plate detection and recognition,” *Sci. Rep.*, vol. 14, no. 1, Dec. 2024, doi: 10.1038/s41598-024-65272-1.
- [24] M. A. Chung, Y. J. Lin, and C. W. Lin, “YOLO-SLD: An Attention Mechanism-Improved YOLO for License Plate Detection,” *IEEE Access*, vol. 12, pp. 89035–89045, 2024, doi: 10.1109/ACCESS.2024.3419587.
- [25] R. Al-batat, A. Angelopoulou, S. Premkumar, J. Hemanth, and E. Kapetanios, “An End-to-End Automated License Plate Recognition System Using YOLO Based Vehicle and License Plate Detection with Vehicle Classification,” *Sensors*, vol. 22, no. 23, Dec. 2022, doi: 10.3390/s22239477.