Journal of Applied Intelligent System (e-ISSN : 2502-9401 | p-ISSN : 2503-0493)
Vol. 9 No. 1 2024, pp. 40 – 50

Optimization of Yolov5 Hyperparameter Using Adam Optimizer in Vehicle Object Detection

Bambang Irawan*[1] , **Pulung Nurtantio Andono***[2]
Universitas Dian Nuswantoro, Semarang, Indonesia
E-mail : p31202102416@mhs.dinus.ac.id[1], pulung@dsn.dinus.ac.id*[2]*
**Corresponding author*

Ruri Suko Basuki[3]
Universitas Dian Nuswantoro, Semarang, Indonesia
E-mail : ruri.basuki@dsn.dinus.ac.id[3]

Abstract - The use of computer vision can be applied in various aspects of daily life, thereby reducing dependence on human labor. One implementation is in the industrial sector, such as in the motor vehicle production process, to sort or classify parts or goods. The computer vision process involves many stages, such as image capture, image processing, image analysis, image recognition, and decision making. In the automotive industry, computer vision has been used in autonomous or driverless electric vehicles, as well as in creating intelligent transportation systems. To detect objects in real-time, one option that can be used is to use the YOLO algorithm which can detect objects in one stage with prediction of bounding boxes and class probabilities simultaneously. However, although YOLO has good performance, its architecture has several shortcomings, such as complexity and complicated hyperparameter configuration. To address this, the Adam optimization algorithm is introduced, which combines the momentum algorithm and RMSprop to adaptively adjust the learning rate and provide faster convergence in model training. This is evidenced by the increase in mAP values in Yolov5. Testing in this study produced the highest mAP value of 67.0%. Meanwhile, for testing Yolov5 without Adam's optimization, the highest mAP value was 56.4%. These results prove that the Yolov5 method with Adam optimization is better than the Yolov5 method without optimization.

Keywords - yolov5, Adam optimization, vehicles, image

1. INTRODUCTION

The development of Computer Vision allows an object or entity to be seen using a computer. The utilization of computer vision can be implemented in various everyday scenarios, enabling us to reduce dependence on human labor. Some applications in the implementation of computer vision, for instance in the industrial field, involve processes in motor vehicle production such as cars and motorcycles, for sorting or classifying specific parts of goods.

In the automotive industry, computer vision has begun to be implemented in electric autonomous vehicles or driverless cars. Intelligent transportation systems can be formed through the utilization of computer vision technology, where real-time information obtained from imaging solutions enhances pedestrian and passenger safety, reduces traffic congestion, and mitigates street crimes. One option to address these solutions is by conducting research on object detection capable of real-time image or video detection. One of the architectures commonly used to tackle this issue with high accuracy is YOLO. YOLO is a convolutional network that can detect objects in a single stage by predicting multiple bounding boxes and class probabilities simultaneously.

The YOLO architecture, despite its capability to detect objects in images and videos effectively, does have several weaknesses due to its large and complex convolutional network. One of the shortcomings of this architecture is the multitude of hyperparameters that need to be configured to improve the model's performance.

To address these issues, several algorithms have been introduced to ease the hyperparameter tuning and facilitate model parameter updates during training. The Adam Optimizer algorithm was introduced in this research due to its popularity among researchers. This algorithm conceptually combines the momentum algorithm and RMSProp. Adam maintains adaptive estimates of the first moment (mean) and the second moment (uncentered) of the parameter gradients. It dynamically adjusts the learning rate based on these estimated moments, resulting in faster convergence.[6]

The contributions of this research can be summarized as follows :

- Aimed to detect vehicle objects based on their types or classes using the YOLOv5 architecture.
- Utilized the Adam optimizer algorithm to optimize hyperparameters within YOLOv5, aiming to enhance mean average precision, accuracy, and precision compared to YOLOv5 architecture without optimization.

2. RESEARCH METHOD

This research focuses on analyzing and identifying various types of vehicles based on their classes. It employs the YOLOv5 architecture for multi-object detection of vehicles, aiming to improve accuracy and precision in recognizing these types of vehicles. The optimization of YOLOv5 hyperparameters using the Adam optimization algorithm is carried out to achieve this goal. The initial method involves using a dataset as input data and annotating the images in the dataset.

The annotation process is carried out on all images in the YOLOv5 format. Annotation involves creating labels by providing bounding boxes along with the class name for each object in every image. Following this, the annotated images in the dataset are divided into three parts: training, validation, and testing.

The vehicle detection process combines the YOLOv5 architecture with the Adam Optimization algorithm. YOLOv5 is utilized for detecting vehicle objects, while the Adam algorithm optimizes the hyperparameters of the YOLOv5 architecture. This optimization aims to enhance the speed and accuracy of the detection process performed by YOLOv5. The research workflow is explained in Figure 1, illustrating the steps involved in the process.

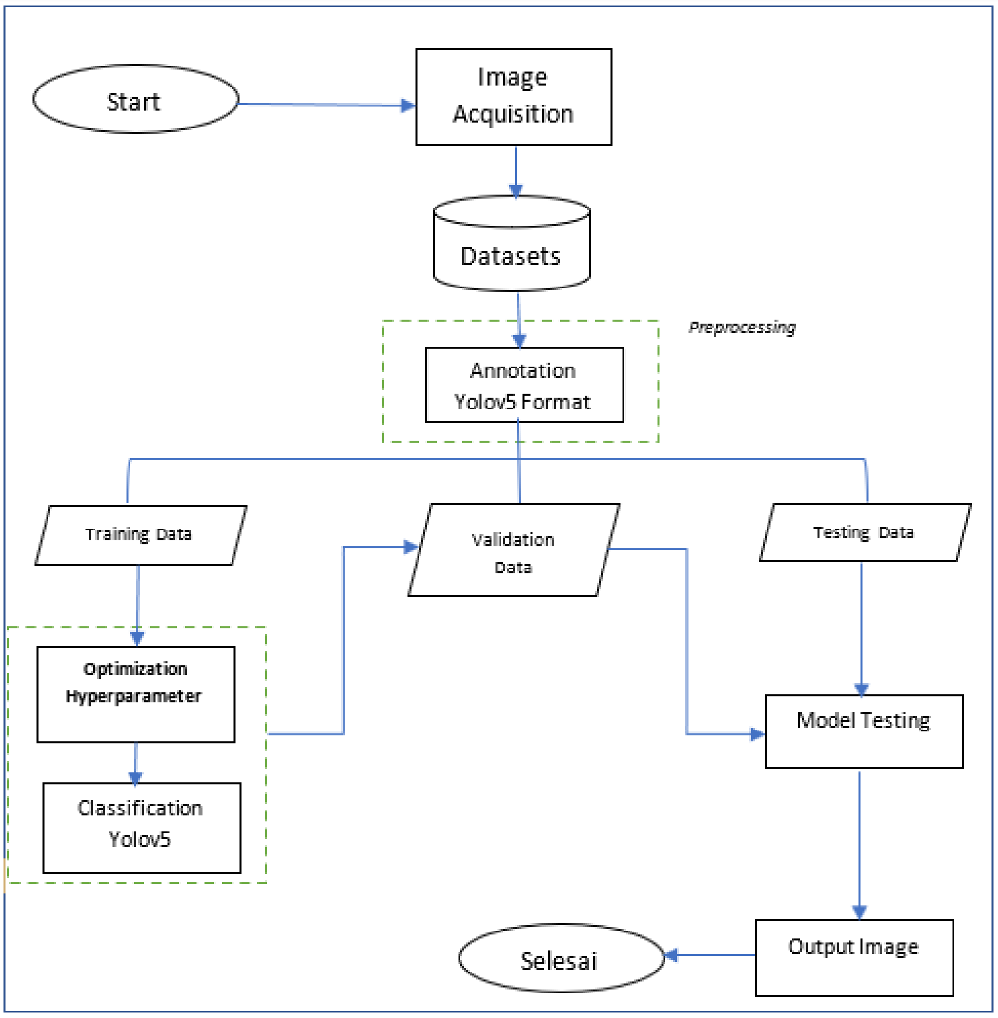

Figure 1. Proposed Method

2.1. *Datasets*

The data utilized in this research is sourced from supervised learning, meaning it contains labeled information. The dataset is obtained from public data, specifically from OpenImages, consisting of 627 images across five object categories: Ambulance, Bus, Car, Motorcycle, and Truck. The dataset is randomly split into 70% for training data, 20% for validation data, and 10% for testing data.

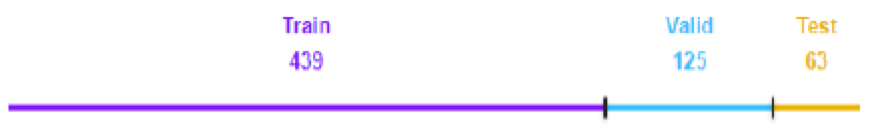

Figure 2. Split Datasets

The dataset splitting is conducted to prevent overfitting and to evaluate the model. Overfitting occurs when a model performs well during training but poorly on unseen data.

2.2. Annotation

After obtaining the dataset, the next step is annotation. The purpose of annotating images before processing is to create labels by providing bounding boxes along with the class name for each object in every image. Subsequently, the annotated images in the dataset are divided into three parts: training, validation, and testing.

2.3. Yolov5

Yolov5 algorithm model training was carried out using the CSPDarknet architecture and using the Pytorch framework. Object detection in yolov5 is carried out with 4 blocks in one stage. This stage starts from Input, Backbone, Neck, Head. The Block Input process will resize the image according to the resolution of the input layer, then in the Block Backbone process feature extraction is carried out by dividing the input feature map into two parts, the next process is Block Neck, in this block the aim is to add a layer between the backbone and the Head, with With this block, object detection can be done in various sizes. Block Head Yolo Layer will determine the bounding box and classify what is in the bounding box.[6]

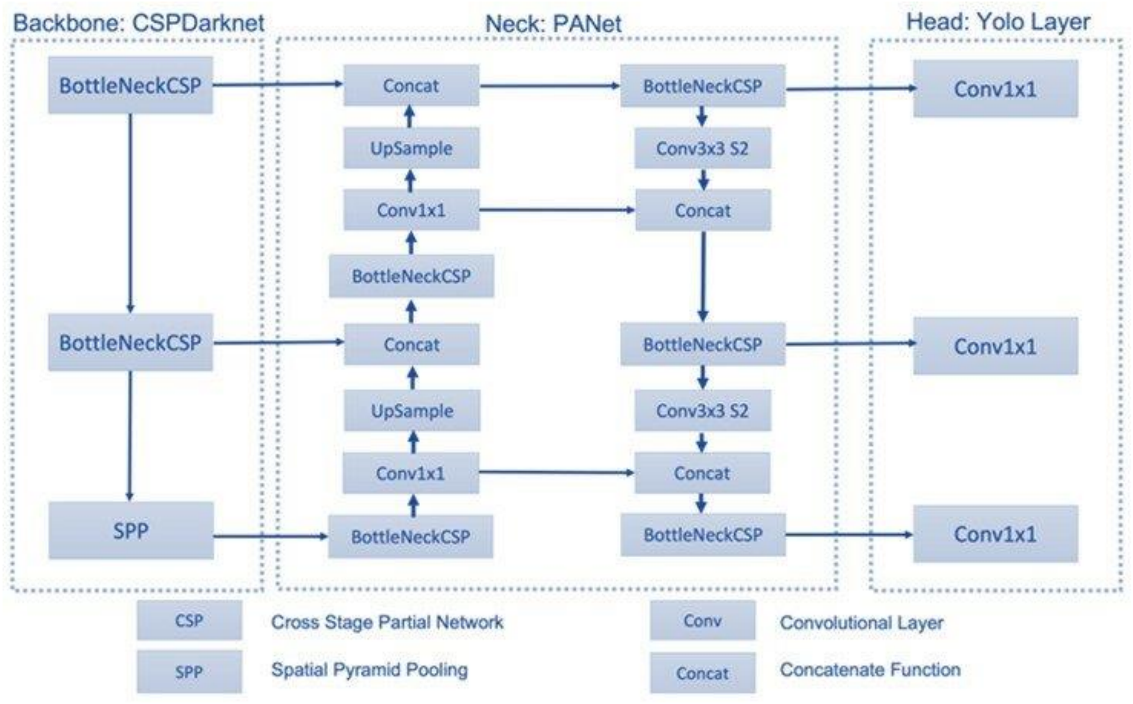

Figure 3. Yolov5 Architecture

2.4. Adam Optimizer

Adaptive Moment Estimation or commonly called Adam is the most popular gradient descent optimization algorithm. This method calculates an adaptive learning rate for each parameter and stores historical averages such as momentum, and average gradient descent similar to RMS-Prop and Adadelta. Adam combines both optimization algorithms[13]. The functioning of Adam optimization can be described as follows :

a. Gradien Computation

At each training iteration, the gradient of the loss function with respect to the model parameters is computed using backpropagation. This gradient provides guidance to minimize the loss.

b. The calculation of the first and second moments

Adam utilizes the first moment (mean) and the second moment (uncentered variance) of gradients to calculate parameter updates. The first moment is computed as an exponentially weighted average of gradients, while the second moment is computed as an exponentially weighted average of squared gradients.

c. Update Parameters

After calculating the first and second moments, the model parameters are updated by considering the adjusted moments along with the learning rate. At this stage, Adam introduces the concept of bias-correction to address the initial bias in moment calculations at the beginning of training.

d. Learning Rate Settings

Adaptive learning rate adjustment adapts the learning rate for each parameter based on the first and second moments. This aids in controlling the size of the steps taken in each iteration.

2.5. Hyperparameter Tuning Optimization

The YOLOv5 algorithm is a neural network with hyperparameters like learning rate, batch size, decay, momentum, and more. To obtain optimal values, these hyperparameters need tuning during the training process. However, manually configuring these hyperparameters can be cumbersome and highly inefficient. Hence, it's crucial to combine them with an optimizer algorithm capable of searching for the most optimal parameter values.

2.6. Testing Method

This research uses Confusion Matrix calculations to measure the performance of the classification model. The confusion matrix can be applied to binary classification or multi-class classification. Confusion Matrix has parameters True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN).

Table 1. Confusion Matrix

		PREDICTION RESULTS	PREDICTION RESULTS
		0	1
REALITY	0	TN	FP
	1	FN	TP

a. Accuracy

The accuracy of a model in categorizing objects reflects its performance. It measures the proportion of correct predictions (whether positive or negative) in relation to the entire dataset. The formula to calculate accuracy is as stated below :[8]

$$Accuracy = \frac{TP + TN}{All\ Data} \quad (1)$$

b. Precision

Precision is used to measure how precise the object detection results are. The formula to measure the precision value is as follows :[15]

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

3. RESULTS AND DISCUSSION

In object detection, YOLO employs a single convolutional network to predict multiple bounding boxes and class probabilities within those boxes simultaneously. The YOLO system divides the input image into an S x S grid, where each grid cell predicts bounding boxes and confidence scores for those boxes. To calculate the value of those bounding boxes, it's obtained using the following equation :[15]

$$\begin{aligned} Ymin &= Xmin * image\_height \\ Xmin &= Ymin * image\_width \\ Ymax &= Xmax * image\_height \\ Xmax &= Ymax * image\_width \end{aligned} \quad (3)$$

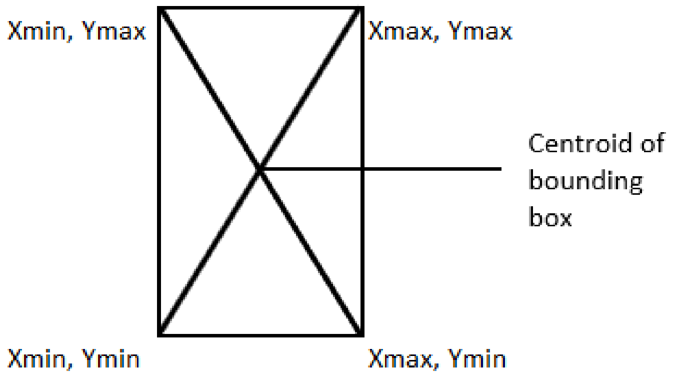

Figure 4. *Bounding Box Format*

In each bounding box, there are 5 predictions: x, y, w, h, and confidence. The coordinates (x, y) represent the center point of the box relative to the grid cell boundaries. Meanwhile, (w, h) or width and height represent the center point of the box relative to the entire image. The confidence value signifies the Intersection over Union (IoU) between the predicted box and the ground truth box. Each grid cell estimates class probabilities, where the probability of a single class is detected in a grid cell regardless of the number of bounding boxes, and the probability depends on the grid cell containing the object.

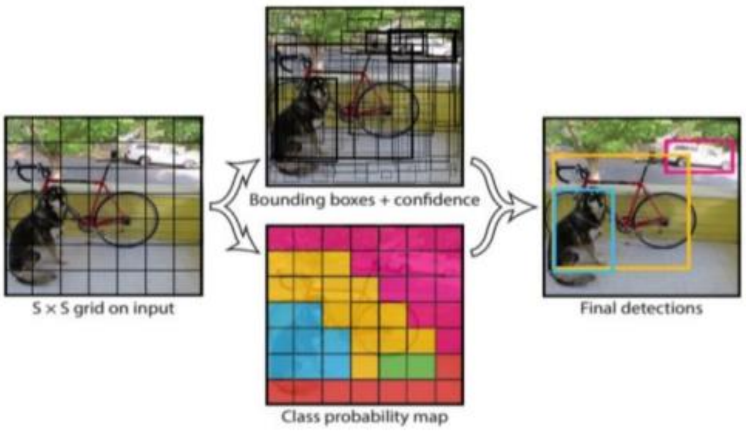

Figure 5. The Yolo Detection Model

The labeling is divided into 5 classes: Ambulance, Bus, Car, Motorcycle, Truck, starting class IDs from 0 for Ambulance, 1 for Bus, 2 for Car, 3 for Motorcycle, and 4 for Truck. After the labeling process, which involves adding bounding boxes to each class of the image, the labeled images will be automatically saved in the same folder.

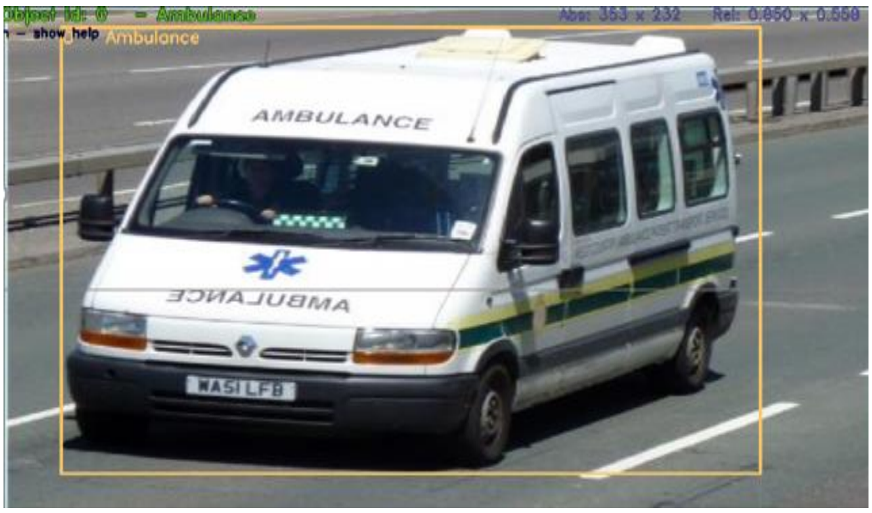

Figure 6. Image annotation based on class

Annotation of figure 6 can be described as follows :

0	: object class (object Ambulance)
0.458984	: midpoint x-coordinate (x)
0.481250	: midpoint y-coordinate (y)
0.783594	: width of the box (width)
0.876389	: height of the box (height)

The values of x, y, width, and height are relative to the image's width and height. The original image values can be obtained by multiplying them with the width or height of the image, as illustrated in table 2.

Table 2. Annotation Calculation

Annotation Format	Annotation Values	Multiplier	Original Value
x	0.458984	353	162
y	0.481250	232	112
width	0.783594	353	276
height	0.876389	232	203

3.1 *Batch Value Determination*

The batch value refers to the number of data samples processed in a single training iteration. This value significantly influences the system's outcomes. In this research, batch values of 16, 32, and 64 were used with 100 epochs on a dataset containing 617 images. This means that, for instance, the first batch trains images 1 to 16, then continues with images 17 to 32, and so forth. The results of the batch value testing conducted are presented in Table 3.

Table 3. Batch Value Testing without Optimizer

No	Batch value	epoch	Model	mAP
1	16	100	Yolov5s.pt	54%
2	32	100	Yolov5s.pt	56.4%
3	64	100	Yolov5s.pt	51.6%

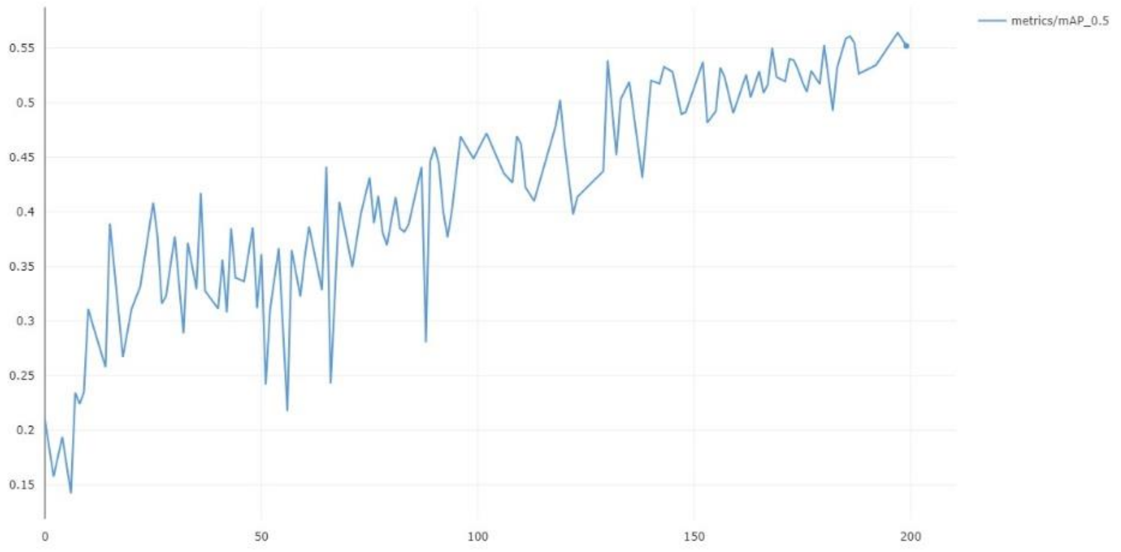

Figure 7. Experimental Batch 32 Values

3.2 *Selection of Adam Hyperparameters*

The array of hyperparameters influencing training results can be classified into two sets: those pertaining to network architecture and those connected with the training process.

Parameters tied to training involve epochs, learning rates, batch sizes, and gradient optimization methods.

This research, testing with an optimizer was conducted by comparing the outcomes between training data using the Adam optimizer and training data without using an optimizer. The testing results of the optimizer with the batch value that produced the highest mAP, namely batch 32, are presented in Table 4.

Table 4. Testing with an Optimizer

No	Batchsize	lr	decay	momentum	epoch	Precision	mAP
1	32	0,01	0,0005	0,937	100	68.6%	63.4%
2	32	0,01	0,0005	0,937	200	64.5%	62.5%
3	32	0,01	0,0005	0,937	300	61.9%	64.9%
4	32	0,001	0,0001	0,99	100	65.5%	59.2%
5	32	0,001	0,0001	0,99	200	63.3%	56.1%
6	32	0,001	0,0001	0,99	300	64.7%	57.3%
7	32	0,0001	0	0,937	100	65.2%	59.2%
8	32	0,0001	0	0,937	200	70.1%	65.9%
9	32	0,0001	0	0,937	300	70.7%	67.0%

In Table 4, the testing was performed using a batch value of 32 in combination with the predetermined Adam hyperparameter settings. The selection of batch value 32 was based on earlier testing without an optimizer, where the highest mAP was achieved using a batch value of 32. Through the conducted research on multi-object vehicle detection, combining the YOLOv5 architecture with the Adam optimizer for hyperparameter optimization successfully recognizes and detects vehicles based on their classes optimally. Starting with image annotation to create per-image labels, followed by the classification process to detect objects per class using the YOLOv5 architecture with optimized hyperparameters employing the Adam optimizer. This testing resulted in the highest precision value of 70.7% and the highest mAP value obtained was 67.0%.

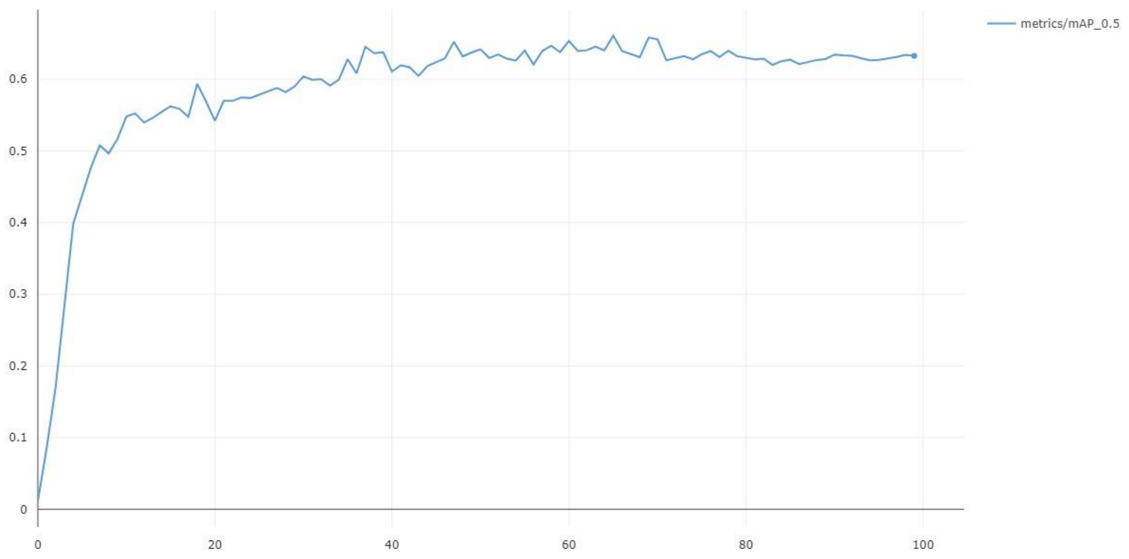

Figure 8. Highest mAP Value

4. CONCLUSION

The batch configuration, or the number of data samples processed in each training iteration, impacts both the speed and stability of data outcomes. In this study, utilizing a batch value of 32 in the YOLO simple architecture (without optimization) resulted in an mAP of 56.4%, performing better compared to batch values of 16 and 64, which achieved mAPs of 54% and 51.6%, respectively. On the other hand, multi-object vehicle detection testing with the YOLOv5 architecture optimized using the Adam Optimizer achieved the highest mAP of 67%. There's still room for improvement in this research. In future studies, extending the training epochs might yield even better mAP values, as this research only conducted training for 300 epochs.

REFERENCES

[1] X. Jiang, K. Sun, L. Ma, Z. Qu, and C. Ren, "Vehicle Logo Detection Method Based on Improved YOLOv4," *Electron.*, vol. 11, no. 20, pp. 1–19, 2022, doi: 10.3390/electronics11203400.

[2] G. Guo and Z. Zhang, "Road damage detection algorithm for improved YOLOv5," *Sci. Rep.*, vol. 12, no. 1, pp. 1–12, 2022, doi: 10.1038/s41598-022-19674-8.

[3] I. A. Elaalami, S. O. Olatunji, and R. M. Zagrouba, "AT-BOD: An Adversarial Attack on Fool DNN-Based Blackbox Object Detection Models," *Appl. Sci.*, vol. 12, no. 4, 2022, doi: 10.3390/app12042003.

[4] S. Sutriawan, A. Z. Fanani, F. Alzami, and R. S. Basuki, "Deep Learning Jaringan Saraf Tiruan Untuk Pemecahan Masalah Deteksi Penyakit Daun Apel," *J. Teknol. Inf. dan Komun.*, vol. 11, no. 1, p. 35, 2023, doi: 10.30646/tikomsin.v11i1.729.

[5] Z. Ren, H. Zhang, and Z. Li, "Improved YOLOv5 Network for Real-Time Object Detection in Vehicle-Mounted Camera Capture Scenarios," *Sensors*, vol. 23, no. 10, 2023, doi: 10.3390/s23104589.

[6] A. Prof, A. Abdulazeez, M. Qazzaz, and E. Faculty, "Car Detection and Features Identification Based on," vol. 7, no. 2, pp. 4049–4056, 2022.

[7] N. Rochmawati, H. B. Hidayati, Y. Yamasari, H. P. A. Tjahyaningtijas, W. Yustanti, and A. Prihanto, "Analisa Learning Rate dan Batch Size pada Klasifikasi Covid Menggunakan Deep Learning dengan Optimizer Adam," *J. Inf. Eng. Educ. Technol.*, vol. 5, no. 2, pp. 44–48, 2021, doi: 10.26740/jieet.v5n2.p44-48.

[8] D. Iskandar Mulyana and M. A. Rofik, "Implementasi Deteksi Real Time Klasifikasi Jenis Kendaraan Di Indonesia Menggunakan Metode YOLOV5," *J. Pendidik. Tambusai*, vol. 6, no. 3, pp. 13971–13982, 2022, doi: 10.31004/jptam.v6i3.4825.

[9] A. A. Hidayah, I. Firdauzi, and J. Prayogi, "Pelatihan Publish or Perish , Vosviewer , Dan Mendeley Pada Mahasiswa Mbkm Riset," vol. 2, no. 1, pp. 8–12, 2023.

[10] S. Mehta, C. Paunwala, and B. Vaidya, "CNN based traffic sign classification using adam optimizer," *2019 Int. Conf. Intell. Comput. Control Syst. ICCS 2019*, no. Iciccs, pp. 1293–1298, 2019, doi: 10.1109/ICCS45141.2019.9065537.

[11] J. Ker, L. Wang, J. Rao, and T. Lim, "Deep Learning Applications in Medical Image Analysis," *IEEE Access*, vol. 6, pp. 9375–9379, 2017, doi: 10.1109/ACCESS.2017.2788044.

[12] C. Zhang, H. Ding, Q. Shi, and Y. Wang, "Grape Cluster Real-Time Detection in Complex

Natural Scenes Based on YOLOv5s Deep Learning Network," *Agric.*, vol. 12, no. 8, 2022, doi: 10.3390/agriculture12081242.

[13] B. Cortiñas-Lorenzo and F. Pérez-González, "Adam and the ants: On the influence of the optimization algorithm on the detectability of DNN watermarks," *Entropy*, vol. 22, no. 12, pp. 1–39, 2020, doi: 10.3390/e22121379.

[14] K. Khairunnas, E. M. Yuniarno, and A. Zaini, "Pembuatan Modul Deteksi Objek Manusia Menggunakan Metode YOLO untuk Mobile Robot," *J. Tek. ITS*, vol. 10, no. 1, 2021, doi: 10.12962/j23373539.v10i1.61622.

[15] Priyanto Hidayatullah,"Buku Sakti Deep Learning" – Stunning Vision AI Academy, Bandung 2021

[16] Abdul Muiz Khalimi, 2018. Cara Menghitung Confusion Matrix 4 Kelas.[Online]. Available: https://www.pengalaman-edukasi.com/2020/01/confusion-matrix-multi-class-menghitung.html

[17] S. Y. Sen and N. Ozkurt, "Convolutional Neural Network Hyperparameter Tuning with Adam Optimizer for ECG Classification," *Proc. - 2020 Innov. Intell. Syst. Appl. Conf. ASYU 2020*, no. 978, 2020, doi: 10.1109/ASYU50717.2020.9259896.