JRAEE

Journal of Robotics, Automation, and Electronics Engineering
Vol. 2, No. 1, 2024, pp. 11-19
e-ISSN 3025-3780
https://journal.uny.ac.id/publications/jraee

# Adaptive Bounding Box Coordinate Adjustment on License Plate Character Detection Using Machine Learning

Ahmad Taufiq Musaddid[a,1,*]

[a] Universitas Negeri Yogyakarta
[1] ahmadtaufiqmusaddid@uny.ac.id
[*] Corresponding Author

ARTICLE INFO

**Article History**
Received 23 Nov. 2023
Revised 28 Jan. 2023
Accepted 25 Feb. 2024

**Keywords**
ANPR;
Machine Learning;
CNN;
Object Detection;
License Plate

ABSTRACT

Effective law enforcement, including the use of the ANPR (Automatic Number Plate Recognition) system, is essential for reducing the number of road traffic accidents. ANPR involves plate localization, character segmentation, and recognition to build a minimum system. This study aims to improve a character segmentation method using a detection approach to address issues like noisy or modified plates. We propose an adaptive improvement on an established sliding window technique, by integrating a CNN (Convolutional Neural Network) for bounding box coordinate adjustment to handle various plate conditions. The proposed method was tested on 280 license plate images and improved the average IoU (Intersection over Union) from 0.4811 to 0.8980. Hence, the recall and precision of the model could be improved to increase any character recognition performance.

Penegakan hukum yang efektif, termasuk penggunaan sistem ANPR (*Automatic Number Plate Recognition*) sangat penting untuk mengurangi jumlah kecelakaan lalu lintas. ANPR melibatkan lokalisasi pelat nomor, segmentasi karakter, dan pengenalan untuk membangun sistem minimum. Penelitian ini bertujuan untuk meningkatkan metode segmentasi karakter yang menggunakan pendekatan deteksi untuk mengatasi masalah seperti pelat nomor yang berderau atau dimodifikasi. Kami mengusulkan peningkatan adaptif pada teknik *sliding wondow* yang sudah ada, dengan mengintegrasikan CNN (*Convolutional Neural Network*) untuk penyesuaian koordinat *bounding box* guna menangani berbagai kondisi pelat nomor. Metode yang diusulkan diuji dengan 280 gambar pelat nomor dan mampu meningkatkan rata-rata IoU (*Intersection over Union*) dari 0,4811 menjadi 0,8980. Sehingga performa model seperti *recall* dan *precision* dapat ditingkatkan untuk mendukung performa pengenalan karakter di masa mendatang.

## 1. Introduction

The rise in accident rates in Indonesia suggests a lack of public awareness regarding traffic safety. According to BPS (Badan Pusat Statistika), there were 109 thousand vehicle accidents in 2018, increasing to 139 thousand cases in 2022 [1]. To curb this trend, consistent monitoring and enforcement of traffic regulations are imperative. Rezapour et al. [2] highlighted that such measures can effectively lower accident rates. Hence, the deployment of automatic surveillance technology is crucial to support law enforcement efforts in this regard, such as ANPR (Automatic Number Plate Recognition).

To implement ANPR, there are 3 general steps to be considered. The first step is to localize the plate that is being analyzed on a vehicle. The second is to segment the characters on the plate. And last is to recognize the characters that are extracted. The whole step should yield all information in the form of a short list of characters or even a string. Hence, the latter step's performance is decided by the performance of the previous steps. In this study, the main focus is the character segmentation step but with a detection approach.

Several studies have conducted character segmentation on license plates using machine learning, with a segmentation or detection approach. In [3], a combination of sliding windows and CNN (Convolutional Neural Network) was employed for character detection and recognition on Taiwanese vehicle plates. The CNN used was a modified version of YOLO (You Only Look Once), with a reduction in the number of CNN layers. The concept utilized was SWSCD (Sliding Window Single Class Detection), where each extracted area is fed into a single class classifier, thus employing multiple classifiers for character detection and recognition. In [4], Mask-RCNN was utilized for character detection and recognition on vehicle plates. The detection implemented in Mask-RCNN operates at the pixel level, using Resnet-50 architecture as the backbone. However, even though this object detection approach could handle plate noise, it wasn't sufficient to handle the overlapping characters. Researchers [5] conducted character segmentation on Chinese vehicle plates with 7 characters, using a concept of detecting areas between characters. They utilized a CNN inspired by Darknet-19 architecture, reducing it to 18 convolutional layers and adding an output layer with 6 neurons to determine 6-character boundaries. The limitation of this study arises if the plate has fewer or more than 7 characters. In [6], sliding windows and CNN were employed for character recognition on Indonesian plates, using a mechanism that detects characters among 36 actual character classes. Variations in sliding window sizes and positions were considered based on average character sizes obtained from measuring character contours on a binary plate image. Additionally, the CR-NET framework, inspired by Fast-YOLO, was adapted for character detection on Brazilian vehicle plates in [7], while a modified version of YOLOv3 was used for Bangladeshi vehicle plates in [8], both employing one-stage detection approaches without any area proposal technique.

One of the problems of license plate character detection is the condition of the observed plates. A noisy or defective plate could reduce the performance of any ANPR system. These plate conditions are commonly found in Indonesia which has high mobility of traffic. Furthermore, in some cases, license plates in Indonesia are modified into nonstandard forms. To tackle this problem, an adaptive approach to detect characters on the variation condition of the captured plate can be used. This research aims to create adaptive adjustment on an established license plate character detection method and to improve the performance. The established method is sliding windows was utilized in some mentioned studies. We added one more step to create a CNN to do regression on RoI (Region of Interest) of detection, such that the fixed-size bounding boxes could be more adaptive and capable of handling various conditions and perspectives of captured plates.

## 2. Methodology

### 2.1 Data Collection and Preparation

To accommodate two main activities, training and testing, the dataset used in this study is divided into two types. The first is the training dataset and the second is a testing dataset. The dataset contains 740 cropped black Indonesian license plate images with a size of 300×94 pixels. For training, 460 images are taken, which contain 3503 characters. And the rest, 280 images which contain 2026 characters are used for testing. The ground truth of the dataset is a collection of $(x_1, x_2, y_1, y_2)$ coordinates of each character on a license plate.

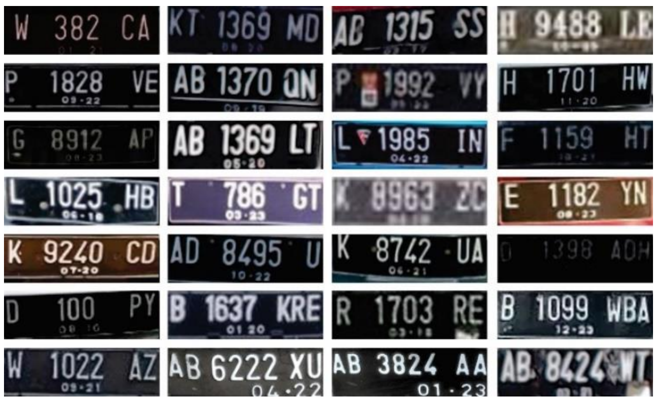

**Fig. 1.** Training set

**Table 1.** Dataset

| Dataset | Total | |
|---|---|---|
| | **Plate** | **Character** |
| Training | 460 | 3503 |
| Testing | 280 | 2026 |

## 2.2 Research Design and Implementation

A sliding window technique is utilized to detect characters on a license plate. This sliding window provides regions to a CNN classifier. This CNN determines whether a region is a character or not. The size of the sliding window is fixed at 25×95 pixels with a stride of 1 pixel. A window sweeps from left to right to extract and propose regions to the classifier.

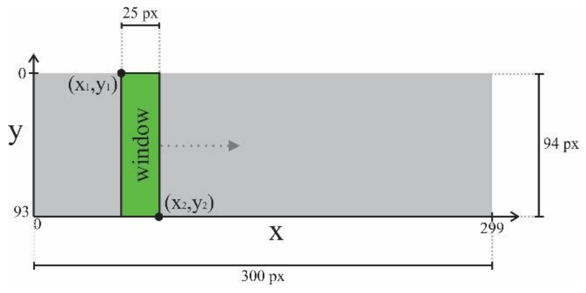

**Fig. 2.** Visualization of sliding window

The classifier (CNN) uses the binary classification concept as it only indicates if a region is considered a character or not. The total number of proposed regions is 274, these regions are fed into the CNN one by one. From these 274 classified regions, we need to associate each potential region with a character's estimated location. To solve this, a clustering algorithm like HAC (Hierarchical Agglomerative Clustering) is used to cluster each potential region into groups of related characters by calculating all distances.

The mentioned detection pipeline is called the detector in this paper from this point. To accomplish the improvement of this detector, a method is proposed as a continuation of our previous work [9]. This method uses additional CNN to do regression on detected characters. The regression is focused on the coordinates of bounding boxes. For further explanation, this method will be discussed in the following sections as a bounding box regressor. For the implementation of all processes, we use the TensorFlow framework and Python programming language.

## 2.3 Proposed Method

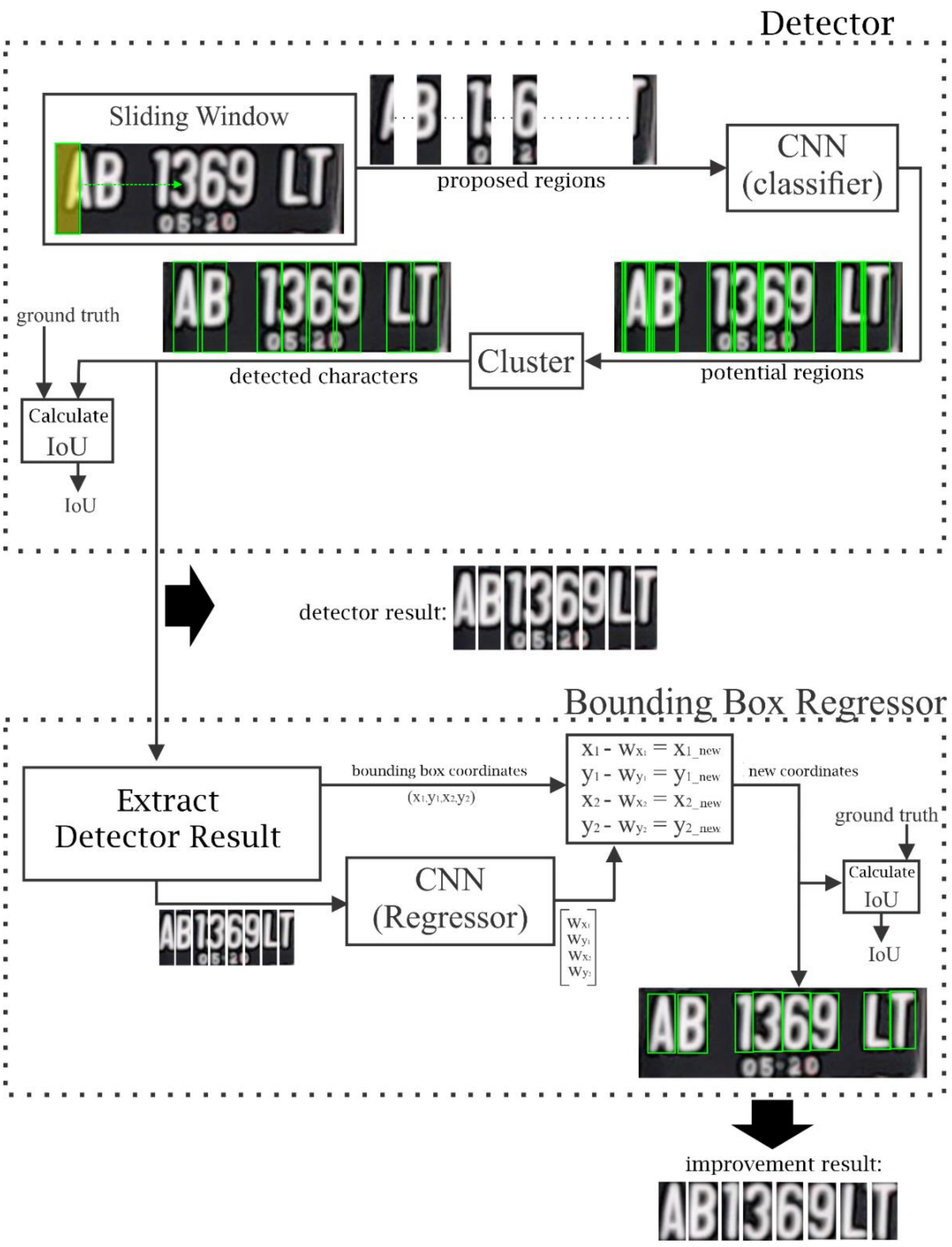

**Fig. 3.** Proposed Method

The bounding box produced by the character detector has a fixed size. This will have an impact on the low level of overlap of the bounding box with the GT (Ground Truth). To handle this problem, bounding box coordinate regression is carried out to increase IoU (Intersection over Union). The input of this regression is the spatial sub-area of a plate image extracted from the previous bounding box results. The output from the regression is four shift values ($W_{x1}, W_{x2}, W_{y1}, W_{y2}$) which are used to improve the bounding box coordinates. The 4 values are designated for ($\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y_1}, \boldsymbol{y_2}$).

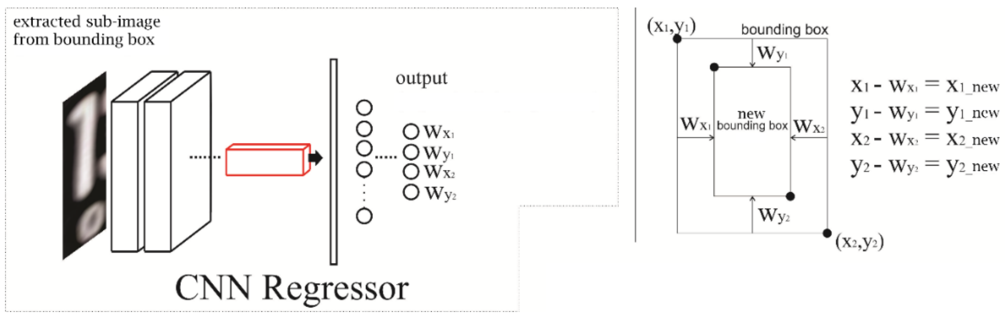
extracted sub-image from bounding box
output
$W_{x_1}$
$W_{y_1}$
$W_{x_2}$
$W_{y_2}$
CNN Regressor
$(x_1,y_1)$
bounding box
$W_{y_1}$
$W_{x_1}$
new bounding box
$W_{x_2}$
$W_{y_2}$
$(x_2,y_2)$
$x_1 - W_{x_1} = x_{1\_new}$
$y_1 - W_{y_1} = y_{1\_new}$
$x_2 - W_{x_2} = x_{2\_new}$
$y_2 - W_{y_2} = y_{2\_new}$

**Fig. 4.** Illustration of predicting 4 shift values and bounding box improvements

Fig. 4 shows the bounding box improvement mechanism utilizing regression results from CNN. Initially, a regressor in the form of a CNN is used to predict the coordinate shift values of a bounding box based on the 25×94 plate sub-image within it. These displacement values are then used to update the coordinate values. $(x_1, x_2, y_1, y_2)$ of the bounding box with a subtraction operation. A value in a coordinate will increase or decrease depending on the predicted shift, whether the shift is positive or negative. Improvements to the bounding boxes on a plate are carried out iteratively based on the number on the plate.

**Table 2.** CNN Classifier Configuration

| *Layer* | Depth | Size | Size of Filter | Stride | Activation |
|---|---|---|---|---|---|
| Input | 1 | 25×94×1 | - | - | - |
| Convolution | 64 | 23×92×64 | 3×3 | 1 | ReLU |
| Convolution | 64 | 21×90×64 | 3×3 | 1 | ReLU |
| Max pooling | 64 | 10×45×64 | 2×2 | 2 | - |
| Convolution | 128 | 8×43×128 | 3×3 | 1 | ReLU |
| Convolution | 128 | 6×41×128 | 3×3 | 1 | ReLU |
| Max pooling | 128 | 3×20×128 | 2×2 | 2 | - |
| Flatten | - | 7680 | - | - | - |
| Dense | - | 100 | - | - | ReLU |
| Output | - | 2 | - | - | Softmax |

**Table 3.** CNN Regressor Configuration

| *Layer* | Depth | Size | Size of Filter | Stride | Activation |
|---|---|---|---|---|---|
| Input | 1 | 25×94×1 | - | - | - |
| Convolution | 32 | 25×94×32 | 3×3 | 1 | ReLU |
| Convolution | 32 | 25×94×32 | 3×3 | 1 | ReLU |
| Max pooling | 32 | 12×47×32 | 2×2 | 2 | - |
| Convolution | 64 | 12×47×64 | 3×3 | 1 | ReLU |
| Convolution | 64 | 12×47×64 | 3×3 | 1 | ReLU |
| Max pooling | 64 | 6×23×64 | 2×2 | 2 | - |
| Convolution | 128 | 6×23×128 | 3×3 | 1 | ReLU |
| Convolution | 128 | 6×23×128 | 3×3 | 1 | ReLU |
| Convolution | 128 | 6×23×128 | 3×3 | 1 | ReLU |
| Max pooling | 128 | 3×11×128 | 2×2 | 2 | - |
| Flatten | - | 8448 | - | - | - |
| Dense | - | 4096 | - | - | ReLU |
| Dense | - | 4096 | - | - | ReLU |
| Output | - | 4 | - | - | - |

The CNN used in the bounding box regressor has a deeper architecture than the character detector CNN (classifier), where 7 convolution layers and 3 max-pooling layers are used. The convolution layers have 32, 64, and 128 filters. The convolution operation uses zero padding on the input image for each convolution layer to maintain the size of the feature map. The reduction in feature map size is only achieved by max pooling. In FCL (Fully Connected Layer), there is one flattening layer, two dense layers, and one output layer. The activation function is ReLU for convolution and dense layers. Meanwhile, for the output layer, no activation function is used, so the input and output relationship at this layer is $f(x) = x$. The CNN regressor size and hyperparameter configuration are shown in Table 2.

The proposed CNN regressor model learning uses data of the extracted character sub-images from the detector model and the original training data. The regression target in learning is the difference between the bounding box coordinate and the closest ground truth coordinate. Hence, by setting this target, the model is expected to be able to produce values for the difference between the bounding box coordinates and the ground truth coordinates in the test data. In the previous explanation, these difference values are called shift values. This regression target is visualized in Fig. 5.

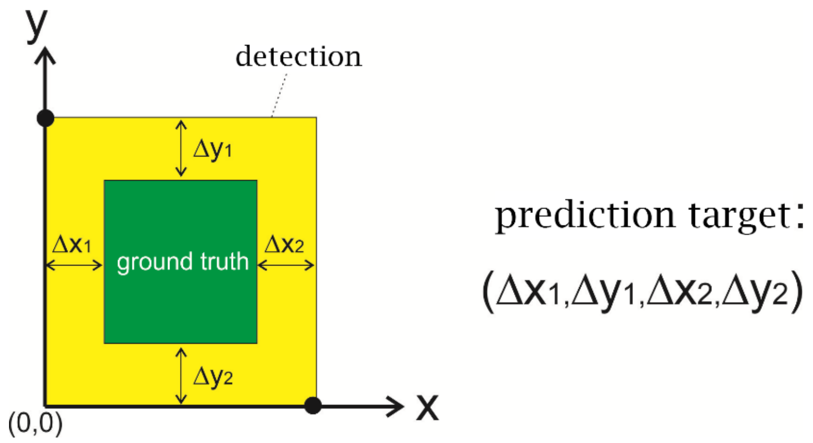

**Fig. 5.** Illustration of prediction target of CNN regressor training

## 2.4 Data Augmentation

To achieve effective character detection, the CNN classifier model should be trained with extensive data, so the initial set of 3,503 character images is augmented by extracting new images within a 25×94 pixel window, positioned within 2 pixels of the ground truth (GT) on the x-axis, yielding 17,439 new character images. Additionally, for classification, non-character images are extracted by defining areas at least 10 pixels away from any GT on the x-axis with an IoU ≤ 3, using a 25×94 pixel window and a stride of 3 pixels, resulting in 20,044 non-character images. The training data for the bounding box regressor model comes from detection results, consisting of 25×94 images. To increase the dataset, image brightness is varied for augmentation, resulting in 17,486 images.

## 3. Model Training

In the training of the CNN classifier, binary cross entropy was used as the loss function, and Adam was the optimizer with a learning rate of 0.001. The training was conducted for 20 epochs with a batch size of 32. After 20 epochs of training, the CNN classifier achieved an accuracy of 0.9989 and a loss of 0.0044. For the CNN regressor training, the MAE (Mean Absolute Error) loss function was used. Adam was also used as the optimizer with a learning rate of 0.001. The training was conducted over 100 epochs with a batch size of 64. The CNN regressor training results showed an accuracy of 0.9992 and a loss of 0.1878.

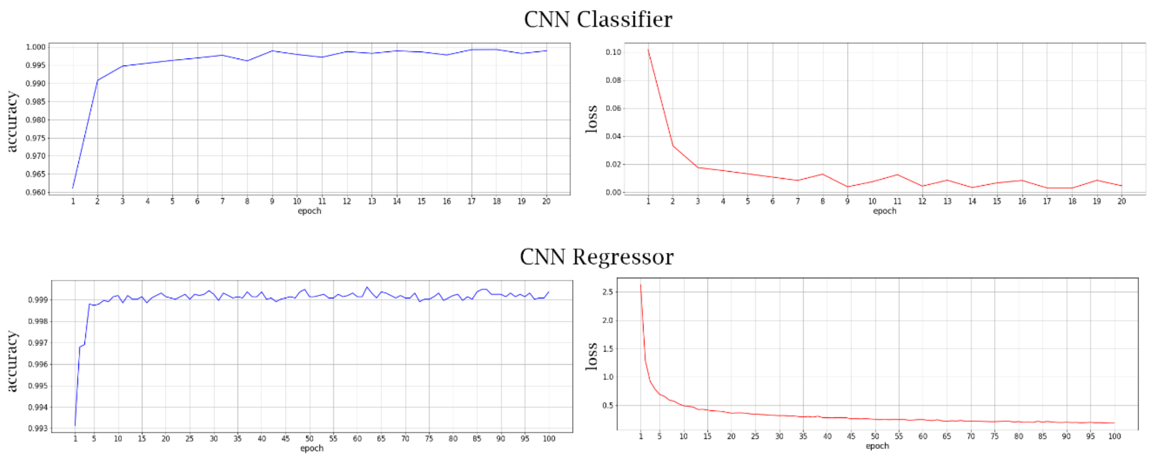

**Fig. 6.** Training results

## 4. Result and Discussion

### 4.1 Testing Results on Detector

The character detector model was tested on test data of 280 Indonesian vehicle plates with a total of 2026 characters. In this test, the results of character detection without involving a bounding box regressor are presented. Of the 2026 characters, the character detector model produced 2030 detections. The obtained average IoU was 0.4811. The average computation of character segmentation on a plate image was 0.0658 seconds. A sample of character segmentation results using this character detector model is visualized in Fig.7, the red box is GT and the green box is the detection result.

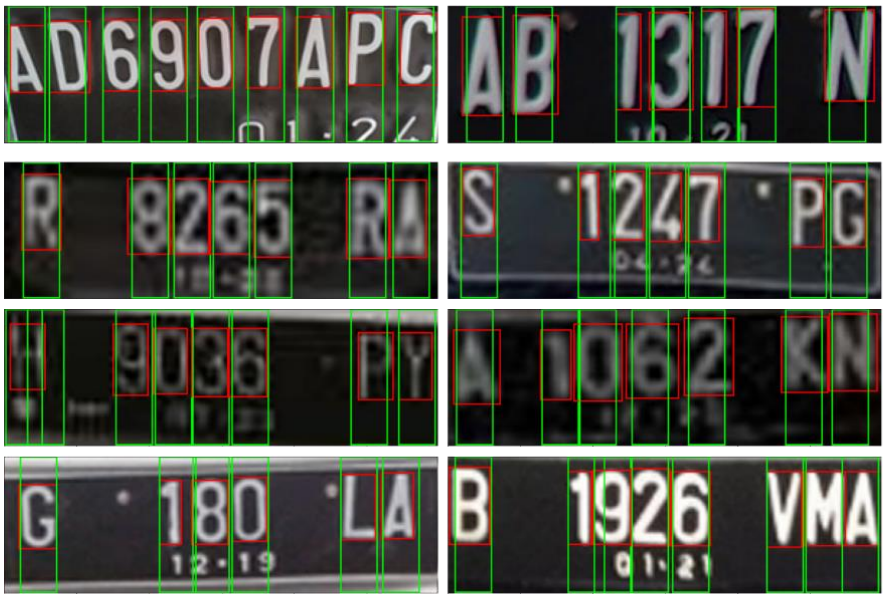

**Fig. 7.** Sample of character detector result

### 4.2 Testing Results with Bounding Box Regressor

After testing the character detector model separately, combining it with a bounding box regressor was carried out and tested. The total bounding box regressed is 2030, which is the bounding box from the character detector test. The results of both tests produced an average IoU of 0.8980. The computation to perform character segmentation for a plate image is 0.0827 seconds on average.

ISSN 3025-3780 (online)
ISSN 3025-4590 (printed)

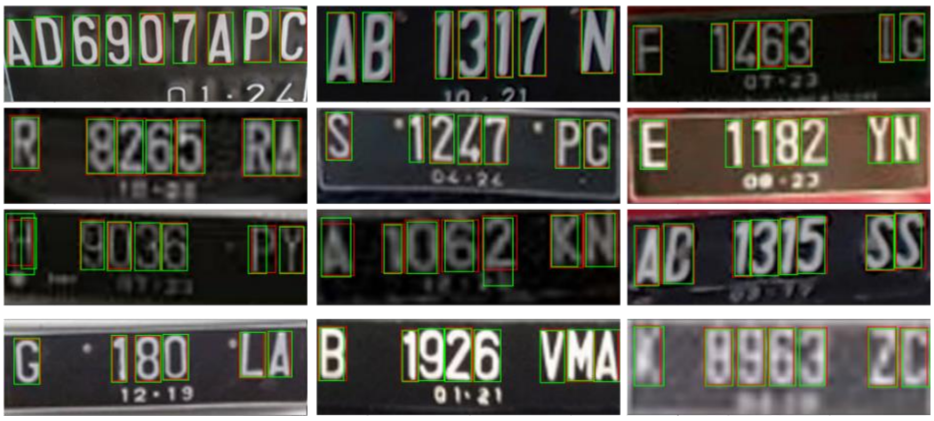

**Fig. 8.** Sample of bounding box regressor result

## 4.3 Performance Comparison

The first comparison is a comparison of the average IoU results produced by the two testing stages that have been carried out along with average computation for a plate image. This comparison is shown in Table 4. In this table, CD is the character detector and BBR is the bounding box regressor. An increase in the average IoU of 0.4169 occurred when BBR was implemented in the system. This can be proven visually by looking at the IoU distribution from the test in Fig. 9. As seen in the figure, the IoU distribution on CD-BBR is centered on higher values.

**Table 4.** IoU and Computation Comparison

| Metrics | Model | |
|---|---|---|
| | CD | CD-BBR |
| Avg. IoU | 0.4811 | 0.8980 |
| Avg. Computation/img | 0.0658 s | 0.0827 s |

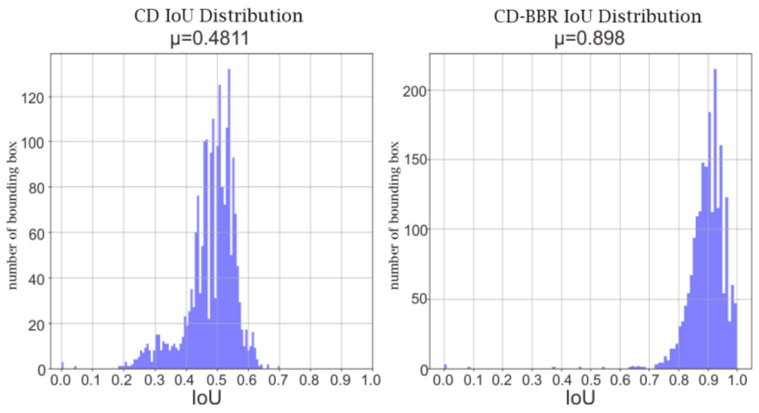

**Fig. 9.** IoU distribution

TP (True Positive), FP (False Positive), and FN (False Negative) in the character segmentation results are determined by the specified IoU threshold. We use 0.8 as the IoU threshold to calculate the precision and recall of the model. Not only our BBR backbone, we also present a test on VGG16 as a bounding box regressor, the comparison of the two is presented in Table 5.

**Table 5.** Comparison of Performance on BBR backbone (IoU threshold of 0.8)

| Model | TP | FP | FN | Avg. IoU | *Precision* | *Recall* | Avg. s/img |
|---|---|---|---|---|---|---|---|
| CD-VGG16[10] | 1856 | 174 | 170 | 0.8818 | 0.9142 | 0.9160 | 0.1154 s |
| CD-BBR | 1943 | 87 | 83 | 0.8980 | 0.9571 | 0.9590 | 0.0827 s |

## 5. Conclusion

From 280 license plate images containing 2026 characters, CD produced character detections with an average IoU of 0.4811. Implementing the BBR increased the average IoU to 0.8980, improving the IoU of CD alone. By comparing bounding box backbone performance, we can see that CD-BBR outperformed CD-VGG16. This is motivated by a statement in [11] that better character segmentation tends to improve recognition. This study's CNN regressor model successfully predicted coordinate shifts, making bounding boxes more adaptive and increasing overlap with characters, thereby reducing the inclusion of non-character areas when used in the recognition phase.

## References

[1] Badan Pusat Statistik, “Jumlah Kecelakaan, Korban Mati, Luka Berat, Luka Ringan, dan Kerugian Materi, 2022.” Accessed: May 15, 2024. [Online]. Available: https://www.bps.go.id/id/statistics-table/2/NTEzIzI=/jumlah-kecelakaan--korban-mati--luka-berat--luka-ringan--dan-kerugian-materi.html

[2] M. M. Rezapour Mashhadi, P. Saha, and K. Ksaibati, “Impact of traffic Enforcement on Traffic Safety,” International Journal of Police Science & Management, vol. 19, no. 4, pp. 238–246, 2017, doi: 10.1177/1461355717730836.

[3] Hendry and R. C. Chen, “A New Method for License Plate Character Detection and Recognition,” in ACM International Conference Proceeding Series 2018, pp. 204–208, doi: 10.1145/3301551.3301592.

[4] C. H. Lin and Y. Li, “A License Plate Recognition System for Severe Tilt Angles Using Mask R-CNN,” *in International Conference on Advanced Mechatronic Systems, ICAMechS*, vol. 2019-Augus, pp. 229–234, doi: 10.1109/ICAMechS.2019.8861691.

[5] A. Abdussalam, S. Sun, M. Fu, H. Sun, and I. Khan, “License Plate Segmentation Method Using Deep Learning Techniques Amr,” Springer Singapore, 2018. [Online]. Available: https://doi.org/10.1007/978-981-13-1733-0_8

[6] I. W. Notonogoro, Jondri, and A. Arifianto, “Indonesian License Plate Recognition Using Convolutional Neural Network,” in 2018 6th International Conference on Information and Communication Technology, ICoICT, doi: 10.1109/ICoICT.2018.8528761.

[7] R. Laroca *et al.*, “A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector,” Proceedings of the International Joint Conference on Neural Networks, vol. 2018-July, 2018, doi: 10.48550/arXiv.1802.09567.

[8] S. Abdullah, M. Mahedi Hasan, and S. Muhammad Saiful Islam, “YOLO-Based Three-Stage Network for Bangla License Plate Recognition in Dhaka Metropolitan City,” in 2018 International Conference on Bangla Speech and Language Processing, ICBSLP 2018, pp. 1–6, doi: 10.1109/ICBSLP.2018.8554668.

[9] A. T. Musaddid, A. Bejo, and R. Hidayat, “Improvement of Character Segmentation for Indonesian License Plate Recognition Algorithm using CNN,” in 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) 2020, pp. 279–283, doi: 10.1109/ISRITI48646.2019.9034614.

[10] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” in 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, pp. 1–14, doi: 10.48550/arXiv.1409.1556.

[11] G. R. Gonçalves, S. P. G. da Silva, D. Menotti, and W. R. Schwartz, “Benchmark for License Plate Character Segmentation,” J Electron Imaging, vol. 25, no. 5, p. 053034, Oct. 2016, doi: 10.48550/arXiv.1607.02937.