Development of Vehicle Detection and Counting Systems with UAV Cameras: Deep Learning and Darknet Algorithms

Abdul Haris Rangkuti *, Varyl Hasbi Athala, and Farrel Haridhi Indallah

Computer Science Department, Bina Nusantara University, Jakarta, Indonesia;
Email: varyl.athala@binus.ac.id (V.H.A.), farrel. Indallah@binus.ac.id (F.H.I.)
*Correspondence: rangku2000@binus.ac.id (A.H.R.)

***Abstract*—This study focuses on identifying and detecting several types of vehicles, with each vehicle's position depicted by drone technology or an Unmanned Aerial Vehicle (UAV) camera. The vehicle's position is captured from a height of 350 to 400 meters above the ground. This study aims to identify the class of vehicles that travel on the highway. The experiment employs several convolutional neural network models, including YOLOv4, YOLOv3, YOLOv7, DenseNet201-YOLOv3, and CSResNext50-Panet-SPP, to identify this type of vehicle. Meanwhile, the Darknet algorithm aids the training process by making it easier to identify the type of vehicle depicted in MP4 movies. Several other Convolution Neural Network (CNN) model experiments were conducted in this study, but due to hardware limitations, only these 5 CNN models could produce an optimal accuracy of up to 70%. Following several experiments, the CSResNext50-Panet-SPP model produced the highest accuracy while detecting 100% of video data using UAV technology, including the volume of vehicles detected while crossing the road. Other CNN models produced high accuracy values, such as DenseNet201-YOLOv3 and YOLOv4 models, which can detect up to 98% to 99% of the time. This research can improve its capabilities by detecting other classes that are affordable by UAV technology but require hardware and peripheral technology to support the training process.**

***Keywords*—unmanned aerial vehicle, vehicle, Convolution Neural Network (CNN), CSResNext50-Panet-SPP, densenet201-YOLOv3, YOLOv3, YOLOv4, YOLOv7**

I. Introduction

All vehicles can be detected when crossing the highway above ground level. The condition of detecting vehicles that are visible above this height is a problem that needs to be studied optimally so that errors do not occur in determining the type of vehicle. When using a camera from an unmanned aerial vehicle (UAV) to capture vehicle-type data, dynamic footage is produced which often contains unsteady camera movement on some moving objects. Drone technology was initially used in the military field and then widely applied in the civilian field [1]. In recent years researchers have become interested in conducting image processing research on datasets generated from drones using deep learning. The application of drones in various fields such as agriculture, aerial surveys, mapping, photography, surveillance, and others has an impact on the data explosion and the abundance of datasets produced by drones [2]. The consequence is that processing datasets to extract information automatically becomes a necessity, and computer vision becomes one of the relevant information technologies to do the job [3]. Drones are capable of acquiring thousands of high-resolution images during a single flight, which the operator must analyze. For example, in the case of an object search operation, it is necessary to find small objects (e.g., cars) in the image. The size of such an object will not exceed 50 × 50 pixels in an image size of 5000 × 3000 pixels. Eventually, this task cannot be completed for one person without automation, due to the accumulation of additional and more complex image data [4]. Therefore, we need a computational engine capable of performing object detection analysis automatically, accurately, and quickly.

Drones were initially used in coastal and marine areas, but their use has expanded to include plantation, forestry, and mining areas. In addition, the use of drones is also increasingly widespread and developing, if initially they were only used for documentary activities, now they can be more analytical [5]. Analyzing drone-captured data becomes quite easy with follow-up information in the form of metadata recorded in each photo produced by the drone. This metadata stores important information in the form of x and y coordinates and relative elevation points. At first, the flying altitude of the drone during the flight preparation stage will affect its quality. As an initial stage of adjustment, the flight altitude can be set to vary from 300 meters to 400 meters above ground level. Conversely, if the drone flies higher it can cover a wider area. But, the spatial resolution will decrease so that the detail of the map scale and the resulting accuracy is also lower [6].

UAV-captured images and their post-analysis are two major categories that fall in commercial applications of aerial vehicles. Applications in aerial images include landslide mapping, search and rescue, wildlife monitoring, the creation of digital elevation maps, and using mounted

Manuscript received October 18, 2022; revised February 3, 2023; accepted March 13, 2023.

cameras for a multitude of purposes. The technology behind innovation in aerial applications is responsible for digital video stabilization, autonomous navigation, and terrain analysis [5]. One of the attractions is that many researchers apply and use deep learning to handle and process drone data acquisition results, as well as to analyze and detect vehicle object class systems. Several researchers state that deep learning technology is one of the state-of-the-art in the field of artificial intelligence and computer vision for the domains of image classification, object detection, and Natural Language Processing (NLP). The data obtained from drone acquisitions are mostly in the form of images and videos, where drones are flown at various altitudes ranging from low altitudes (10–99 m) and medium altitudes (100–400 m) [7].

Object detection in low-altitude UAV datasets has been performed using deep learning with some CNN models (example of object detections in Fig. 1). Object detection is a technique of identifying variable objects in a given image and inserting a boundary around them to provide localization coordinates. Object detection in aerial images has gained the attention of researchers working in this field as aerial vehicles provide stereo views from a camera mounted on them. Deep learning-based object detection approaches are revolutionizing autonomous navigation vehicles' capabilities [8].

The work presented in the paper is intended to offer detection accuracy in a wide-ranging indication of the use of deep learning-based object detection approaches specifically on low-altitude aerial datasets. It will serve as a repository of all current developments in deep learning-based object detection in low-altitude datasets and also help young researchers to consult research issues for further perusal in this field.

Figure 1. Examples of object detection in UAV datasets [9, 10].

A. Focus and Motivation

Our research is focused on the need to find several methods based on the convolution network model for object detection in low-altitude UAV datasets and group them into three classes. This research is expected to find out the exact accuracy based on the height of the object depicted on the drone/UAV dataset. So that when determining the class for each object, it becomes optimal and reduces errors due to objects that are not too clearly depicted on the UAV Dataset.

In addition, from the deepening of several experiments in detecting objects, the role of the deep learning method becomes very dominant to increase the recognition of objects in the UAV Dataset. Such as the average accuracy in recognizing the detected object. Although, in general, visible objects range between 500–600 meters above the ground surface.

The main objectives of this research are as follows:

1) The system reviewed the taxonomy of deep learning object detection algorithms using several Convolution Neural Network (CNN) models on UAV Camera
2) The system can better detect vehicle types from a height of 300–400 Meters above sea level more optimally by using a UAV camera.
3) The system could produce optimal MAP from each CNN model used so that it can be a guideline for other researchers in detecting objects including the calculation of their volume from the UAV camera.
4) This research can be a new approach, including being one of the raw models in detecting visible objects using a UAV camera.

In addition, carrying out object detection research using a UAV camera for the detection of vehicle objects using deep learning algorithms needs to consider two main factors, namely the accuracy of object detection and calculating its volume as well as producing optimal data processing speed. Some of the advanced deep learning algorithms in the object detection model are YOLOv4, YOLOv3, YOLOv7, Densenet201-YOLOv3, and CSResNext50-Panet-SPP. The display of the detection results using one of the CNN models from the Deep learning algorithm is shown in Fig. 2.

Figure 2. MAP results from object class detection from UAV datasets on public highways using DenseNet201-YOLOv3.

Figs. 2–3 illustrate the MAP results from processing UAV Datasets in determining the object's class that passes

through public highways. With the training process carried out by the Darknet algorithm, it will classify the object class that is detected in an object in the bounding box. The concept of object class detection from UAV Datasets in the video form is a new approach that can continue to be developed for MAP. The difference between the two figures occurs at the initial time detection of objects of each class for the CSResNext50-Panet-SPP model. The percentage of accuracy starts at more than 88% for the CSResNext50-Panet-SPP model, while in the CNN model, the average accuracy percentage starts at 60–65%. After that, it only reaches 100%. Especially when detecting motorcycles, the dataset from the drone in video form is too small.

Figure 3. MAP results from object class detection from UAV datasets on public highways using CSResNext50-Panet-SPP.

II. Literature Review

Drones or Unmanned Aircraft have been developing and the existence of UAV technology in the world of aviation continues to experience increasing development in recent years. However, as air transportation, it is also used in commercial and military circles, including a technology that has other functions such as regional mapping, the film industry, maritime patrols, disaster, medical assistance, and forest fire detection [8]. One of the technologies mentioned is the presence of UAVs (Unmanned Aerial Vehicles) [11]. UAV is a pilotless aircraft that is operated using a remote control or automatic control. UAVs can be remotely controlled and have various shapes, sizes, configurations, and characters. The results of data collection from Drones/UAVs are used for the implementation of several deep learning algorithms for object detection in the processing of datasets generated by drones. The researcher also uses a UAV Dataset that is flown at low altitudes, and in his research, the dataset used is the result of drone acquisition at an altitude of 350m [12]. There are at least four categories in computer vision on the UAV Dataset, namely image object detection, video object detection, single object tracking, and multi-object tracking. The characteristics of the dataset are categorized into two types, namely urban and sub-urban areas. Several researchers said that there were several issues and problems with the UAV Dataset, namely small objects, occlusion, variations in spatial scale and resolution, and class imbalance [13, 14].

Another research where there are so many obstacles that are mostly faced by UAVs is the difficulty of landing on a base. This difficulty can be solved by the renewal of UAVs which is the development of landing vision by detecting the helipad to prevent the risk of accidents that could be harmful and could lead to death [15]. Another study uses UAVs to detect forest and land fires so that they have an impact on ecosystem damage, besides that, forests in Indonesia continue to shrink every year due to forest fires. One solution to this problem is to use a Unmanned Aerial Vehicle (UAV) to make direct observations through the camera. The detection of fire produced an average accuracy of 0.92. The best accuracy was obtained on the 3rd test with a precision score of 0.96, a recall score of 0.98, and an accuracy score of 0.96, so this research can continue to be developed [16]. In another paper, an approach for vehicle detection is presented with virtual line-based sensors which are just straight detection lines that are first set on-road lanes. The proposed method has an outstanding advantage in any condition such as excellent traffic jams as well as sunny, cloudy, and rainy days, or nighttime, or even tunnels with complex illumination [17].

Several studies on UAVs have been widely published in international journals and conferences in different application areas such as search and rescue [18], air security and monitoring [19], disaster management planning [20], plant management vision [21], and mission communication [22]. The vehicle has the ability to fly at different speeds to hover over the target, perform outdoor flights, and maneuver at a close range of objects over a suitable place [23]. These features make it suitable for operations where human intervention becomes difficult to perform completely and can replace humans. Some of the major challenges in low-altitude UAV-based object detection when compared to standard images such as large-scale variety, dense distribution of objects, arbitrary orientation, object relative motion, and turbulence of atmospheric conditions cause objects to become blurry [24]. All these challenges lead to an object development detection approach in low-altitude aerial images using low-level scene features as well as immersive features to process. There are some other important critical issues in object detection on drone platforms due to differences in mAP can be seen [25]. An overview of the percentage of drone technology utilization in several types of activities supported by deep learning algorithms. Such implementations can be seen in Fig. 4.

It has been quite evident in recent years that there has been a boost in research publications due to the emergence of deep learning-based object detection, but achieving a high level of accuracy is challenging in the case of low-altitude UAVs. The domain of object detection was infinite

in nature if we consider each development, we would strictly stick to algorithms that have scope in low-altitude aerial images [26]. The literature on object detection in aerial images had been classified into two categories: classical and modern object detection approaches. The classical categorization includes conventional techniques which include vision-based as well as machine classifier-based approaches. Whereas modern deals with deep learning-based algorithms which are our focus area. Classical approaches to object detection include all major developments made in the field of aerial images using handcrafted features-based machine learning approaches [27].

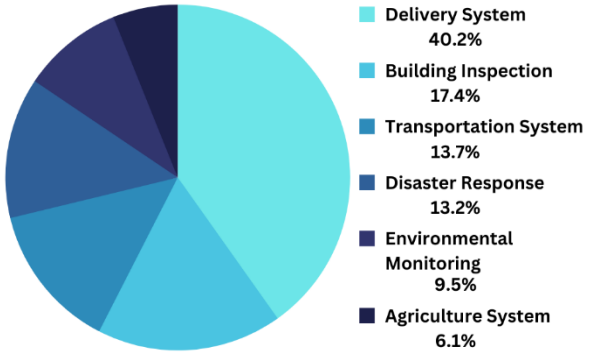


Figure 4. Some examples in some field of the use and utilization of object detection [1].

III. Materials and Methods

This study is for vehicle object recognition on UAV datasets with distances between surfaces ranging from 300 to 400 m. Detected objects are classified into three types: cars, trucks, and motorcycles. There are two types of trucks: trailer trucks and regular trucks. The car is not classified by vehicle type. The motorcycle object appears to be very small, making it difficult to distinguish it from a bicycle.

The preparation stage, training stage, and testing stage are the three major parts of this research. The preparation stage consists of gathering video data and analyzing the video dataset. All collected data is processed for the training stage, and the weight of each data is calculated so that it can be recognized during the testing process. The testing process involves recognizing test data based on previously collected training data. Furthermore, a more detailed explanation can be found in the sections that follow.

A. Collecting Materials And Preparation

An overview of the processes that occur at the preparation stage can be seen in Fig. 5.

Fig. 5 depicts the preparation stage. At this point, video data is collected using Drone/UAV technology, and the drone video dataset is used as training data. However, the object detection algorithm cannot detect objects in a raw video without first being trained. As a result, videos captured by drones must be processed first for the machine to understand them. Each sample video from the drone is processed with FFMPEG to obtain screenshots of the video every few seconds. This process's goal is to prepare the image for the image labeling process. Image labeling is required to teach the machine which objects to identify. As shown in Fig. 6, the drone videos have been converted into new images. There is also a text file for each image which contains data of objects for the machine to learn in the image.

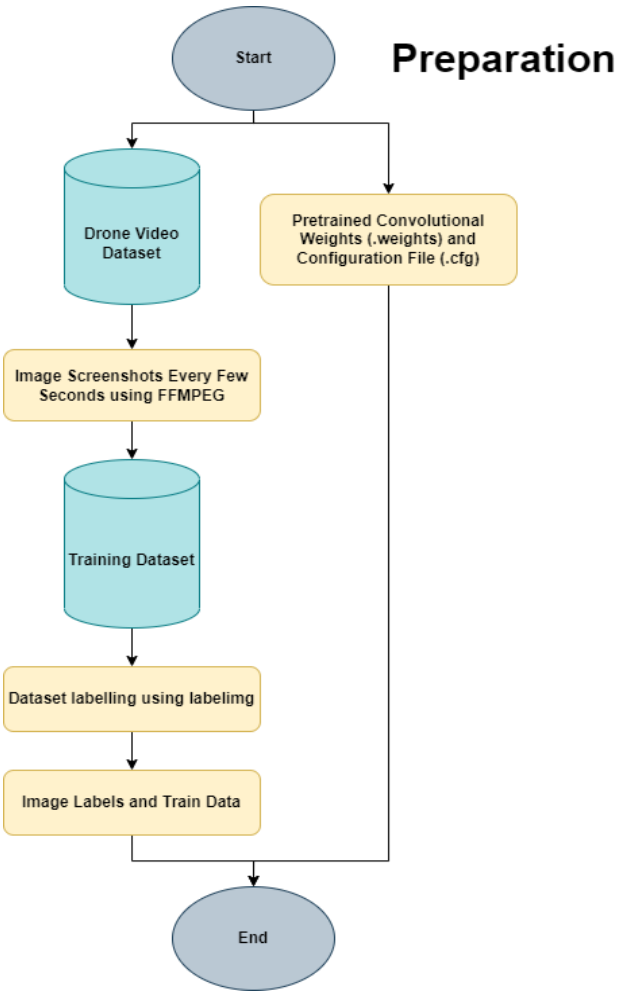


Figure 5. The preparation stages in vehicle object detection.

After converting and labeling the images, another file must be prepared. A file called trainer. Data containing labeled drone images, total class value, and training output folder must be created. Another process in this stage is to obtain pre-trained convolutional weight and configuration of the CNN model from the internet. Then, the configuration file must be configured depending on the total class. These configurations can be seen in the following section as training hyperparameters.

Fig. 7 illustrates the training stage, which generates weight files. As stated before, it requires files which are images for training, image labels, class identity, training data, pre-trained model weight, and a matching model configuration before training. To decrease the variance when training, the training configuration of some hyperparameters for all CNN models is equivalent to one

another. The equated training hyperparameters are shown in Table I.

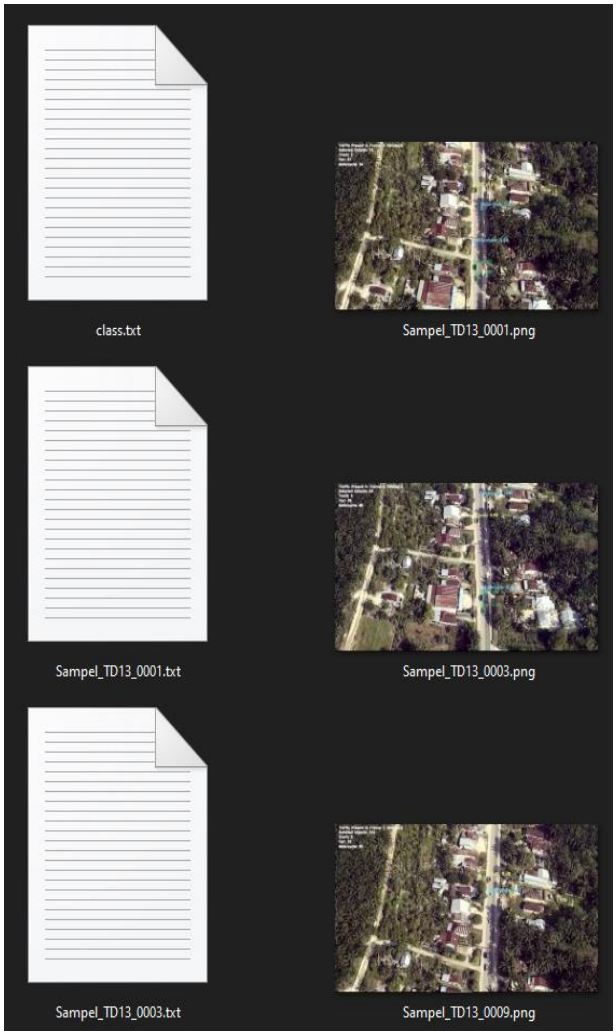


Figure 6. Labeled drone image samples.

TABLE I. DARKNET TRAINING CONFIGURATION

Hyperparameters	Value
Batch	32
Subdivisions	32
Max_Batch	6000
Classes	3

After preparing the required files, the training stage can be started. The training stage iterates until the iteration modulus is either completed or until it finishes the 1000th iteration modulus. This stage aims to use the drone image data and supporting files to be trained for each iteration using the Darknet algorithm. The results of the training phase are weight files. Weights are generated by the choice of a target-object-space, which depends heavily on the nature of the objects in the training set and the predicted property". To obtain the most accurate data possible, the training process is conducted using the same device and training dataset. The result would be in a less ambiguous weight file. To support the experiment, the Darknet framework was used to help only in the training process which trained some CNN models.

B. *Training Dataset*

The next stage is the training process which can be seen in Fig. 7.

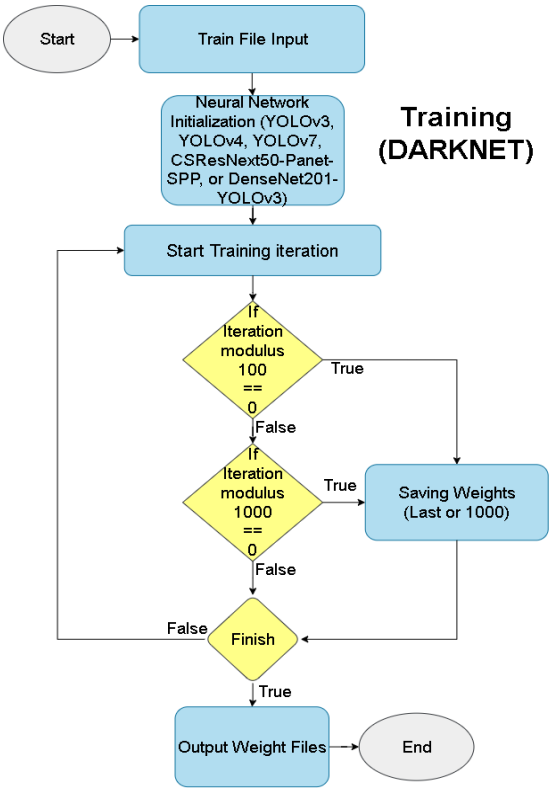


Figure 7. The training stage in vehicle object detection.

TABLE II. BASIC INFORMATION OF CNNS

CNN Model	Total Layers	BFLOPs	Network Size
YOLOv3	81	65.319	416
YOLOv4	137	59.578	416
YOLOv7	132	43.625	416
DenseNet201- YOLOv3	300	44.438	416
CSResNext50-Panet-SPP	112	99.411	608

Table II shows the basic information on CNNs that are used for the experiment. YOLOv3 has the least layers with only 81 layers which could indicate the smallest model in the experiment. DenseNet201- YOLOv3 on the other hand has a total layer of 300. DenseNet201-YOLOv3 was the largest model tested in the experiment according to the number of layers. However, information about the model's architecture alone cannot justify the size and real-world performance of the model. BFLOPs and network size are parameters that can determine the model performance. In this case, the table indicates that YOLOv7 has the lowest required performance to run on a single image (BFLOPs) even with a total layer of 132. This is followed by DenseNet201-YOLOv3 with a required performance of 44.438 BFLOPs. Meanwhile, the highest BFLOPs value in the experiment is 99.411 BFLOPs, which is required to run CSResNext50-Panet-SPP. To prove this performance fact, tests need to be done which will be explained further in the paper.

C. Testing Process

After the training process was carried out, it was continued with the testing process on vehicle object data taken from other UAV Datasets. Based on the training data, all testing data would be detected optimally. In this experiment, four CNN models were also used to know which one is optimal for carrying out the vehicle object detection process. The stages of the testing process can be seen in Fig. 8.

Fig. 8 illustrates that after the training is complete, the system can start the testing phase by importing the required files for testing. These files are the trained weight file, configuration file, and trainer data file. The testing process starts by taking a frame from the UAV Dataset in video form. Then, the system calculates the prediction using non-max suppression (NMS). The other process is drawing the bounding box and the system will determine the call traffic type and calculate the volume of the object class as the target object. The bounding box usually comes with other information like class and coordinate. This information is important for vehicle object detection and keeping ID. This is to generate the appropriate virtual key for the UI to receive after the system succeeds to detect the vehicle object. Then, an algorithm using conditional cases decides which behavior to apply within the case including showing the accuracy starting with the lowest percent will grow up after detecting the vehicle clearly. Then, the system draws a line to see the direction of the object if it has moved in the frame. Finally, the bounding box of the vehicle should respond to the ID key and accuracy percent according to the predefined bindings.

If the different objects are too close and have the same ID, then two choices are made, if yes, then a new ID is assigned, but if not, the old id is assigned. However, for the assigned ID and old ID. At the same time, the object is shown from the processed frame. In this testing stage, the output is accuracy, precision, F1 Score, and others, so that the accuracy percentage in vehicles detection in the video form can be acquired by the UAV Dataset in the show frame process with the output testing is the performance of vehicle detection on the video UAV Dataset. The outputs are metrics that would be used to measure the performance of every CNN model.

The types of vehicle detection studied use different CNN models to improve recognition, each model would result in a different performance. This phenomenon happens because each CNN model had a different architecture that made each model unique. This is why some models could work well when in a certain situation especially to detect the vehicle object. Therefore, by using the data collected throughout the training process, it will be possible to compare the five CNN models. Several metrics, such as mAP, precision, recall, and F1-score are used as comparison variables. The comparison variables between each CNN model will determine which approach is the best to detect vehicle objects on the highway based on UAV Dataset in Video form. The samples needed to measure the CNN model's performance are split into two categories. The first category is the positive samples, which have the targeted object in them. The second category is the negative samples, which have none of the targeted objects in them.

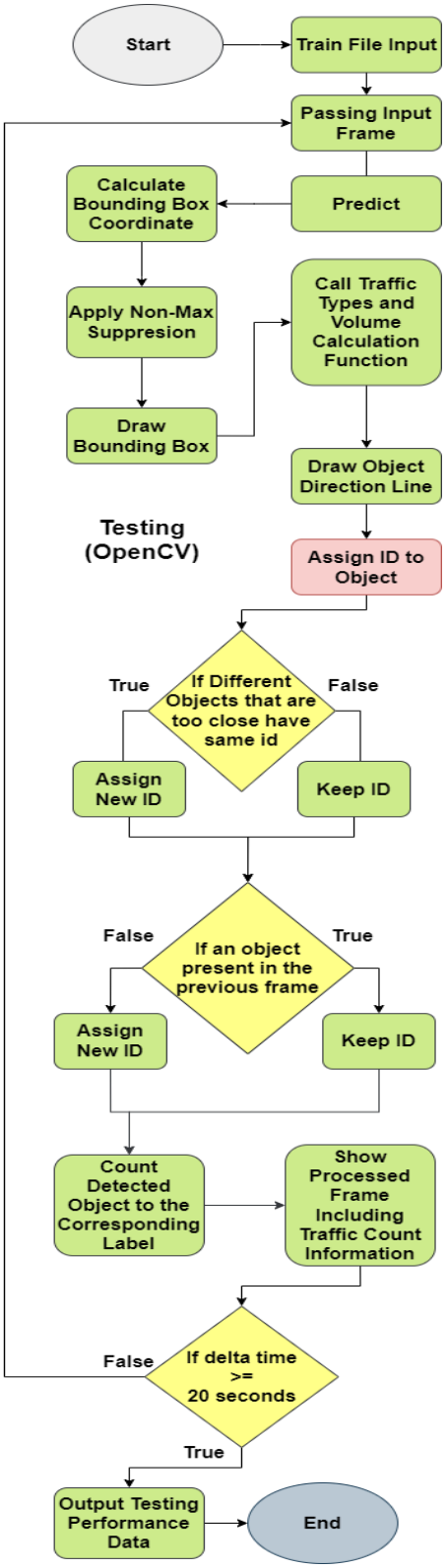


Figure 8. The testing stage in vehicle object detection.

1) Precision and recall

To calculate the precision value, there are two necessary variables. The first variable is the number of positive

samples that the model correctly classified and the last one is the total number of samples that are classified as positive samples (whether the model correctly classified them or not). The range value of precision is from 0 to 1, with 0 as its lowest score and 1 as its highest score. This precision value reflects how reliable the model is when classifying the positive samples. The result of precision is obtained by dividing only correctly classified positive samples by the total number of positive samples. Compared to precision, a recall is calculated by dividing the number of positive samples that the model correctly classified and the number of total positive samples [28, 29]. Recall completely ignores the negative samples and only focuses on the result of the positive samples. With the range the same as precision, a recall measures how many positive samples are correctly classified by the model. Both precision and recall formulas are illustrated below:

$$\text{Precision} = \frac{TRUE_{positive}}{TRUE_{positive} + FALSE_{positive}} \quad (1)$$

$$\text{Recall} = \frac{TRUE_{positive}}{TRUE_{positive} + FALSE_{negative}} \quad (2)$$

where:

$TRUE_{positive}$ = total of positive samples that the model correctly classified

$FALSE_{positive}$ = total of negative samples that the model mistakenly classified as positive samples

$FALSE_{negative}$ = total of positive samples that the model couldn't classified

2) Intersection over union (IoU)

In Intersection over Union or IoU, there are two things that need to be addressed because the two are defined later in the IoU formula. Those two values are the predicted bounding box and the truth bounding box. The predicted bounding box is the box that the model predicts on having one of the targeted objects or items. Meanwhile, the truth bounding box is the box that the tester initially marked as the targeted object before the measuring process. Finally, the definition of IoU is the ratio between the intersection of the predicted bounding box and the truth bounding box with the combined area or union of the two boxes (see Fig. 9). The more the predicted box overlay the area of the truth box, the higher the accuracy of the model. In return, the IoU score would be near the value of 1, which is the highest accuracy score [28].

3) F1-Score and mAP (mean average precision)

F1 Score used the two previous metrics, which are precision and recall, F1-Score is a metric that combines the precision and recall metrics into a single metric. The formula for the F1-score is defined as the average of precision and recall [28, 29]. The output of the formula will give an F1-score value ranging from 0 to 1, where 1 is the highest accuracy value.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

In addition to the F1-score that summarizes the two previous metrics, the mean Average Precision(mAP) is the metric that shows the mean value of average precision for the detection process of all the previously determined classes [17]. Performance accuracy test results of object detection on the UAV Dataset have been carried out. The test results were done using a performance metric called the mean average precision (mAP) [11]. The formula is as follows:

$$\text{mAP} = \frac{\sum_{q=1}^{Q} \text{AveP}_{(q)}}{Q} \quad (4)$$

where Q is the number of queries in the dataset and AveP(q) is the Average Precision (AP) for a particular query, q for a given query, q, the corresponding AP is computed, and then the average of all these AP scores will give you a single number, called MAP which measures how well our model performs querying. Average Precision, or AP in short, is the average of the precision metric across all recall values between 0 and 1 at various IoU thresholds [28]. The mAP model will be one of the core metrics to determine which model has the best overall performance because it takes considers all previously mentioned metrics.

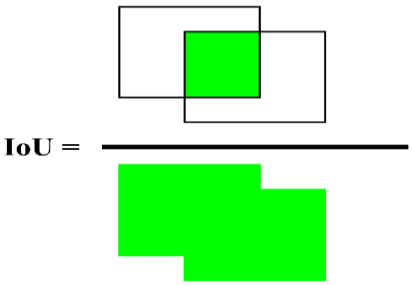


Figure 9. Illustration of intersection over union (IoU).

IV. Result and Discussion

A. Training Results

The training uses PCs with the latest CPU and GPU technology. The use of PC technology will not affect the aftermath use of the models. This method has a benefit in accelerating the training duration of the model because the Darknet framework supports the GPU Acceleration method for the training phase which reduces the training time when compared to using the CPU for training. If the training process is done with less advanced technology, then it would take a lot more time to finish the study because the output weight file would result in the same file as the advanced CPU. The hardware Specifications needed to support the vehicle detection and the counting system with UAV Camera which can be seen in Table III.

TABLE III. HARDWARE SPECIFICATION ON THE EXPERIMENT

No.	Component	Specification
1	Processor	Ryzen 5 5600H
2	RAM	16 GB DDR4
3	GPU	NVIDIA RTX 3060
3	VRAM	6 GB

The training process uses the Darknet algorithm model. As the model architecture increases in size, the machine's training speed decreases. The size of the model also affects the model output size. Fortunately, all of the tested models resulted in a good loss average result that are displayed in the form of CNN model training graphs, which can be seen in Fig. 10.

Fig. 10 (a) illustrates the training process of YOLOv4 in a graph. The YOLOv4 training graph shows a significant decrease in loss percentage during the first 1200 iterations. But then it would steadily decrease with a few little increases and decrease until the last iteration. In Fig. 10 (b) shows the training process of YOLOv3. Unlike YOLOv7, which has a consistent loss percentage for the majority of its iterations, both DenseNet201-YOLOv3 and CSResNext50-Panet-SPP show an unsteady decrease loss percentage. YOLOv3 exhibits a similarly steady decrease in loss percentage as YOLOv7.

Fig. 10 (c) shows the training process of YOLOv7 in a graph. The loss percentage began to fall dramatically after about the 200th iteration and continued to fall until about the 540th iteration. Between the 540th and 700th iterations, the loss percentage began to rise again and reached a maximum of 2%. After the 700th iteration, the loss percentage once again decreased dramatically and continued to do so until the 800th iteration. The loss percentage began to decrease steadily after the 800th iteration as the iterations progressed. Fig. 10 (d) shows the training process of CSResnext50-Panet-SPP in a graph. In Fig. 10 (e), a graph depicts the training process of DenseNet201-YOLOv3.

At first, most of the model's loss declined in the first 600 iterations. YOLOv7 on the other hand declined faster than other models. However, this decline will not have any significant impact as long as the loss declines to an acceptable level. After a steep decline at the start, the loss starts to stabilize in a gentle curve. It shows that the model is starting to understand the given dataset. Finally, the graph shows that the loss stabilizes until the end of the iteration. This result suggests that the trained model has effectively learned the provided vehicle dataset. As stated, each model architecture is unique and has its own distinct benefits in particular circumstances.

Therefore, the experiment can go on using the generated train weights. A more detailed training result can be seen in Table IV.

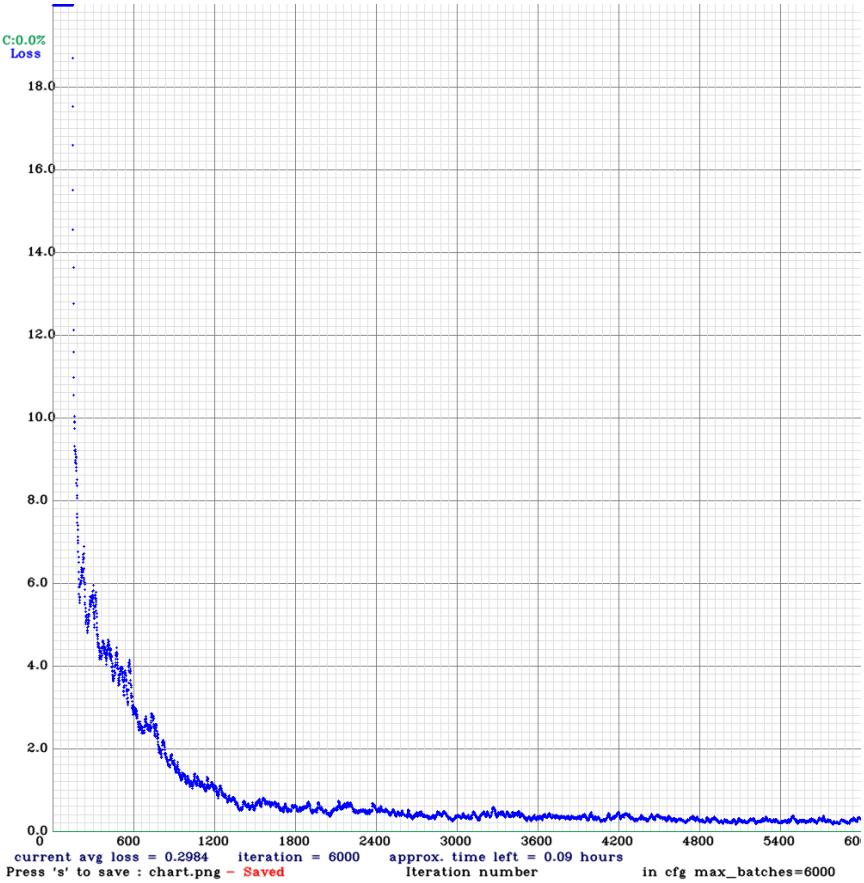
Table IV showed the average loss of the CNN model to determine how it will perform. Thus, carrying out is one of the parameters that could affect the test results. As explained before, the lower the average loss, the better the machine in understanding the dataset. This way, it could potentially affect the performance of detecting objects. If the machine doesn't understand how to differentiate one different vehicle type from another, then the machine wouldn't be able to detect the object as well as it would expect. As explained before, the lower the average loss, the better the machine in understanding the dataset. Table IV shows that all of the models have an average loss below 0.3. This value is low enough and acceptable for the experiment. DenseNet201-YOLOv3 has the lowest average loss of 0.1358 with a total of 5.49 h of training time. When detecting vehicle objects on highways, CNN model YOLOv3 has the second lowest average loss of 0.1546 with an approximate training time of 3.05 h. Next, YOLOv7 has an average loss of 0.1887 and an approximation training time of 3.13 h. However, the difference between them is only 0.0341. The YOLOv4 model comes second to last with an average loss of 0.2984 with an approximation training time of 4.51 h. Lastly, the CSREsNext50-Panet-SPP model has an average loss of 0.2985 with an approximate training time of 7.46 h. All CNN models must be properly tested and analyzed; the performance of a model can not be determined by using only the average loss value. Therefore, the following section will explain more about the performance in other aspects.

TABLE IV. TRAINING RESULT OF EACH CNN MODELS USING AVERAGE LOSS PARAMETER

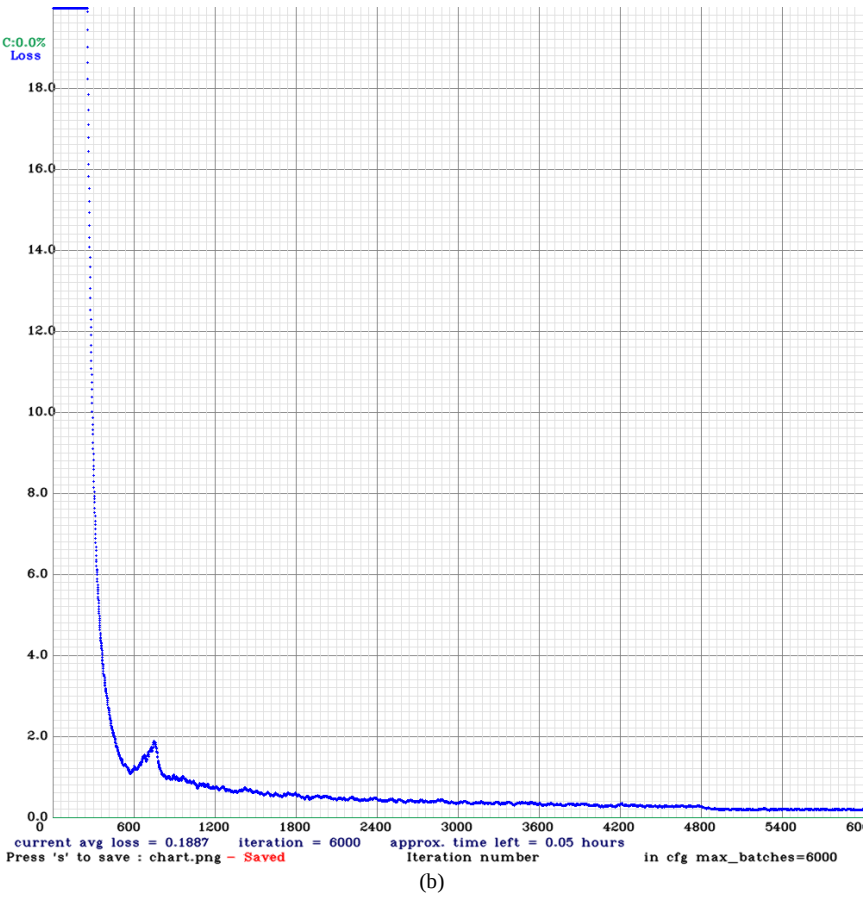
Model	Average Loss (%)
YOLOv3	0.1546
YOLOv4	0.2984
YOLOv7	0.1887
DenseNet201-YOLOv3	0.1358
CSResNext50–Panet-SPP	0.2985

B. Simulation and Results

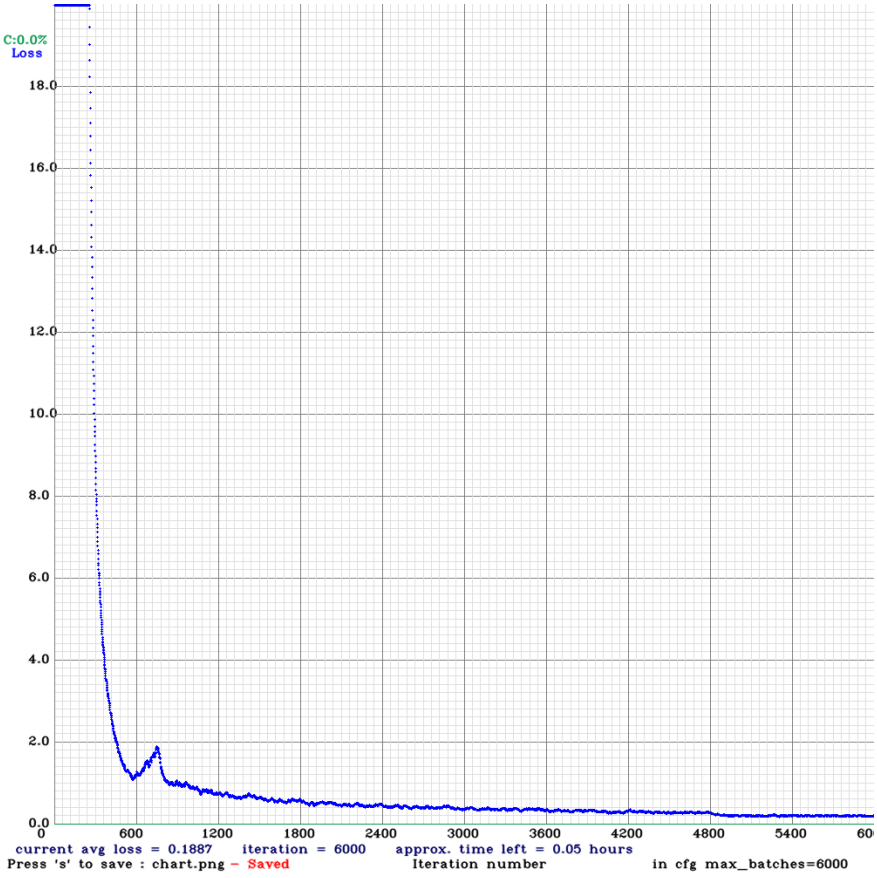
Before testing the weights trained in the self-service application, the vehicle object detection algorithm needs to import the supporting files. The supporting files are the training label, image path, model configuration, and a .data file type called the trainer.data. These supporting files are necessary for executing the testing process which uses the OpenCV library as the inference. OpenCV itself is an open-source library mainly used for image processing [20]. Then, the self-service module for accuracy and vehicle class detection will be automatically initiated. Next, an examination of vehicle object detection is performed. This examination was needed to carry out if the trained object detection works properly. In this case, the frames that are captured and processed by the machine were examined in a separate window. Fig. 11 provides an overview of the simulation used to detect vehicle class objects in the UAV dataset.


C:0.0%
Loss
18.0
16.0
14.0
12.0
10.0
8.0
6.0
4.0
2.0
0.0
0
600
1200
1800
2400
3000
3600
4200
4800
5400
60
current avg loss = 0.2984 iteration = 6000 approx. time left = 0.09 hours
Press 's' to save : chart.png - Saved
Iteration number
in cfg max_batches=6000

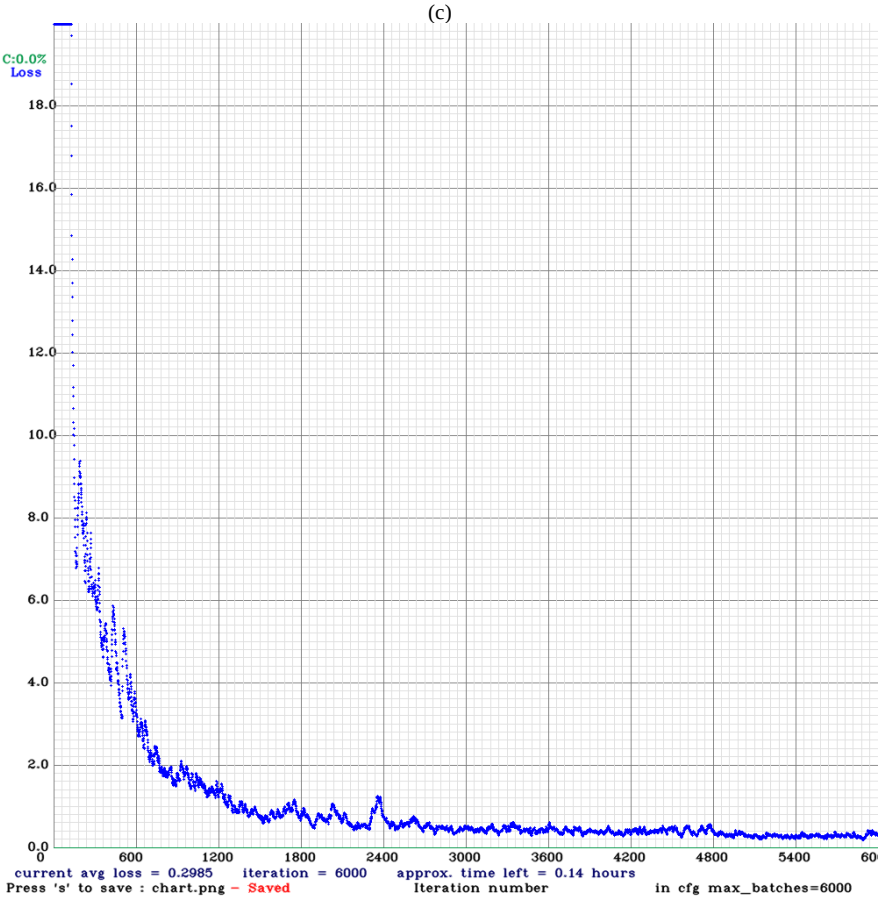
(a)


C:0.0%
Loss
18.0
16.0
14.0
12.0
10.0
8.0
6.0
4.0
2.0
0.0
0
600
1200
1800
2400
3000
3600
4200
4800
5400
60
current avg loss = 0.1887 iteration = 6000 approx. time left = 0.05 hours
Press 's' to save : chart.png - Saved
Iteration number
in cfg max_batches=6000

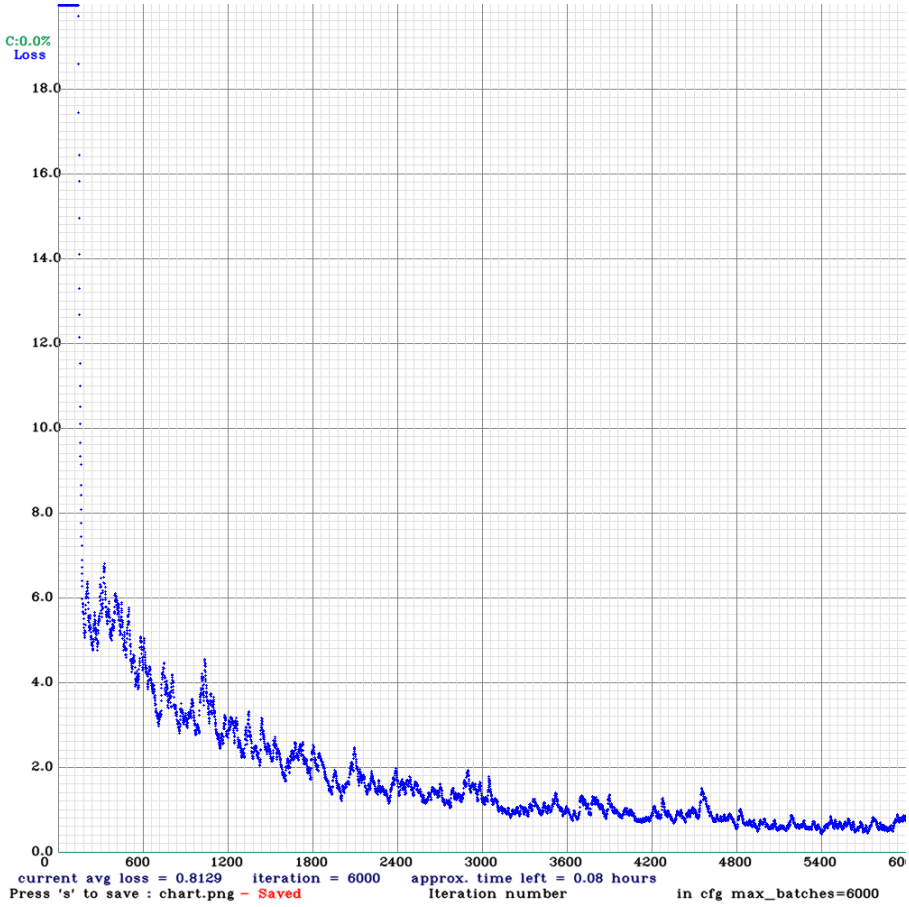
(b)


C:0.0%
Loss
18.0
16.0
14.0
12.0
10.0
8.0
6.0
4.0
2.0
0.0
0
600
1200
1800
2400
3000
3600
4200
4800
5400
60
current avg loss = 0.1887 iteration = 6000 approx. time left = 0.05 hours
Press 's' to save : chart.png - Saved
Iteration number
in cfg max_batches=6000

(c)


C:0.0%
Loss
18.0
16.0
14.0
12.0
10.0
8.0
6.0
4.0
2.0
0.0
0
600
1200
1800
2400
3000
3600
4200
4800
5400
60
current avg loss = 0.2985 iteration = 6000 approx. time left = 0.14 hours
Press 's' to save : chart.png - Saved
Iteration number
in cfg max_batches=6000

(d)



(e)

Figure 10. Training graph of CNN Models: YOLOv4 (a), YOLOv3 (b), YOLOv7 (c), CSResnext50-Panet-SPP (d), and DenseNet201-YOLOv3 (e).

Fig. 11 showed that when the application detects vehicles on the highway, it can run some objects and calculations flawlessly with above-acceptable performance. That means a great response feel and fast processing speed. It is required for real-time object detection to make sure everything is processed without delay between one frame to another frame and interactions around the bounding box with accuracy number. However, When the system was tested to run the same object detection, the response speed and processing time were not acceptable. In the experiment, all of the model's network input sizes depend on object classes such as trucks, cars, and motorcycles. This is to reduce the processing load which could increase processing time. The result of the bounding box calculations, using the combined area or union of the two boxes are presented in Fig. 12.

Fig. 12 showed the result of calculating the truth bounding box with the combined area or union of the two boxes. Calculating the IoU performance for the highest on the CNN model, namely CSResNext50-Panet-SPP at 91.42%, followed by the CNN YOLOv4 model at 86.11%. On the other hand, the CNN DenseNet201-YOLOv3 model had the lowest IoU performance at 68.94%. Therefore, this experiment shows that the CSResNext50-Panet-SPP model has the highest IoU performance and can serve as a guideline for future research.

Fig. 13 informs the inference time of all five CNN models relative to their mAP@0.50 percentage or their accuracy. Inference time calculates the processing time between the captured frame and its results in a form of prediction. The higher the inference time, then the slower the detection process becomes. This also worsens the experience of using this object detection technology because of its slow processing capability. In Fig. 15, CNN Model shows an additional percent more accuracy than half the amount of the original inference time. In the vehicle object detection process, the UAV Dataset shows that the average inference time is more than 50% compared to the average accuracy of the image predicted by the system. For example, the CSResNext50-Panet-SPP model has 100% accuracy, but the inference time does not result in 50%.

Instead, the inference time that the model has is more than 100% which is 173.44 ms. Other models have the same results as CSResNext50-Panet-SPP. The YOLOv4 model has an accuracy of 99.19%, while the inference time is more than 50% which is 83.84 ms. The YOLOv3 model has an average accuracy of 95.73%, but the inference time is more than 50% which is 81.32 ms. YOLOv7 and DenseNet201-YOLOv3 have an inference time of less than 50 ms. DenseNet201- YOLOv3 has the lowest inference time with 38.65 ms but also has the lowest mAp@0.50 percentage which is 92.39%. On the other hand, YOLOv7 has a low inference time (44.98 ms) but high compatibility (99.61% mAp@0.50). Calculating recall, precision, and F1 scores with deep learning algorithms on several CNN models can be seen in Fig. 14.

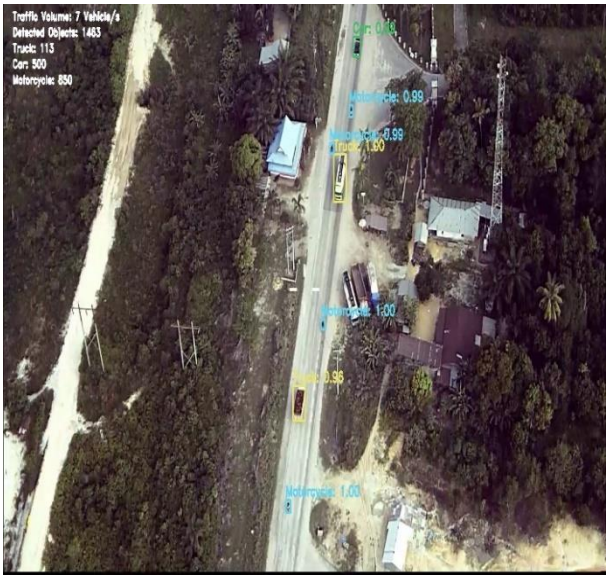


Figure 11. Vehicle class object detection simulation on UAV dataset.

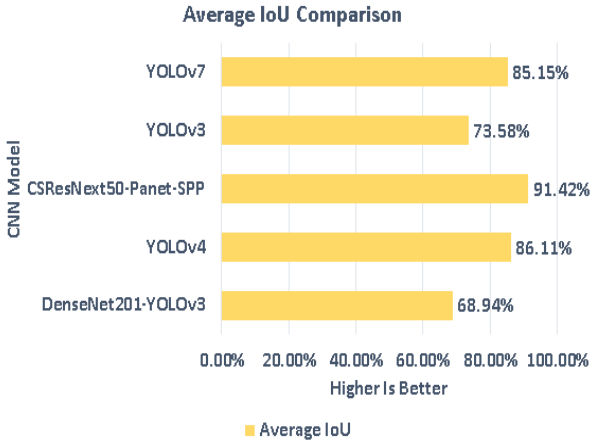


Figure 12. CNN models average IoU after optimization comparison

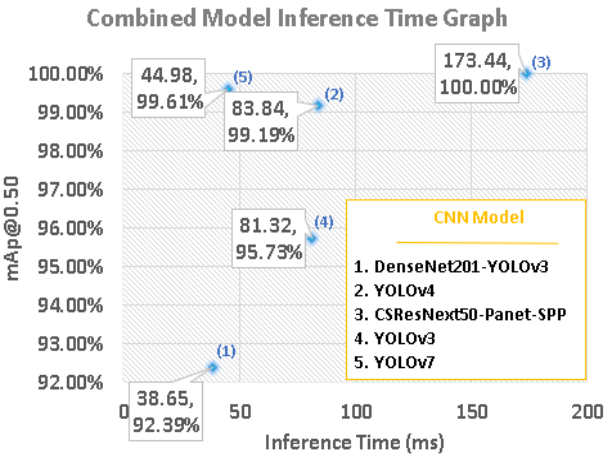


Figure 13. Combined inference time model graph before and after optimization.

Fig. 14 described an experiment using a UAV Dataset when detecting three object classes, namely the truck class which includes trailer trucks, and several models of cars, and the motorcycle class which includes bicycles because the object looks very similar. Based on this experiment, the highest accuracy is in the CNN CResNext50-Panet-SPP model where the percentage of precision, recall, and F1 Score reaches 100%. Followed by the YOLOv4 model where the average is up to 98.67% and the lowest accuracy is DenseNet201-YOLOv3 with an average value of 91.67%. So, it can be concluded that the approach of the deep learning algorithm with several CNN models in the testing process would be supported by the Darknet algorithm. The minimum average accuracy is only 91%. So, it can be concluded that object detection with this approach has worked well and could detect all the objects almost perfectly.

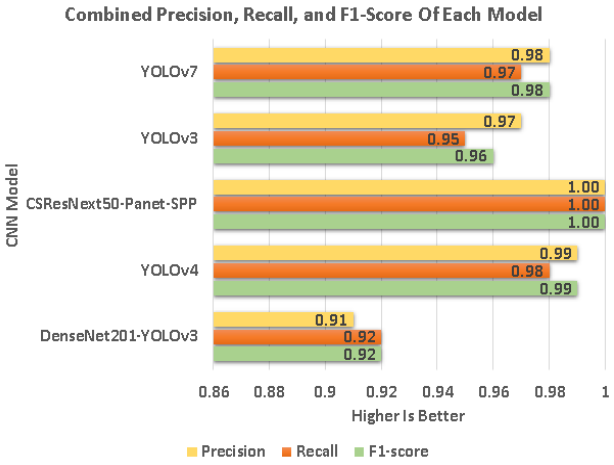


Figure 14. Precisions, recall and F1-score of each model CNN.

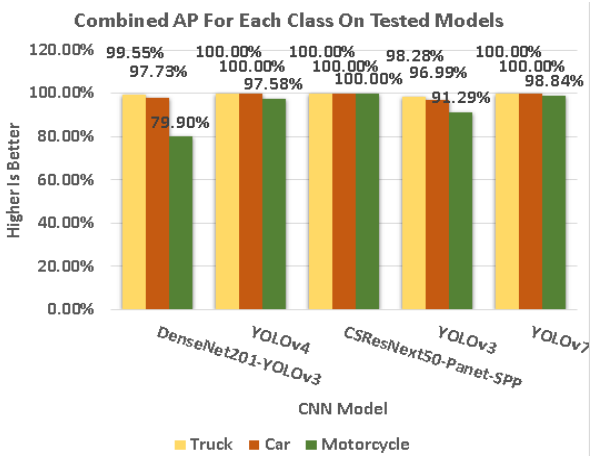


Figure 15. The combination of the average accuracy for each vehicle class on the CNN model.

Fig. 15 showed the average accuracy when recognizing three vehicle class objects using several CNN models for this experiment. The CSResNext50-Panet-SPP model can recognize all vehicle objects ranging from trucks (including trailers), cars (several types of cars), and motorcycles (including bicycles) on the UAV Dataset where the distance between the surface and the top position of the drone is between 300-400 meters with the ground moving vehicles for each class up to 100%. The YOLOv4 and YOLOv7 model can also detect all classes of vehicle objects very well, such as trucks and cars up to 100%,

while motorcycles (bikes) can detect up to 97.58% for YOLOv4 and 98.84% for YOLOv7.

The introduction continued by using the DenseNet201-YOLOv3 CNN model where the accuracy of trucks can be recognized well. But for cars, there are only a few errors where the accuracy reaches 99%, while motorcycle objects can be recognized only up to 79.90% which is the lowest percentage of all types. For the recognition of moving objects on the UAV Dataset, the YOLOv3 model is used. Although its accuracy for motorcycle objects is only 91.29%, it is still robust in recognizing images with an accuracy of more than 90%. This experiment also proves that the approach for vehicle object recognition on the UAV Dataset can be recognized on average more than 90%.

Figure 16. Approximately identical image comparison of detected objects for each model.

The five models are being used to determine their performance to detect cars, motorcycles, and trucks as their targeted objects. Each model is confronted with a test video on the same road. The model's job is to detect as many targeted objects while also being precise about them. Creating two published paper that are currently in the publication process.

Fig. 16 showed the comparison of each model to detected the model. Each image has two cars, one truck, and two motorcycles in it. In each frame, not every model could detect all vehicles that are on the road. Only CSResNext50-Panet-SPP and YOLOv7 were able detect all the vehicles. YOLOv3 and Yolov4 could not detect any motorcycles and DenseNet201-YOLOv3 can only detect one truck and one motorcycle. To determine how consistently these CNN models can detect vehicles on a road using UAVs, each CNN model needs to use the same test video sample.

Fig. 17 showed to detect the objects using a video, five parameters were shown on the top left of the video. These five parameters were: traffic present in the frame, total objects detected, total trucks detected, total cars detected, and total motorcycles detected. Fig. 19 informs the prediction on the test video using the seven model. YOLOv7 prediction managed to detect a total of 73 objects consisting of 37 cars, 34 motorcycles, and 2 trucks.

Figure 17. Cropped YOLOv7 prediction.

Figure 18. Cropped YOLOv4 prediction.

Fig. 18 showed the prediction of YOLOv4 on the test video. YOLOv4 detected a total of 64 vehicles. Yolov4 detects 9 fewer vehicles than YOLOv7, with a total of 42 cars, 17 motorcycles, and 5 trucks. YOLOv4 managed to detect a lot more cars and trucks but couldn't detect as many motorcycles as YOLOv7.

Figure 19. Cropped DenseNet201-YOLOv3 prediction.

Fig. 19 showed the prediction of DenseNet201 YOLOv3 on the test video. DenseNet201-Yolov3 detected a total of 102 vehicles with a total of 32 cars, 65 motorcycles, and five trucks. DenseNet201-YOLOv3 detected the most motorcycles out of all the CNN models, but it couldn't detect as many cars as other models.

Figure 20. Cropped CSResNext50-Panet-SPP prediction.

Fig. 20 showed the prediction on the video test used the CSResNext50-Panet-SPP model. CSResNext50-Panet-SPP detected a total of 84 vehicles. CSResNext50-Panet-SPP has the highest number of vehicles detected, with a total of 39 cars, 40 motorcycles, and 3 trucks. The number of motorcycles detected is the highest among all the CNN models.

Fig. 21 showed the prediction on the test video using the YOLOv3 model. YOLOv3 detected a total of 44 vehicles. YOLOv3 has the lowest number of objects detected with a total of 35 cars, 7 motorcycles, and 2 trucks. The number of motorcycles detected by YOLOv3 is fewer than 10.

Figure 21. Cropped YOLOv3 prediction.

TABLE V. TOTAL OBJECTS DETECTED IN THE LAST FRAME (VOLUME)

	YOLOv7	YOLOv4	DenseNet201-YOLOv3	CSResNext50-Panet-SPP	YOLOv3
Car (Mobil)	2	2	2	2	2
Motorcycle (Motor)	2	0	1	2	0
Truck (Truk)	1	1	0	1	1
SUM	5	3	3	5	3

All of the prediction images were taken from the last frame of the video. The result of the total objects detected in the last frame from each CNN model is shown in Table II. Besides the prediction data from the last frame of the test video, there is also data on the total objects detected in the last 12 seconds of the video.

Table V showed the objects totals detected in the last frame of the video. In Table V, CSResNext50-Panet-SPP and YOLOv7 had the most object detected out of all five CNN models. Both models managed to detect two cars, two motorcycles, and one truck. The rest of the models had the same total number of vehicles detected which is three vehicles. YOLOv4 and YOLOv3 could not detect any motorcycle, while DenseNet201-YOLOv3 couldn't detect a single truck.

TABLE VI. TOTAL OBJECTS DETECTED IN THE APPROXIMATE LAST 12 SECONDS

	Control Data	YOLOv7	YOLOv4	DenseNet201-YOLOv3	CSResNext50-Panet-SPP	YOLOv3
Car (Mobil)	5	37	42	35	39	35
Motorcycle (Motor)	4	34	17	30	40	7
Truck (Truk)	2	2	5	6	5	2
SUM	11	73	64	71	84	44

Table VI showed the data of the total objects that were detected in the last 12 seconds of the test video. Table VI shows that CSResNext50-Panet-SPP detects the most vehicles with a total of 84 vehicles which consists of 39 cars, 40 motorcycles, and five trucks. The lowest number of objects detected is the YOLOv3 with a total of 44 vehicles that consist of only 35 cars, seven motorcycles, and two trucks detected.

Each CNN model has a unique architecture, and they produce different metric values from each other. This difference will be a key component for comparing the four CNN models and determining which model is most suitable for detecting vehicle object classes. In Fig. 13, all CNN models depicted the average values of Precision, Recall, F1-Score, and IoU. The CSResNext50-Panet-SPP model got the highest average IoU value, followed by YOLOv4, densenet201-YOLOv3, and YOLOv3, thus affecting Precision, Recall, and F1-Score on each CNN model.

C. Evaluation

In the experiment, there were a total of five CNN models tested for their performance in object detection technology on the UAV dataset. Although all of the CNN models managed to detect more than 40 vehicles. The total number of vehicles in the video is only 11 vehicles which consist of five cars, four motorcycles, and two trucks. Table VI gave an insight into why the numbers are so much different compared to control data. It can be identified that the problem is caused by the movement of the camera and the size of the objects that need to be detected. The result of this bug would sometimes make the same object counted as a new object by the system. Another factor is that when using video from a UAV, the targeted objects tend to be very small and the camera moves a lot. This makes it hard for the system to keep track of the targeted objects. It can be seen from Table V that smaller objects are harder for the CNN model to detect. But the number of objects detected could be an indicator that the model has a good performance. If the models could detect a high number of vehicles, then the model can detect the objects within a frame better than another model.

V. CONCLUSION

In this study, the right approach is needed to optimize the ability to detect the three classes of vehicle objects depicted in the UAV dataset, such as a motorcycle class that is similar to a bicycle, a car class that is almost similar to several types of cars and a truck class that is almost the same for trailers and general trucks.

When detecting three vehicle class objects on the UAV dataset, this study utilized a deep learning algorithm with five CNN models and Darknet algorithms to support the training process. The experimental results can be concluded that the CSResNext50-Panet-SPP is the CNN model that can be used as a solution to recognize the three-vehicle class in a UAV dataset such as car, truck, and Motorcycle.

Based on the results of the experiments on the UAV dataset, it was illustrated that the CsResNext50-Panet-SPP model has produced Precision, recall, and F1 Scores with a percentage of up to 100% followed by an average IoU of more than 90%. Furthermore, the YOLOv4 model had an accuracy percentage is more than 98% with an average IoU of more than 85%. Lastly, YOLOv7 had the best potential when using a UAV dataset because of its low inference time of 44.98 ms and high mAp@0.50 accuracy percentage of 99.61%.

As stated before, the most optimal model used in object detection with the UAV dataset is the YOLOv7 CNN model. Because of the model's low inference time and high accuracy, YOLOv7 is potentially the best to be implemented in real-time object detection cases using a UAV video stream on a power constraint processor device. Another option was by using CSResNext50-Panet-SPP because of its high accuracy ability. But due to its high inference time, CSResNext50-Panet-SPP is only recommended when using high-power devices. For further research, a more complex clearer dataset and improved object tracking system could potentially increase the performance of this kind of research on the topic of real-time object detection performance, especially UAV.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Abdul Haris Rangkuti devised and supervised the research while providing suggestions and recommendations along the way include wrote the revision of papers. Varyl Hasbi Athala conducted the experiment, analyzed the resulting data, and wrote the paper with support from Farrel Haridhi Indallah. All authors approved the final version of this paper.

References

[1] P. Mittal, A. Sharma, and R. Singh, "Deep learning-based object detection in low-altitude UAV datasets: A survey," *Image and Vision Computing*, vol. 104, pp. 212–227, 2020.

[2] S. L. Madawalagama, N. Munasinghe, S. D. P. J. Dampegama, and L. Samarakoon, "Low cost aerial mapping with consumer-grade drones," in *Proc. 37th Asian Conference on Remote SensingAt: Colombo*, Sri Lanka, 2016, vol. I, pp. 1–8.

[3] C. Martin, S. Parkes, Q. Zhang, X. Zhang, M. F. McCabe, and C. M. Duarte, "Use of unmanned aerial vehicles for efficient beach litter monitoring," *Marine Pollution Bulletin*, vol. 131, pp. 662–673, 2018.

[4] G. V. Konoplich, E. O. Putin, and A. A. Filchenkov, "Application of deep learning to the problem of vehicle detection in UAV images," in *Proc. 19th International Conference on Soft Computing and Measurement*s, pp. 4–6, 2016.

[5] I. Colomina and P. Molina, "Unmanned aerial systems for photogrammetry and remote sensing: A review," *ISPRS J. Photogramm. Remote Sens*., vol. 92, pp. 79–97, 2014.

[6] W. S. Udin and A. D. Ahmad, "Assessment of photogrammetric mapping accuracy based on variation flying altitude using unmanned aerial vehicle," in *Proc. IOP Conference Series: Earth and Environmental Science*, 2014, vol. 18, no. 12027.

[7] B. Dipert. Vision processing opportunities in drones. [Online] Available: https://www.embedded-vision.com/platinum-members/embedded-vision-alliance/embedded-visiiontraining/documents/pages/drones

[8] A. R. Pathak, M. Pandey, and S. Rautaray, "Application of deep learning for object detection," *Procedia Comput. Sci.*, vol. 132, pp. 1706–1708, 2018.

[9] K. Mok. Deep learning drone detects fights, bombs, shootings in crowds. [Online]. Available: https://thenewstack.io/deep-learning-drone-detects-Fights-bombsshootings-in-crowds/

[10] Detection and counting of arabian oryx from aerial image. [Online]. Available: https://blogs.flytbase.com/ai-drones/

[11] R. O. Prakash and C. Saravanan, "Autonomous robust helipad detection algorithm using computer vision," in *Proc. Int. Conf. Electr. Electron. Optim. Tech. ICEEOT 2016*, 2016, pp. 2599–2604.

[12] M. Tzelepi and A. Tefas, "Human crowd detection for drone flight safety using convolutional neural networks," in *Proc. 25th European 2017 Signal Processing Conference (EUSIPCO*, 2017, pp. 743–747.

[13] A. Nazerdeylami, B. Majidi, and A. Movaghar, "Autonomous litter surveying and human activity monitoring for governance intelligence in coastal eco-cyber-physical systems," *Ocean and Coastal Management*, vol. 200, no. 105, 2021.

[14] Y. Lin, M. Wang, W. Chen, W. Gao, L. Li, and Y. Liu, "Multiple object tracking of drone videos by a temporal-association network with separated-tasks structure," *Remote Sensing*, vol. 14, no. 16, pp. 3862–2874, 2022.

[15] A.Wahyudin and Z. Husin, "Sistem deteksi objek helipad pada pergerakan autonomous fixed wing drone berbasis algoritma YOLO," Doctoral dissertation, Sriwijaya University, 2020.

[16] M. R. Prasanta, M. Y. Pranata, M. A. Firnanda, and S. Sendari, "Rancang bangun quadcopter drone untuk deteksi api menggunakan YOLOv4," *Cyclotron*, vol. 5, no. 1, 2022.

[17] Y. Tian, Y. Wang, R. Song, and H. Song, "Accurate vehicle detection and counting algorithm for traffic data collection," in *Proc. International Conference on Connected Vehicles and Expo (ICCVE)*, 2015, pp. 285–290.

[18] S. Rathinam *et al.*, "Autonomous searching and tracking of a river using an UAV," in *Proc. 2007 American Control Conference 2007*, 2017, pp. 359–364.

[19] P. M. Olsson, J. Kvarnström, P. Doherty, O. Burdakov, and K. Holmberg, "Generating UAV communication networks for monitoring and surveillance," in *Proc. 11th International Conference on Control Automation Robotics & Vision*, 2010, pp. 1070–1077.

[20] S. M. Adams and C. J. Friedland, "A survey of unmanned aerial vehicle (UAV) usage for imagery collection in disaster research and management," in *Proc. 9th International Workshop on Remote Sensing for Disaster Response*, vol. 8, 2011, pp. 1–8.

[21] E. A. George, G. Tiwari, R. N. Yadav, E. Peters, and S. Sadana, "UAV systems for parameter identification in agriculture," in *Proc. IEEE Global Humanitarian Technology Conference: South Asia Satellite (GHTC-SAS) 2013*, 2013, pp. 270–273.

[22] B. Xu, X. Xu, and C. M. Own, "On the feature detection of nonconforming objects with automated drone surveillance," in *Proc. 3rd International Conference on Communication and Information Processing*, 2017, pp. 484–489.

[23] S. Campbell, W. Naeem, and G. W. Irwin, "A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres," *Annu. Rev. Control*., vol. 36, no. 2, pp. 267–283, 2012.

[24] J. Zhou, C. M. Vong, Q. Liu, and Z. Wang, "Scale adaptive image cropping for UAV object detection," *Neurocomputing*, vol. 366, pp. 305–313, 2019.

[25] Y. Cao *et al.*, "VisDrone-DET2021: The vision meets drone object detection challenge results," in *Porc. 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, Montreal, BC, Canada, no. 35, 2021, pp. 2847–2854.

[26] S. Sun and C. Salvaggio, "Aerial 3D building detection and modeling from airborne LiDAR point clouds," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens*., vol. 6, no. 3, pp. 1440–1449, 2013.

[27] Z. Li, W. Shi, P. Lu, L. Yan, Q. Wang, and Z. Miao, "Landslide mapping from aerial photographs using change detection-based Markov random field, Remote Sensing," *Environ*, vol. 187, pp. 76–90, 2016.

[28] Esri. How the Compute Accuracy For Object Detection tool works. ArcGIS Pro, Esri. [Online]. Available: https://pro.arcgis.com/en/pro-app/latest/tool-reference/image-analyst/how-compute-accuracy-for-object-detection-works.htm

[29] A. F. Gad, Accuracy, precision, and recall in deep learning. [Online]. Available: https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/