IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 November 2023, accepted 19 November 2023, date of publication 21 November 2023,
date of current version 5 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3335826

APPLIED RESEARCH

Object Detection in Dense and Mixed Traffic for Autonomous Vehicles With Modified Yolo

ARI WIBOWO[1], BAMBANG RIYANTO TRILAKSONO[1,2], (Member, IEEE), EGI MUHAMMAD IDRIS HIDAYAT[1], AND RINALDI MUNIR[1]

[1]School of Electrical Engineering and Informatics, Institut Teknologi Bandung, Bandung 40132, Indonesia
[2]University Center of Excellence—Artificial Intelligence on Vision, NLP and Big Data Analytics (U-CoE AI-VLB), Bandung 40132, Indonesia

Corresponding author: Bambang Riyanto Trilaksono (bambang.riyanto@itb.ac.id)

This work was supported in part by the Ministry of Finance of the Republic of Indonesia under the Lembaga Pengelola Dana Pendidikan Riset Inovatif Produktif (LPDP RISPRO) Research Funding and the Indonesian Ministry of Education and Culture under the World Class Research (WCR) Program.

ABSTRACT Autonomous vehicles rely on the accurate detection and recognition of objects in their surroundings, a critical requirement for safe operation, especially in congested traffic with diverse vehicle types. This study presents a novel dataset collected in various road conditions in Indonesia, and it focuses on the detection and classification of visual objects around autonomous vehicles. Object recognition is achieved through the use of YOLOv7-based deep learning, adapted to identify small, faint, and partially concealed objects. Key enhancements include the integration of a deformable layer and the transition from Non-Maximum Suppression (NMS) to Soft Non-Maximum Suppression (softNMS). The dataset comprises eight predefined custom classes commonly encountered in Indonesian traffic. We collected video data recordings of heavy traffic scenarios featuring a wide range of vehicle types as training and testing data. The object detection model is fine-tuned through transfer learning, with multiple learning configurations explored for comparison. Experimental results demonstrate that deep learning models trained with transfer learning outperform those trained from scratch. Specifically, the modified YOLOv7, referred to as YOLOv7-MOD, incorporates a deformable convolution layer for up-sampling, leading to a remarkable performance of 94.68% in Recall, 96.87% in Precision, and 95.76% in F1-score. The modification resulted in an additional performance increase of 1.05% on average compared to the original model. The findings indicate that YOLOv7-MOD enhances the precision of object detection and recognition compared to the original YOLOv7, making it a promising solution for autonomous vehicle perception systems.

INDEX TERMS Object detection, deformable convolution, autonomous vehicle, deep learning.

I. INTRODUCTION

In the last few years, self-driving car has been started up by industry and the research community. It is designed to assist the human life. Some of them, used for industrial purpose, and the other are used for daily needed. There are several levels in autonomous vehicle, it starts from partially manual control, until automatically drive itself [1]. The most important ability that vehicle needed is visual understanding in environment information. Detecting and recognition object surrounding the car is related in perception system on autonomous vehicle.

The associate editor coordinating the review of this manuscript and approving it for publication was Yongjie Li.

In developed countries, most traffic is filled by cars. However, in developing countries, such as Indonesia, traffic is more congested and filled with various types of vehicles, most of which are motorbikes. In addition, Indonesia has many types of roads, such as one-way, two-way, and local. Each type has a different object diversity and density. Therefore, to avoid accidents under any road condition, autonomous vehicles need to know their surroundings by detecting and recognizing all objects in their surroundings, especially in crowded areas.

Perception system in autonomous vehicle consists of object detection and object recognition. Object detection aims to understand location and the class of the object on

the frame. Furthermore, deep learning is widely used in various applications such as detecting and recognizing visual objects. Different from original machine learning which mostly use simple neural network, most of deep learning use convolutional neural network (CNN). In traditional machine learning, feature extraction from images often requires in-depth domain knowledge and a manual approach to design relevant features. This can be a complex and time-consuming task, and the results are not always as good as those provided by CNNs. There are many CNN models exists, but YOLO is one of the most efficient deep learning models that have both great precision and speed. This research will use deep learning model based on YOLO for object detection.

Finally, training deep-learning models with datasets does not always require large datasets. In addition, manual labeling is costly. Transfer learning allows us to use knowledge from well-known datasets such as the COCO datasets [4], [5]. Transfer learning aims to solve the problem of learning different objects by extracting useful information from the data of the associated object and transferring it for other purposes [4]. Some information from an existing dataset may be similar to that from a new one. For example, pedestrians in an existing dataset have features similar to those of the new dataset. To enrich the features of the model built for mixed traffic environments, we collected data on the roads in the city of Bandung, Indonesia.

YOLO is built on top of the darknet framework. However, several deep learning frameworks are available in addition to darknet. Tensorflow with keras was used as the deep-learning framework. Tensorflow can also leverage the GPU to accelerate the deep-learning computation process. YOLO predicts multiple bounding boxes for each grid cell. During training, we only want one bounding box predictor to be responsible for each object. We assign one predictor to be ''responsible'' for predicting an object, based on which prediction has the highest current IOU with the ground truth. This leads to specialization between bounding box predictors. Each predictor improves the prediction of specific sizes, aspect ratios, or classes of objects, thereby improving the overall recall. This study used a modified YOLO-based deep learning model for object detection. In this paper, we propose a modified YOLOv7 for object detection in dense and mixed traffic for autonomous vehicles. We also modified YOLOv3 and YOLOv5 as a baseline. The YOLOv7 can detect objects quickly because YOLO built-in on the top darknet. However, there are some cases where a small object can not be detected. Meanwhile, the YOLO can be modified with an additional deformable convolutional layer and the use of softNMS to improve algorithm performance. The proposed algorithm will modify the YOLO-based deep learning model for object detection. This is important to overcome the problem of detecting small objects and overcomes the detection of objects that look faint visually. As a result, it can increase precision, recall, and F1 score. The proposed model is referred to as YOLOv7-MOD. The main contributions of this paper are as follows:

(1) We designed a deep learning network for object detection (YOLOv7-MOD), where the constructed model is capable of detecting small-sized and faintly visible objects.

(2) We compiled a local dataset for object detection in low-light conditions collected on the streets of Bandung city in the late evening, representing typical dense and mixed traffic in developing countries.

(3) We demonstrated the performance of the designed network architecture through various experiments. The experimental results show that our network outperforms existing methods on synthetic images both quantitatively and qualitatively.

The remainder of this paper is organized as follows. Related work and an overview of object detection methods is presented in Section II. Our main contribution, YOLOv7-MOD to deal with small objects and objects that appear blurry. presented in Section III. Section IV is the generation of traffic datasets in urban areas and the experimental results are presented and analyzed in Section V. The summary and conclusion of the present work are drawn in Section VI.

II. PRELIMINARIES

Numerous techniques are available for object detection, one of which is the classical HOG algorithm that enjoyed widespread use prior to the era of deep learning. HOG finds application in computer vision and image processing for object detection. This approach involves tallying occurrences of gradient orientations within localized image regions. It shares similarities with edge orientation histograms and scale-invariant feature transformation (SIFT). The HOG descriptor emphasizes the structural and shape characteristics of objects, leading to object detection within frames by assessing gradients across each frame segment. However, it's important to note that this technique is less suitable for deployment in autonomous vehicles. A study in [3] underscores that HOG's performance lags behind deep learning methods primarily due to issues related to accuracy.

A CNN is a deep learning algorithm for detecting an object from an image [6]. Instead of depending on the hand working mathematically, the CNN algorithm is based on a neural network. Currently there are a number of CNN-based models commonly used for object detection and classification, e.g., ResNet, VGGNet, and Inception. CNN typically detects and classifies only one object per image [7]. Consequently, advancements in CNN have led to the development of more sophisticated features. Various techniques have been employed to enhance CNN's ability to detect and classify multiple objects within a single image. One of the earliest methods for achieving this was RCNN, which employed a two-stage detection approach to identify multiple objects within a single frame [3]. RCNN utilizes region proposals to detect objects, systematically scanning from the top-left to the bottom-right corner of an image. The CNN is applied within each of these proposal regions. However, RCNN's

drawback is its time-consuming nature, as it employs a Region Proposal Network to segment the image, resulting in approximately 2000 partitions per-image [8]. In response to this speed limitation, Fast R-CNN and Faster R-CNN were developed, offering incremental improvements and significantly outperforming the original RCNN [4]. Fast R-CNN streamlines the process by first applying CNN to the original image and then generating region proposals. Faster R-CNN follows a similar order to Fast R-CNN but employs a neural network to determine the region of the proposal. Notably, RCNN takes approximately 50 seconds to process an image, whereas Fast RCNN processes an image in just two seconds, and Faster RCNN can do so in a mere 0.2 seconds [5]. These neural network models are referred to as two-stage detectors because they initially search for objects within an image using region proposals before assigning their respective classes.

Even though Faster R-CNN features faster inference time compared to R-CNN and Fast R-CNN, it is not fast enough to be implemented in an autonomous vehicle due to the rapid dynamics of surrounding objects movement. Based on the generated inference rate, YOLO produces a speed that can be used for object detection in autonomous vehicles [3]. Faster object detection is currently known as a one-stage detector. YOLO and SSD are two of many one-stage detectors [9]. YOLO can process the image in one image without sliding the region of the proposal. YOLO performs object detection by predicting coordinates relative to the location of a grid cell [10]. YOLO has several versions with better accuracy. YOLO extracts feature at different scales and use anchor boxes to predict bounding boxes [11]. By doing this, YOLO can identify objects of various sizes and appearances, making it versatile and effective in detecting objects under different conditions. Anchor boxes are predefined boxes of various shapes and sizes that serve as reference points during the object detection process. YOLO uses these anchor boxes to help predict the position and size of objects within an image. By comparing the characteristics of detected objects to the anchor boxes can make accurate predictions about the location and dimensions of those objects. Bounding boxes are rectangles that enclose objects within an image. YOLO's main goal is to predict these bounding boxes for each detected object. By using the features extracted at different scales and anchor boxes as references can accurately determine the position and dimensions of objects in the image, creating bounding boxes that tightly enclose the detected objects. In this paper, we employ several performance evaluation metrics, including precision, recall, F1-score, and average precision, as referenced in [6].

TABLE 1. Advantages and disadvantages method.

Method	Research	Advantages	Disadvantages
R-CNN	Rich feature hierarchies for accurate object detection and semantic segmentation (Girshick et al., 2014)	High accuracy	Computationally expensive
Fast R-CNN	Fast R-CNN (Girshick, 2015)	Faster than R-CNN	Still computationally expensive
Faster R-CNN	Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks (Ren et al., 2016)	Fast and accurate	Requires region proposal generation
YOLO	You Only Look Once: Unified, Real-Time Object Detection (Redmon et al., 2016)	Fast and able to handle multi-scale object detection in real-time	Lower accuracy compared to R-CNN and Fast R-CNN
RetinaNet	Focal Loss for Dense Object Detection (Lin et al., 2017)	High accuracy	Computationally expensive
YOLOv3	YOLOv3: An Incremental Improvement (Redmon and Farhadi, 2018)	Fast and accurate	Lower accuracy compared to RetinaNet
YOLOv5	YOLOv5: Improved Real-Time Object Detection with One-Stage Object Detector	Fast, accurate, and real time	Limited in detecting heavily occluded objects
YOLOv7	YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors	Recognize a broader range of object categories or have the ability to detect objects with better accuracy	More complex architecture, it might demand greater resource investments

III. MODIFIED YOLO MODEL

A. YOLOv3-MOD MODEL

In YOLOv3, the inserted layers play a crucial role in enhancing the object detection capabilities of the model. One of the types of layers used is the pooling layer, which is commonly employed to reduce the feature dimensions within the network. Pooling layers assist in extracting essential features and reducing computational load. These layers can

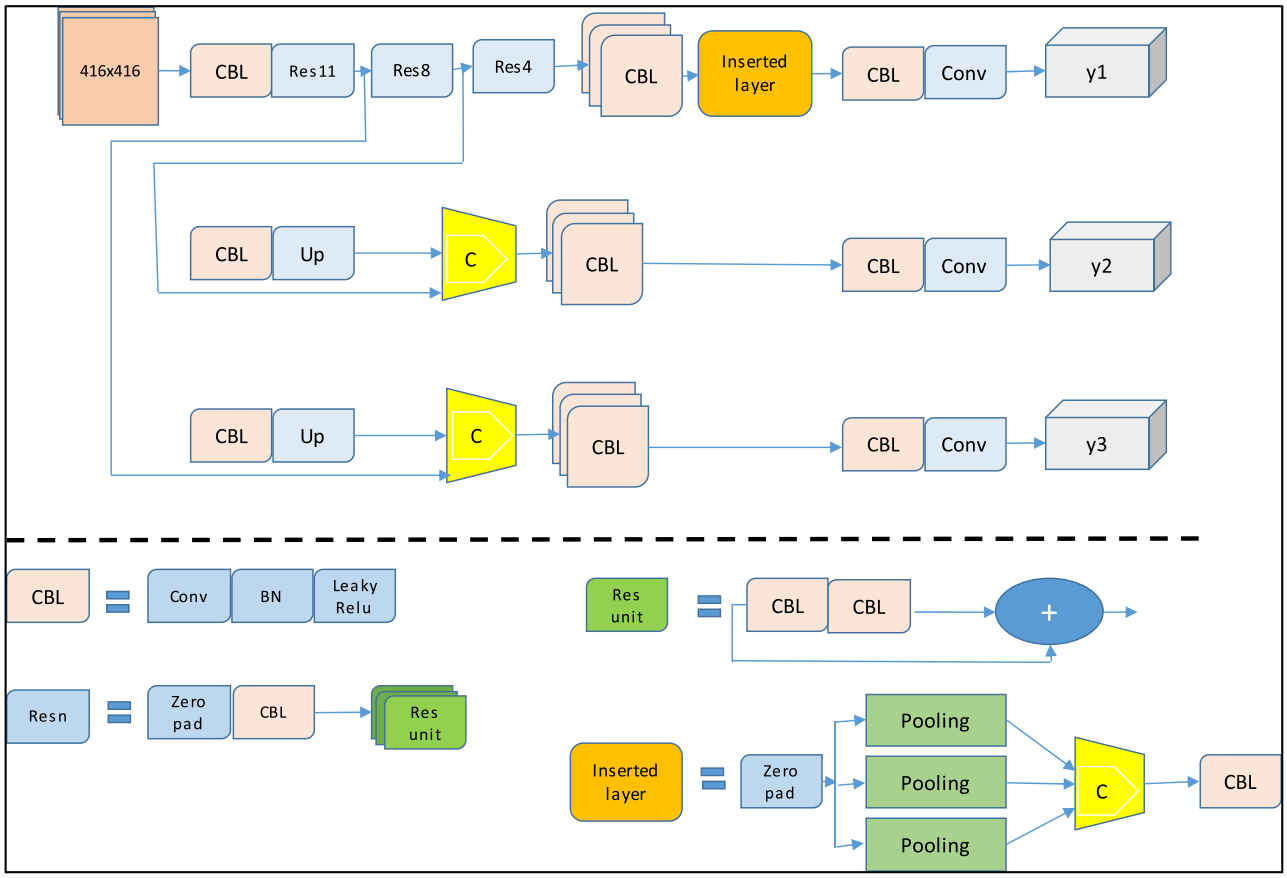

FIGURE 1. YOLOv3-MOD architecture.

shrink spatial resolution while retaining critical information. Zero padding is a technique used to fill the area around the edges of an image with zeros. It helps in preserving information along the image's edges and prevents the loss of vital data. Often, zero padding layers are used before convolutional layers to maintain consistent output sizes. Additionally, concatenation is a merging operation used to combine outputs from multiple layers at a single point within the network. This allows the model to fuse information from various levels of feature extraction, which can enhance object detection capabilities in complex environments. Hence, by utilizing inserted layers such as pooling, zero padding, and concatenation in YOLOv3, the model can obtain stronger feature representations, preserve edge information, and integrate features from different extraction levels. This contributes to improving the model's performance in object detection tasks within diverse environments.

There have been several efforts made to improve the performance of YOLOv3, some of which are described in [14]. Similarly, in this paper, we have developed YOLOv3-MOD, YOLOv5-MOD, and YOLOv7-MOD. The network on YOLOv3-MOD was slightly improved compared with the basic YOLOv3. Compared to other detection algorithms, YOLOv3 can run quite quickly without significantly sacrificing detection accuracy. YOLOv3 uses darknet53 as the backbone to perform feature extractors [15]. It contained 53 convolutional layers. Batch normalization is applied to stabilize training, speed up convergence, and regularize the model [10]. The last layer uses a linear activation function, whereas the other layers use leaky rectification.

Model development on YOLOv3-MOD is shown in Figure 1, where an additional layer consisting of zero padding, 3 pooling, and concatenning is inserted. An additional layer was added after the backbone feature extraction layer. The purpose of the incremental layer is to unify and combine the multiscale local region features, which leads to increased robustness. Hence, the network becomes more robust to various object deformations and spatial layouts [10].

The additional layer contains a few max-pooling layers. The filters of these pooling layers have sizes of 5×5, 9×9, and 13×13, and have one stride. All these max-pooling layers have the same size because of the padding operations. The outputs of the max pooling become the input of the next convolutional layer. Furthermore, combining different feature sizes allows the network to exploit more spatial data in the convolutional layer [10].

Similar to YOLOv3, YOLOv3-MOD predicts the bounding boxes at three different levels. The features extracted

from the backbone and obtained through the additional layer are extracted by these three levels using a concept similar to that of feature pyramid networks [10]. There are three steps in the up-sampling feature. Each step of the up-sampled features is coupled with features extracted from the backbone, of the same size, allowing better detection of objects with different sizes. In the same manner, YOLOv5-MOD was developed following a similar approach as the development of YOLOv3-MOD.

B. MODIFIED YOLOv7

1) DEFORMABLE CONVOLUTIONAL LAYER

Deformable Convolution (DC) is a variant of the Convolutional Neural Network (CNN) which allows the convolution to deform the convolutional kernel to adapt the kernel position to the features in the image. In the context of feature extraction, DC is useful for capturing more complex and differently shaped features in images. Convolutional kernels usually have a fixed shape, such as a box or circle, and cannot model changes in the shape of the observed object. DC can adjust its position and shape according to the distribution of features in the image so that it can capture features that are more complex and have different shapes.

The deformable convolution adds a spatial offset to each point in the receptive field of the convolution operation. The offset is learned from the feature map input by the convolution layer through a parallel network. The receptive field of the convolution operation after offset can be consistent with the actual shape of the object. Even if the target is deformed, the convolution receptive field region can still focus on the target. The Deformable Convolutional Layer introduces a level of flexibility and adaptability to the standard convolutional layers commonly found in deep learning networks. Unlike traditional convolutional layers that employ fixed and rigid kernels for feature extraction, the deformable convolutional layer allows the network to adjust its convolutional kernels or filters based on the characteristics of the objects in the image.

This adaptability is particularly valuable when dealing with objects that may have varying shapes, sizes, or positions within the image. In scenarios such as autonomous driving, where the vehicle encounters diverse traffic conditions and objects, the deformable convolutional layer can precisely capture and analyze the details of these objects by flexibly adjusting its convolutional kernels. For example, if the system detects an object with a unique shape or orientation, the deformable convolutional layer can deform its kernels to align with the object's characteristics, improving the feature extraction process. This adaptability ensures that the model can accurately identify and classify objects under challenging conditions, ultimately enhancing the reliability and safety of the autonomous vehicle's perception system. In essence, the Deformable Convolutional Layer in Figure 2 represents an innovative feature that empowers the deep learning model to adapt and better understand the complex and dynamic visual information captured by the vehicle's cameras. It plays a crucial role in improving the model object detection and recognition capabilities, making it a valuable component in the quest for safer and more reliable autonomous driving systems.

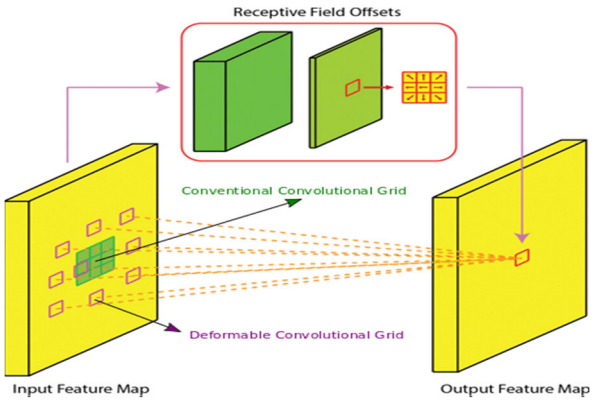

FIGURE 2. Deformable convolutional layer [27].

The comparison of sampling points of conventional convolution and deformable convolution network is shown in Figure 3.

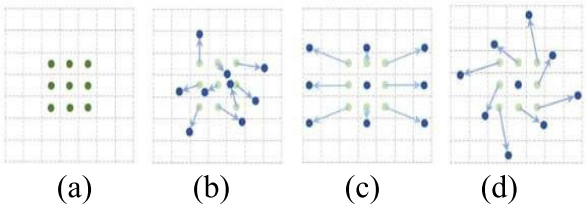

FIGURE 3. Comparison of sampling points between conventional convolution and deformable convolution [27].

(a) is the traditional 3 × 3 convolution kernel, where the region containing green points is the receptive field of a convolution operation, and you can see that the receptive field is a fixed 3 × 3 rectangle. (b), (c) and (d) are deformable convolutions. The blue points are the input pixels of the migrated convolution kernel. On the basis of the original convolution, a learning offset is added to each sampling point of the convolution kernel, so that the convolution kernel is irregular in shape, increasing the receptive field and adapting the convolution kernel to different morphological changes and scale changes. Assuming that there is a 3 × 3 convolution kernel R={(−1, −1), (−1, 0), · · · ,(0,1), (1,1)} with nine sets of offsets, and two values in each set of offsets represent the corresponding horizontal and vertical offsets, then R represents a standard 3 × 3 convolution kernel. For the input feature graph x and the output feature graph y, the traditional convolution calculation process is as follows: for the value of each position p_0 on the output feature graph y, it is calculated by the formula:

$$y\left(p_0\right) = \sum w\left(p_{npn} \in R\right) .x(p_0 + p_n) \tag{1}$$

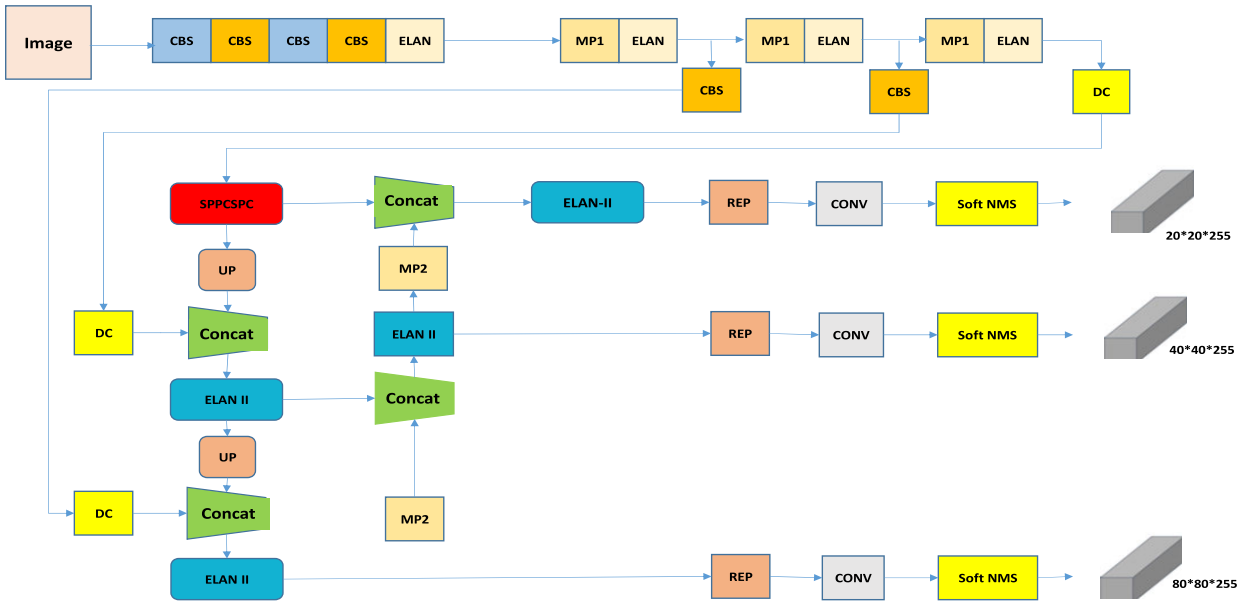

FIGURE 4. YOLOv7-MOD architecture.

p_n is an enumeration of all positions in R, and $w(p_n)$ represents the weight of convolution kernel. The formula represents a convolution operation in image processing, where it's used to process one signal (x) with another signal (w) to produce an output signal (y). Here's an explanation:

$y\,(p_0)$: This is the value of the output signal at position p0.

$\sum$: This symbol indicates that we are summing up values over a range of positions.

$w\,(p_n)$: This is the weight or kernel used in the convolution operation. It represents the filter or mask that is applied to the input signal x.

$x(p_0+p_n)$: This is the input signal x shifted by the position pn and evaluated at the position p0 + pn. In other words, it's the value of the input signal at a specific position determined by adding p0 and pn.

pn is an enumeration of all positions in R: This means that pn takes on different values corresponding to all possible positions within the range R. R represents the set of all positions where the convolution is applied.

The formula calculates the output signal y at position p0 by aggregating the weighted contributions from the input signal x, originating from various positions determined by adding p0 to different values of pn. These weights, denoted as w(pn), govern the impact of each input signal value on the resulting output, and the summation encompasses all potential positions within the specified range R. This operation finds widespread application in diverse signal processing and image processing tasks, including but not limited to edge detection, filtering, and feature extraction.

In reality, the formulas used not only appear at different scales but are also located in various regions within a single image, making it challenging to capture all the features during training. To address this issue, [23] proposed the use of a deformable convolutional network in the one-stage object detection network YOLOv5 to enhance the modeling capability of the geometric transformation of detected objects. Furthermore, [24] the deformable convolution layers can effectively detect key features with arbitrary scales, aspect ratios, and rotation angles. However, deformable convolution requires intensive memory usage due to the explicit offset generation for each location in the feature map [25]. Therefore, [26] simply added deformable convolutional layers to the network after the Convolution Batch SiLU (CBS) layers in the backbone of our detection model to assist in multi-scale feature extraction.

2) YOLOv7-MOD MODEL

YOLOv7, as a key member of the YOLO model family, is renowned for its excellence in object detection tasks. This model harnesses the robust CSPDarknet53 network, previously utilized in YOLOv4. The CSPDarknet53 is meticulously crafted to capture features across various scales. This remarkable architecture comprises a sequence of convolution layers interconnected with shortcut connections and inter-stage connections. In this study, we further developed the YOLOv7 architecture by introducing the deformable convolution layer, as depicted in Figure 4, which we refer to as YOLOv7-MOD. Deformable convolution serves as the cornerstone in the architecture of YOLOv7-MOD that we have crafted. In contrast to traditional convolution, which rigidly employs a fixed grid for point sampling, deformable convolution endows the network with the remarkable capability to dynamically adjust the sampling grid to the unique characteristics of the input data. This adaptability has proven to be highly advantageous, particularly in the detection of partially occluded objects or objects with irregular shapes.

In YOLOv7-MOD, the Deformable Convolution (DC) layer is strategically positioned within the network architecture. The DC layer is placed at multiple network levels, including mid-level to near the end, allowing the network to dynamically adjust the sampling grid based on the characteristics of detected objects in the image. This enables the network to better capture the details of objects that may have irregular shapes or be partially occluded. The careful and multi-level placement of the DC layer provides additional flexibility in the object feature extraction process, resulting in a significant improvement in object detection and recognition capabilities in various complex and dynamic environmental scenarios. The DC layer is a key component that sets YOLOv7-MOD apart and enables it to be more adaptive in handling a wide range of object detection tasks.

Positioning DC after the backbone facilitates additional feature extraction prior to the computation of the final object positions. This strategically addresses potential early interference during the feature extraction stage, ultimately enhancing detection accuracy. DC empowers the model to identify objects that could be partially obscured by other objects, contributing to improved object detection performance. Integrating DC after the backbone equips the model with the capability to more effectively handle such scenarios, even when these objects are not distinctly visible to conventional convolutional layers.

IV. DATASET AND TRAINING

A. DATASET

In many fields, datasets that provide sufficient number of samples for specific problems have proven to be important catalyst for rapid improvement of solutions. They facilitate fast iteration of algorithms based on quantitative evaluation of their performance, expose potential weakness, and enable fair comparison. In computer vision, there are always datasets for individual fundamental problems, such as image classification [17]. Instead of using an existing dataset from the Internet, such as [18] and [19], the dataset is adopted based on Indonesian traffic conditions, which have dense and mixed traffic. The video was captured in Bandung between two areas, around Institute Technology of Bandung (ITB) and along the way from Cileunyi to ITB. Several objects appeared frequently while the data were collected, i.e., motorcyclists, pedestrians, cars, buses, trucks, mini pick up, mini truck boxes, and mini busses. The number of datasets for each class is not the same. Motorcyclists were the most frequent objects appearing in videos, followed by cars. The rarest objects in the dataset were mini truck boxes and mini pickups.

The setting of the device was set to be as close as the Berkeley deep drive dataset setting [19]. The detected object was also adjusted, although the dataset geography, environment, and weather were not as diverse as those of the Berkeley dataset. The dataset was collected in Bandung City during fine weather. Similar to the Berkeley dataset, a single camera was placed on the car's dashboard. The camera used was the FLIR blackfly camera with a resolution of 1.3 Megapixels. The labels were placed in two ways. For object detection, bounding box annotation was provided for each image. Each bounding box was labeled with a specific class. The dataset contained eight class categories. The labels for object detection and tracking were annotated using a labeling tool [20]. One of the processes for labeling and creating datasets is shown in Figure 5.

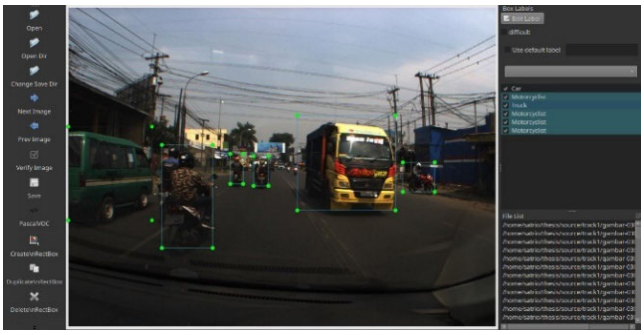

FIGURE 5. Labeling images for object detection and classification.

Images are labeled in the VOC format for models other than YOLO. The total image annotated for the object detection dataset comprised 15717 images. The annotation XML files are converted to keras format files that contain directory images, bounding boxes, and classes for detection that can utilize this dataset. This text file can then be used to train the YOLO model.

B. TRAINING

Several techniques have been tested for training multiple versions of YOLO and their modifications. The difference between the original YOLO and the modified YOLO lies in the number of layers used. In the modified YOLO, additional layers are used to improve accuracy. These models are trained in two ways: training without pre-trained weights; and training with pre-trained weights. All methods used 200 epochs and 16 batches with different training separations. For the first category, the model without pre-trained weights was trained, and all layers were unfrozen for 200 steps. Two methods were utilized for the following categories: (1) transfer learning with a frozen backbone and (2) transfer learning with fine-tuning. Darknet53 pre-trained weights from [15] were applied as our backbone layer weights for these two models.

In the training process binary cross-entropy is applied as the loss function. Another loss function, Focal Loss, handles dense object detection. It was introduced [21] to handle dense object detection. However, based on the experiment conducted in [22], the performance decreased significantly when the focal loss was implemented. Therefore, the focal loss is unsuitable for YOLO because it only has a single-stage detector. This is more suitable for using a two-stage detector, such as the RCNN family.

V. EXPERIMENT

The object detection dataset comprises 15,717 images categorized into eight classes, including motorcyclists, pedestrians, cars, trucks, buses, mini truck boxes, mini pickups, and mini busses. To facilitate model training, the dataset was thoughtfully divided into training, validation, and testing subsets, with 12,719 images allocated for training, 1,413 for validation, and 1,585 for testing. A variety of learning methods were explored to train both the YOLOv7 and YOLOv7-MOD models, with the goal of identifying the most effective training approach that balances precision, dataset compatibility, and epoch optimization. Once the models were adequately trained on the dataset, they underwent rigorous testing using the designated test dataset, applying an IOU threshold of 0.5 and a confidence score of 0.1 to assess their performance.

The test scenario involved four distinct categories of objects: evident objects, small objects, ambiguous objects, and occluded objects. Evident objects were characterized by medium-sized, clearly visible images. Small objects encompassed instances where objects appeared either small or were situated at a considerable distance from the camera. Ambiguous objects exhibited reduced visibility due to factors like high noise or inadequate lighting conditions. Lastly, occluded objects denoted situations in which the object to be detected was partially concealed by other objects. The test aimed to evaluate and compare the performance of the modified YOLO method against the original YOLO method. To assess performance, metrics such as F1 scores and area under the precision-recall curve were employed. The results of these tests are presented in Figures 6 to 13 for detailed analysis.

Figures 6 to 9 depict the performance measurements using the F1 Score for the YOLOv3, YOLOv5, YOLOv7, and YOLOv7-MOD models, respectively. Notably, when comparing the measurements of the modified YOLOv7 model, as illustrated in Figure 10, it becomes evident that the addition of layers results in higher F1 scores. Furthermore, in Figure 9, where the modified YOLOv7 model incorporates a deformable layer, superior measurement results are observed compared to the original YOLO model. This enhancement can be attributed to the specific layers within the YOLOv7-Mod model designed to extract and handle subtle and small objects.

These F1 Score results are consistent with precision-recall measurements, revealing that the inclusion of additional layers leads to an increase in precision-recall values, as indicated in Figure 13. The performance measurements for the original YOLOv7 are presented in Figure 13. The precision-recall visualization of detection and classification results for the four methods—YOLOv3, YOLOv5, YOLOv7, and YOLOv7-MOD can be seen in Figures 10 to 13. These figures illustrate instances where the original YOLO model fails to detect certain objects, while YOLOv7-MOD successfully identifies them. Detection results are provided in the form of bounding boxes and object class names. It's important to note that YOLOv7-MOD faces challenges in extracting frames with small and faint images, as compared to the original YOLO model.

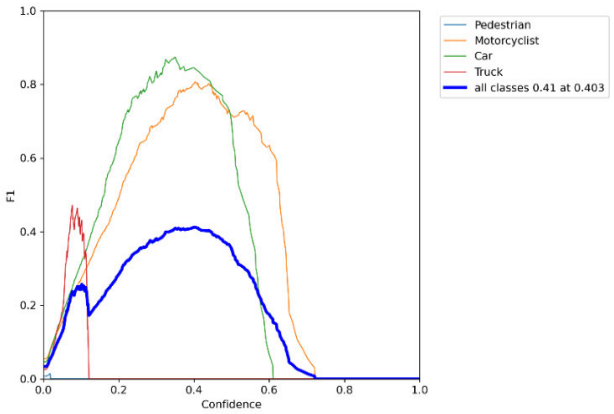

FIGURE 6. F1 score of YOLOv3.

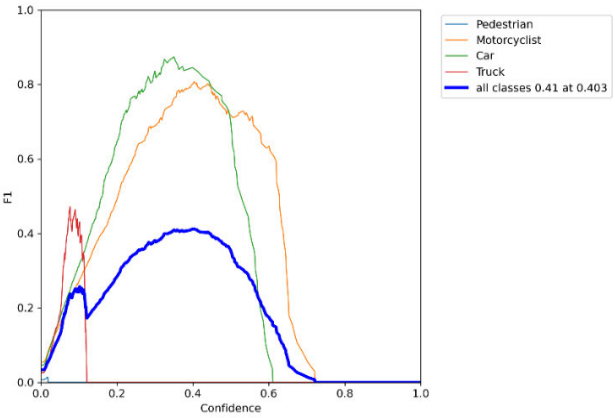

FIGURE 7. F1 score of YOLOv5.

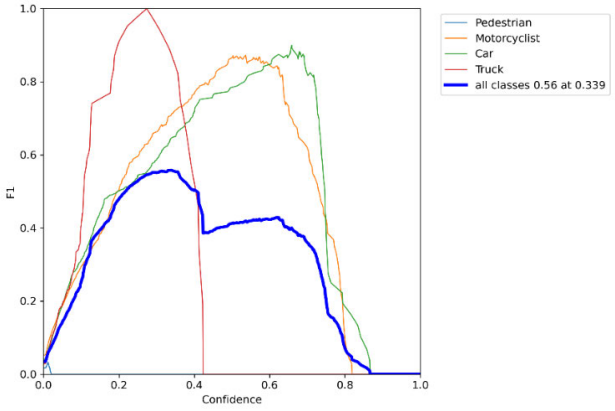

FIGURE 8. F1 score of YOLOv7.

Subsequently, the precision-recall curves are calculated to produce a single metric for object detection. This single metric can be used to compare the best object detection models. The calculation was based on the VOC challenge

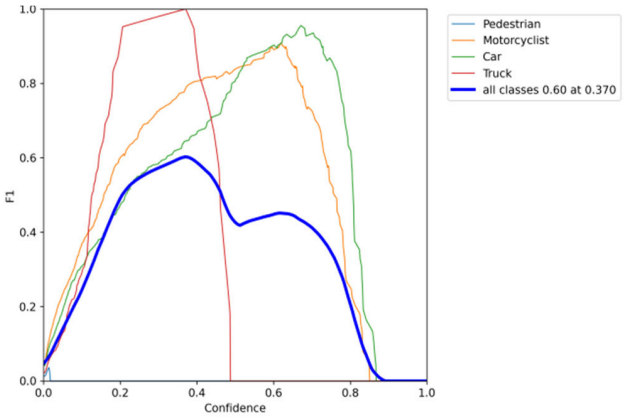

FIGURE 9. F1 score of YOLOv7-MOD.

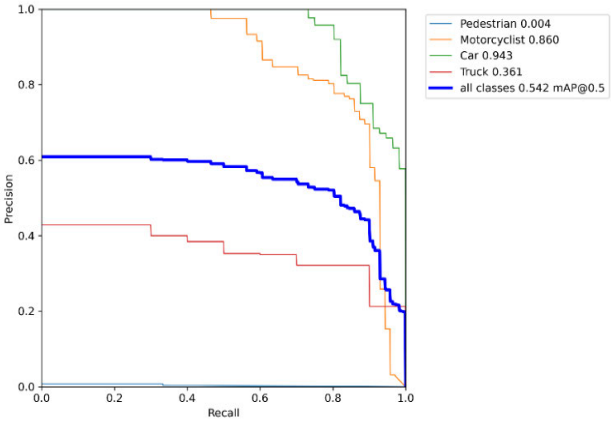

FIGURE 10. Precision-Recall curve for YOLOv3.

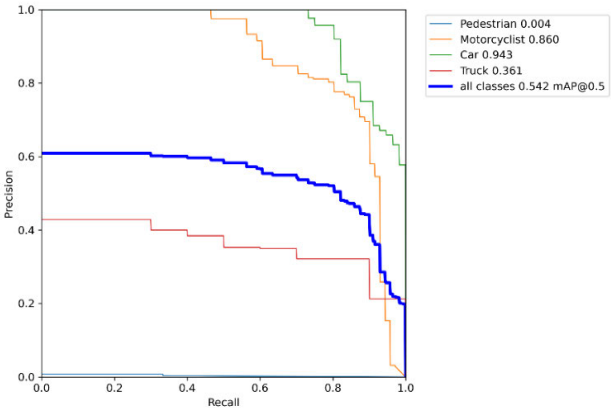

FIGURE 11. Precision-Recall curve for YOLOv5.

2012 metric [18]. We compared our methods with the original YOLOv3, YOLOv5, and YOLOv7 method for image object detection. The results presented in Table 2 demonstrate that YOLOv7-MOD exhibits a notable improvement in mAP, with an increase of 1.05% when compared to the original YOLOv7 algorithm. However, it is worth noting that this enhancement comes at a cost of a 16 FPS decrease. This reduction in

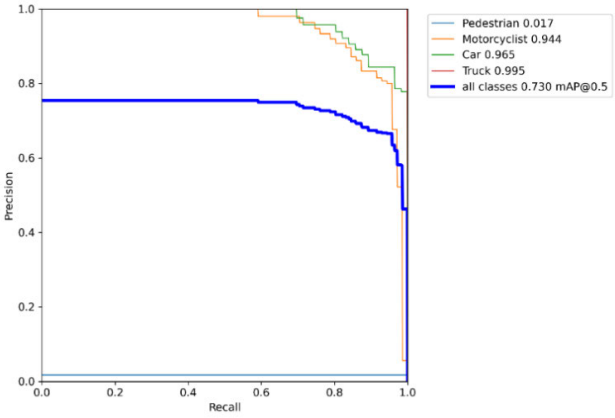

FIGURE 12. Precision-Recall curve for YOLOv7.

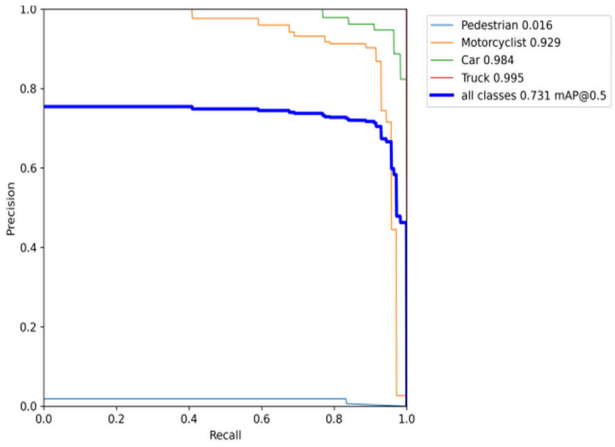

FIGURE 13. Precision-Recall curve for YOLOv7-MOD.

frames per second indicates that the deformable convolution module plays a crucial role in adaptively adjusting the convolution kernel's shape to effectively extract features, consequently leading to higher detection accuracy. These findings are further illustrated in Figure 14, which clearly depicts YOLOv7-MOD outperforming other methods in the test results.

Figure 15 shows comparison between the improved YOLOv7-MOD target detection algorithm and the original YOLO algorithm. From the comparison, it can be seen that for target with large scale change, original YOLOv7 extracts more background interference due to the low discrimination between the target and the background, which leads to the inaccurate detection box. YOLOv7-MOD can better distinguish the background and the result is more accurate. YOLOv7-MOD can successfully detect small targets. In the left image (a) there are several objects that are not detected by YOLOv7 but can be detected by YOLOv7-MOD, and the detection results can be shown in the right image (b). With the addition of a deformable layer, YOLOv7-MOD is able to detect small and occluded objects. However, the introduction of deformable convolution increases the calculation amount

FIGURE 14. Object detection result using YOLOv3 (a), YOLOv5 (b), YOLOv7 (c), YOLOv7-MOD (d).

FIGURE 15. Object detection result using YOLOv7 (a) and YOLOv7-MOD (b).

of the network, which will lead to the decrease of FPS. The more the deformable convolution layers are added, the higher the accuracy is and the slower the speed is. In general, the overall performance of the YOLOv7-MOD is better.

The results of precision, recall, F1-Score and mAP metric measurements obtained are shown in Table 2, where the Yolo modification with the addition parallel pooling and concatenation layer can improve F1-Score by 1.80% and mAP by 1.67% on YOLOv3. The experimental results further demonstrate that incorporating layer addition significantly enhances the performance of YOLOv5 and YOLOv7, yielding improvements of 0.33% and 0.49% in F1-Score, and 0.41% and 1.06% in mAP, respectively. With the addition of a deformable layer on Yolov7 it can produce a significant increase in performance when compared to the baseline of the object detection model.

Given that the discussion revolves around autonomous vehicles, in addition to using local datasets, we have also implemented the BDD100K dataset. The use of the BDD100K dataset with YOLOv7-MOD yields highly satisfactory object detection results as shown in Table 3. The extensive and diverse BDD100K dataset empowers the model to recognize objects effectively under various weather and traffic conditions. YOLOv7-MOD, with its efficient object detection architecture, delivers fast and accurate detections. During testing, the model exhibits resilience to significant environmental variations, including heavy traffic situations, adverse weather, and fluctuating lighting conditions. Evaluation results demonstrate high precision and

TABLE 2. Performance evaluation.

Model	P.	R	F1-Score	mAP	FPS
YOLOv3	92.26%	84.04%	88.34%	89.82%	22
YOLOv3-MOD	93.12%	87.35%	90.14%	91.49%	24
YOLOv5	95.50%	88.70%	92.32%	94.90%	**62**
YOLOv5-MOD	95.15%	91.23%	92.65%	95.31%	53
YOLOv7	95.80%	93.64%	95.17%	97.57%	60
YOLOv7-MOD	**96.87%**	94.68%	**95.76%**	**98.63%**	44
CenterNet	92.16%	**95.12%**	94.09%	95.41%	33
SSD	86.03%	82.40%	84.03%	89.03%	39
EfficientDet	87.58%	92.94%	90.11%	95.88%	26
RetinaNet	88.09%	89.27%	89.06%	94.07%	13

recall for various object classes, reflecting the model's ability to identify objects accurately. However, utilizing this dataset also demands substantial computational resources for swift training and inference. Additionally, fine-tuning of the model may be necessary for highly specific tasks. Nevertheless, the results of using the BDD100K dataset with YOLOv7-MOD showcase significant potential for applications in traffic monitoring, object recognition, and various other computer vision tasks. In conclusion, this combination proves effective for object detection in diverse real-world scenarios. Table 3 presents a model comparison between YOLOv7 and YOLOv7-mod. It is evident from this table that the YOLOv7-MOD model outperforms YOLOv7 when utilizing the BDD100K dataset.

TABLE 3. Performance evaluation between YOLOv7 and YOLOv7-mod using BDD100K.

Model	P	R	F1-Score	mAP
YOLOv7	85.73%	84.52%	85.15%	85.57%
YOLOv7-MOD	**85.81%**	**86.64%**	**86.27%**	**86.33%**

VI. CONCLUSION

This paper proposes a method to detect and classify objects in dense and mixed traffic areas based on YOLOv7-MOD, which is trained using transfer learning. In addition, the custom dataset was adapted to mixed and dense traffic, characterising a typical traffic condition in developing countries. The dataset was then used for object detection training, validation, and testing. In summary, there are two steps used for this experiment. First, object detection is performed using the YOLOv7-MOD model. By adding deformable layers and using the transfer learning that uses pre-trained kitty dataset, the model can achieve a higher precision score than by retraining the model from the beginning.

To enhance the accuracy of the YOLO algorithm in detecting irregular targets with varying scales and shapes, and to improve its capability in detecting small targets, we seek to address these challenges. This paper proposes a YOLO target detection algorithm with deformable convolution kernel which can adaptively adjust the shape of convolution kernel to extract features more effectively. The experimental results show that the improved model proposed in this paper has better effect than the original model, and effectively compensates for the detection defects of the original model. In the future, more efficient models can be explored to improve the detection speed while ensuring accuracy. The system can be trained with more dataset images to reduce imbalance data. Because the YOLOv7-MOD model has a few false positives, to increase the higher score on object detection, we might further improve the deep learning model to one that has better precision detection, in particular when encountering visually small, severely occluded, and blurry objects.

REFERENCES

[1] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, May 1996, doi: 10.1613/jair.301.
[2] T. Turay and T. Vladimirova, "Toward performing image classification and object detection with convolutional neural networks in autonomous driving systems: A survey," *IEEE Access*, vol. 10, pp. 14076–14119, 2022, doi: 10.1109/ACCESS.2022.3147495.
[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.
[4] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.
[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.
[6] M. Elgendy, *Deep learning for Vision Systems*. Shelter Island, NY, USA: Manning, 2020.
[7] A. A. Yilmaz, M. S. Guzel, I. Askerbeyli, and E. Bostanci, "A vehicle detection approach using deep learning methodologies," 2018, *arXiv:1804.00429*.
[8] W. Li, "Analysis of object detection performance based on faster R-CNN," *J. Phys., Conf.*, vol. 1827, no. 1, Mar. 2021, Art. no. 012085, doi: 10.1088/1742-6596/1827/1/012085.
[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Computer Vision—ECCV*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0ssss_2.
[10] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.
[11] Z. Liu and S. Wang, "Broken corn detection based on an adjusted YOLO with focal loss," *IEEE Access*, vol. 7, pp. 68281–68289, 2019, doi: 10.1109/ACCESS.2019.2916842.
[12] R. Qian, X. Lai, and X. Li, "3D object detection for autonomous driving: A survey," *Pattern Recognit.*, vol. 130, Oct. 2022, Art. no. 108796, doi: 10.1016/j.patcog.2022.108796.
[13] U. Nepal and H. Eslamiat, "Comparing YOLOv3, YOLOv4 and YOLOv5 for autonomous landing spot detection in faulty UAVs," *Sensors*, vol. 22, no. 2, p. 464, Jan. 2022, doi: 10.3390/s22020464.
[14] Z. Huang, J. Wang, X. Fu, T. Yu, Y. Guo, and R. Wang, "DC-SPP-YOLO: Dense connection and spatial pyramid pooling based YOLO for object detection," *Inf. Sci.*, vol. 522, pp. 241–258, Jun. 2020, doi: 10.1016/j.ins.2020.02.067.
[15] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[16] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944, doi: 10.1109/CVPR.2017.106.
[17] S. Liu, L. Li, J. Tang, S. Wu, and J.-L. Gaudiot, *Creating Autonomous Vehicle Systems* (Synthesis Lectures on Computer Science). Cham, Switzerland: Springer, 2020, doi: 10.1007/978-3-031-01805-3.
[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010, doi: 10.1007/s11263-009-0275-4.
[19] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "BDD100K: A diverse driving dataset for heterogeneous multitask learning," 2018, *arXiv:1805.04687*.
[20] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision—ECCV*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham, Switzerland: Springer, 2014, pp. 740–755, doi: 10.1007/978-3-319-10602-1_48.
[21] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.
[22] Q. Yang, Y. Zhang, W. Dai, and S. J. Pan, *Transfer Learning*, 1st ed. Cambridge, U.K.: Cambridge Univ. Press, 2020, doi: 10.1017/9781139061773.
[23] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.
[24] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable ConvNets v2: More deformable, better results," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9300–9308.
[25] S. A. Siddiqui, M. I. Malik, S. Agne, A. Dengel, and S. Ahmed, "DeCNT: Deep deformable CNN for table detection," *IEEE Access*, vol. 6, pp. 74151–74161, 2018.
[26] D. Cao, Z. Chen, and L. Gao, "An improved object detection algorithm based on multi-scaled and deformable convolutional neural networks," *Hum.-Centric Comput. Inf. Sci.*, vol. 10, no. 1, pp. 1–22, Dec. 2020.
[27] H. Whang, S. Zhang, and C. She, "YOLO target detection algorithm with deformable convoluiton," in *Proc. IEEE Int. Conf. Mechatronics Automat. (ICMA)*, Aug. 2021, pp. 768–772.

ARI WIBOWO received the bachelor's and master's degrees in electrical engineering and informatics from Institut Teknologi Bandung (ITB), Bandung, Indonesia, where he is currently pursuing the Ph.D. degree with the School of Electrical and Informatics Engineering.

Previously, he was a Lecturer with Batam State Polytechnic, with a focus on software engineering, data mining, and artificial intelligence. His research interests include computer vision, data mining, and artificial intelligence.

BAMBANG RIYANTO TRILAKSONO (Member, IEEE) received the degree in electrical engineering from Institut Teknologi Bandung (ITB), Bandung, Indonesia, and the master's and Ph.D. degrees in electrical engineering from Waseda University, Tokyo, Japan.

He is currently a Professor with the Control and Computer System Research Group, School of Electrical Engineering and Informatics, ITB. His research interests include control systems, robotics, and artificial intelligence.

EGI MUHAMMAD IDRIS HIDAYAT received the degree in electrical engineering from Institut Teknologi Bandung (ITB), Bandung, Indonesia, the master's degree in control and information systems from Universität Duisburg-Essen, and the Ph.D. degree in electrical engineering from Uppsala Universitet.

He has been teaching electrical engineering undergraduate and graduate courses with the School of Electrical Engineering and Informatics, ITB. His research interests include robotics and artificial intelligence.

RINALDI MUNIR received the degree, master's, and Ph.D. degrees in electrical engineering and informatics from Institut Teknologi Bandung (ITB), Bandung, Indonesia.

He has been teaching electrical engineering undergraduate and graduate courses with the School of Electrical Engineering and Informatics, ITB, since 1992. His research interests include computer vision, data structure, and digital image processing.