A Study on the Implementation of YOLOv4 Algorithm with Hyperparameter Tuning for Car Detection in Unmanned Aerial Vehicle Images

2023 11th International Conference on Information and Communication Technology (ICoICT) | 979-8-3503-2198-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICoICT58202.2023.10262651

1st Muhammad Alfian Ramadhani
Informatics
University of Muhammadiyah Malang
Malang, Indonesia
fianrmdni@gmail.com

2nd Yufis Azhar
Informatics
University of Muhammadiyah Malang
Malang, Indonesia
yufis@umm.ac.id

3rd Galih Wasis Wicaksono
Informatics
University of Muhammadiyah Malang
Malang, Indonesia
galih.w.w@umm.ac.id

***Abstract*—Unmanned Aerial Vehicles (UAVs) for surveillance and monitoring have become more prevalent due to their versatility and mobility. These vehicles capture high-resolution images that provide a broad field of view in real-time. Today, enhancing object detection accuracy on images captured by unmanned aerial vehicles (UAVs) has become a significant challenge. Through extensive research, it has been established that the correct setup of hyperparameters is imperative to achieving the highest accuracy in machine learning. Our study introduces a technique that utilizes hyperparameter tuning to implement the YOLOv4 algorithm, enabling the detection of cars in unmanned aerial vehicle images. In general, all scenarios of this study have different accuracy results, which have implications for their detectability. Thus, scenario 3 of YOLOv4 hyperparameter tuning is the best model accuracy. Our approach utilizes the PSU Aerial Car Images Dataset from previous studies. During this research, accuracy values were obtained through testing at the model validation stage rather than at the testing stage. In this study, we achieved a validation performance of the detection model by using a validation dataset proportion of 20%. Based on our research, it has been revealed that the YOLOv4 algorithm is a highly efficient car detection system when it comes to unmanned aerial vehicle images. Through rigorous testing of multiple hyperparameter tuning scenarios, we achieved an exceptional accuracy of 99.02% in the optimal model scenario, which utilized YOLOv4. Similarly, in replicating a research paper's hyperparameter tuning methods on YOLOv3, the highest accuracy of 98.40% was attained in scenario 2.**

Keywords— Hyperparameter, Tuning, YOLOv4, Drone, UAV

I. INTRODUCTION

Due to their low cost and easy operation, drones (Unmanned Aerial Vehicles (UAVs)) have been used in many military and civilian areas to meet explosive consumption growth [1]. UAVs are increasingly used in surveillance and monitoring because of their flexibility and mobility. The UAV also produces high-resolution images for a wide field of view in real-time. The low-quality object detection algorithms with traditional machine-learning approaches could have improved the UAVs. However, since the advent of deep learning algorithms and especially convolutional neural networks, object recognition and detection have shown a marked increase in accuracy [2]. One of the problems handled by computer vision, part of deep learning, is the multi-object tracking of frame sequences [3]. This is an early indication of the massive use of UAVs for data acquisition and analysis in many engineering fields [2]. Detecting car objects accurately using aerial images and calculating them in real-time for traffic monitoring is a major challenge. It is important for traffic control and supervision to track the path and direction of objects. The system can help reduce traffic congestion and direct vehicles to less crowded areas [3].

Recently, several object detection problems have been better addressed using Convolutional Neural Networks (CNNs). Computer vision research for intelligent transportation systems also follows the trend, and many such works are finding the practicality in their use [4]. The deep learning algorithms that work on this method are RCNN, Fast RCNN, and Faster RCNN. They are very accurate but so slow that they cannot be used in real-time applications. In a single-stage approach, object detection is taken as a regression problem, and within its scope, object classification and bounding box coordinates are given. One deep learning algorithm that works using a single-stage approach is You Only Look Once (YOLO) [5]. Several studies have been conducted regarding YOLO. First, research was conducted by Liao et al. [6] entitled Real-Time UAV Trash Monitoring System. The main objective of his study is to build UAV and IoT architectures using the YOLOv4-Tiny-3l model, which is deployed to embedded systems so that UAVs have the mobility to obtain beach images and map garbage information in real-time for further analysis [6]. The advantages of this research are hyperparameter adjustment and model evaluation to get the best model and the lowest generalization error. However, it has deficiencies related to system accuracy that need to be improved. Then, another study conducted by Liu et al. [7] proposed using the YOLO-Tomato detector for tomato detection based on the YOLOv3 model. The strengths of this study are the high accuracy using YOLOv3 and the ability to detect tomatoes of various sizes, positions, and shapes. However, it has the disadvantage of not having a system performance comparison with other tomato detection algorithms.

Furthermore, research by Setyaningsih et al. [8]. Significantly, YOLOv4 outperforms the Mask R-CNN with an accuracy rate of up to 98.6%. The advantage of this research is the use of two algorithms, namely Mask R-CNN and YOLOv4 so that comparisons can be made, where the test results show that both algorithms provide fairly good accuracy. The drawback is the limited dataset. Subsequent research, which became the main basis for this research, was conducted by Bilel et al. [2]. This study considers the use of Faster R-CNN and YOLOv3, CNN-based algorithms for car detection in UAV imagery, and compares the two. YOLOv3 results outperform Faster R-CNN in sensitivity which means that YOLOv3 is better able to extract all the cars in the image with 99.07% accuracy, while Faster R-CNN is only 79.40%. Based on previous research, hyperparameter tuning is an essential technique for achieving high accuracy in machine

learning. Still, it requires a large allocation of processing resources [9].

By using the dataset contained in the study [2]. This research aims to compare the results of the validation performance of which detection model is better for car detection, with the YOLOv3 and YOLOv4 algorithms using hyperparameter tuning whether or not they receive data augmentation treatment. With the contribution of this study, a pattern of improvement was obtained for YOLOv4 using hyperparameter tuning and data augmentation, while for YOLOv3, the opposite result was obtained.

II. Literature Review

A. Object Detection

Object Detection is one of the fundamental topics in computer vision [10]. Object detection has the task of predicting the location of objects in the image through the bounding box and classifying the objects contained in each bounding box [11]. The main focus of research on most object detectors based on deep learning is to develop better neural network architectures and feature extraction and improve classification and localization accuracy [10].

B. YOLOv3

You Only Look Once (YOLO) is one of the most representative single-stage target detection algorithms. YOLO has been updated for several versions, bringing significant performance improvements. YOLOv2 removes the full connection layer in YOLO. Then anchor boxes are introduced to predict the bounding box. Furthermore, YOLOv3 replaced the YOLOv2 backbone with Darknet53 and adopted multiscale prediction, which significantly increased the accuracy and speed of detection [12].

C. YOLOv4

YOLOv4 is an advanced version of the YOLO series algorithm, which has been improved. YOLOv4 is based on YOLOv3 and has incorporated many excellent detection steps to improve accuracy. As of YOLOv4, the feature extraction enhancement was located in Darknet53, later renamed CSPDarknet53. The activation function changed from LeakyRelu to Mish. Then, the SPP (Spatial Pyramid Pooling) module was added to strengthen the feature extraction network [13]. YOLO includes a One-Stage detector, so object detection is performed directly on the input image using CSPDarknet53. Meanwhile, in the Two-Stage object detector, the approach is carried out in two stages, with the object proposition and classification stage and refinement of the bounding box, providing higher precision but requiring longer processing time [14]. Fig. 1 is related to object detector architecture based on its type: One-Stage Detector and Two-Stage Detector.

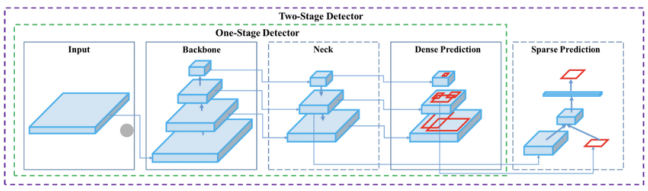

Fig. 1. Object Detection

YOLO architecture has 2 main parts, Backbone and Head. The first detection is carried out by entering an image or video in the input section; then, it will be forwarded to the Backbone section. The backbone is used to improve accuracy before detection at the Head [15].

D. Data Augmentation

Data augmentation is a technique that can reduce overfitting by increasing the dataset size with minimum effort[16]. Data augmentation generates more training samples by adjusting the angle of rotation, exposure, saturation, hue, and mosaic of the image dataset for training. Data is usually augmented by carrying out transformations on the data, or it can be interpreted as making a copy of the data source without changing the labels printed on each part of the data [16]. Data augmentation can improve the model's adaptability to images and improve the model's generalization ability.

E. Hyperparameter Tuning

Hyperparameter tuning is a configuration process in machine learning where a set of possible values is selected for each parameter, and the model is trained using every possible combination. Hyperparameter tuning has an important role in machine learning and deep learning algorithms because the resulting parameters significantly affect the performance of the CNN model [17]. The goal is to optimize which hyperparameter values have the best performance to use to produce the best final model.

F. Maintaining the Integrity of the Specifications

At this stage, the performance of the detection object model is evaluated. The YOLO model trained on the PSU Aerial Car Images dataset using hyperparameter tuning was used to compare the model's performance with previous research [2]. Evaluation is done by calculating four classification values, including False Positive (FP), True Positive (TP), False Negative (FN), and True Negative (TN) [18]. Then based on the four classification values above, we can calculate performance metrics using a classification report which contains recall, accuracy, precision, and f1-score to evaluate the performance of the proposed model. The following is the calculation formula.

$$Recall\ (Sensitivity) = \frac{TP}{TP+FN} \tag{1}$$

$$Accuracy = \frac{TP+TN}{P+N} \tag{2}$$

$$Precision = \frac{TP}{TP+FP} \tag{3}$$

$$F1\text{ - }Score = 2 * \frac{1}{\frac{1}{precission}+\frac{1}{recall}} \tag{4}$$

In addition to the four classification reports above, mAP (mean average precision) is usually used to evaluate model performance. Mean Average Precision (mAP) is a metric used to measure the accuracy of the object detection model for all classes in a particular database [19].

The confusion matrix is also used, which represents the comparison results of the classification, namely true positive (TP), false positive (FP), false negative (FN), and true negative (TN). The confusion matrix is used to measure the performance of an algorithm at the evaluation stage [20].

III. System Design

The flowchart in Fig. 2 sequentially shows the system begins by collecting datasets, pre-processing, splitting data, then augmenting data with a proportion of 80% for data train and 20% data validation, then forming a detection model is done with a pre-trained model from mscoco to YOLOv4 algorithm initial training. Then, during the detection model training stage, the YOLOv4 algorithm will use CSP-Darknet53 as the backbone variant for feature extraction [21]. Before being inserted into the feature extraction network, images will be converted to a size of 416x416 pixels according to the model configuration. The weighting results of the YOLOv4 algorithm will be stored in a different way and format, namely by storing the weighting results in every 100 iterations in the .weights format [8]. If the model is optimal, then the flow ends. However, if not, then the hyperparameter tuning process according to predetermined scenarios will continue to be carried out until the best model is found.

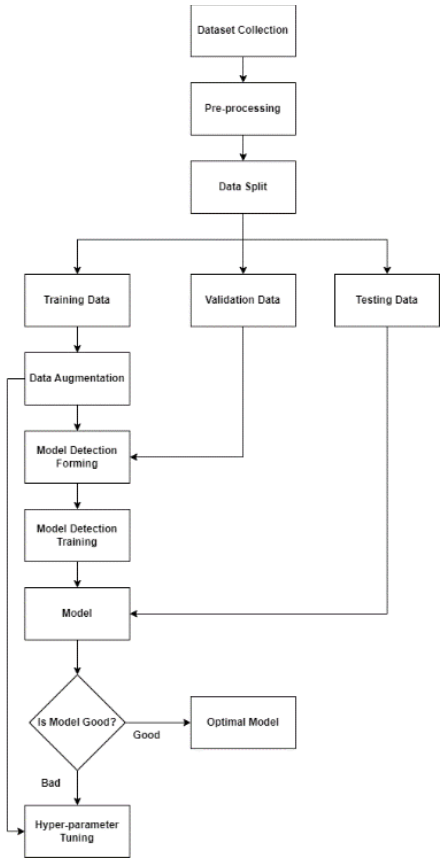

Fig. 2. Flowchart of the research method

IV. Result & Discussion

A. Dataset Collection

We are currently utilizing the PSU Aerial Car Images Dataset [22] sourced from public domains to replicate the procedures of prior studies. [2]. The available dataset comprises a single class, which is further divided into 218 data train and 52 data validation sets. Along with image data, each image data also comes with an annotation file in .xml format containing bounding box coordinates. Furthermore, this research has also utilized the Aerials Car Dataset [23], [2]. The Aerials Car Dataset has 89 images with characteristics similar to the dataset before. A dataset will be selected for testing purposes. Fig. 3 illustrates a car dataset utilized in drone imagery as an example.

Fig. 3. Example of a car dataset on drone imagery

B. Pre-processing

We eliminate duplicated data between the training and test sets during this phase to ensure optimal training. As a result, the complete dataset consists of 266 image data and annotations, with 214 assigned to training and 52 to validation.

It's important to note that the standard format for dataset annotation is typically in .xml, which needs to be converted to .txt to be used with the YOLO algorithm [24][25]. Utilizing the xml_to_yolo_bbox custom function is the ideal solution for this conversion process.

C. Data Split

The PSU Aerial Car Images Dataset undergoes a data split at this stage. This process aims to divide the data into training and validation sets. The goal is to achieve a ratio of approximately 80 to 20, where 80% of the dataset is used for training and 20% for validation. The aim is to ensure optimal use of the data. We require 213 training data and 53 validation data to adhere to the data split plan. Therefore, we must manually move 1 data from the training to the validation set. The combined dataset will comprise 213 training data, 53 validation data, and 1 test data. Afterward, the custom functions "create_train_txt" and "create_valid_txt" will be utilized to produce a .txt file that lists the names of the image datasets utilized for training and validation purposes. Data Augmentation

At this stage, we perform data augmentation on the train data that was previously split. Our study utilizes a custom function called "rotateYolobbox," followed by "rotate_image." The outcome is the addition of 7 new data for each dataset in the train data, with rotations of 45, 90, 135, 180, 225, 270, and 315. This process results in 1704 datasets for training, 53 for validation, and 1 for testing.

D. Model Detection Forming

Once data augmentation is complete, the next stage involves configuring the initial parameters for model building using the YOLOv3 and YOLOv4 algorithms. This is also where each algorithm's pre-trained model and backbone selection take place. Table I provides a view of the initial values for configuring the YOLOv3 and YOLOv4 tuning hyperparameters.

The YOLOv3 and YOLOv4 algorithm detection models are configured based on the number of object classes to be detected, which, in this case, is the car class. The configuration is also tailored to the graphics card's capabilities used in this study, the NVIDIA RTX 3070 TI. The parameters that affect the graphics card's abilities are the batches and subdivisions, with initial values of 64 and 16, respectively. This means that the graphics card can process 64 data in one step by dividing it into 64 data.

When processing an image, the initial configuration considers various parameters, including color saturation, exposure, and hue. Additionally, each algorithm requires a

specific input size, with YOLOv3 requiring 416x416 pixels and YOLOv4 requiring 608x608 pixels.

TABLE I. INITIAL CONFIGURATION OF YOLOV3 AND YOLOV4 TUNING HYPERPARAMETERS.

Parameter	YOLOv3 Value	YOLOv4 Value
Batch	64	64
Subdivision	64	64
Width	416	608
Height	416	608
Channels	3	3
Momentum	0,9	0,949
Decay	0,0005	0.0005
Angle	0	0
Saturation	1,5	1.5
Exposure	1,5	1,5
Hue	0,1	0,1
Learning rate	0,001	0,01
Burn_in	1000	1000
Max_batches	6000	6000
Policy	Steps	Steps
Steps	4800,5400	4800,5400
Scales	0,1;0,1	0,1;0,1

E. Model Detection Training

During this stage, the YOLOv3 and YOLOv4 algorithms will begin training with the initial hyperparameter configuration, pre-trained model, and backbone utilized. The training dataset makes up 80% of the data, while the validation dataset makes up 20%. The length of the training process is dependent on the hyperparameter configuration selected. To view the accuracy and loss graph for YOLOv3, refer to Fig. 4; for YOLOv4, refer to Fig. 5.

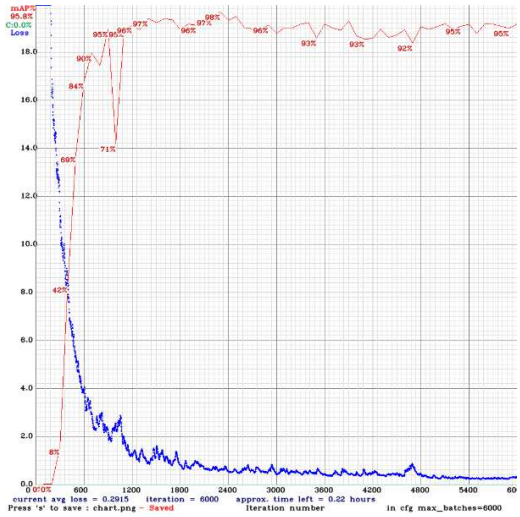

Fig. 4. YOLOv3 training data process

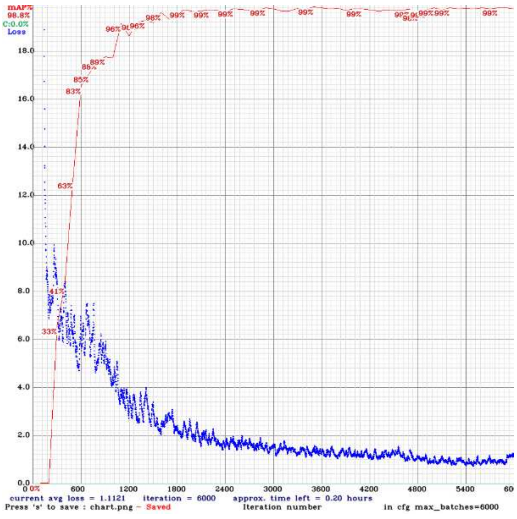

Fig. 5. YOLOv4 training data process

F. Hyperparameter Tuning

In this stage, we will fine-tune the hyperparameters to identify the model with the best performance. Once we have established the detection and training models, we will evaluate and compare the new model with the previous one. Suppose the new model performs worse or not better than the previous model. In that case, we will continue with hyperparameter tuning as per the planned scenario at the beginning of the study. Please refer to Table II and Table III for the hyperparameter configuration constraints and their respective values used in some of the initial scenarios for YOLOv3 and YOLOv4.

TABLE II. INITIAL SCENARIO FOR YOLOV3 HYPERPARAMETER TUNING CONSTRAINTS.

Input	Momentum	Decay	Data Augmentation	Base
416	0.9	0.0005	No	Default YOLOv3
608	0.9	0.005	No	Paper [2]
416	0.9	0.0005	Yes	Default YOLOv3
608	0.9	0.005	Yes	Paper [2]

TABLE III. INITIAL SCENARIO FOR YOLOV4 HYPERPARAMETER TUNING CONSTRAINTS.

Input	Momentum	Decay	Data Augmentation	Base
608	0.949	0.0005	No	Default YOLOv4
608	0.9	0.005	No	Paper [2]
608	0.949	0.0005	Yes	Default YOLOv4
608	0.9	0.005	Yes	Paper [2]

G. Detection Model Results

We have completed the validation performance of the detection model using a 20% validation dataset proportion. We will assess the best and last models in each hyperparameter tuning scenario to evaluate the model's performance. Our evaluation will consider precision, recall, quality, f1-score, and processing time. To calculate the evaluation limit formula, we will use the True Positive (TP), False Positive (FP), and False Negative (FN) results for each

detection model. The YOLO algorithm is used in the odd-numbered schemes, while in the even-numbered schemes, we replicate the reference paper's YOLO algorithm. The accuracy values for YOLOv3 and YOLOv4 are shown in Table IV and V, respectively.

TABLE IV. RESULTS OF ACCURACY, TP, FP, AND FN VALUES IN YOLOV3.

Algorithm	Scheme	Epoch	MAP	TP	FP	FN
YOLOv3	1	1300	96,94%	737	58	40
YOLOv3	1	6000	93,98%	720	53	57
YOLOv3	2	2300	98,40%	749	63	28
YOLOv3	2	6000	95,78%	742	32	35
YOLOv3	3	1600	81,15%	291	15	486
YOLOv3	3	6000	60,34%	138	5	639
YOLOv3	4	1300	70,49%	192	18	585
YOLOv3	4	6000	22,37%	17	3	760

According to research, when default hyperparameters were used in YOLOv3 scenario 1, the best model achieved 96.94% accuracy, while the last model achieved 93.98% accuracy. The hyperparameter tuning process was replicated from previous research on YOLOv3 scenario 2 to improve accuracy. This resulted in a higher accuracy rate for the best model at 98.40% and 95.78% for the last model, compared to scenario 1.

In previous studies, the YOLOv3 algorithm was replicated in scenarios 1 and 2. Now, it's time to evaluate the results of the YOLOv4 algorithm. It's worth noting that YOLOv4 scenario 1 achieved better accuracy results than YOLOv3 scenario 2, with the best model reaching 98.69% and the last model reaching 96.95%. Therefore, it's necessary to run YOLOv4 scenario 2 to compare its results.

TABLE V. RESULTS OF ACCURACY, TP, FP, AND FN VALUES IN YOLOV4.

Algorithm	Scheme	Epoch	MAP	TP	FP	FN
YOLOv4	1	1500	98,69%	756	65	21
YOLOv4	1	6000	96,95%	753	56	24
YOLOv4	2	2200	99,01%	756	79	21
YOLOv4	2	6000	96,63%	757	60	20
YOLOv4	3	5200	99,02%	761	66	16
YOLOv4	3	6000	98,79%	752	63	25
YOLOv4	4	4900	98,99%	735	63	22
YOLOv4	4	6000	98,71%	749	63	28

In scenario 2 of YOLOv4, the best model has shown an increase in accuracy compared to scenario 1, achieving 99.01%. However, the last model experienced a decrease of 96.63%. The training process will include data augmentation in each algorithm's remaining initial scenario plans to improve the results. The goal is to achieve an optimal model outcome.

After implementing data augmentation, the YOLOv4 scenario 3 algorithm showed a significant improvement in the best model's performance, with a 99.02% increase compared to the previous scenario. However, in YOLOv4 scenario 4, both models showed a decrease in performance compared to YOLOv4 scenario 3, with the best model achieving 98.99% and the last model achieving 98.71%. This indicates that YOLOv4 scenario 3 with augmented data had the highest performance among the four scenarios. The classification reports for YOLOv3 and YOLOv4 are presented in Tables VI and Table VII, respectively.

TABLE VI. RESULTS OF PRECISION, RECALL, F-1 SCORE, QUALITY, AND PROCESSING TIME IN YOLOV3.

Algorithm	Scheme	Precision	Recall	F1-Score	Quality
YOLOv3	1	92,60%	94,85%	93,77%	88,26%
YOLOv3	1	93,14%	92,66%	92,90%	86,75%
YOLOv3	2	92,24%	96,40%	94,27%	89,17%
YOLOv3	2	95,87%	95,50%	95,68%	91,72%
YOLOv3	3	95,10%	37,45%	53,74%	36,74%
YOLOv3	3	96,50%	17,76%	30,00%	17,65%
YOLOv3	4	91,43%	24,71%	38,91%	24,15%
YOLOv3	4	85,00%	2,19%	4,27%	2,18%

TABLE VII. RESULTS OF PRECISION, RECALL, F-1 SCORE, QUALITY, AND PROCESSING TIME IN YOLOV4.

Algorithm	Scheme	Precision	Recall	F1-Score	Quality
YOLOv4	1	92,08%	97,30%	94,62%	89,79%
YOLOv4	1	93,08%	96,91%	94,96%	90,40%
YOLOv4	2	90,54%	97,30%	93,80%	88,32%
YOLOv4	2	92,66%	97,43%	94,98%	90,44%
YOLOv4	3	92,02%	97,94%	94,89%	90,27%
YOLOv4	3	92,27%	96.78%	94,47%	89,52%
YOLOv4	4	92,11%	97,09%	94,53%	89,63%
YOLOv4	4	92,24%	96,40%	94,27%	89,17%

Upon thorough analysis of the YOLOv4 results, we have successfully replicated YOLOv3 hyperparameter tuning scenarios 1 and 2 into scenarios 3 and 4, respectively, utilizing the same dataset post-augmentation. However, it has come to our attention that scenarios 3 and 4 exhibited lower accuracy than the original scenarios before augmentation. The best model in scenario 3 achieved an accuracy of 81.15%, while the last model reached 60.34%. On the other hand, scenario 4 produced a best model accuracy of 70.49% and a previous model accuracy of 22.37%.

The model's performance can be controlled and optimized within predefined boundaries by effectively tuning hyperparameters. In both YOLOv3 scenario two and YOLOv4 scenario three, the top-performing models were identified based on their exceptional accuracy and recall values, outperforming all other scenarios.

H. Detection Model Testing Results

The final step is choosing and evaluating the top-performing model among all YOLOv4 scenarios to detect car

objects. The testing data set comprises selected samples from the aerial cars data set. These samples share similar traits with the training and validation data sets but have never been utilized during the training or validation phases, making them unseen data. In scenario 2 of the YOLOv3 model, the testing data successfully detects a car with an indication of its presence through a labeled bounding box. Fig. 6 displays the relevant test results.

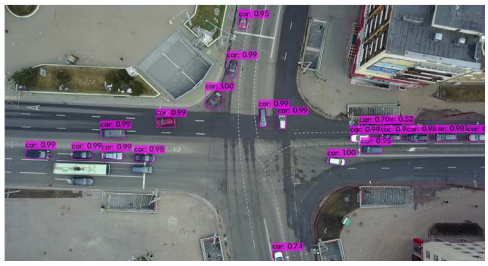

Fig. 6. The results of testing the YOLOv3 detection model in scenario 2

Fig. 7 displays the test results from the YOLOv4's top model in scenario 3, where a car label and its bounding box indicate its detection in the testing data.

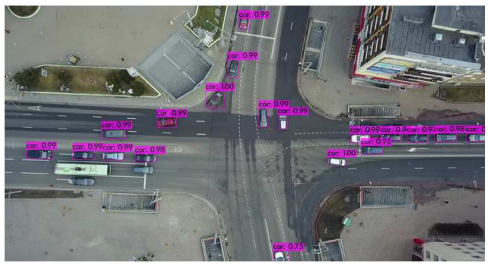

Fig. 7. The results of testing the YOLOv4 detection model in scenario 3

V. CONCLUSION

After conducting numerous hyperparameter tuning scenarios, the YOLOv4 algorithm was implemented as a car detection system on unmanned aerial vehicle images, achieving a remarkable accuracy rate of 99.02% in the best model scenario 3 YOLOv4. Moreover, we replicated the hyperparameter tuning from previous research on YOLOv3 and achieved the highest accuracy value in the best model scenario 2, with a value of 98.40%. Our detection model table analysis revealed that YOLOv4 experienced an increase in accuracy after data augmentation, whereas YOLOv3's accuracy decreased. We recommend replicating the experiment with some modifications to hyperparameter tuning and adding new data through data augmentation techniques to further enhance our findings. This will provide valuable insights into the patterns of accuracy increase and decrease in general for YOLOv3 and YOLOv4 after applying hyperparameter tuning and data augmentation.

REFERENCES

[1] M. Wu, W. Xie, X. Shi, P. Shao and Z. Shi, "Real-time drone detection using deep learning approach", International Conference on Machine Learning and Intelligent Communications, pp. 22-32, 2018.

[2] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, “Car detection using unmanned aerial vehicles: Comparison between faster R-CNN and YOLOv3,” in Proc. 1st Int. Conf. Unmanned Vehicle Syst.-Oman (UVS), Muscat, Oman, Feb. 2019, pp. 1–6.

[3] D. M. Jiménez-Bravo, Á. Lozano Murciego, A. Sales Mendes, H. Sánchez San Blás, and J. Bajo, “Multi-object tracking in traffic environments: A systematic literature review,” Neurocomputing, vol. 494. Elsevier B.V., pp. 43–55, Jul. 14, 2022.

[4] Shehan P. Rajendran, Linu Shine, R. Pradeep and Sajith Vijayaraghavan, "Real-time traffic sign recognition using YOLOv3 based detector", 2019 10th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp. 1-7, 2019.

[5] N. Aswini, S. V. Uma and V. Akhilesh, "Drone to obstacle distance estimation using YOLO V3 network and mathematical principles", J. Phys. Conf. Ser., vol. 2161, Jan. 2022.

[6] Y. H. Liao and J. G. Juang, “Real-Time UAV Trash Monitoring System,” Applied Sciences (Switzerland), vol. 12, Feb. 2022.

[7] G. Liu, J. C. Nouaze, P. L. T. Mbouembe, and J. H. Kim, “YOLO-tomato: A robust algorithm for tomato detection based on YOLOv3,” Sensors (Switzerland), vol. 20, Apr. 2020.

[8] E. R. Setyaningsih and M. S. Edy, “YOLOv4 dan Mask R-CNN Untuk Deteksi Kerusakan Pada Karung Komoditi,” Teknika, vol. 11, pp. 45–52, Mar. 2022.

[9] U. Misra et al., "Rubberband: Cloud-based hyperparameter tuning", Proc. 16th Eur. Conf. Comput. Syst., pp. 327-342, 2021.

[10] Z. Xia, C. Zhang, and J. Kim, “Improving Object Detection Using Weakly-Annotated Auxiliary Multi-Label Segmentation,” IEEE Access, vol. 9, pp. 161283–161291, 2021.

[11] Elgendy, M. Deep Learning for Vision Systems; Simon and Schuster: New York, NY, USA, 2020.

[12] X. Ma, K. Ji, B. Xiong, L. Zhang, S. Feng, and G. Kuang, “Light-YOLOv4: An Edge-Device Oriented Target Detection Method for Remote Sensing Images,” IEEE J Sel Top Appl Earth Obs Remote Sens, vol. 14, pp. 10808–10820, 2021.

[13] H. Linglin, L. Qiang, H. Xianzhen, and L. Maosong, “Research on pruning algorithm of target detection model with YOLOv4,” in Proceedings - 2020 Chinese Automation Congress, CAC 2020, Institute of Electrical and Electronics Engineers Inc., pp. 3283–3287, Nov. 2020.

[14] A. Faustine and G. P. Kusuma, “DETECTING OBJECTS FROM AERIAL IMAGES USING SINGLE-STAGE DETECTION METHOD,” J Theor Appl Inf Technol, vol. 31, 2022.

[15] M. Furqan Rasyid, M. Syukri Mustafa, A. Asvin Mahersatillah Suradi, “Menghitung Akurasi Sistem Deteksi Masker Berdasarkan Arah Pandangan Kepala Objek Calculating Mask Detection System Accuracy Based on Object Head View Direction,” Computer Science Research and Its Development Journal, vol. 14, pp. 203–214, 2022.

[16] R. Z. Fadillah, A. Irawan, M. Susanty, “Data Augmentasi Untuk Mengatasi Keterbatasan Data Pada Model Penerjemah Bahasa Isyarat Indonesia (BISINDO),” JURNAL INFORMATIKA, vol. 8, pp. 208-2014, 2021.

[17] A. E. MINARNO, M. H. C. MANDIRI, and M. R. ALFARIZY, “Klasifikasi COVID-19 menggunakan Filter Gabor dan CNN dengan Hyperparameter Tuning,” ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika, vol. 9, p. 493, Jul. 2021.

[18] Y. Song, W. Taylor, Y. Ge, M. Usman, M. A. Imran, and Q. H. Abbasi, “Evaluation of deep learning models in contactless human motion detection system for next generation healthcare,” Sci Rep, vol. 12, Dec. 2022.

[19] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. Da Silva, “A comparative analysis of object detection metrics with a companion open-source toolkit,” Electronics (Switzerland), vol. 10, pp. 1–28, Feb. 2021.

[20] S. Lasniari, J. Jasril, S. Sanjaya, F. Yanto, and M. Affandes, “Pengaruh Hyperparameter Convolutional Neural Network Arsitektur ResNet-50 Pada Klasifikasi Citra Daging Sapi dan Daging Babi,” J. Nas. Komputasi dan Teknol. Inf., vol. 5, pp. 474–481, 2022.

[21] X. Jiang, T. Gao, Z. Zhu and Y. Zhao, "Real-time face mask detection method based on YOLOv3", Electronics, vol. 10, no. 7, pp. 837, 2021.

[22] PSU Car Dataset. Available online: https://github.com/aniskoubaa/psu-car-dataset (accessed on 20 July 2023).

[23] Aerial-Car-Dataset. Available online: https://github.com/jekhor/aerial-cars-dataset (accessed on 20 July 2023).

[24] W. Widyawati and R. Febriani, “Real-time detection of fruit ripeness using the YOLOv4 algorithm,” Teknika: Jurnal Sains dan Teknologi, vol. 17, p. 205, Nov. 2021.

[25] A. Bochkovskiy, C. Y. Wang and H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection", arXiv preprint arXiv:2004.10934, 2020.