International Journal of Computing and Digital Systems
2025, VOL. 17, NO.1, 1-13

http://dx.doi.org/10.12785/ijcds/1571020530

Modified YOLOv5-based License Plate Detection: an Efficient Approach for Automatic Vehicle Identification

Rifqi Alfinnur Charisma[1] **and Suharjito**[2]

[1]*Computer Science Department, BINUS Graduate Program-Master of Computer Science, Bina Nusantara University, Jakarta 11480, Indonesia*
[2]*Industrial Engineering Department, BINUS Graduate Program-Master of Industrial Engineering, Bina Nusantara University, Jakarta 11480, Indonesia*

Received 18 April 2024, Revised 16 June 2024, Accepted 29 August 2024

Abstract: Indonesia witnesses a continual annual surge in the vehicle count, with the Central Statistics Agency (BPS) projecting a total of 148.2 million vehicles in 2022, marking a 6.3 million increase from the preceding year. This growth underscores the escalating challenges associated with traffic management and violations. Hence, the development of a robust vehicle number plate image recognition system becomes paramount for effective traffic control, accurate parking records, and streamlined identification of vehicle owners. In this study, we introduce a modified YOLO v5 algorithm, enhancing its backbone to achieve richer feature extraction from vehicle images. This modification aims to improve the model's capacity to handle diverse conditions, such as low lighting, intricate viewing angles, and blurred license plates. Utilizing the AOLP dataset, the modified YOLO v5 algorithm demonstrates remarkable performance metrics, boasting a recall value of 99.7%, precision reaching 99.1%, mAP50 of 99.4%, and mAP50-95 of 84.8%. The enhanced precision signifies the model's proficiency in minimizing identification errors, while the commendable recall highlights its adeptness in accurately locating number plates. Additionally, the Optical Character Recognition (OCR) model, dedicated to character recognition on number plates, achieves an accuracy level of 92.85%, underscoring its efficacy in deciphering alphanumeric characters. This integrated approach leverages advanced algorithms and our backbone modifications to tackle the intricacies of real-world scenarios, affirming its viability for enhancing traffic management systems and bolstering the efficiency of vehicle-related processes.

Keywords: YOLO, EasyOCR, Object detection, Plate number, Character recognition

1. INTRODUCTION

The annual growth of the vehicle population in Indonesia persists, as indicated by statistics from the Central Statistics Agency (BPS). In 2022, the total number of vehicles is projected to reach 148.2 million, reflecting a surge of 6.3 million compared to the preceding year. Every vehicle, whether motorbike or car, must have a number plate as proof that the vehicle has received permission from the police and is being used on the road [1], [2]. A number plate is a motor vehicle identification mark the police give when the vehicle is first used [3].

As the number of vehicles on the road increases, problems with traffic violations also increase. This makes a vehicle number plate image recognition system important [4]. The vehicle number plate image recognition system can be carried out using pattern recognition techniques or Deep Learning and Computer Vision [5].

There are several studies on the use of computer vision for number plate detection, such as the [6] research, which uses the YOLO-Darknet algorithm with datasets from AOLP [7]. The same algorithm was also studied by [8], but the results of recognizing the object of this research were better, namely 98.22% compared to the [6] research which only produced an accuracy of 97.1%. In the following year, there was research conducted by [9] using the YOLO v4 method, followed by research from [10], which used YOLO v5 with the RHNP dataset. This research resulted in mAP of 98.8% on YOLO v4 and 90.8% on YOLO v5.

From this research, it can be concluded that the performance of the YOLO algorithm can obtain high accuracy regarding number plate detection. However, this research does not take into account traffic problems such as plate image conditions with low lighting, complex viewing angles, and small or large number plates. It was blurred so that it could reduce the performance of the number plate recognition system as a whole. According to research

E-mail address: rifqi.charisma@binus.ac.id, suharjito@binus.edu

conducted by [11] and [12], inconsistent lighting can affect object recognition because it can change the appearance of objects in the image. If the lighting is too bright or dark, the object may look different from the image referenced by the object recognition model. This can cause the model to fail to recognize objects with low accuracy. Therefore, lighting consistency is critical in image capture for accurate object recognition.

Based on this problem, research was conducted to detect number plates using the YOLO v5 algorithm with a vehicle dataset called AOLP. The AOLP dataset includes vehicle images under various imaging conditions. These include low-light images and varying levels of image clarity. These diverse imaging conditions reflect frequently occurring situations in traffic and pose a challenge for object detection models to recognize license plates accurately. YOLO v5's performance reached 67.24% AP (Average Precision) on the DOTA (emergency landing spot) dataset using an Nvidia RTX 2070 GPU. This result is better than YOLO v4 with an AP value of 64.818%. Therefore, this research will use YOLO v5 because it was proven superior to YOLO v4 [13].

This research contributes to the development of object detection techniques using YOLO v5 with two main approaches. First, through modifications to the YOLO v5 backbone, this research introduces the addition of new convolution layers to improve feature representation without significantly increasing the number of parameters. Second, this research performs careful hyperparameter tuning, optimizing critical parameters in the model training process to improve detection performance. Thus, the contribution of this research is expected to increase the accuracy and reliability of object detection systems, especially in the context of vehicle number plate detection in poor lighting conditions and varying viewing angles.

2. RELATED WORK

Various research has been carried out in number plate detection using multiple methods and models in computer vision.

For example, in research by [14], CNN, RNN, and LSTM models were used with the AOLP dataset consisting of three image categories. The CNN model was trained with four layers to classify license plate images, while RNN with LSTM was used for character detection. The model in this research produced a precision of 97.18%, recall of 97.19% in object recognition, and accuracy of 86.22%. However, the approach had limitations in handling diverse environmental conditions and different license plate formats.

Another study by [15] proposed a hybrid CNN and SVM approach for license plate classification and recognition, the segmented characters were scaled to 28×28 images for subsequent processing. The CNN model, consisting of seven layers—an input layer, convolutional layer, ReLU layer, max-pooling layer, fully connected layer, classification layer, and a softmax layer—was employed for feature extraction from the segmented regions. This method achieved an impressive recognition accuracy of 98.45%. However, one limitation of this approach is that it primarily relies on a fixed image resolution (28×28), which may not fully capture finer details in the characters, potentially affecting the model's performance on more complex or noisy images. Furthermore, [16] used Fast-YOLO v2 with adjustments to the input image and convolution kernel size. This modification significantly improved the results of object detection and character recognition, achieving 99.45% recall in object detection and 96.9% accuracy in character recognition. The main drawback was the complexity of the modifications, making the model less adaptable to real-time applications.

Sun et al. [17] compared YOLO v2 and YOLO v3 for number plate detection and used CRNN-12 to read number plates. The dataset was collected manually from various locations with varying lighting conditions. Although both YOLO models achieved high levels of accuracy, YOLO v3 showed slightly better performance in terms of Intersection Over Union (IOU). Meanwhile, CRNN-12 managed to achieve 98.86% accuracy in reading number plates. This study highlighted the robustness of YOLO v3, but it also showed that the model struggled with smaller license plates and different angles. Further research was conducted by [18] used CNN for license plate classification, the proposed architecture was evaluated on three popular benchmarks, namely the Stanford Cars dataset, the Indian License Plates Dataset, and the Car License Plate Detection Dataset. The model achieved an accuracy of 98% across these three datasets. However, a potential limitation of this study is that while it reports high accuracy, it does not provide detailed insights into the model's performance under varying conditions such as different lighting, angles, or occlusions, which are common challenges in real-world scenarios.

Lastly, [9] adopted YOLO v4 for vehicle type and license plate character detection. Using this model on high-resolution video datasets produced a high success rate in detecting number plates, even on tiny number plates. However, the high-resolution requirement limited its application to certain types of cameras and environments. Meanwhile, research by [10] explored SG-YOLO v5, a modification of YOLO v5 with improved performance in license plate detection. SG-YOLO v5 managed to achieve a mean average precision (mAP0.5) of 94.5%.

YOLO v5 has been shown to have a balance between object detection accuracy and computational speed, making it suitable for license plate detection tasks in real-time scenarios. This advantage is supported by research results from [10], which show the superior performance of YOLO v5 in terms of mean average precision (mAP0.5) of 94.5%. YOLO v5 also has high adaptability to variations in object and environmental conditions, such as low lighting, complex viewing angles, and small or blurred objects on vehicle license plates. This capability meets the challenges of license plate detection in various traffic situations.

Thus, choosing YOLO v5 as a number plate detection method has advantages in terms of accuracy, adaptability, and speed, in accordance with the needs of this research.

3. RESEARCH METHODOLOGY

A. *Research Flowcart*

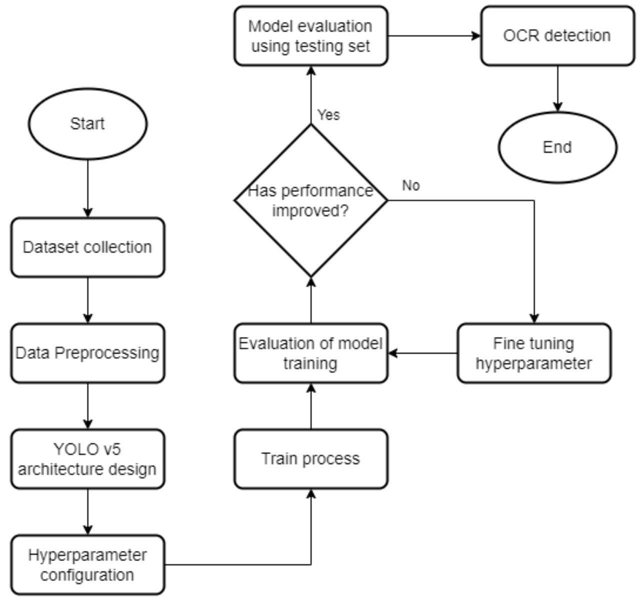

Start
Dataset collection
Data Preprocessing
YOLO v5 architecture design
Hyperparameter configuration
Train process
Evaluation of model training
Has performance improved?
Yes
No
Model evaluation using testing set
OCR detection
End
Fine tuning hyperparameter

Figure 1. Research flowchart

Figure 1 illustrates the research stages that will be carried out in this research. The initial stage of this research begins with collecting relevant datasets to train and test the object detection model. After the dataset is collected, a pre-processing process is carried out, including dividing training and testing data and adding labels or annotations to objects in the image. Next, the researchers designed the YOLO v5 object detection model architecture, which was adapted to the characteristics of the dataset and research problems. This process is followed by hyperparameter adjustments, such as learning rate and batch size, to improve model performance. Model training is carried out using a training dataset involving loss calculations, backpropagation, and weight updates to minimize prediction errors.

Once training is complete, the model is evaluated using a validation dataset to measure object detection performance, with metrics such as precision, recall, and mAP. Evaluation results are used to determine whether model performance is adequate or requires adjustment. If it is satisfactory, the researcher continues to the testing stage using dataset testing. However, if performance still needs to be improved, hyperparameter fine-tuning is performed for better configuration. The trained and validated model is tested on a testing dataset to test performance in realistic situations. The process of detecting number plate characters using OCR is involved in identifying the characters in the bounding box that have been detected as vehicle number plates.

B. *Dataset Preprocessing*

The dataset processing process initiates with the careful selection of a dataset tailored to meet the specific research requirements. In this context, the researchers have opted for the AOLP dataset, as introduced in the work by [7]. Comprising 2049 images captured under diverse lighting conditions, this dataset proves particularly well-suited for the development of object detection models capable of robust performance across varying illumination scenarios. The AOLP dataset includes car images taken under varying conditions. The conditions represented in this dataset include dark images taken from CCTV and at night, as well as images taken in parking lots and on roads, thus providing different points of view. Figure 2 visually illustrates a representative sample from the AOLP dataset, providing a glimpse into its diverse and challenging image conditions.

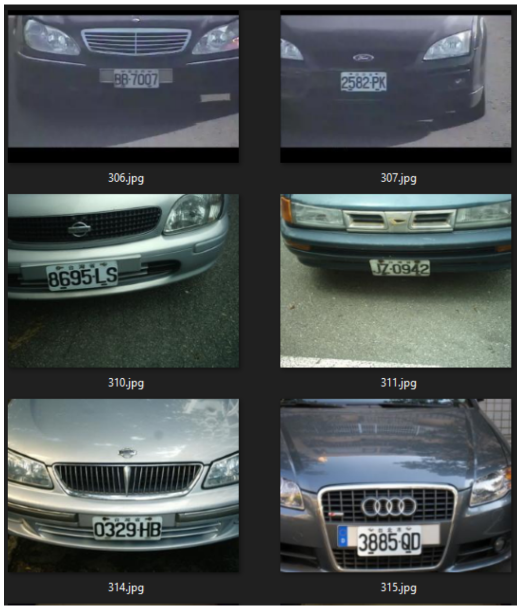


Figure 2. AOLP Dataset

The next step is to conduct the labelling process on the images in the dataset. This labelling process is important to provide information to the model about the location and class of objects in the image. Tools such as CVAT (Computer Vision Annotation Tool) can make labelling easier. The output of labelling using CVAT is a file containing information about the location and class of objects marked on the images in the dataset [19]. CVAT will produce a label file in the appropriate format. In this study, the output from CVAT is a file with the extension .txt, which corresponds to the model used in YOLO v5 [20]. In Figure 3, the illustration visually elucidates the dataset labelling process through CVAT, portraying the interface and functionality of the annotation tool. The labelled objects, marked with bounding boxes and corresponding class labels, serve as an essential input for training the object detection model.

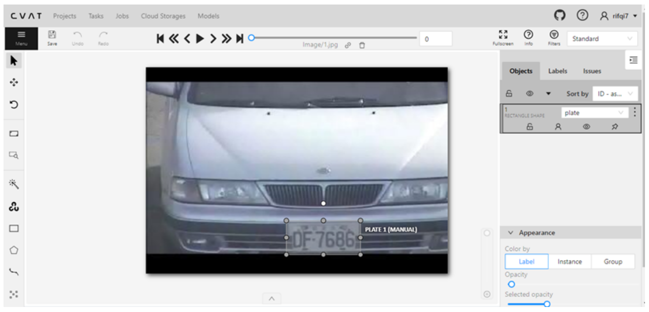


Figure 3. Labelling Process

Once the labelling process is finished, the dataset will be categorized into three groups, namely training data, validation data, and testing data. This categorization will follow a split ratio of 7:2:1, ensuring a substantial portion for training, followed by validation and testing sets. This segmentation strategy aims to facilitate comprehensive model training, robust validation, and accurate testing for optimal performance evaluation.

C. *YOLO v5 Architecture*

Figure 4 is the proposed model configuration. The YOLO v5 framework employs a mosaic data augmentation technique, which involves the processes of zooming, cropping, arrangement, and stitching of input data. This methodology is implemented to enhance the performance of small object detection within the model. During the model training process, the input from YOLO v5 is changed to a uniform size, namely 416×416 pixels. The core network (backbone) consists of two parts, namely Focus and CSP. Focus is used to crop the image before inserting it into the core network [21]. As shown in Figure 5, the original image measuring 416×416×416 is cropped to 208×208×12, and then fed into a convolution operation using the feature map of 32 kernels. The Focus operation allows changing the size of the image to a smaller one without using additional parameters and still maintaining the information of the original image [22], [23].

In Figure 6, the architectural illustration delineates the YOLO v5 backbone. Initiated by the Focus-CSP1 1 process, it unfolds through subsequent stages, including CSP1 3-CSP1 3-SPP (Spatial Pyramid Pooling). The CSP1 modules represent the cross-stage partial networks, which are integral for feature extraction and information propagation across stages of the network. The SPP stage incorporates spatial pyramid pooling, contributing to the model's capability to capture features at different scales.

The neck segment functions as a network layer that is responsible for merging image features and transmitting them to the prediction layer. In YOLO v5, the neck section adopts the FPN+PAN architecture [24], [25], [26]. As shown in figure 7, FPN is tasked with developing high-level feature information and combining it gradually from top to bottom to obtain a feature map used for predictions. Meanwhile, PAN is like a deep pyramid because it forms a path that connects various levels of image resolution from the bottom to the top [27].

The prediction layer or head functions to predict image features and create bounding boxes to identify the object type. YOLO v5 uses GIOU-Loss as a function to measure the bounding box prediction error. In Figure 8, the head architecture of YOLO v5 is depicted, comprising bottleneck CSP (Cross-Stage Partial) and a 1×1 convolutional layer. This architecture is designed to generate predictions and facilitate the efficient identification of object types. The bottleneck CSP module contributes to feature extraction and information flow, while the 1×1 convolutional layer aids in refining the predictions.

D. *Improvement Backbone YOLO v5*

In the development of an object detection model using the YOLOv5 architecture, the backbone assumes significance in feature extraction from the input image. Modifications to the backbone section aim to enhance the model's capacity to capture pertinent and intricate features, thereby enabling improved object detection across diverse conditions, including low lighting and varying viewing angles.

Based on Figure 9, the YOLO v5 backbone modification involves adding a 1×1 convolutional layer after block C3. The following is a complete explanation regarding the modification of adding layers to the YOLO v5 backbone.

- Added convolutional layer after layer 3×C3 (128): Convolutional Layer (128, 1×1) is a convolutional layer with 1×1 kernel and 128 filters. A 1×1 kernel indicates that this layer only processes one pixel at a time without changing the spatial dimensions of the feature map. 128 filters indicate that this layer produces 128 new feature maps from the input. This layer is added after Block C3 to perform two main functions. Firstly, this layer performs a linear transformation of the feature map from Block C3. This transformation can be seen as a matrix multiplication operation, where the kernel weight matrix acts as a linear transformation. This transformation allows the model to learn complex non-linear relationships between feature maps, resulting in richer and more informative representations. The second function is that this layer can help in adjusting the dimensions of the feature map. In the YOLOv5 architecture, the feature map dimensions are changed at several stages. The addition of a 1×1 convolutional layer allows the model to flexibly adjust the dimensions of the feature map, ensuring compatibility between stages and improving computational efficiency.
- Added convolutional layer after layer 9×C3 (512):

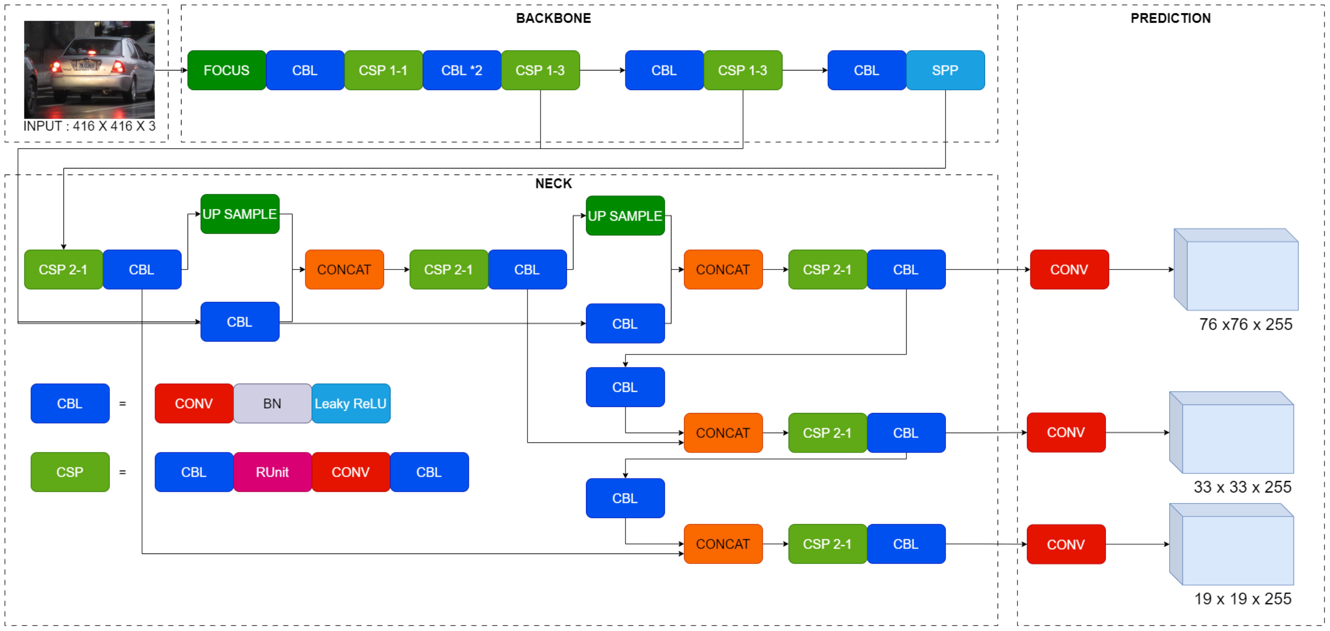


Figure 4. YOLO v5 Architecture

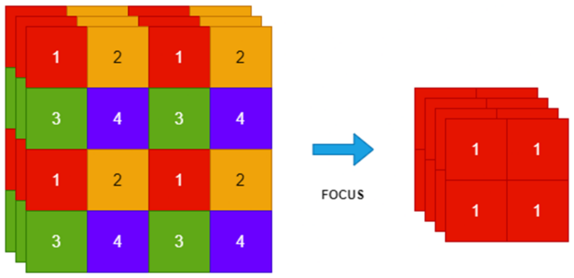


Figure 5. Focus Process

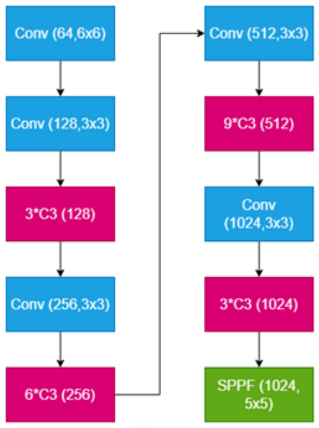


Figure 6. YOLO v5 Backbone

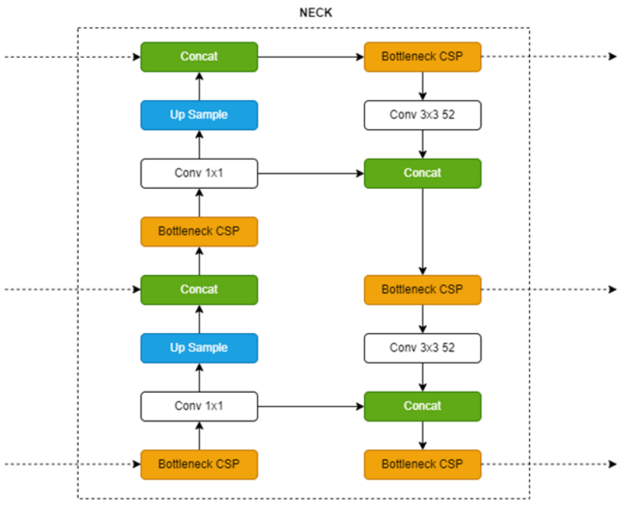


Figure 7. YOLO v5 neck illustration

Convolutional layer (512 filters, kernel size 1×1) consists of a 1×1 kernel applied to the previous feature map. This kernel acts as a filter that processes each pixel individually. This layer produces a new feature map with 512 channels, where the number of 512 filters determines the output dimensions of this layer. This layer has two main functions. The first is that it helps change the dimensions of the feature map before proceeding to the next stage in the architecture. Although the 1×1 kernel does not change the spatial dimensions (height and width) of the feature map, this layer can change the number of feature map channels. This allows the model to produce a more concise or richer representation of the data. Second, although the 1×1 kernel does not pro-

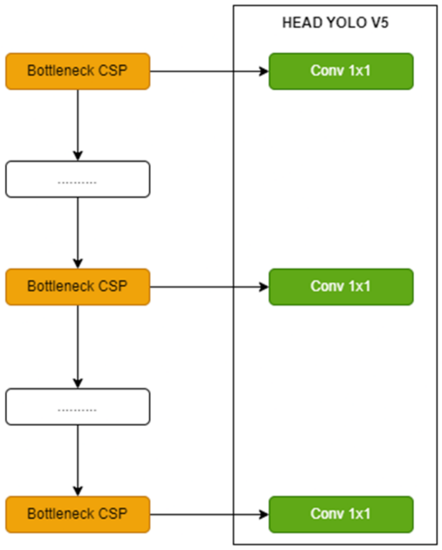


Figure 8. YOLO v5 head illustration

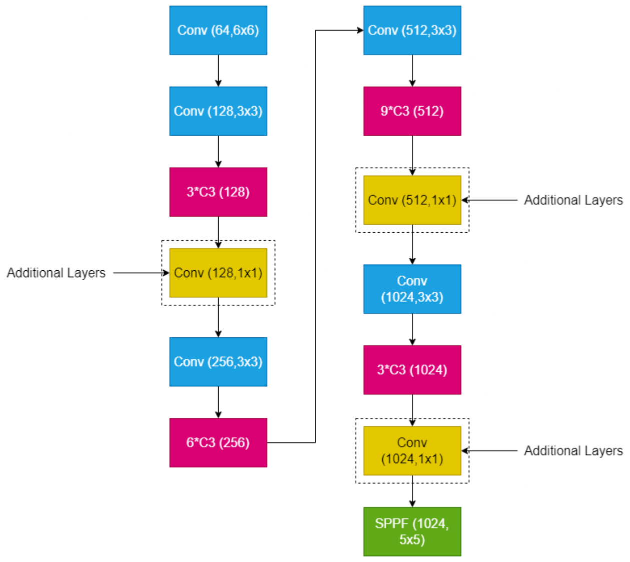


Figure 9. Modifications to the YOLO v5 Backbone

cess spatial information, this layer is able to extract high-level features from the feature map. Through matrix multiplication operations with trained kernels, this layer identifies non-linear relationships between channels and produces more complex and informative feature representations.

- Added convolutional layer after layer 3×C3 (1024): Convolutional Layer (1024, 1×1) is a convolution layer with 1×1 kernel and 1024 filters. A 1×1 kernel indicates that this layer only processes one pixel at a time without changing the spatial dimensions of the feature map. 1024 filter indicates that this layer produces 1024 new feature maps from the input. This layer is added after Block C3 to perform two main functions; the first is that this layer performs a linear transformation of the feature map from Block C3. This transformation can be seen as a matrix multiplication operation, where the kernel weight matrix acts as a linear transformation. This transformation allows the model to learn complex non-linear relationships between feature maps, resulting in richer and more informative representations. Second, this layer can help in adjusting the dimensions of the feature map. In the YOLO v5 architecture, the feature map dimensions are changed at several stages. The addition of a 1×1 convolutional layer allows the model to flexibly adjust the dimensions of the feature map, ensuring compatibility between stages and improving computational efficiency.

These enhancements improve detection accuracy and robustness compared to the standard YOLOv5 backbone, which, while efficient, may not capture intricate features as effectively.

E. Optical Character Recognition (OCR) Architecture

This study employs EasyOCR as the Optical Character Recognition (OCR) architecture, which is an accessible tool for extracting text from images. EasyOCR relies on deep learning models and employs a variety of techniques to analyze images and extract textual information. The utilized EasyOCR model is pre-trained, having undergone training with diverse datasets to enhance its capability in recognizing text across different visual scenarios. Figure 10 illustrates the architecture of the EasyOCR model.

1) Image Preprocessing

During the image pre-processing stage, which aims to enhance its quality and suitability for text recognition, EasyOCR executes various operations, including noise elimination, binarization, and correction of skewness. These operations are crucial in priming the image for a more precise text recognition process.

2) Detect Regions in Images that Contain Characters

CRAFT (Character-Region Awareness For Text detection) is a highly effective text detection system renowned for its efficiency and precision in identifying text of diverse sizes, orientations, and letter variations. Employing a Convolutional Neural Network (CNN) architecture, this model generates two distinct output maps, namely one for character region scores and another for affinity scores.

3) Preparing Character Recognition

The CRNN architecture includes convolutional layers, recurrent layers, and decoding via CTC (Connectionist Temporal Classification). Convolution Layers function to extract visual features from input images. Typically, multiple convolution layers with different filters are used to capture different levels of visual information. The Recurrent Layer receives the output from the convolution layer and

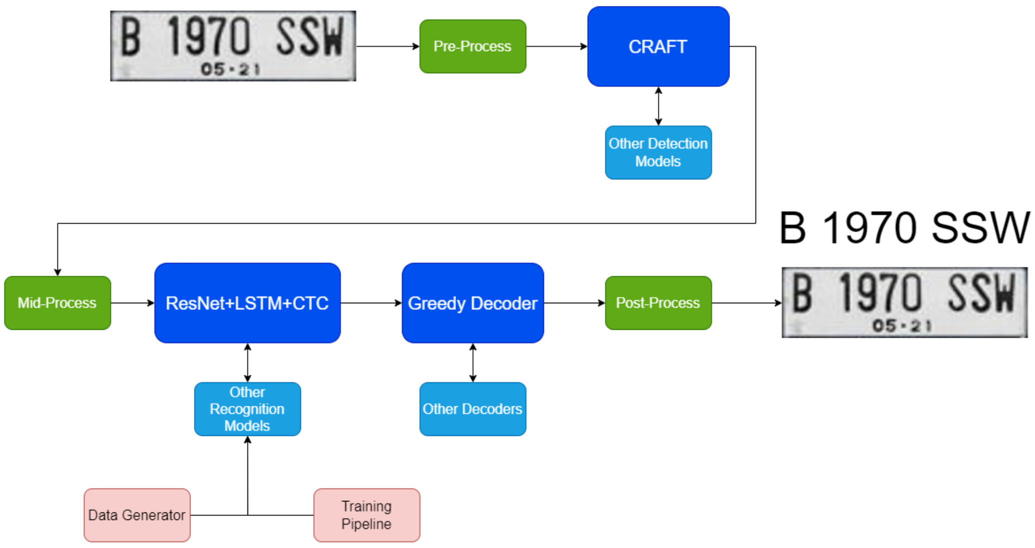


Figure 10. EasyOCR Architecture

processes it sequentially. Recurrent layers, such as LSTM, are able to learn temporal dependencies between characters in a license plate. Decoding converts the output from the recurrent layer into probabilities for each character. CTC (Connectionist Temporal Classification) is a commonly used algorithm for decoding text from images.

4) *Decode Results from Character Recognition*

Decoding, especially in CTC-based decoders, is commonly utilized to determine the most probable character sequence based on the generated output probability. The Greedy decoder employed in this model operates by selecting the character with the highest probability at each sequence step. The recognition model generates a probability distribution across all feasible characters for every character within the sequence. The Greedy decoder straightforwardly selects the character with the highest probability at each step and adds it to the eventually recognized text.

5) *Post-Process Character Recognition*

Post-processing is an important stage in OCR that aims to correct errors and inconsistencies in recognized text and improve the overall quality of the text. Post-processing techniques used include spell-checking, language modelling, and text normalization.

F. *Model Evaluation*

The data that will be tested in this research is image data with various lighting conditions and viewing angles. Image testing is carried out using a resolution level of 416×416 pixels so that the computing process is light, and this process will be carried out on the Google Colaboratory service. The acquired findings include of metrics such as precision, recall, and mAP. In evaluating the performance of the model, researchers compared it with research conducted by [14], [16], [28], [6], [29], and [30]. Then, it will also be compared with the YOLO v5 Flat model (without modification), which has been trained using the AOLP dataset with the same hyperparameter configuration. The results of related studies can be a benchmark for researchers to determine the extent to which the proposed model can excel in vehicle number plate detection and recognition.

4. EXPERIMENTAL AND RESULT ANALYSIS

A. *Experimental Environment and Parameter Settings*

Experiments were conducted using the Google Colab platform, which provides a cloud computing-based research environment with powerful computing resources. Google Colab facilitates research and development in various fields, including deep learning, without the need for expensive local computing infrastructure. In this experiment, Google Colab provides access to system resources that include 51 GB of system RAM and a V-100 type graphics processing unit (GPU) with a RAM capacity of 16 GB.

In the Modified-YOLO v5 training phase, the model is trained using 100 iterations (epochs). These iterations reflect how often the entire training dataset is provided to the model to update and adjust. Additionally, the input image size during training is set to 416×416 pixels with three colour channels (R, G, B), creating an input tensor the size of 416x416x3. In this research, the YOLO hyperparameter configuration was carried out to improve the model's performance in detecting number plate objects in the dataset. Training is carried out using 6 different hyperparameter configurations by changing the learning rate, batch size, momentum, and decay values as shown in Table I. (table) The choice of learning rate is based on balancing model convergence and preventing overshooting. A lower learning rate, such as 0.0001, achieves more accurate convergence. Batch size affects how many samples are used to calculate the gradient. Smaller batch sizes, such as 4, can provide

TABLE I. MODIFIED-YOLO V5 HYPERPARAMETER TUNING

Model	Learning Rate	Batch Size	Momentum	Decay
1	0.01	4	0.9	0.0005
2 [10]		10	0.937	
3		16	0.95	
4 [31]	0.001	4	0.9	
5 [32]		10	0.937	
6		16	0.95	

more accurate results but at a higher computational cost.

Momentum determines how quickly the model accumulates information from previous gradients. Experiments are used to select momentum values that achieve convergence acceleration without significant overshooting. Decay reduces the learning rate over time and prevents the model from overfitting. A low decay value, such as 0.0005, is used to maintain a balance between convergence and generalization.

It is imperative to acknowledge that the hyperparameter configuration for model 2 has been derived from the scholarly work of [10], while the configuration for model 4 is adopted from [31], and the configuration for model 5 is sourced from [32]. In contrast, the hyperparameter configurations for models 1, 3, and 6 represent the proposed settings introduced in the course of this research.

B. *License Plate Detection*

This chapter will present the findings from the performance assessment of the Modified-YOLO v5 model on the specific dataset employed in this study. The YOLO model underwent 100 epochs of training using the training dataset. The evaluation results of the Modified-YOLO v5 model for detecting number plates in six different experiments have been measured using precision, recall, mAP50, and mAP50-95 metrics. The evaluation produces exciting and relevant data for understanding model performance.

TABLE II. VALIDATION RESULTS ON HYPERPARAMETER TUNING MODIFICATION-YOLO V5

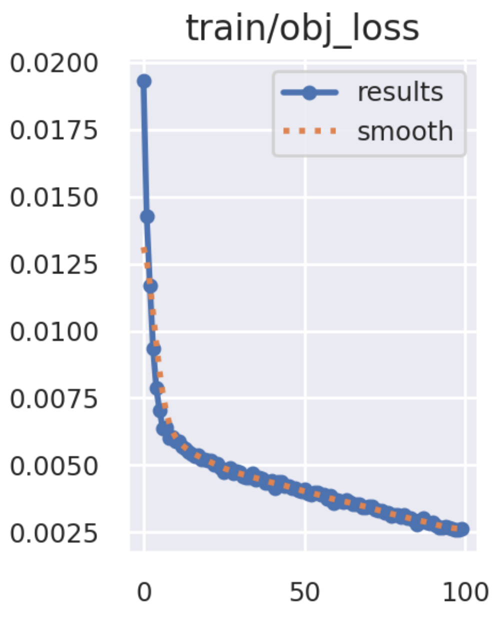
Number of Model	Precision	Recall	mAP50	mAP50-95
1	0.983	0.973	**0.994**	**0.848**
2	**0.994**	0.96	0.993	0.845
3	0.991	**0.997**	**0.994**	**0.848**
4	0.983	0.971	0.993	0.827
5	0.976	0.977	**0.994**	0.833
6	0.977	0.983	**0.994**	0.837

The results from the experiments presented in Table II highlight the impressive efficacy of the Modified-YOLO v5 model in identifying license plates. The remarkably high precision levels, ranging from 0.976 to 0.994, indicate that the majority of detections performed by the model are accurate. High precision means the model rarely gives false alarms. This is important in applications such as law enforcement, where detection errors can cause inconvenience or law enforcement errors.

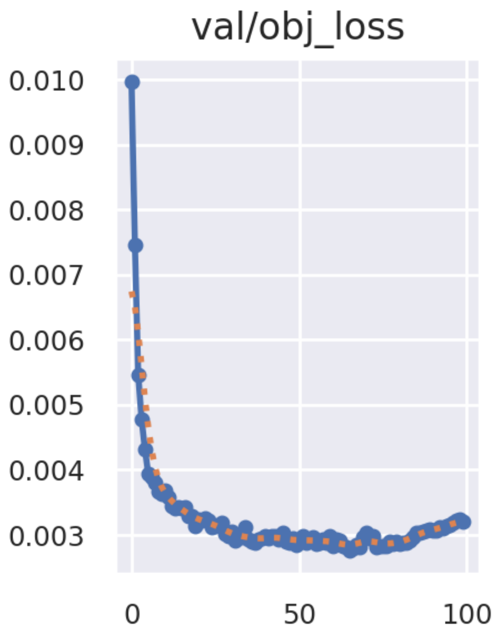
The high recall rate, ranging from 0.96 to 0.997, indicates that the model tends to find most existing license plate instances. High recall means the model can detect almost all existing objects. This is especially important in applications such as traffic monitoring, where failure to detect violations can have fatal consequences. The very high level of accuracy of mAP50, reaching a range of 0.993 to 0.994, shows that the model effectively identifies and determines bounding boxes with a high level of accuracy. This confirms the model's ability to provide predictions with a high level of confidence in license plate detection at an IoU of 0.5. The mAP50-95 results remain high, ranging from 0.827 to 0.848, indicating the model's ability to detect license plates at various IoU levels. This shows the model's consistency in providing good predictions even at higher IoU levels, which is often a challenge in object detection tasks. In the experimental results, one configuration stands out, namely in the third experiment. This experiment achieved the highest performance with a recall value of 0.997, mAP50 of 0.994, and mAP50-95 of 0.848. These results indicate that the hyperparameter configuration in the third experiment made a positive contribution to improving the model's ability to detect number plates.

In Figure 11(a), it can be seen that the training loss graph shows a significant decreasing trend from the start of training. Starting with a value of 0.02, there was a rapid decline until it reached a value of 0.006 at the 10th epoch. Next, a slower decline was seen until it reached a value of 0.0025 at the 100th epoch. This indicates that the model effectively learns from the training data during the training process. Meanwhile, in Figure 11(b), the validation loss graph also shows a very fast decline from the initial value of 0.10 to a value of 0.004 in the epoch 1-10 range. However, after the 10th epoch, the validation loss graph experienced insignificant fluctuations. Even though there were fluctuations, the lowest value was recorded at the 60th epoch with a value of 0.001 before then increasing to reach a value of 0.0034 at the 100th epoch. This indicates that the model is able to generalize well to data that has never been seen before. Judging from the comparison of the two graphs, there are slight signs of overfitting in the training data, namely when the model focuses its learning too much on specific training data and loses the ability to generalize to new data. However, the impact of overfitting does not directly affect the overall quality of the model.

After completing training, the next stage involves testing the Modified-YOLO v5 model using testing data. The dataset used consists of 206 images, which are taken from the AOLP dataset and have never been seen by the model before. To match the input size of the Modified YOLO v5 model, each image was resized to 416×416 pixels. In the



(a)



(b)

Figure 11. (a) Training Loss Results and (b) Validation Loss Results

test results, the model was able to detect number plates well in 200 images out of a total of 206 images, resulting in a testing accuracy rate of 97.08%.

Figure 12 visually presents sample detection outcomes, showcasing the model's capability to recognize and annotate number plates in test images. In Figure 12(a), the model succeeds in detecting number plates well in busy traffic, showing its ability to detect relatively small plates. Meanwhile, in Figure 12(b), the model is able to identify number plates on quieter roads, showing its reliability in detecting up to three vehicles at once. Figure 12(c) shows the success of the model in detecting number plates in the parking area from a side view. Not only that, in Figure 12(d), the model remains effective in detecting number plates even in low light conditions, such as in the image with two cars and one number plate exposed to inadequate light.

C. Comparative Study

A series of comparison experiments were carried out to evaluate the performance of the Modified-YOLOv5 model proposed in this study regarding license plate detection. In this context, Modified-YOLO v5 is compared with a number of other YOLO algorithms. Experimental results, including precision and recall values, are presented in Table III.

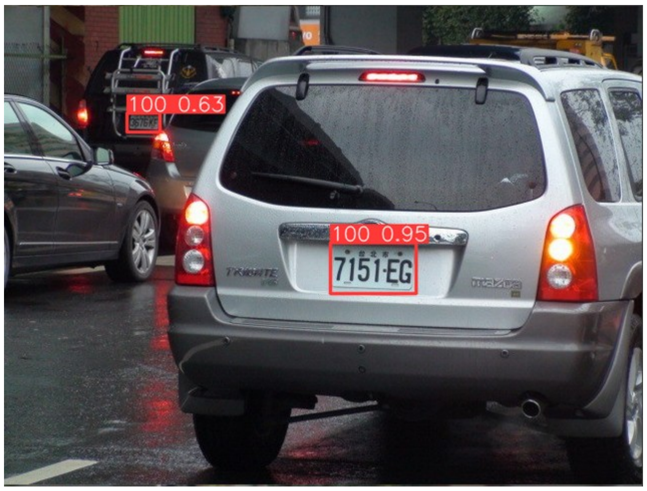
When compared to the CNN model, the modified-YOLO v5 model shows a significant performance increase, with Precision increasing by 1.92% and Recall increasing by 2.51%. This means that the modified-YOLO v5 model is better able to produce accurate predictions and detect most of the objects that should be identified.

TABLE III. COMPARISON OF MODIFIED-YOLO V5 RESULTS WITH BASELINE

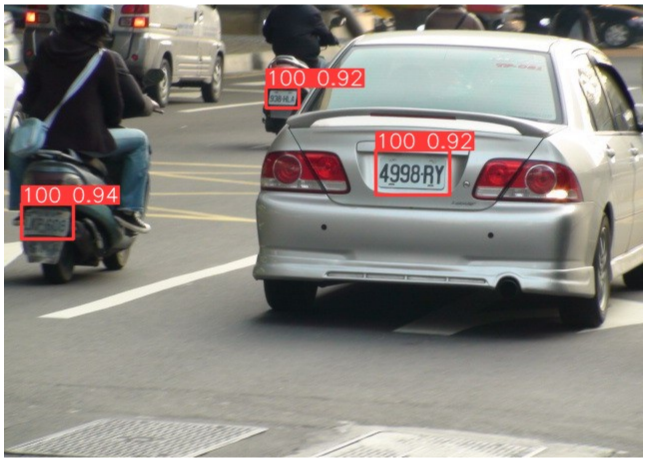
Model	Precision	Recall
CNN	97.18%	97.19%
Fast-YOLO v2	-	99.45%
SWSCD-YOLO Darknet	98.2%	97.9%
YOLO v5	98.6%	96.7%
Modified-YOLO v5 (proposed)	**99.1%**	**99.7%**

Compared with the Fast-YOLO v2 model, the proposed model, namely Modified-YOLO v5, shows an increase in Recall of 0.25%. These results indicate that Modified-YOLO v5 is more effective in detecting and recognizing objects overall when compared to Fast-YOLO v2. Meanwhile, when compared with the SWSCD-YOLO Darknet model, the Modified-YOLO v5 model produces an increase in both Precision by 0.9% and Recall by 1.8%. This indicates that Modified-YOLO v5 is not only more accurate in providing positive predictions but also more efficient in detecting the majority of existing target objects when compared to SWSCD-YOLO Darknet. The Modified-YOLO v5 model was also compared with the YOLO v5 original (without modification), which had been trained using the AOLP dataset with the same hyperparameter configuration, Results of the Modified-YOLO v5 showed an increase in precision of 0.5%, and recall of 3%.

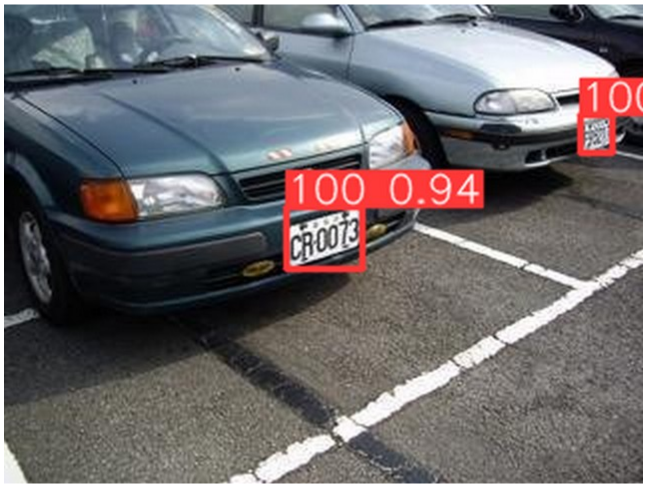
Through this comparison, it can be seen that Modified-YOLO v5 as the proposed model consistently shows a higher level of accuracy compared to the baseline model,



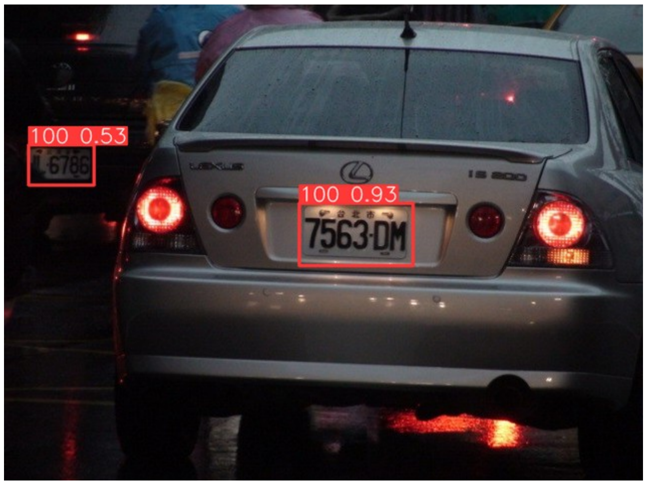
(a)



(b)



(c)



(d)

Figure 12. Number Plate Detection Testing

indicating its potential in license plate detection. These results provide a solid basis to highlighting the contribution and advantages of Modified-YOLO v5 in the context of license plate object detection.

D. License Plate Recognition

Before applying EasyOCR, we applied license plate image extraction using the Modified-YOLO v5 model. This process aims to improve image quality and ensure the number plate region is clearer. After obtaining the license plate image, the following preprocessing steps are applied to improve the writing detection and recognition performance:

- Resize: The image is resized to 600×480 pixels. This is done so that the image size matches the input size expected by the EasyOCR model. Resizing helps ensure consistency and suitability to detection model requirements.
- Grayscale: The image is converted to grayscale format. Conversion to a grayscale is carried out to reduce the complexity of the writing detection process. In grayscale format, color information is no longer needed, and the basic structure of objects, such as writing, can be easily identified. In Figure 13, the outcome of this grayscale operation is visually depicted. The converted grayscale image serves as a foundational step in enhancing the efficiency of subsequent writing detection algorithms.
- Binarization: involves converting the image into binary format through a specific method. This transformation results in an image where the object, such as text, is isolated from the background, creating a clear contrast. The primary objective is to facilitate the detection and recognition of text by enhancing its visibility against the background. As depicted in

Figure 14, this binarization technique enables a distinct separation between the text and its surroundings, thus providing a solid foundation for improved text detection algorithms.

- Text Detection: At this stage, text detection is carried out using EasyOCR, where the model produces output in the form of text that matches the image. Figure 15 shows the result of the text detection process, clearly showing the inscription on the number plate as ”7456TH”. EasyOCR managed to detect it with good accuracy.

The final stage was the testing stage, using 70 images that had not previously been seen by the YOLO model. From the test results, 69 images were successfully read correctly by the EasyOCR model, achieving an accuracy level of 98.57% as shown in Table IV. The EasyOCR model shows significant improvements when compared to several baseline models used.

TABLE IV. COMPARISON OF EASYOCR RESULTS WITH BASELINE

Model	Accuracy
EasyOCR [29]	80%
EasyOCR [30]	95%
EasyOCR (proposed)	**98.57%**

Furthermore, as shown in Table IV, the proposed EasyOCR model achieves a remarkable accuracy of 98.57%, significantly outperforming previous implementations. Specifically, the proposed EasyOCR model demonstrates an 18.57% improvement in accuracy compared to the implementation by [29], which achieved an accuracy of 80%. Additionally, it surpasses the accuracy reported by [30], which was 95%, showing a notable increase of 3.57%. These substantial improvements indicate that the proposed EasyOCR model is far more effective in recognizing license plate characters than earlier versions.

The results of text detection and recognition using the EasyOCR model can be seen in Figure 4.14. Figure 16(a) with the number plate ”DL2229” is correctly predicted by the model, while figure 16(b) with the number plate ”LI8850” is incorrectly predicted as ”LF8850” by the EasyOCR model. The error in predicting the right image is caused by the presence of black noise under the letter ”I.” This noise causes the EasyOCR model to predict the character as the letter ”f.” The presence of noise in the image can affect the performance of the EasyOCR model by making character interpretation less accurate.

5. Conclusions and Future Work

The conclusion of a series of experiments carried out in this research shows positive results in the development of a number plate detection and recognition system using the Modified-YOLO v5 and EasyOCR models. The implemented YOLO v5 model modifications succeeded in achieving a very good level of accuracy, with a recall value of 0.997, precision of 0.991, mAP50 of 0.994, and mAP50-95 of 0.848. In object detection, the Modified-YOLO v5 model experienced an increase in precision by 0.9% and recall by 0.25% compared to the baseline models, namely SWSCD-YOLO Darknet and Fast-YOLO v2. In addition, the test results also show that the Modified-YOLO v5 model is able to overcome the challenges of detecting small license plate objects, confirming the model’s reliability in handling real-world scenarios where license plates can appear in various sizes and conditions. Meanwhile, the EasyOCR model used for character recognition on number plates achieved an accuracy level of 98.57%, which experienced an increase in accuracy of 3,57% compared to the baseline model.

For future research, considering computational limitations, it is recommended to train the YOLO model with a more extensive and more diverse dataset. Large datasets that include license plates from various countries will enable license plate detection models to become more reliable and able to cope with a wide variety of license plates around the world. Thus, the use of a more diverse dataset will help increase the generalization of the model so that the model can be more effective in detecting and recognizing license plates in a variety of different conditions and environments.

References

[1] S. Jagtap, “Analysis of feature extraction techniques for vehicle number plate detection,” *International Journal of Computer Science and Information Technologies*, vol. 6, no. 6, pp. 5342–5346, 2015.

[2] S. K. Satapathy, S. Mishra, R. S. Sundeep, U. S. R. Teja, P. K. Mallick, M. Shruti, and K. Shravya, “Deep learning based image recognition for vehicle number information,” *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 8, pp. 52–55, 2019.

[3] P. Rathore, P. Gupta, S. Jain, and Y. Shrivastava, “A study of the automated vehicle number plate recognition system,” *i-manager’s Journal on Pattern Recognition*, vol. 9, no. 2, p. 30, 2022.

[4] K. M. Babu and M. Raghunadh, “Vehicle number plate detection and recognition using bounding box method,” in *2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*. IEEE, 2016, pp. 106–110.

[5] S. T. Bow, *Pattern recognition and image preprocessing*. CRC press, 2002.

[6] R.-C. Chen *et al.*, “Automatic license plate recognition via sliding-window darknet-yolo deep learning,” *Image and Vision Computing*, vol. 87, pp. 47–56, 2019.

[7] G.-S. Hsu, J.-C. Chen, and Y.-Z. Chung, “Application-oriented license plate recognition,” *IEEE transactions on vehicular technology*, vol. 62, no. 2, pp. 552–561, 2012.

[8] B. Setiyono, D. A. Amini, and D. R. Sulistyaningrum, “Number plate recognition on vehicle using yolo-darknet,” in *Journal of Physics: Conference Series*, vol. 1821, no. 1. IOP Publishing, 2021, p. 012049.

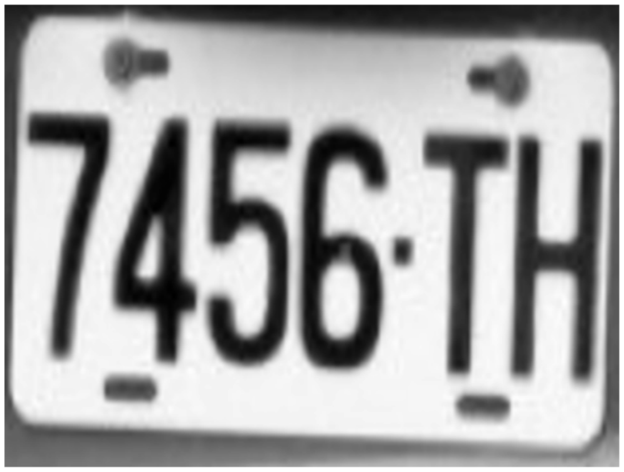


Figure 13. Grayscale Image

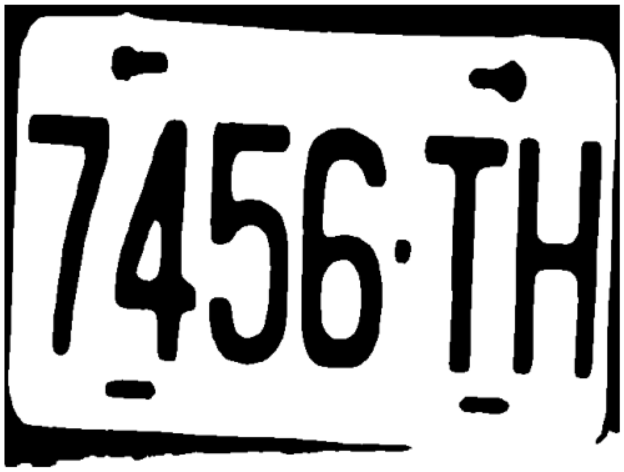


Figure 14. Binarization Image

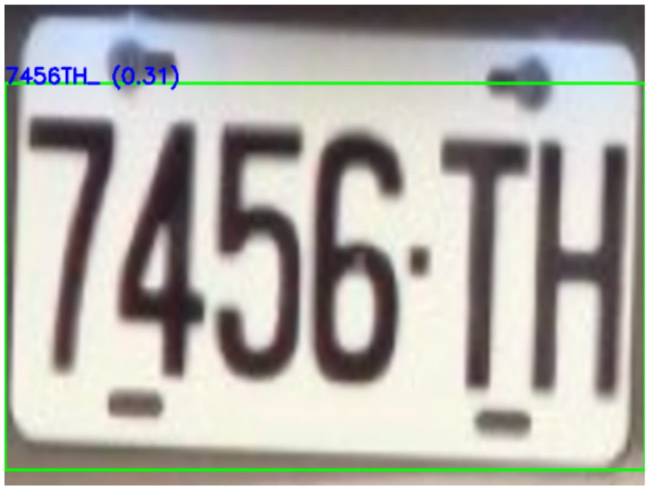


Figure 15. Text Detection

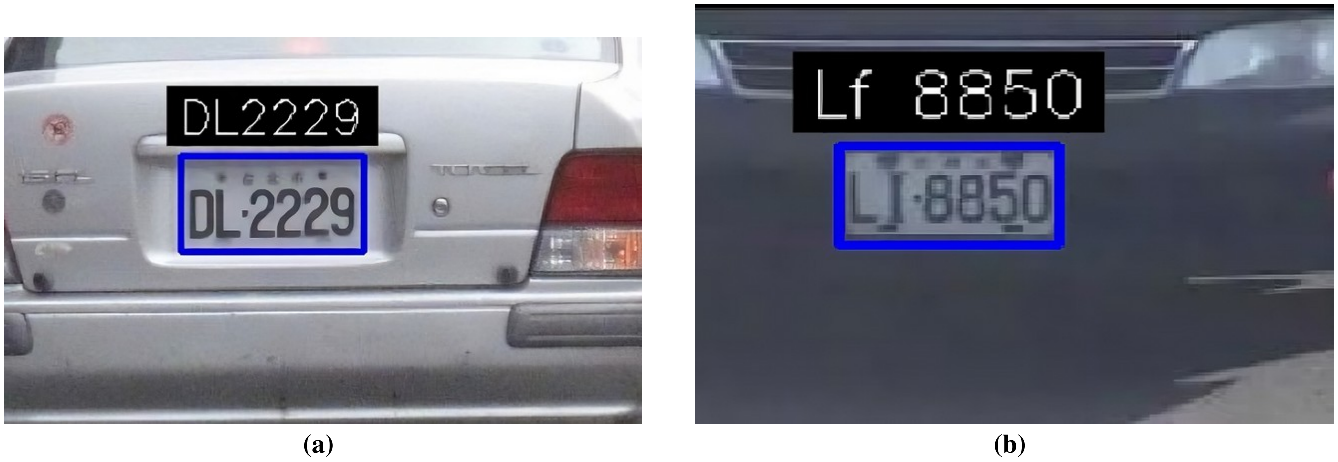


Figure 16. Number Plate Detection Testing

[9] S.-H. Park, S.-B. Yu, J.-A. Kim, and H. Yoon, "An all-in-one vehicle type and license plate recognition system using yolov4," *Sensors*,

vol. 22, no. 3, p. 921, 2022.

[10] C. Wei, Z. Tan, Q. Qing, R. Zeng, and G. Wen, “Fast helmet and license plate detection based on lightweight yolov5,” *Sensors*, vol. 23, no. 9, p. 4335, 2023.

[11] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, “Imagenet large scale visual recognition challenge,” *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[12] A. John and D. Meva, “A comparative study of various object detection algorithms and performance analysis,” *International Journal of Computer Sciences and Engineering*, vol. 8, no. 10, pp. 158–163, 2020.

[13] U. Nepal and H. Eslamiat, “Comparing yolov3, yolov4 and yolov5 for autonomous landing spot detection in faulty uavs,” *Sensors*, vol. 22, no. 2, p. 464, 2022.

[14] H. Li and C. Shen, “Reading car license plates using deep convolutional neural networks and lstms,” *arXiv preprint arXiv:1601.05610*, 2016.

[15] N. Sharma, M. A. Haq, P. K. Dahiya, B. Marwah, R. Lalit, N. Mittal, and I. Keshta, “Deep learning and svm-based approach for indian licence plate character recognition.” *Computers, Materials & Continua*, vol. 74, no. 1, 2023.

[16] R. Laroca, L. A. Zanlorensi, G. R. Gonçalves, E. Todt, W. R. Schwartz, and D. Menotti, “An efficient and layout-independent automatic license plate recognition system based on the yolo detector,” *IET Intelligent Transport Systems*, vol. 15, no. 4, pp. 483–503, 2021.

[17] H. Sun, M. Fu, A. Abdussalam, Z. Huang, S. Sun, and W. Wang, “License plate detection and recognition based on the yolo detector and crnn-12,” in *Signal and Information Processing, Networking and Computers: Proceedings of the 4th International Conference on Signal and Information Processing, Networking and Computers (ICSINC) 4th*. Springer, 2019, pp. 66–74.

[18] S. Arora, R. Mittal, D. Arora, and A. K. Shrivastava, “A robust approach for licence plate detection using deep learning,” *Inteligencia Artificial*, vol. 27, no. 73, pp. 129–141, 2024.

[19] D. J. Pangal, G. Kugener, S. Shahrestani, F. Attenello, G. Zada, and D. A. Donoho, “A guide to annotation of neurosurgical intraoperative video for machine learning analysis and computer vision,” *World Neurosurgery*, vol. 150, pp. 26–30, 2021.

[20] R. P. Thangaraj Sundaramurthy, Y. Balasubramanian, and M. Annamalai, “Real-time detection of fusarium infection in moving corn grains using yolov5 object detection algorithm,” *Journal of Food Process Engineering*, vol. 46, no. 9, p. e14401, 2023.

[21] S. Li, Y. Li, Y. Li, M. Li, and X. Xu, “Yolo-firi: Improved yolov5 for infrared image object detection,” *IEEE access*, vol. 9, pp. 141 861–141 875, 2021.

[22] Y. Chen, Z. Xiao, L. Zhao, L. Zhang, H. Dai, D. W. Liu, Z. Wu, C. Li, T. Zhang, C. Li *et al.*, “Mask-guided vision transformer (mg-vit) for few-shot learning,” *arXiv preprint arXiv:2205.09995*, 2022.

[23] B. Kovalenko, V. Lukin, S. Kryvenko, V. Naumenko, and B. Vozel, “Bpg-based automatic lossy compression of noisy images with the prediction of an optimal operation existence and its parameters,” *Applied Sciences*, vol. 12, no. 15, p. 7555, 2022.

[24] X. Fu, A. Li, Z. Meng, X. Yin, C. Zhang, W. Zhang, and L. Qi, “A dynamic detection method for phenotyping pods in a soybean population based on an improved yolo-v5 network,” *Agronomy*, vol. 12, no. 12, p. 3209, 2022.

[25] J. Zhang, Y. Liu, D. Zhang, H. Guo, M. Huang, W. Wang, C. Lin, and C. Zhang, “Detection method of the secondary protective rope for electric power workers based on uav image and yolo algorithm,” in *Proceedings of the 2023 6th International Conference on Signal Processing and Machine Learning*, 2023, pp. 182–189.

[26] M. Yasir, L. Shanwei, X. Mingming, S. Hui, M. S. Hossain, A. T. I. Colak, D. Wang, W. Jianhua, and K. B. Dang, “Multi-scale ship target detection using sar images based on improved yolov5,” *Frontiers in Marine Science*, vol. 9, p. 1086140, 2023.

[27] S. Wu, X. Wang, and C. Guo, “Application of feature pyramid network and feature fusion single shot multibox detector for real-time prostate capsule detection,” *Electronics*, vol. 12, no. 4, p. 1060, 2023.

[28] S. M. Silva and C. R. Jung, “License plate detection and recognition in unconstrained scenarios,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 580–596.

[29] V. F. W. Benfano Soewito, “Efficient license plate detection and recognition with yolov7 and ocr,” *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 3, p. 1598–1605, Mar. 2024. [Online]. Available: https://ijisae.org/index.php/IJISAE/article/view/5558

[30] N. M. D. S. B. Rayudu Sushma, Madhuri Rithika Devi, “Automatic license plate recognition with yolov5 and easy-ocr method,” *International Journal of Innovative Research in Technology*, vol. 9, no. 1, 2022.

[31] R. V. Iyer, P. S. Ringe, and K. P. Bhensdadiya, “Comparison of yolov3, yolov5s and mobilenet-ssd v2 for real-time mask detection,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 08, no. 07, 2021.

[32] I. S. Isa, M. S. A. Rosli, U. K. Yusof, M. I. F. Maruzuki, and S. N. Sulaiman, “Optimizing the hyperparameter tuning of yolov5 for underwater detection,” *IEEE Access*, vol. 10, p. 52818–52831, 2022. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2022.3174583